MINISTRY OF EDUCATION
AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Order Management System for Multi-Brand Kitchen Center

GFA23SE12	
Group Members	Thái Quốc Toàn – Team Leader – SE151272 Trần Phú Sơn – Team Member – SE150552 Lê Hồng Thành – Team Member – SE150023 Võ Vỹ Khang – Team Member – SE150103 Lê Xuân Bách – Team Member – SE150031
Supervisor	Mrs. Nguyễn Thị Cẩm Hương Mr. Lâm Hữu Khánh Phương
Ext Supervisor	N/A
Capstone Project code	FA23SE021

- Ho Chi Minh, Nov 2023-

Table Of Contents

Table Of Contents 1
Acknowledgements 24
Definition and Acronyms 25
I. Project Introduction 26
1. Overview 26
1.1. Project Information 26
1.2 Project Team 26
1.2.1. Supervisors 26
1.2.2. Team Members 26
2. Product Background 26
3. Existing Systems 27
4. Business Opportunity 27
5. Software Product Vision 27
6. Project Scope & Limitations 27
6.1. Major features 27
6.2. Limitations & Exclusions 28
6.2.1. Limitations 28
6.2.2. Exclusions 28
II. Project Management Plan 29
1. Overview 29
1.1. Scope & Estimation 29
1.2. Project Objectives 32
1.3. Project Risks 32
2. Management Approach 33
2.1. Project Process 33
2.2. Quality Management 34
2.3. Training Plan 34
3. Project Deliverables 35
4. Responsibility Assignments 35
5. Project Communications 36

6. Configuration Management 36
6.1. Document Management 36
6.2. Source Code Management 37
6.3. Tools & Infrastructures 37
6.3.1. Tools 37
6.3.2. Infrastructures 37
III. Software Requirement Specification 38
1. Product Overview 38
2. User Requirements 38
2.1. Actors 38
2.2. Use Case Diagram 39
2.2.1. MBKC Admin Use Case Diagram 39
2.2.2. Brand Manager Use Case Diagram 40
2.2.3. Kitchen Center Manager Use Case Diagram 41
2.2.4. Store Manager Use Case Diagram 42
2.2.5. Cashier Use Case Diagram 43
2.2.6. MBKC Bot Use Case Diagram 44
2.3. List of Use Case Descriptions 45
2.3.1. MBKC Admin Use Case Description 45
2.3.2. Brand Manager Use Case Description 46
2.3.3. Kitchen Center Manager Use Case Description 49
2.3.4. Store Manager Use Case Description 51
2.3.5. Cashier Use Case Description 52
2.3.6. MBKC Bot Use Case Description 54
3. Functional Requirements 55
3.1. System Functional Overview 55
3.1.1. Screen Flow 55
3.1.1.1. MBKC Admin Screen Flow 55
3.1.1.2. Kitchen Center Manager Screen Flow 56
3.1.1.3. Brand Manager Screen Flow 57
3.1.1.4. Cashier Screen Flow 58
3.1.1.5. Store Manager Screen Flow 59

3.1.2. Screen Descriptions 60
3.1.2.1. MBKC Admin Screen Description 60
3.1.2.2. Kitchen Center Manager Screen Description 61
3.1.2.3. Brand Manager Screen Descriptions 63
3.1.2.4. Cashier Screen Descriptions 66
3.1.2.5. Store Manager Descriptions 67
3.1.3. Screen Authorization 68
3.1.4. Non-Screen Functions 70
3.1.5. Entity Relationship Diagram 71
3.1.5.1. Entities Description 71
3.2. Functional requirements 73
3.2.1. Accounts Feature 73
3.2.2. Authentications Feature 73
3.2.3. Banking Accounts Feature 74
3.2.4. Brands Feature 75
3.2.5. Cashiers Feature 78
3.2.6. Categories Feature 80
3.2.7. Configurations Feature 82
3.2.8. Dashboards Feature 82
3.2.9. Kitchen Centers Feature 84
3.2.10. Money Exchanges Feature 86
3.2.11. Orders Feature 88
3.2.12. Partner Products Feature 92
3.2.13. Partners Feature 93
3.2.14. Products Feature 94
3.2.15. Shipper Payments Feature 98
3.2.16. Store Partners Feature 99
3.2.17. Stores Feature 101
3.2.18. User Devices Feature 102
3.2.19. Verifications Feature 103
3.2.20. Wallets Feature 103
4. Non-Functional Requirements 104

4.1. External Interfaces 104
4.1.1. User Interfaces 104
4.1.2. Communications Interfaces 104
4.2. Quality Attributes 104
4.2.1. Usability 104
4.2.2. Reliability 104
4.2.3. Performance 104
4.2.4. Security 104
5. Requirement Appendix 104
5.1. Business Rules 104
5.2. Application Messages List (Partial) 107
IV. Software Design Description 111
1. System Design 111
1.1. System Architecture Overview 111
1.1.1 Overall System Architecture 111
1.1.2 Detail System Architecture 112
1.1.2.1. Web Application Sub-system 112
1.1.2.2. Mobile Application Sub-system 113
1.1.2.3. MBKC API Sub-system 113
1.1.2.4. MBKC Private API Sub-system 114
1.1.2.5. MBKC Bot Sub-system 114
1.2. Package Diagram 115
1.2.1. Web Application Package Diagram 115
1.2.2. Mobile Application Package Diagram 117
1.2.3. MBKC API Package Diagram 119
1.2.4. MBKC Private API Package Diagram 121
1.2.2. MBKC Bot Package Diagram 123
2. Database Design 124
2.1. Physical Database 124
2.2. Data Dictionary 125
2.2.1. Accounts Table 125
2.2.2. Roles Table 125

2.2.3. UserDevices Table 125
2.2.4. BrandAccounts Table 126
2.2.5. Brands Table 126
2.2.6. Cashiers Table 126
2.2.7. CashierMoneyExchanges Table 127
2.2.8. Wallets Table 127
2.2.9. Configurations Table 127
2.2.10. Categories Table 127
2.2.11. ExtraCategories Table 128
2.2.12. StoreAccounts Table 128
2.2.13. KitchenCenters Table 128
2.2.14. KitchenCenterMoneyExchanges Table 129
2.2.15. MoneyExchanges Table 129
2.2.16. Transactions Table 130
2.2.17. Products Table 130
2.2.18. Stores Table 131
2.2.19. StoreMoneyExchanges Table 132
2.2.20. Orders Table 132
2.2.21. OrderDetails Table 133
2.2.22. PartnerProducts Table 133
2.2.23. OrderHistories Table 134
2.2.24. ShipperPayments Table 135
2.2.25. BankingAccounts Table 135
2.2.26. Partners Table 136
2.2.27. StorePartners Table 136
3. Detailed Design 137
3.1. Brand's Participation In MBKC System 137
3.1.1. Class Diagram 137
3.1.1.1. Class Specification 138
3.1.1.1.1. KitchenCentersController class 138
3.1.1.1.2. BrandsController class 138
3.1.1.1.3. StoresController class 138

3.1.1.1.4. IKitchenCenterService interface 139
3.1.1.1.5. CreateKitchenCenterRequest class 139
3.1.1.1.6. IBrandService interface 140
3.1.1.1.7. PostBrandRequest class 140
3.1.1.1.8. RegisterStoreRequest class 140
3.1.1.1.9. IStoreService interface 141
3.1.1.1.10. ConfirmStoreRegistrationRequest class 141
3.1.1.1.11. KitchenCenterService class 141
3.1.1.1.12. KitchenCenterStatus enumeration 142
3.1.1.1.13. KitchenCenterService class 142
3.1.1.1.14. BrandStatus enumeration 142
3.1.1.1.15. StoreService class 142
3.1.1.1.16. StoreStatus enumeration 143
3.1.1.1.17. IUnitOfWork interface 143
3.1.1.1.18. FirebaseStorageModel class 143
3.1.1.1.19. FirebaseStorageRepository class 144
3.1.1.1.20. UnitOfWork class 144
3.1.1.1.21. KitchenCenterRepository class 146
3.1.1.1.22. StoreRepository class 146
3.1.1.1.23. BrandRepository class 147
3.1.1.1.24. Email class 147
3.1.1.1.25. EmailRepository class 148
3.1.1.1.26. RoleRepository class 148
3.1.1.1.27. Role class 148
3.1.1.1.28. AccountRepository class 149
3.1.1.1.29. Account class 149
3.1.1.1.30. BrandAccount class 150
3.1.1.1.31. Brand class 150
3.1.1.1.32. Store class 151
3.1.1.1.33. KitchenCenter class 152
3.1.1.1.34. StoreAccount class 153
3.1.1.1.35. Wallet class 153

3.1.2. Activity Diagram 154
3.1.2.1. Create Kitchen Center 154
3.1.2.2. Create Brand 155
3.1.2.3. Register Store 156
3.1.2.4. Confirm Store Registration 157
3.1.3. Sequence Diagram 158
3.1.3.1. Create Kitchen Center 158
3.1.3.2. Create Brand 159
3.1.3.3. Register Store 160
3.1.3.4. Confirm Store Registration 161
3.1.4. State Machine Diagram 162
3.1.4.1. Store State Machine Diagram 162
3.2. Product Creation and Mapping 163
3.2.1. Class Diagram 163
3.2.1.1. Class Specifications 164
3.2.1.1.1. PostCategoryRequest class 164
3.2.1.1.2. CategoryRequest class 164
3.2.1.1.3. ExtraCategoryRequest class 164
3.2.1.1.4. CreateProductRequest class 164
3.2.1.1.5. PostStorePartnerRequest class 165
3.2.1.1.6. PartnerAccountRequest class 165
3.2.1.1.7. PostPartnerProductRequest class 166
3.2.1.1.8. CategoriesController class 166
3.2.1.1.9. ProductsController class 167
3.2.1.1.10. StorePartnersController class 167
3.2.1.1.11. PartnerProductsController class 167
3.2.1.1.12. ICategoryService interface 168
3.2.1.1.13. ProductStatus enumeration 168
3.2.1.1.14. IProductService interface 168
3.2.1.1.15. IStorePartnerService interface 168
3.2.1.1.16. IPartnerProductService interface 168
3.2.1.17. PartnerProductStatus enumeration 169

3.2.1.1.18. StoreService class 169
3.2.1.1.19. ProductType enumeration 169
3.2.1.1.20. ProductService class 170
3.2.1.1.21. StorePartnerService class 170
3.2.1.1.22. PartnerProductService class 170
3.2.1.1.23. IUnitOfWork interface 171
3.2.1.1.24. CategoryStatus enumeration 171
3.2.1.1.25. CategoryType enumeration 171
3.2.1.1.26. FirebaseStorageModel class 171
3.2.1.1.27. FirebaseStorageRepository class 172
3.2.1.1.28. Email class 172
3.2.1.1.29. EmailRepository class 172
3.2.1.1.30. UnitOfWork class 173
3.2.1.1.31. CategoryRepository class 175
3.2.1.1.32. Category class 175
3.2.1.1.33. ExtraCategoryRepository class 176
3.2.1.1.34. Brand class 177
3.2.1.1.35. ExtraCategory class 177
3.2.1.1.36. GrabFoodRepository class 178
3.2.1.1.37. ProductRepository class 178
3.2.1.1.38. Product class 178
3.2.1.1.39. Wallet class 180
3.2.1.1.40. StoreRepository class 180
3.2.1.1.41. StorePartnerRepository class 180
3.2.1.1.42. PartnerProductRepository class 181
3.2.1.1.43. Store class 181
3.2.1.1.44. Partner class 182
3.2.1.1.45. StorePartner class 182
3.2.1.1.46. PartnerProduct class 183
3.2.1.1.47. StoreAccount class 184
3.2.1.1.48. Account class 184
3.2.1.1.49. BrandAccount class 185

3.2.2. Activity Diagram 186
3.2.2.1. Create Normal Category 186
3.2.2.2. Create Extra Category 187
3.2.2.5. Add Extra Category to Normal Category 188
3.2.2.4. Create Product 189
3.2.2.5. Create Store Partner 190
3.2.2.6. Create Partner Product 191
3.2.3. Sequence Diagram 192
3.2.3.1. Create Normal / Extra Category 192
3.2.3.2. Add Extra Category to Normal Category 193
3.2.3.3. Create Product 194
3.2.3.4. Create Store Partner 195
3.2.3.5. Create Partner Product 196
3.2.4. State Machine Diagram 197
3.2.4.1. Partner Product State Machine Diagram 197
3.3. Getting, Parsing, and Confirming Orders from Partners 198
3.3.1. Class Diagram 198
3.3.1.1. MBKC Bot 198
3.3.1.1.1. Class Specification 199
3.3.1.1.1.1. Worker class 199
3.3.1.1.1.2. IconfigurationService interface 199
3.3.1.1.1.3. IRabbitMQService interface 200
3.3.1.1.1.4. IstoreService interface 200
3.3.1.1.1.5. IAuthenticationService interface 200
3.3.1.1.1.6. IOrderService interface 200
3.3.1.1.1.7. IEmailService interface 201
3.3.1.1.1.8. IuserDeviceService interface 201
3.3.1.1.1.9. ConfigurationService class 201
3.3.1.1.1.10. RabbitMQService class 201
3.3.1.1.1.11. AuthenticationService class 202
3.3.1.1.1.12. OrderService class 202
3.3.1.1.1.13. EmailService class 203

3.3.1.1.1.14. UserDeviceService Class 203
3.3.1.1.1.15. UnitOfWork class 204
3.3.1.1.1.16. ConfigurationRepository class 205
3.3.1.1.1.17. RabbitMQRepository class 205
3.3.1.1.1.18. StoreRepository class 205
3.3.1.1.1.19. GrabFoodRepository class 205
3.3.1.1.1.20. OrderRepository class 206
3.3.1.1.1.21. EmailRepository class 206
3.3.1.1.1.22. UserDeviceRepository class 206
3.3.1.1.1.23. Configuration class 207
3.3.1.1.1.24. RabbitMQ class 207
3.3.1.1.1.25. Store Class 207
3.3.1.1.1.26. StorePartner Class 207
3.3.1.1.1.27. Partner class 208
3.3.1.1.1.28. PartnerProduct class 208
3.3.1.1.1.29. Product Class 208
3.3.1.1.1.30. Order Class 209
3.3.1.1.1.31. OrderDetail Class 210
3.3.1.1.1.32. Email Class 211
3.3.1.2. MBKC Private API 212
3.3.1.2.1. Class Specification 213
3.3.1.2.1.1. UserDevicesController class 213
3.3.1.2.1.2. StoresController class 213
3.3.1.2.1.3. ConfigurationsController class 213
3.3.1.2.1.4. OrdersController class 214
3.3.1.2.1.5. IUserDevicceService interface 214
3.3.1.2.1.6. IStoreService interface 214
3.3.1.2.1.7. IConfigurationService interface 214
3.3.1.2.1.8. IOrderService interface 215
3.3.1.2.1.9. UserDevicceService class 215
3.3.1.2.1.10. StoreService class 215
3.3.1.2.1.11. ConfigurationService class 216

3.3.1.2.1.12. OrderService class 216
3.3.1.2.1.13. IUnitOfWork interface 216
3.3.1.2.1.14. Configuration class 217
3.3.1.2.1.15. ConfigurationRepository class 217
3.3.1.2.1.16. UnitOfWork class 217
3.3.1.2.1.17. UserDeviceRepository class 218
3.3.1.2.1.18. UserDevice class 218
3.3.1.2.1.19. Store class 219
3.3.1.2.1.20. OrderRepository class 219
3.3.1.2.1.21 Order class 220
3.3.1.2.1.22. OrderDetail class 222
3.3.1.2.1.23. Partner class 222
3.3.1.2.1.24. StorePartner class 223
3.3.1.2.1.25. OrderHistory class 223
3.3.1.2.1.26. PartnerProduct class 223
3.3.1.2.27 Product class 224
3.3.1.3. MBKC API 226
3.3.1.3.1. Class Specification 227
3.3.1.3.1.1. OrdersController class 227
3.3.1.3.1.2 IOrderService interface 227
3.3.1.3.1.3 OrderService class 227
3.3.1.3.1.4. IUnitOfWork interface 228
3.3.1.3.1.5. UnitOfWork class 228
3.3.1.3.1.6. OrderRepository class 228
3.3.1.3.1.7. Order class 229
3.3.1.3.1.8. OrderHistory class 230
3.3.2. Activity Diagram 232
3.3.2.1. Get and Parse Order Data from Partner 232
3.3.2.2. Confirm Finished of Preparing Orders 233
3.3.2.3. Confirm Status Order to Ready Delivery 234
3.3.3. Sequence Diagram 235
3.3.3.1. Get and Parse Order Data from Partner 235

3.3.3.2. Create New Order 236
3.3.2.3. Update Existed Order 237
3.3.3.4. Confirm Finished of Preparing Orders 238
3.3.3.5. Confirm Status Order to Ready Delivery 239
3.3.4. State Machine Diagram 240
3.3.4.1. Order State Machine Diagram 240
3.4. Transferring and Withdrawing Money from a Wallet 241
3.4.1. Class Diagram 241
3.4.1.1. Class Specifications 242
3.4.1.1.1. OrdersController class 242
3.4.1.1.2. MoneyExchangesController class 242
3.4.1.1.3. IOrderService interface 242
3.4.1.1.4. IMoneyExchangeService interface 243
3.4.1.1.5. OrderService class 243
3.4.1.1.6. MoneyExchangeService class 243
3.4.1.1.7. IUnitOfWork interface 244
3.4.1.1.8. UnitOfWork class 244
3.4.1.1.9. OrderHistoryRepository class 246
3.4.1.1.10. OrderRepository class 246
3.4.1.1.11. FirebaseStorageModel class 247
3.4.1.1.12. FirebaseStorageRepository class 247
3.4.1.1.13. StoreMoneyExchangeRepository class 247
3.4.1.1.14. KitchenCenterMoneyExchangeRepository class 248
3.4.1.1.15 CashierMoneyExchangeRepository class 248
3.4.1.1.16. KitchenCenterMoneyExchange class 248
3.4.1.1.17. StoreMoneyExchange class 249
3.4.1.1.18. WalletRepository class 249
3.4.1.1.19. CashierRepository class 249
3.4.1.1.20. CashierMoneyExchange class 250
3.4.1.1.21. Cashier class 250
3.4.1.1.22. ShipperPaymentRepository class 251
3.4.1.1.23. BankingAccount class 251

3.4.1.1.24. Order class 252
3.4.1.1.25. OrderHistory class 254
3.4.1.1.26. Wallet class 254
3.4.1.1.27. KitchenCenter class 255
3.4.1.1.28. ShipperPayment class 256
3.4.1.1.29. Transaction class 257
3.4.1.1.30. MoneyExchange class 258
3.4.2. Activity Diagram 259
3.4.2.1. Create Shipper Payment 259
3.4.2.2. Transfer Money from Cashier's Wallet to Kitchen Center 260
3.4.2.3. Withdraw money for Store 261
3.4.3. Sequence Diagram 262
3.4.3.1. Create Shipper Payment 262
3.4.3.2. Transfer Money from Cashier's wallet to Kitchen Center 263
3.4.3.3. Withdraw money for Store 264
V. Software Testing Documentation 265
1. Overall Description 265
1.1. Test Model 265
1.2. Test Levels 265
1.3. Testing Types 265
2. Test Plan 265
2.1. Test Stages 265
2.2. Test Environments 265
2.2. Human Resources 266
2.3. Test Milestones 266
3. Test Cases and Test Reports 266
VI. Release Package & User Guides 267
1. Deliverable Package 267
2. Installation Guides 267
2.1. System Requirements 267
2.2. Installation Instruction 268
2.2.1. Environment Setup 268

2.2.2. Run Backend Server 269
2.2.3. Run Frontend web application 271
2.2.4. Run Frontend mobile application 272
2.2.5. Run MBKC Bot 274
3. Uer Manual 276
3.1. Overview 276
3.2. Web Application 276
3.2.1. Authentication 276
3.2.1.1 Login 276
3.2.1.1.1 Description 276
3.2.1.1.2. Guide Steps 276
3.2.1.2. Reset Password 277
3.2.1.2.1. Description 277
3.2.1.2.2. Guide Steps 278
3.2.1.3. Update Password in First Time Login 283
3.2.1.3.1. Description 283
3.2.1.3.2. Guide Steps 284
3.2.1.4. Update Password 285
3.2.1.4.1. Description 285
3.2.1.4.2. Guide Steps 285
3.2.2. Kitchen center management 287
3.2.2.1. Create new Kitchen Center 287
3.2.2.1.1. Description 287
3.2.2.1.2. Guide Steps 287
3.2.2.2. View list Kitchen Center 291
3.2.2.2.1. Description 291
3.2.2.2.2. Guide Steps 291
3.2.2.3. View details Kitchen Center 294
3.2.2.3.1. Description 294
3.2.2.3.2. Guide Steps 294
3.2.2.4. Update Status Kitchen Center 297
3.2.2.4.1. Description 297

3.2.2.4.2. Guide Steps 297
3.2.2.5. Update Information Kitchen Center 299
3.2.2.5.1. Description 299
3.2.2.5.2. Guide Steps 299
3.2.2.6. Delete Kitchen Center 305
3.2.2.6.1. Description 305
3.2.2.6.2. Guide Steps 305
3.2.3. Brand management 309
3.2.3.1. Create new Brand 309
3.2.3.1.1. Description 309
3.2.3.1.2. Guide Steps 309
3.2.3.2. View list Brand 313
3.2.3.2.1. Description 313
3.2.3.2.2. Guide Steps 313
3.2.3.3. View details Brand 316
3.2.3.3.1. Description 316
3.2.3.3.2. Guide Steps 316
3.2.3.4. Update Status Brand 318
3.2.3.4.1. Description 318
3.2.3.4.2. Guide Steps 318
3.2.3.5. Update Information Brand 321
3.2.3.5.1. Description 321
3.2.3.5.2. Guide Steps 321
3.2.3.6. Delete Brand 327
3.2.3.6.1. Description 327
3.2.3.6.2. Guide Steps 327
3.2.4. Store management 331
3.2.4.1. Register new Store 331
3.2.4.1.1. Description 331
3.2.4.1.2. Guide Steps 331
3.2.4.2. View list Store 335
3.2.4.2.1. Description 335

3.2.4.2.2. Guide Steps 335
3.2.4.3. View details Store 338
3.2.4.3.1. Description 338
3.2.4.3.2. Guide Steps 338
3.2.4.4. Confirm Store Registration 341
3.2.4.4.1. Description 341
3.2.4.4.2. Guide Steps 341
3.2.4.5. Update Status Store 346
3.2.4.5.1. Description 346
3.2.4.5.2. Guide Steps 346
3.2.4.6. Update Information Store 349
3.2.4.6.1. Description 349
3.2.4.6.2. Guide Steps 349
3.2.4.7. Delete store 355
3.2.4.7.1. Description 355
3.2.4.7.2. Guide Steps 355
3.2.5. Partner management 359
3.2.5.1. View list Partner 359
3.2.5.1.1. Description 359
3.2.5.1.2. Guide Steps 359
3.2.5.2. View details Partner 362
3.2.5.2.1. Description 362
3.2.5.2.2. Guide Steps 363
3.2.5.3. Update Status Partner 365
3.2.5.3.1. Description 365
3.2.5.3.2. Guide Steps 366
3.2.5.4. Update Information Partner 368
3.2.5.4.1. Description 368
3.2.5.4.2. Guide Steps 368
3.2.6. Normal Category management 373
3.2.6.1. Create new Normal Category 373
3.2.6.1.1. Description 373

3.2.6.1.2. Guide Steps 373
3.2.6.2. View list of Normal Categories 377
3.2.6.2.1. Description 377
3.2.6.2.2. Guide Steps 377
3.2.6.3. View details Normal Category 381
3.7.3.1. Description 381
3.2.6.3.2. Guide Steps 381
3.2.6.4. Add Extra Category to Normal Category 385
3.2.6.4.1. Description 385
3.2.6.4.2. Guide Steps 385
3.2.6.5. Update Status Normal Category 393
3.2.6.5.1. Description 393
3.2.6.5.2. Guide Steps 393
3.2.6.6. Update Information Normal Category 396
3.2.6.6.1. Description 396
3.2.6.6.2. Guide Steps 396
3.2.6.7. Delete Normal Category 402
3.2.6.7.1. Description 402
3.2.6.7.2. Guide Steps 402
3.2.7. Extra category management 406
3.2.7.1. Create new Extra Category 406
3.2.7.1.1. Description 406
3.2.7.1.2. Guide Steps 406
3.2.7.2. View list of Extra Categories 410
3.2.7.2.1. Description 410
3.2.7.2.2. Guide Steps 410
3.2.7.3. View details Extra Category 414
3.2.7.3.1. Description 414
3.2.7.3.2. Guide Steps 414
3.2.7.4. Update Status Extra Category 417
3.2.7.4.1. Description 417
3.2.7.4.2. Guide Steps 417

3.2.7.5. Update Information Extra Category 420
3.2.7.5.1. Description 420
3.2.7.5.2. Guide Steps 420
3.2.7.6. Delete Extra Category 426
3.2.7.6.1. Description 426
3.2.7.6.2. Guide Steps 426
3.2.8. Product management 430
3.2.8.1. Create new Product 430
3.2.8.1.1. Description 430
3.2.8.1.2. Guide Steps 430
3.2.8.2. View list of Products 437
3.2.8.2.1. Description 437
3.2.8.2.2. Guide Steps 437
3.2.8.3. View details Product 441
3.2.8.3.1. Description 441
3.2.8.3.2. Guide Steps 441
3.2.8.4. Update Status Product 444
3.2.8.4.1. Description 444
3.2.8.4.2. Guide Steps 444
3.2.8.5. Update Information Product 447
3.2.8.5.1. Description 447
3.2.8.5.2. Guide Steps 447
3.2.8.6. Delete Product 452
3.2.8.6.1. Description 452
3.2.8.6.2. Guide Steps 452
3.2.9. Store Partner management 456
3.2.9.1. Create new Store Partner 456
3.2.9.1.1. Description 456
3.2.9.1.2. Guide Steps 456
3.2.9.2. View list Store Partner 462
3.2.9.2.1. Description 462
3.2.9.2.2. Guide Steps 462

3.2.9.3. Update Status Store Partner 466
3.2.9.3.1. Description 466
3.2.9.3.2. Guide Steps 466
3.2.9.4. Update Information Store Partner 471
3.2.9.4.1. Description 471
3.2.9.4.2. Guide Steps 471
3.2.9.5. Delete Store Partner 477
3.2.9.5.1. Description 477
3.2.9.5.2. Guide Steps 477
3.2.10. Partner Product management 483
3.2.10.1. Create new Partner Product 483
3.2.10.1.1. Description 483
3.2.10.1.2. Guide Steps 483
3.2.10.2. View list of Partner Products 487
3.2.10.2.1. Description 487
3.2.10.2.2. Guide Steps 487
3.2.10.3. View details Partner Product 490
3.2.10.3.1. Description 490
3.2.10.3.2. Guide Steps 490
3.2.10.4. Update Status Partner Product 493
3.2.10.4.1. Description 493
3.2.10.4.2. Guide Steps 493
3.2.10.5. Update Information Partner Product 497
3.2.10.5.1. Description 497
3.2.10.5.2. Guide Steps 497
3.2.10.6. Delete Partner Product 503
3.2.10.6.1. Description 503
3.2.10.6.2. Guide Steps 503
3.2.11. Cashier management 508
3.2.11.1. Create new Cashier 508
3.2.11.1.1. Description 508
3.2.11.1.2. Guide Steps 508

3.2.11.2. View list of Cashiers 512
3.2.11.2.1. Description 512
3.2.11.2.2. Guide Steps 512
3.2.11.3. View details Cashier 515
3.2.11.3.1. Description 515
3.2.11.3.2. Guide Steps 515
3.2.11.4. Update Status Cashier 518
3.2.11.4.1. Description 518
3.2.11.4.2. Guide Steps 518
3.2.11.5. Update Information Cashier 521
3.2.11.5.1. Description 521
3.2.11.5.2. Guide Steps 521
3.2.11.6. Delete Cashier 527
3.2.11.6.1. Description 527
3.2.11.6.2. Guide Steps 527
3.2.12. Banking Account management 531
3.2.12.1. Create new Banking Account 531
3.2.12.1.1. Description 531
3.2.12.1.2. Guide Steps 531
3.2.12.2. View list of Banking Accounts 535
3.2.12.2.1. Description 535
3.2.12.2.2. Guide Steps 535
3.2.12.3. View details Banking Account 539
3.2.12.3.1. Description 539
3.2.12.3.2. Guide Steps 539
3.2.12.4. Update Status Banking Account 542
3.2.12.4.1. Description 542
3.2.12.4.2. Guide Steps 542
3.2.12.5. Update Information Banking Account 545
3.2.12.5.1. Description 545
3.2.12.5.2. Guide Steps 545
3.2.12.6. Delete Banking Account 550

3.2.12.6.1. Description 550
3.2.12.6.2. Guide Steps 550
3.2.13. Order management 554
3.2.13.1. View list of Orders 554
3.2.13.1.1. Description 554
3.2.13.1.2. Guide Steps 554
3.2.13.2. View details Order 559
3.2.13.2.1. Description 559
3.2.13.2.2. Guide Steps 559
3.2.13.3. Confirm Order Ready to Delivery 562
3.2.13.3.1. Description 562
3.2.13.3.2. Guide Steps 562
3.2.13.4. Create Shipper Payment 566
3.2.13.4.1. Description 566
3.2.13.4.2. Guide Steps 567
3.2.14. Wallet management 573
3.2.14.1. View wallet of cashier 573
3.2.14.1.1. Description 573
3.2.14.1.2. Guide Steps 573
3.2.14.2. View wallet of kitchen center 576
3.2.14.2.1. Description 576
3.2.14.2.2. Guide Steps 576
3.2.14.3. View list of transactions 579
3.2.14.3.1. Description 579
3.2.14.3.2. Guide Steps 579
3.2.14.4. View list of shipper payments 586
3.2.14.4.1. Description 586
3.2.14.4.2. Guide Steps 586
3.2.14.5. Transfer Money to Kitchen Center 592
3.2.14.5.1. Description 592
3.2.14.5.2. Guide Steps 592
3.2.14.6. Create Payment for Store 596

3.2.14.5.1. Description 596
3.2.14.5.2. Guide Steps 596
3.2.15. Update Configuration 600
3.2.15.1. Description 600
3.2.15.2. Guide Steps 601
3.3. Mobile Application 606
3.3.1. Authentication 606
3.3.1.1 Login 606
3.3.1.1.1 Description 606
3.3.1.1.2. Guide Steps 606
3.3.1.2. Reset Password 608
3.3.1.2.1. Description 608
3.3.1.2.2. Guide Steps 608
3.3.1.3. Update Password in First Time Login 612
3.3.1.3.1. Description 612
3.3.1.3.2. Guide Steps 612
3.2.1.4. Update Password 613
3.2.1.4.1. Description 613
3.2.1.4.2. Guide Steps 613
3.3.2. Category management 615
3.3.2.1. View list of Categories 615
3.3.2.1.1. Description 615
3.3.2.1.2. Guide Steps 615
3.3.2.2. View list of Extra Categories 619
3.3.2.2.1. Description 619
3.3.2.2.2. Guide Steps 619
3.3.2.3. View details Category 621
3.3.2.3.1. Description 621
3.3.2.3.2. Guide Steps 621
3.3.3. Product management 624
3.3.3.1. View list of Products 624
3.3.3.1.1. Description 624

3.3.3.1.2. Guide Steps 624
3.3.3.2. View details Product 629
3.3.3.2.1. Description 629
3.3.3.2.2. Guide Steps 629
3.3.4. Order management 634
3.3.4.1. View list of Orders 634
3.3.4.1.1. Description 634
3.3.4.1.2. Guide Steps 634
3.3.4.2. View details Order 637
3.3.4.2.1. Description 637
3.3.4.2.2. Guide Steps 637
3.3.4.3. Confirm Order to Ready 640
3.3.4.3.1. Description 640
3.3.4.3.2. Guide Steps 640
3.3.4.4. Cancel Order 643
3.3.4.4.1. Description 643
3.3.4.4.2. Guide Steps 643
3.3.3. Wallet management 647
3.3.3.1. View wallet 647
3.3.3.1.1. Description 647
3.3.3.1.2. Guide Steps 647
3.3.3.2. View details Transaction 650
3.3.3.2.1. Description 650
3.3.3.2.2. Guide Steps 650

Acknowledgements

After four years of studying and practicing at FPT University with a major in Software Engineering, this capstone project is an important milestone that marks the completion of our team members' duties in the university lecture hall.

First and foremost, we are extremely grateful for our family. They are the driving force behind our daily efforts. Without their sacrifices, we would not be where we are today. We are much obliged to them for nurturing, teaching, caring for, and investing in us. We want to dedicate this achievement to them, and we want them to know that their sacrifices throughout the years were not in vain.

At the same time, we would like to offer our heartfelt appreciation to the two supervisors on our graduation project. Mrs. Nguyễn Thị Cẩm Hương and Mr. Lâm Hữu Khánh Phương are their names. They have been an endless supply of inspiration and support, as well as constructive feedback and useful suggestions. They were also incredibly generous with their time and knowledge, and they were always accessible to answer our inquiries and resolve our concerns. We are quite grateful for their guidance and expertise.

Besides, we would like to thank Mrs. Võ Thị Thanh Vân, Mr. Trần Thanh Nguyên, Mr. Nguyễn Ngọc Lâm, Mr. Đỗ Tấn Nhàn, and all the other teachers at FPT University for helping us from zero in the past four years. Now, we have enough knowledge and qualifications to complete our work. This product is not only the result of our efforts but also of all the teachers' efforts. We are beyond grateful.

Finally, my name is Thái Quốc Toàn, and I am the team leader for the MBKC project. I truly thank everyone on the team. Thank you for giving it your all, despite the fact that our job is extremely stressful and difficult. Thank you for persevering through the most difficult stage of the project. I am extremely grateful and proud of you guys. I am really happy and excited to work with you guys.

Definition and Acronyms

Acronym	Definition
AWS	Amazon Web Services
BA	Business Analysis
BR	Business Rule
ERD	Entity Relationship Diagram
GUI	Graphical User Interface
PM	Project Manager
SDD	Software Design Description
SPMP	Software Project Management Plan
SRS	Software Requirement Specification
UAT	User Acceptance Test
UC	Use Case
API	Application Program Interface
Multi-Brand Kitchen Center Model	A multi-brand kitchen center model is a kitchen center model where the kitchen space is used by many brands. This kitchen center model is just a place to carry out processing of brands.

Table 1: Definition and Acronyms

I. Project Introduction

1. Overview

1.1. Project Information

- Project name: Order Management System for Multi-Brand Kitchen Center
- Project code: FA23SE021
- Group name: GFA23SE12
- Software type: Web application, Mobile application, Worker service, WPF Application

1.2 Project Team

1.2.1. Supervisors

Full Name	Phone Number	Email	Title
Nguyễn Thị Cẩm Hương	0909912347	huongntc2@fpt.edu.vn	Mrs.
Lâm Hữu Khánh Phương	0915353001	phuonglhk@fpt.edu.vn	Mr.

Table 2: Supervisors

1.2.2. Team Members

Full Name	Phone Number	Email	Role
Thái Quốc Toàn	0886448660	toantqse151272@fpt.edu.vn	Leader
Trần Phú Sơn	0914360736	sontpse150552@fpt.edu.vn	Member
Lê Hồng Thành	0372227898	thanhlhse150023@fpt.edu.vn	Member
Võ Vỹ Khang	0944586962	khangvvse150103@fpt.edu.vn	Member
Lê Xuân Bách	0982856649	bachlxse150031@fpt.edu.vn	Member

Table 3: Team Members

2. Product Background

Currently, in Vietnam, the need for online business on e-commerce sites is becoming a trend and is being used by many businesses in the culinary field to assist grow revenue and bring their products closer to their customers.

A food business may be using multiple e-commerce systems, such as GrabFood, Baemin, etc., at the same time. Therefore, order management on each e-commerce system is being done separately, with no consistency in the process of receiving orders from customers.

Besides, with the kitchen center model, businesses in the culinary field can quickly open a new store because the kitchen center model supports them with a kitchen space with full kitchen equipment. This can help them reduce the cost of renting space and buying kitchen appliances.

Therefore, this MBKC (Multi-Brand Kitchen Center) system was born with the purpose of helping businesses in the culinary field solve the most important problem, which is unifying the order management process from e-commerce systems. Besides, the multi-brand kitchen center model is applied in the system, which helps them reduce costs when opening a new store.

3. Existing Systems

Currently, in Vietnam, there is still no system to support businesses in the culinary field in optimizing the order management process from e-commerce systems. In addition, businesses in the culinary field are also supported in opening stores quickly when agreeing to participate in the system with the applied kitchen center model.

4. Business Opportunity

Currently, in Vietnam, there are more and more large and small business units appearing in the culinary field. They have become partners and conduct sales on e-commerce sites such as GrabFood, Baemin, etc. Accessing and managing order information on multiple e-commerce systems at the same time is still limited. The order management system for the MBKC system with the multi-brand kitchen center model is a bridge between business units and the e-commerce system they use to support order management.

5. Software Product Vision

MBKC system was built to serve the multi-brand kitchen center model with the aim of providing kitchen space for business units that are lacking or want to quickly open a store without worrying about costs.

MBKC is an order management system with the desire to connect business units in the culinary field with a system of e-commerce such as Grabfood, Baemin, etc. MBKC System helps business units that they can unify the order and product management processes on the MBKC system without having to perform on each e-commerce system individually as they did before.

6. Project Scope & Limitations

6.1. Major features

As the MBKC system administrator, I want to manage the kitchen centers so that I can know how many kitchen centers are operating in the system. I want to manage brands that are business units so that I can know how many brands have joined the system. I want to manage registered stores so that I can know the list of stores that brands have requested to open in a specific kitchen center. I want to manage the e-commerce partners so that I can know how many the e-commertce partner system can be connected to the MBK System.

As a kitchen center manager, I want to manage the stores so that I can know how many stores are operating inside the kitchen center. I want to manage the kitchen center's cashiers so that I can know the information of cashiers about those who are working in the kitchen center. I want to manage the bank accounts that are used by the kitchen center so that the shipper from the e-commerce system can pay when they receive an order. I want to manage all order information of stores that are operating in the kitchen center so that I can know how many orders that kitchen center collected from e-commerce systems. I want to manage the wallet of the kitchen center so that I can keep track of how much money is received from cashiers and how much money is sent to stores in the kitchen center. I want to create remittance information for the stores in the kitchen center so that I can record that I have transferred money to the stores they sold today.

As a brand manager, I want to manage the registered stores so that I can know the store information at kitchen centers where stores are registered successfully. I want to manage the categories and products so that I can know about all the categories and products that the brand has in its inventory. I want to manage product

mapping so that I can map between products in the MBKC system and products in the e-commerce systems, which helps the MBKC system collect orders from e-commerce systems correctly. I want to manage the partners of the store that are e-commerce systems used by the store so that I can provide accounts to help the MBKC System access the e-commerce systems to collect orders.

As a cashier, I want to manage an order information so that I can give the finished order to the shipper of the e-commerce systems. I want to create payment for orders when orders were not paid by customers before so that I can stand for stores in the kitchen center to receive money from the shipping unit of e-commerce systems. I want to end my shift so that I can check the money I have collected today and confirm the end of the shift to send the money to the wallet of kitchen center. I want to manage my wallet so that I can keep track of how much money I have collected from shipping units each day.

As a store manager, I want to view products to know what products the store is selling. I want to manage order information so that I can perform accurate processing following order and know the amount of money collected from the order. I want to manage wallet of store so that I can track balance of wallet. At the same time, I can also view transactions receiving money from the kitchen center. I want to confirm the order has been finished so that I can notify the shipping unit from the e-commerce systems so they can come to pick up the order.

6.2. Limitations & Exclusions

6.2.1. Limitations

- This system is not the best and most optimal solution.
- The mobile application only supports Android OS.
- There is only one admin in the system.
- The MBKC system is currently unable to connect to several e-commerce services. Currently, the system can only connect to GrabFood.
- GrabFood has only granted the MBKC system a few APIs, such as Login and Getting Menu. GrabFood has secured all of the remaining APIs.

6.2.2. Exclusions

- The system does not manage the registration process of brands with MBKC admin.
- The MBKC system does not calculate commissions from store orders.

II. Project Management Plan

1. Overview

1.1. Scope & Estimation

No	WBS Item	Complexity	Est. Effort (man-days)
1	**Project Initiating**		**13**
1.1	Define background and context of project	Complex	7
1.2	Find problems and solutions of project	Medium	3
1.3	Determine project scope	Medium	2
1.4	List requirements	Medium	1
2	**Project Planning**		**55**
2.1	Technology Training		18
2.1.1	Learn ASP.NET Core 6 Web API with Entity Framework	Complex	6
2.1.2	Learn React Framework	Complex	6
2.1.3	Learn Flutter	Complex	6
2.2	Document writing		35
2.2.1	Write project introduction	Medium	2
2.2.2	Write project management plan	Medium	3
2.2.3	Write software requirement specification	Complex	6
2.2.4	Write software design document	Complex	6
2.2.5	Write test document	Medium	4
2.2.6	Write user guide document	Medium	4
2.2.7	Write final document	Complex	10
2.3	Team meeting	Simple	1
2.4	Supervisor meeting	Medium	1
2.5	Create Source Base	Complex	6
3	**Project Executing**		**169**
3.1	Implement authentication		10
3.1.1	Login and logout	Simple	3
3.1.2	Forgot password	Simple	3
3.1.3	Re-generate tokens	Simple	4
3.2	Implement brand management		11
3.2.1	Create a new brand	Medium	3
3.2.2	Update an existed brand information	Medium	3
3.2.3	View list of brands and view brand details	Simple	1
3.2.4	Update a brand status	Medium	2
3.2.5	Delete an existed brand	Medium	2

3.3	Implement kitchen center management		11
3.3.1	Create a new kitchen center	Medium	3
3.3.2	Update an existed kitchen center information	Medium	3
3.3.3	View list of kitchen centers and view kitchen center details	Simple	1
3.3.4	Update a kitchen center status	Medium	2
3.3.5	Delete an existed kitchen center	Medium	2
3.4	Implement partner management		5
3.4.1	Update an existed partner information	Medium	2
3.4.2	View list of partners and view partner details	Simple	1
3.4.3	Update a partner status	Medium	2
3.5	Implement store management		13
3.5.1	Register a new store	Medium	3
3.5.2	Confirm a new store registration	Medium	2
3.5.3	View list of stores and view store details	Simple	1
3.5.4	Update an existed store information	Medium	3
3.5.5	Update a store status	Medium	2
3.5.6	Delete an existed store	Medium	2
3.6	Implement category management		11
3.6.1	Create a new category	Medium	2
3.6.2	View list of categories and view category details and get extra category details	Medium	2
3.6.3	Update an existed category information	Medium	2
3.6.4	Update a category status	Medium	1
3.6.5	Add extra category in normal category	Medium	2
3.6.6	Delete an existed category	Medium	2
3.7	Implement product management		16
3.7.1	Create a new product	Complex	4
3.7.2	Update an existed product information	Complex	4
3.7.3	View list of products and product details	Medium	3
3.7.4	Update a product status	Medium	2
3.7.5	Delete an existed product	Medium	2
3.8	Implement store partner management		9
3.8.1	Create a new store partner	Complex	4
3.8.2	View list of store partners and view store partner detail	Simple	1
3.8.3	Update an existed store partner information	Medium	2
3.8.4	Update an existed store partner status	Medium	1
3.8.5	Delete an existed store partner	Medium	1
3.9	Implement account management		4
3.9.1	Get account profile	Medium	2
3.9.2	Update account profile	Medium	2
3.10	Implement cashier management		9

3.10.1	Create a new cashier	Medium	3
3.10.2	View list of cashiers and view cashier details	Simple	1
3.10.3	Update an existed cashier information	Medium	3
3.10.4	Update a cahier status	Medium	1
3.10.5	Delete an existed cashier	Mediun	1
3.11	Implement partner product management		10
3.11.1	View list of partner products and view partner product details	Medium	2
3.11.2	Create a new partner product	Complex	3
3.11.3	Update a partner product status	Medium	2
3.11.4	Update a partner product information	Medium	2
3.11.5	Delete an existed partner product	Simple	1
3.12	Implement order management		25
3.12.1	MBKC bot collect order from GrabFood	Complex	6
3.12.2	MBKC bot parse order from GrabFood to MBKC System	Complex	4
3.12.3	Confirm finished order	Medium	2
3.12.4	Cancel order	Medium	2
3.12.5	Confirm ready delivery	Medium	2
3.12.6	View list orders and view order details	Simple	2
3.12.7	Create Shipper payment	Complex	4
3.12.8	Create End of Shift	Medium	3
3.13	Implement wallet management		14
3.13.1	View wallet information	Medium	2
3.13.2	Send money from cashier to kitchen center automatically	Medium	4
3.13.3	Send money from kitchen center to store automatically	Medium	4
3.13.4	Withdraw money for store	Medium	4
3.14	Multiple language for web application	Medium	6
3.15	Send notification	Medium	6
3.16	Send Email	Medium	4
3.17	Design UI	Complex	15
4	**Testing**		**32**
4.1	Create test cases	Medium	8
4.2	API testing	Complex	10
4.3	System testing	Complex	10
4.4	Create testing document	Complex	5
5	**Closing**		**10**
5.1	Create Final Project Report	Complex	10
		Total Estimated Effort (man-days)	***279***

Table 4- Scope & Estimation

1.2. Project Objectives

No	Testing Stage	Test Coverage	No. of Defects	% of Defect	Notes
1	Reviewing	100	10	10%	
2	Integration Test	150	20	13,3%	
3	System Test	200	25	12,5%	

Table 5- Project Objectives

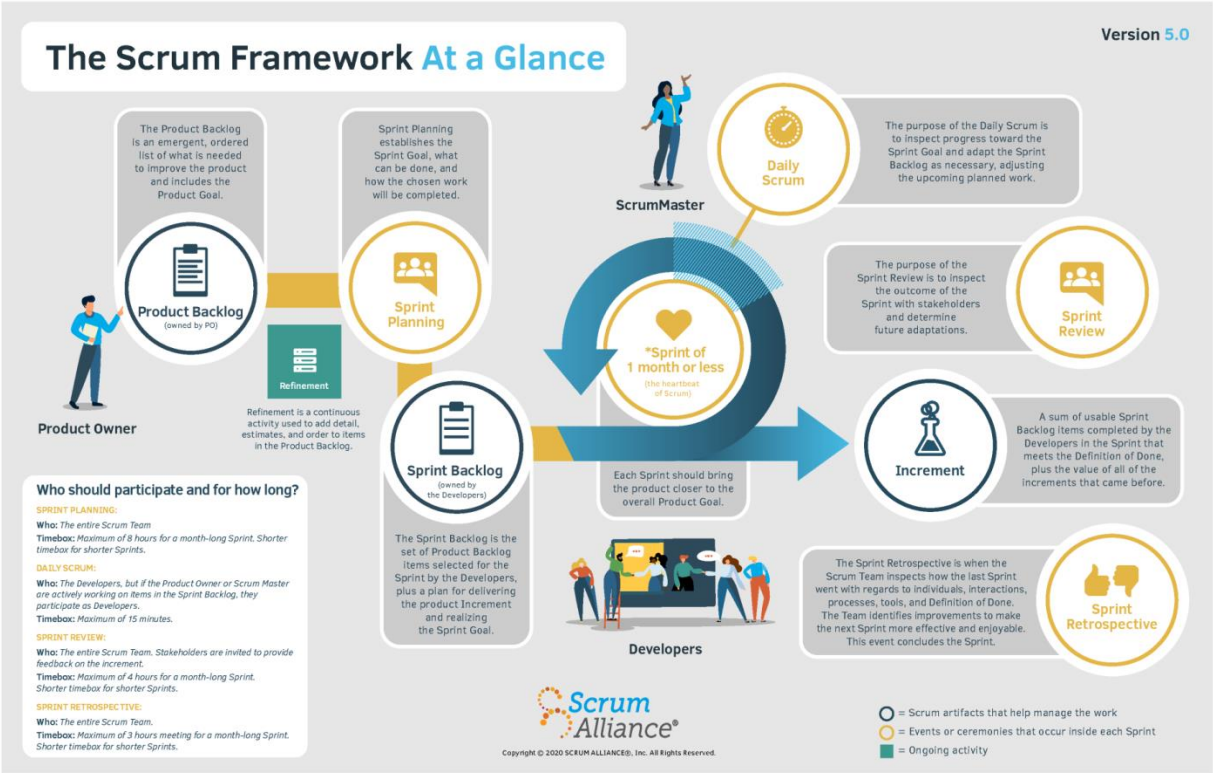
1.3. Project Risks

No	Risk Description	Impact	Possibility	Response Plans
1	Team member is illness	Medium	Low	Give tasks for each member, not too big or too small. So that they can catch up when they recover.
2	The team member's computer is possible to be damaged	Medium	Low	Need to fix the computer or buy a new one if it cannot be fixed.
3	Conflict between team members	High	Medium	Ask the team leader or supervisor for advice.
4	Requirement changes: requirements unexpected changes	High	High	Chose another way of solution to respond to those changes.
5	Technology risk: cannot find the suitable solution for technique problems, the software cannot fulfill the expectation	High	Medium	Do more research on official resources, discuss with mentor.

Table 4: Project Risks

2. Management Approach

2.1. Project Process



Source*: https://www.scrumalliance.org/about-scrum*

Agile methodology's Scrum framework is used to construct this project. We chose this process model because:

- Customers rarely know exactly how they want their system to look; their requirements change often, and we need to develop from part to part to produce the best process for our system to satisfy the customers.
- Our team is using Agile for the system testing to receive feedback daily and respond to it right away.
- Because no one on our team has any prior experience developing a system following a business model and connecting it with other e-commerce systems, we must speak with clients directly, conduct research, and work quickly to produce it so that we may receive input for improvement.

2.2. Quality Management

In this project, we employ the following strategy to raise project quality:

- API convention:
  - Name the API Route that corresponds to the system entity.
  - The response of APIs must be returned following:
    - 200, 400, 401, and 404 are the status codes used in the system.
    - Message body must be returned in JSON format.
- Coding convention:
  - Set the meaningful names for variables.
  - Add comments to code can be confusing.
  - Use Pascal case for naming Class, Interface, Method, Property, Constant.
  - Use Camel case for naming Variable.
  - Use Screaming snake case for naming Enum.
- API testing:
  - Member of the server-side team who have finished creating one module or screen must conduct API testing to guarantee the module can function properly by Postman tool.
- System testing:
  - All team members must run all test cases and free tests again after completing the code for the entire system to make sure the system is operating smoothly.

2.3. Training Plan

Training Area	Participants	When, Duration	Waiver Criteria
ASP.NET Web API Core 6 with Entity Framework	Thái Quốc Toàn Lê Xuân Bách	Week 1 – 6 days	Mandatory
ReactJS	Trần Phú Sơn Võ Vỹ Khang	Week 1 – 6 days	Mandatory
Flutter	Lê Hồng Thành	Week 1 – 6 days	Mandatory
SQL Server	All members	Week 1 – 6 days	Mandatory
GitHub	All members	Week 1 – 6 days	Mandatory
Jira	All members	Week 1 – 6 days	Mandatory
Figma	All members	Week 1 – 6 days	Mandatory
Research connecting to e-commerce systems	All members	Week 2 – 3 days	Mandatory

Table 5: Training Plan

3. Project Deliverables

Sprint	Sprint objective	Duration	Deliverable
1	Project Plan Document, SRS Document	14 days	Project Planning, Software Requirement
2	Design Document	14 days	Architecture Design, Basic Design, Detail Design and Database Design
3	Authentication module, Brand management module, Kitchen center management module	14 days	Code & API test cases & System test cases
4	Category module, Store module, Partner module, Banking account module	14 days	Code & API test cases & System test cases
5	Product module, Store partner module, Mapping product module, Cashier module	14 days	Code & API test cases & System test cases
6	Orders module, Wallet and Transactions Module	14 days	Code & API test cases & System test cases
7	Test Reports, Guide Documents, Final Project Package	14 days	User manual guides, Final code, documents,

Table 6: Project Deliverables

4. Responsibility Assignments

Fullname	Role	Responsibility
Nguyễn Thị Cẩm Hương Lâm Hữu Khánh Phương	Supervisor	• Instruct project team • Supervise project status • Review deliverables • Answer questions about the project
Thái Quốc Toàn	Leader	• Managing process • Track backlog • Clarifying requirements • Prepare documents • Quality management • Support other team members • Arrange meeting

Thái Quốc Toàn Trần Phú Sơn Lê Hồng Thành Võ Vỹ Khang Lê Xuân Bách	Team members	• Designing database • Clarifying requirements • Prepare documents • Coding • Create test plan • Testing • GUI design • Support other team members

Table 7: Responsibility Assignments

5. Project Communications

Communication Item	Who/ Target	Purpose	When, Frequency	Type, Tool, Method(s)
Working with supervisor	Supervisor 1 (Nguyen Thi Cam Huong) and team members	• Review documentation • Demonstrate features • Evaluate progress and result	1 time per week	Face to face, Google Meet, Zalo
Working with supervisor	Supervisor 2 (Lam Huu Khanh Phuong) and team members	• Review documentation • Demonstrate features • Evaluate progress and result	1 time per week	Face to face, Google Meet, Zalo
Working in team	Team members	• Raise opinions, problems • Ask for help from other members • Report working status to leader	Always	Face to Face, Messenger, Google Meet, Zalo

Table 8:Project Communications

6. Configuration Management

6.1. Document Management

We use OneDrive and Office 365 to manage documents since they supply real-time data synchronization. Team members can easily collaborate in real-time thanks to OneDrive. We use the document template to build documents on OneDrive, then assign writing duties to individuals following their predetermined roles. Each participant can at once view the outcomes of the others. We can also evaluate document updates in Office 365 so that we can do so whenever necessary.

6.2. Source Code Management

We decided to manage our source code on GitHub. The version control system makes it possible for team members to work together on source code efficiently, conveniently, and easily to resolve conflict. It is especially useful for looking back in time and at once recognizing changes a collaborator made.

6.3. Tools & Infrastructures

6.3.1. Tools

Type	Tools
IDEs/Editors	Visual Studio Code, Visual Studio
UML tools	lucid.app
Source Version Control	GitHub
Project Management Tool	OneDrive

Table 11: Tools

6.3.2. Infrastructures

Type	Infrastructure
Framework	ReactJS (Frontend), Entity Framework Core 6 (Backend), Flutter (Mobile)
Programming Languages	Typescript, C#, Dart
DBMS	SQL Server
Deployment Server	Microsoft Azure
Storage	Firebase Storage

Table 12: Infrastructures

III. Software Requirement Specification

1. **Product Overview**

The MBKC (Multi-Brand Kitchen center) System is a new software system that replaces the current operating process of the multi-brand kitchen center. The current operating process is performed entirely manually, which brings many disadvantages and inefficiencies. The MBKC System was born to support optimal operational processes in managing orders from e-commerce systems that the system can connect to.

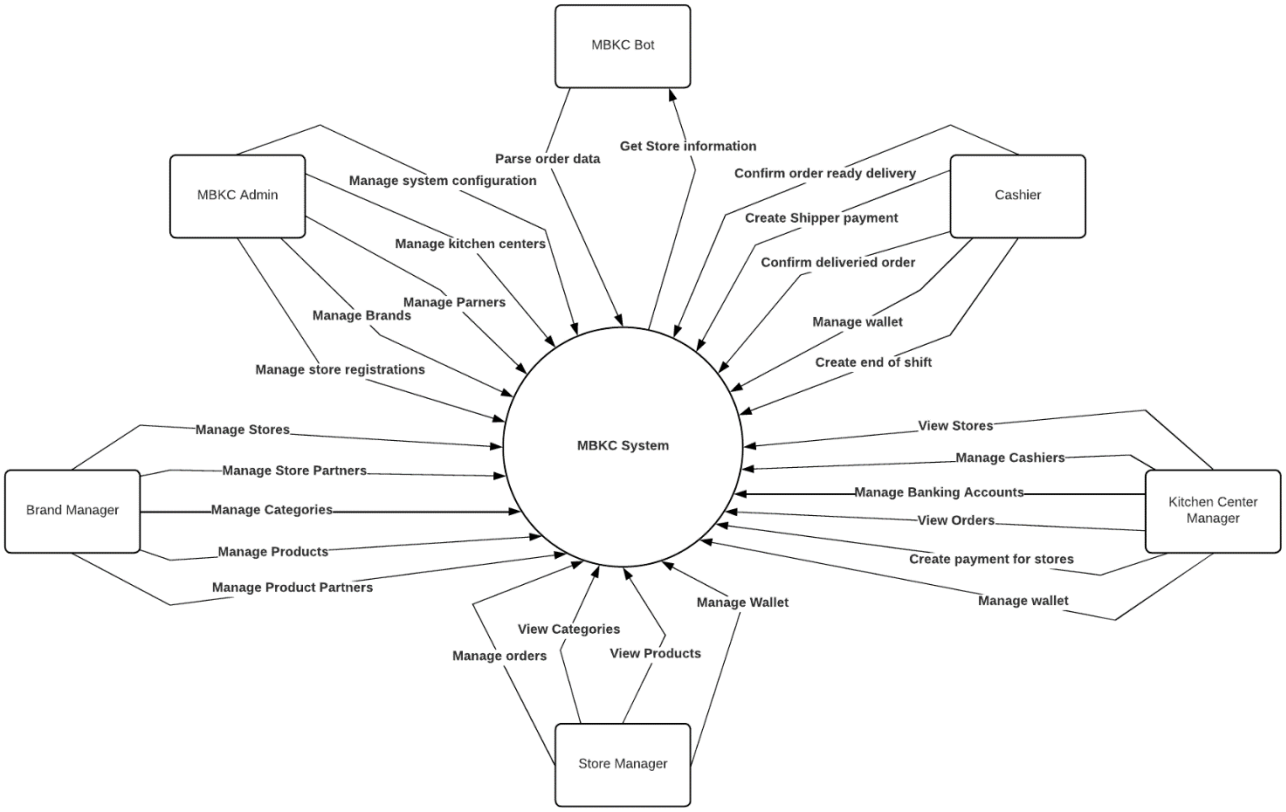


Figure 1: Context Diagram of MBKC System

2. User Requirements

2.1. Actors

No	Actor	Description
1	MBKC Admin	a person who manages the MBKC system.
2	Brand Manager	a person who registers to join in the space of the kitchen centers.
3	Store Manager	a person who is assigned to manage a store by the brand manager.
4	Kitchen Center Manager	a person who is assigned to manage a kitchen center by MBKC admin.
5	Cashier	a person who is a member of the specific kitchen center that plays the role of representative for all the Stores in the kitchen center to give the finished order to the deliveries and take the cash from them.
6	MBKC Bot	a worker service that automatically gets orders from the e-commerce systems for active stores in the MBKC system following the configuration time.

Table 13: Actors

2.2. Use Case Diagram

2.2.1. MBKC Admin Use Case Diagram

Figure 4: MBKC Admin Use Case Diagram

2.2.2. Brand Manager Use Case Diagram

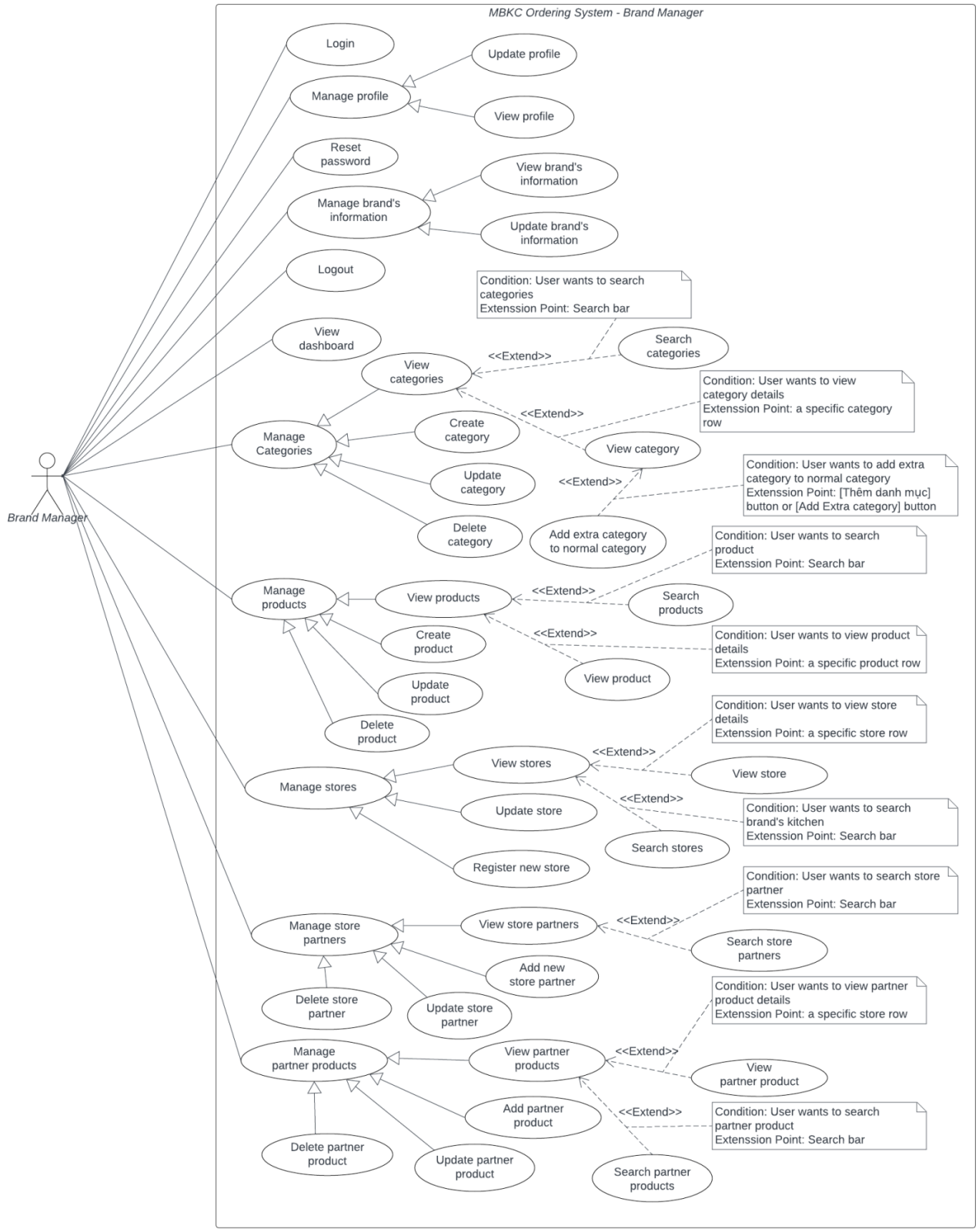


Figure 5: Brand Manager Use Case Diagram

2.2.3. Kitchen Center Manager Use Case Diagram

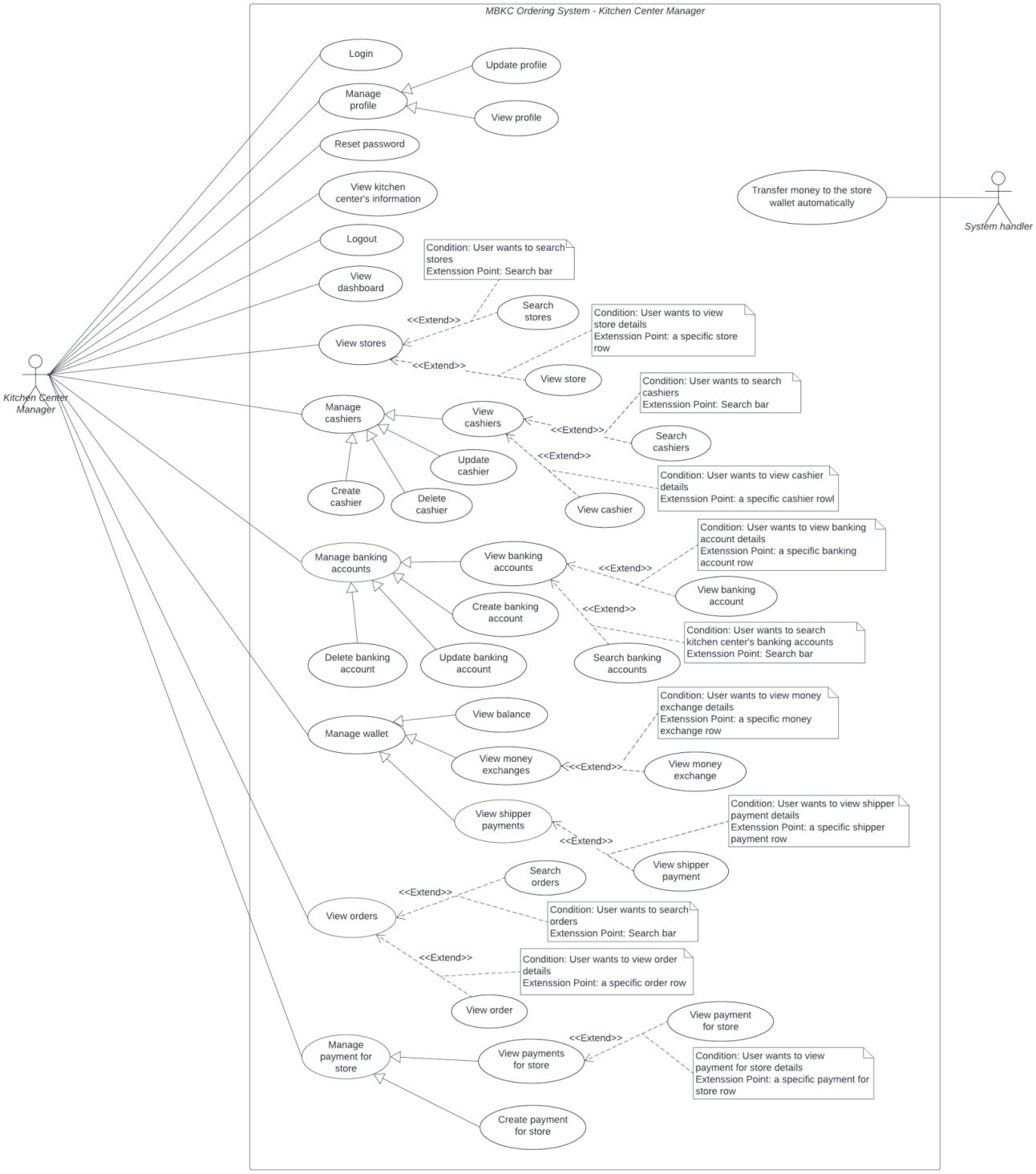


Figure 6: Kitchen Center Manager Use Case Diagram

2.2.4. Store Manager Use Case Diagram

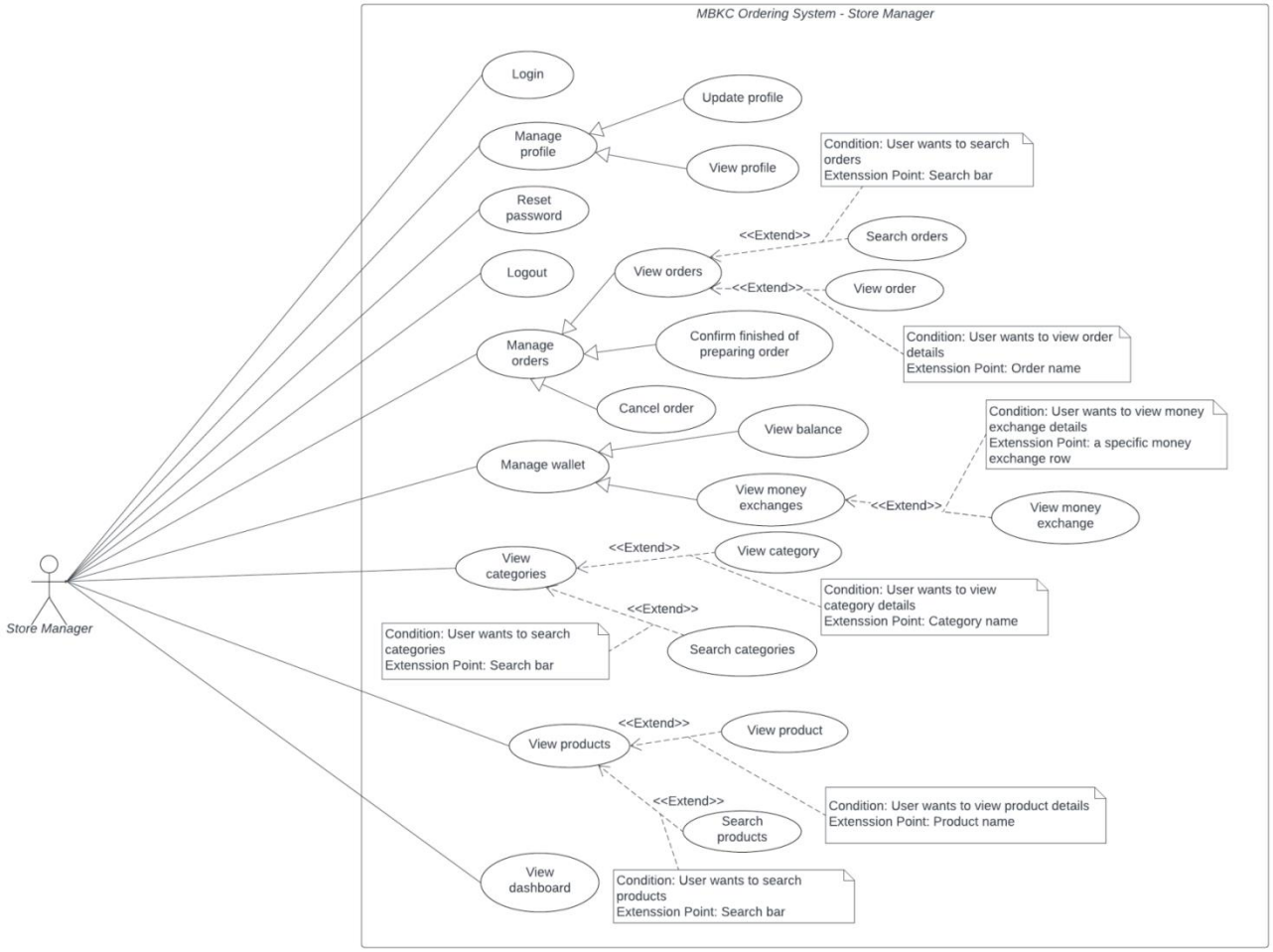


Figure 7: Store Manager Use Case Diagram

2.2.5. Cashier Use Case Diagram

Figure 8: Cashier Use Case Diagram

2.2.6. MBKC Bot Use Case Diagram

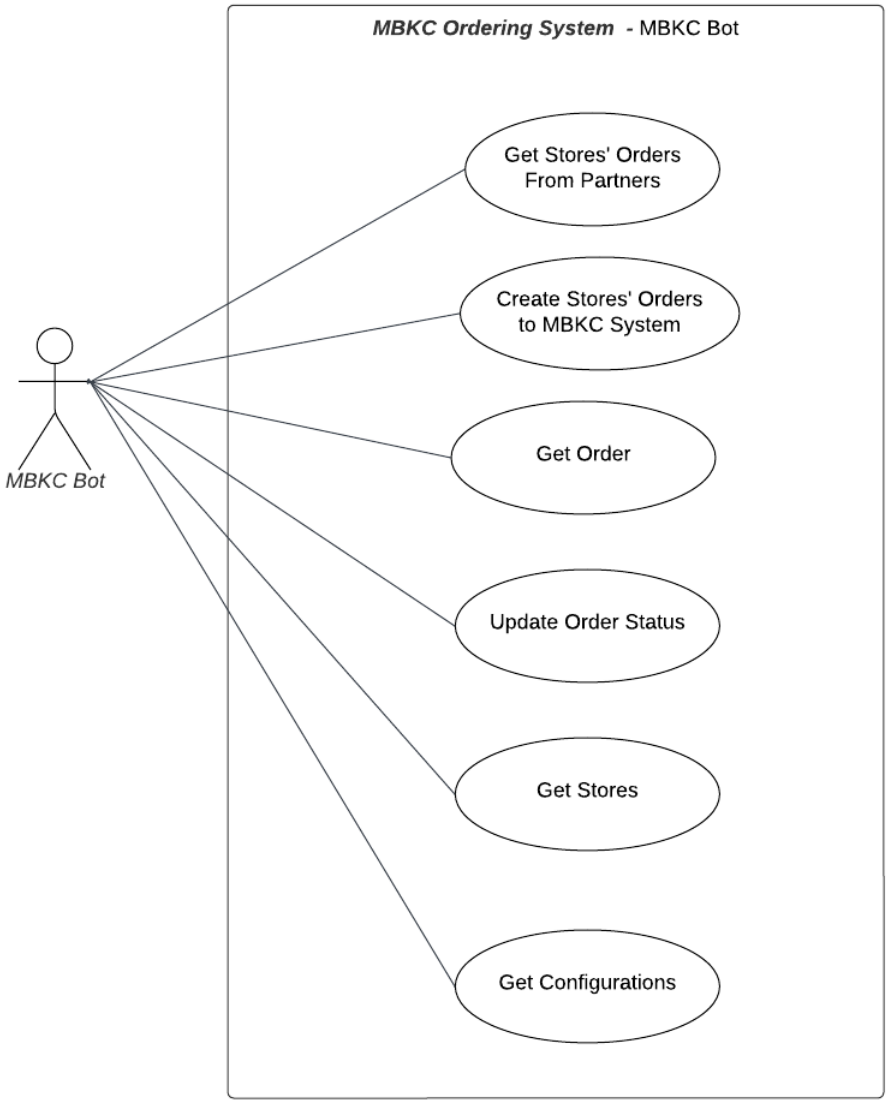


Figure 9: MBKC Bot Use Case Diagram

2.3. List of Use Case Descriptions

2.3.1. MBKC Admin Use Case Description

No	Actor	Use case	Use case description
1	MBKC Admin	Login	This use case allows “MBKC Admin” to login to the MBKC Web application by username and password.
2	MBKC Admin	Logout	This use case allows “MBKC Admin” to logout from MBKC Web application.
3	MBKC Admin	View dashboard	This use case allows “MBKC Admin” to view a few information such as: number of kitchen centers, number of brands, number of stores, number of partners, list of new kitchen centers, list of new brands, list of active stores.
4	MBKC Admin	View kitchen centers	This use case allows “MBKC Admin” to view kitchen center list.
5	MBKC Admin	View kitchen center	This use case allows “MBKC Admin” to view kitchen center details.
6	MBKC Admin	Search kitchen centers	This use case allows “MBKC Admin” to search kitchen centers by name.
7	MBKC Admin	Create kitchen center	This use case allows “MBKC Admin” to create a new kitchen center.
8	MBKC Admin	Update kitchen center	This use case allows “MBKC Admin” to update kitchen center information and status.
9	MBKC Admin	Delete kitchen center	This use case allows “MBKC Admin” to delete kitchen center.
10	MBKC Admin	View brands	This use case allows “MBKC Admin” to view brand list.
11	MBKC Admin	View brand	This use case allows “MBKC Admin” to view brand details.
12	MBKC Admin	Search brands	This use case allows “MBKC Admin” to search brands by name.
13	MBKC Admin	Create brand	This use case allows “MBKC Admin” to create a new brand.
14	MBKC Admin	Update brand	This use case allows “MBKC Admin” to update brand information and status.
15	MBKC Admin	Delete brand	This use case allows “MBKC Admin” to delete brand.

16	MBKC Admin	View stores	This use case allows “MBKC Admin” to view store list.
17	MBKC Admin	View store	This use case allows “MBKC Admin” to view store details.
18	MBKC Admin	Search stores	This use case allows “MBKC Admin” to search stores by name.
19	MBKC Admin	Confirm store registration	This use case allows “MBKC Admin” to confirm store registration of the brand.
20	MBKC Admin	Delete store	This use case allows “MBKC Admin” to delete store.
21	MBKC Admin	View partners	This use case allows “MBKC Admin” to view partner list.
22	MBKC Admin	View partner	This use case allows “MBKC Admin” to view partner details.
23	MBKC Admin	Search partners	This use case allows “MBKC Admin” to search partners by name.
24	MBKC Admin	Update partner	This use case allows “MBKC Admin” to update partner information and status.
25	MBKC Admin	View system configuration	This use case allows “MBKC Admin” to view system configuration information.
26	MBKC Admin	Update system configuration	This use case allows “MBKC Admin” to update system configuration information.

Table 14: MBKC Admin Use Case Description

2.3.2. Brand Manager Use Case Description

No	Actor	Use case	Use case description
1	Brand Manager	Login	This use case allows “Brand Manager” to login to the MBKC Web application by username and password.
2	Brand Manager	Reset password	This use case allows “Brand Manager” to reset their password.
3	Brand Manager	View profile	This use case allows “Brand Manager” to view their profile.
4	Brand Manager	Update profile	This use case allows “Brand Manager” to update their profile.
5	Brand Manager	Logout	This use case allows “Brand Manager” to logout from MBKC Web application.
6	Brand Manager	View dashboard	This use case allows “Brand Manager” to view a few

			information such as: number of stores, number of normal categories, number of extra categories, number of products, statistical chart of the store's income in the last 7 days, list of active stores, list of products width number of products sold.
7	Brand Manager	View brand's information	This use case allows “Brand Manager” to view information of brand.
8	Brand Manager	Update brand's information	This use case allows “Brand Manager” to update information of brand.
9	Brand Manager	View categories	This use case allows “Brand Manager” to view category list.
10	Brand Manager	View category	This use case allows “Brand Manager” to view category details.
11	Brand Manager	Add extra category to normal category	This use case allows “Brand Manager” to add extra category list to normal category.
12	Brand Manager	Search categories	This use case allows “Brand Manager” to search categories by name.
13	Brand Manager	Create category	This use case allows “Brand Manager” to create a new category.
14	Brand Manager	Update category	This use case allows “Brand Manager” to update category information and status.
15	Brand Manager	Delete category	This use case allows “Brand Manager” to delete category.
16	Brand Manager	View products	This use case allows “Brand Manager” to view product list.
17	Brand Manager	View product	This use case allows “Brand Manager” to view product details.
18	Brand Manager	Search products	This use case allows “Brand Manager” to search products by name.
19	Brand Manager	Create product	This use case allows “Brand Manager” to create a new product.
20	Brand Manager	Update product	This use case allows “Brand Manager” to update product information and status.
21	Brand Manager	Delete product	This use case allows “Brand Manager” to delete product.
22	Brand Manager	View stores	This use case allows “Brand Manager” to view store list.

23	Brand Manager	View store	This use case allows “Brand Manager” to view store details.
24	Brand Manager	Search stores	This use case allows “Brand Manager” to search stores by name.
25	Brand Manager	Register new store	This use case allows “Brand Manager” to register a new store.
26	Brand Manager	Update store	This use case allows “Brand Manager” to update store information and status.
27	Brand Manager	View store partners	This use case allows “Brand Manager” to view store partner list
28	Brand Manager	Search store partners	This use case allows “Brand Manager” to search store partners by store name
29	Brand Manager	Add new store partners	This use case allows “Brand Manager” to add one or more partners to store
30	Brand Manager	Update store partner	This use case allows “Brand Manager” to update store partner information and status
31	Brand Manager	Delete store partner	This use case allows “Brand Manager” to delete store partner
32	Brand Manager	View partner products	This use case allows “Brand Manager” to view partner product list
33	Brand Manager	View partner product	This use case allows “Brand Manager” to view partner product details
34	Brand Manager	Search partner products	This use case allows “Brand Manager” to search partner products by product name
35	Brand Manager	Add partner product	This use case allows “Brand Manager” to add new partner product
36	Brand Manager	Update partner product	This use case allows “Brand Manager” to update partner product information and status
37	Brand Manager	Delete partner product	This use case allows “Brand Manager” to delete partner product.

Table 15: Brand Manager Use Case Description

2.3.3. Kitchen Center Manager Use Case Description

No	Actor	Use case	Use case description
1	Kitchen Center Manager	Login	This use case allows “Kitchen Center Manager” to login to the MBKC Web application by username and password.
2	Kitchen Center Manager	Reset password	This use case allows “Kitchen Center Manager” to reset their password.
3	Kitchen Center Manager	View profile	This use case allows “Kitchen Center Manager” to view their profile.
4	Kitchen Center Manager	Update profile	This use case allows “Kitchen Center Manager” to update their profile.
5	Kitchen Center Manager	Logout	This use case allows “Kitchen Center Manager” to logout from MBKC Web application.
6	Kitchen Center Manager	View dashboard	This use case allows “Kitchen Center Manager” to view a few information such as: total balances in wallet daily, number of stores, number of cashiers, statistical chart of the kitchen center's income in the last 7 days, list of active stores, list of cashiers.
7	Kitchen Center Manager	View kitchen center's information	This use case allows “Kitchen Center Manager” to view information of kitchen center.
8	Kitchen Center Manager	View stores	This use case allows “Kitchen Center Manager” to view store list.
9	Kitchen Center Manager	View store	This use case allows “Kitchen Center Manager” to view store details.
10	Kitchen Center Manager	Search stores	This use case allows “Kitchen Center Manager” to search stores by name.
11	Kitchen Center Manager	View cashiers	This use case allows “Kitchen Center Manager” to view cashier list.
12	Kitchen Center Manager	View cashier	This use case allows “Kitchen Center Manager” to view cashier details.
13	Kitchen Center Manager	Search cashiers	This use case allows “Kitchen Center Manager” to search cashiers by name.
14	Kitchen Center Manager	Create cashier	This use case allows “Kitchen Center Manager” to create a new cashier.

15	Kitchen Center Manager	Update cashier	This use case allows “Kitchen Center Manager” to update cashier information and status.
16	Kitchen Center Manager	Delete cashier	This use case allows “Kitchen Center Manager” to delete cashier.
17	Kitchen Center Manager	View banking accounts	This use case allows “Kitchen Center Manager” to view banking account list.
18	Kitchen Center Manager	View banking account	This use case allows “Kitchen Center Manager” to view banking account details.
19	Kitchen Center Manager	Search banking accounts	This use case allows “Kitchen Center Manager” to search banking accounts by name.
20	Kitchen Center Manager	Create banking account	This use case allows “Kitchen Center Manager” to create a new banking account.
21	Kitchen Center Manager	Update banking account	This use case allows “Kitchen Center Manager” to update banking account information and status.
22	Kitchen Center Manager	Delete banking account	This use case allows “Kitchen Center Manager” to delete banking account.
23	Kitchen Center Manager	View orders	This use case allows “Kitchen Center Manager” to view order list.
24	Kitchen Center Manager	View order	This use case allows “Kitchen Center Manager” to view order details.
25	Kitchen Center Manager	Search orders	This use case allows “Kitchen Center Manager” to search order by id order partner.
26	Kitchen Center Manager	View balance	This use case allows “Kitchen Center Manager” to view a few information such as: balances in wallet, total daily receive, total daily send, list of money exchanges.
27	Kitchen Center Manager	View money exchanges	This use case allows “Kitchen Center Manager” to view money exchange list.
28	Kitchen Center Manager	View money exchange	This use case allows “Kitchen Center Manager” to view money exchange details.
29	Kitchen Center Manager	View shipper payments	This use case allows “Kitchen Center Manager” to view shipper payment list.
30	Kitchen Center Manager	View shipper payment	This use case allows “Kitchen Center Manager” to view shipper payment details.

31	Kitchen Center Manager	View payments for store	This use case allows “Kitchen Center Manager” to view payments for store list.
32	Kitchen Center Manager	View payment for store	This use case allows “Kitchen Center Manager” to view payments for store details.
33	Kitchen Center Manager	Create payment for store	This use case allows “Kitchen Center Manager” to create payments to withdraw money to the store.
34	System handler	Transfer money to the store wallet automatically	This use case allows “System handler” to transfer money to the store wallet automatically.

Table 16: Kitchen Center Manager Use Case Description

2.3.4. Store Manager Use Case Description

No	Actor	Use case	Use case description
1	Store Manager	Login	This use case allows “Store Manager” to login to the MBKC Mobile application by username and password.
2	Store Manager	Reset password	This use case allows “Store Manager” to reset their password.
3	Store Manager	View profile	This use case allows “Store Manager” to view their profile.
4	Store Manager	Update profile	This use case allows “Store Manager” to update their profile.
5	Store Manager	Logout	This use case allows “Store Manager” to logout from MBKC Mobile application.
6	Store Manager	View dashboard	This use case allows “Store Manager” to view a few information such as: total upcoming orders, total preparing orders, total ready orders, total completed orders, total revenues daily.
7	Store Manager	View categories	This use case allows “Store Manager” to view category list.
8	Store Manager	View category	This use case allows “Store Manager” to view category details.
9	Store Manager	Search categories	This use case allows “Store Manager” to search categories by name.
10	Store Manager	View products	This use case allows “Store Manager” to view product list.

11	Store Manager	View product	This use case allows “Store Manager” to view product details.
12	Store Manager	Search products	This use case allows “Store Manager” to search products by name.
13	Store Manager	View orders	This use case allows “Store Manager” to view order list.
14	Store Manager	View order	This use case allows “Store Manager” to view order details.
15	Store Manager	Search orders	This use case allows “Store Manager” to search order by id order partner.
16	Store Manager	Confirm finished of preparing order	This use case allows “Store Manager” to confirm finished of preparing order.
17	Store Manager	Cancel order	This use case allows “Store Manager” to cancel order.
18	Store Manager	View balance	This use case allows “Store Manager” to view a few information such as: balances in wallet, list of money exchanges.
19	Store Manager	View money exchanges	This use case allows “Store Manager” to view money exchange list.
20	Store Manager	View money exchange	This use case allows “Store Manager” to view money exchange details.

Table 17: Store Manager Use Case Description

2.3.5. Cashier Use Case Description

No	Actor	Use case	Use case description
1	Cashier	Login	This use case allows “Cashier” to login to the MBKC Web application by username and password.
2	Cashier	Reset password	This use case allows “Cashier” to reset their password.
3	Cashier	View profile	This use case allows “Cashier” to view their profile.
4	Cashier	Update profile	This use case allows “Cashier” to update their profile.
5	Cashier	Logout	This use case allows “Cashier” to logout from MBKC Web application.

6	Cashier	View dashboard	This use case allows “Cashier” to view a few information such as: total revenue of today, total orders of today, list of orders delivered successfully, list new shipper payments, list new transactions.
7	Cashier	View orders	This use case allows “Cashier” to view order list.
8	Cashier	View order	This use case allows “Cashier” to view order details.
9	Cashier	Search orders	This use case allows “Cashier” to search order by id order partner.
10	Cashier	Confirm status order to ready delivery	This use case allows “Cashier” to confirm status of order to ready delivery.
11	Cashier	Create shipper payment	This use case allows “Cashier” to create shipper payment to deliver the order.
12	Cashier	View balance	This use case allows “Cashier” to view a few information such as: balances in wallet, total daily transactions, total daily shipper payments, list of money exchanges, list of shipper payments.
13	Cashier	View money exchanges	This use case allows “Cashier” to view money exchange list.
14	Cashier	View money exchange	This use case allows “Cashier” to view money exchange details.
15	Cashier	View shipper payments	This use case allows “Cashier” to view shipper payment list.
16	Cashier	View shipper payment	This use case allows “Cashier” to view shipper payment details.
17	Cashier	Create end of shift	This use case allows “Cashier” to create end of shift to send money collected from orders back to the kitchen center.
18	System handler	Transfer money to the kitchen center’s wallet automatically	This use case allows “System handler” to transfer money to the kitchen center’s wallet automatically.

Table 18: Cashier Use Case Description

2.3.6. MBKC Bot Use Case Description

No	Actor	Use case	Use case description
1	MBKC Bot	Get stores' orders from partners	This use case allows “MBKC Bot” to access to partner’s API to collect orders.
2	MBKC Bot	Create stores' orders to MBKC system	This use case allows “MBKC Bot” to parse order data and create a new order to the MBKC System.
3	MBKC Bot	Get order	This use case allows “MBKC Bot” to get order information from MBKC System.
4	MBKC Bot	Update order status	This use case allows “MBKC Bot” to update partner order status.
5	MBKC Bot	Get stores	This use case allows “MBKC Bot” to get all of stores in the MBKC System.
6	MBKC Bot	Get configurations	This use case allows “MBKC Bot” to get configuration information to decide the start time and end time of receiving orders.

Table 19: MBKC Bot Use Case Description

3. Functional Requirements

3.1. System Functional Overview

3.1.1. Screen Flow

3.1.1.1. MBKC Admin Screen Flow

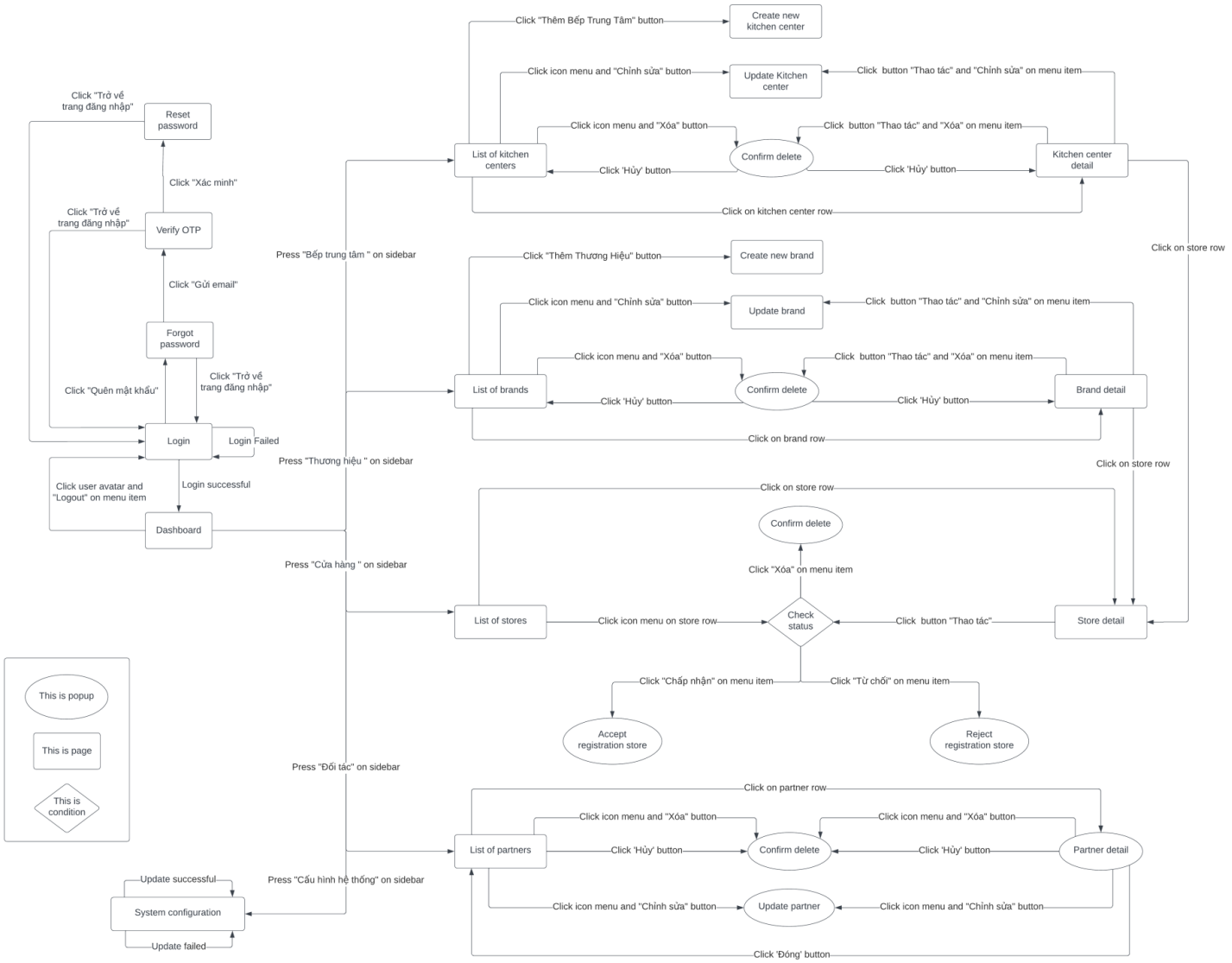


Figure 10: MBKC Admin on Web Application Screen Flow

3.1.1.2. Kitchen Center Manager Screen Flow

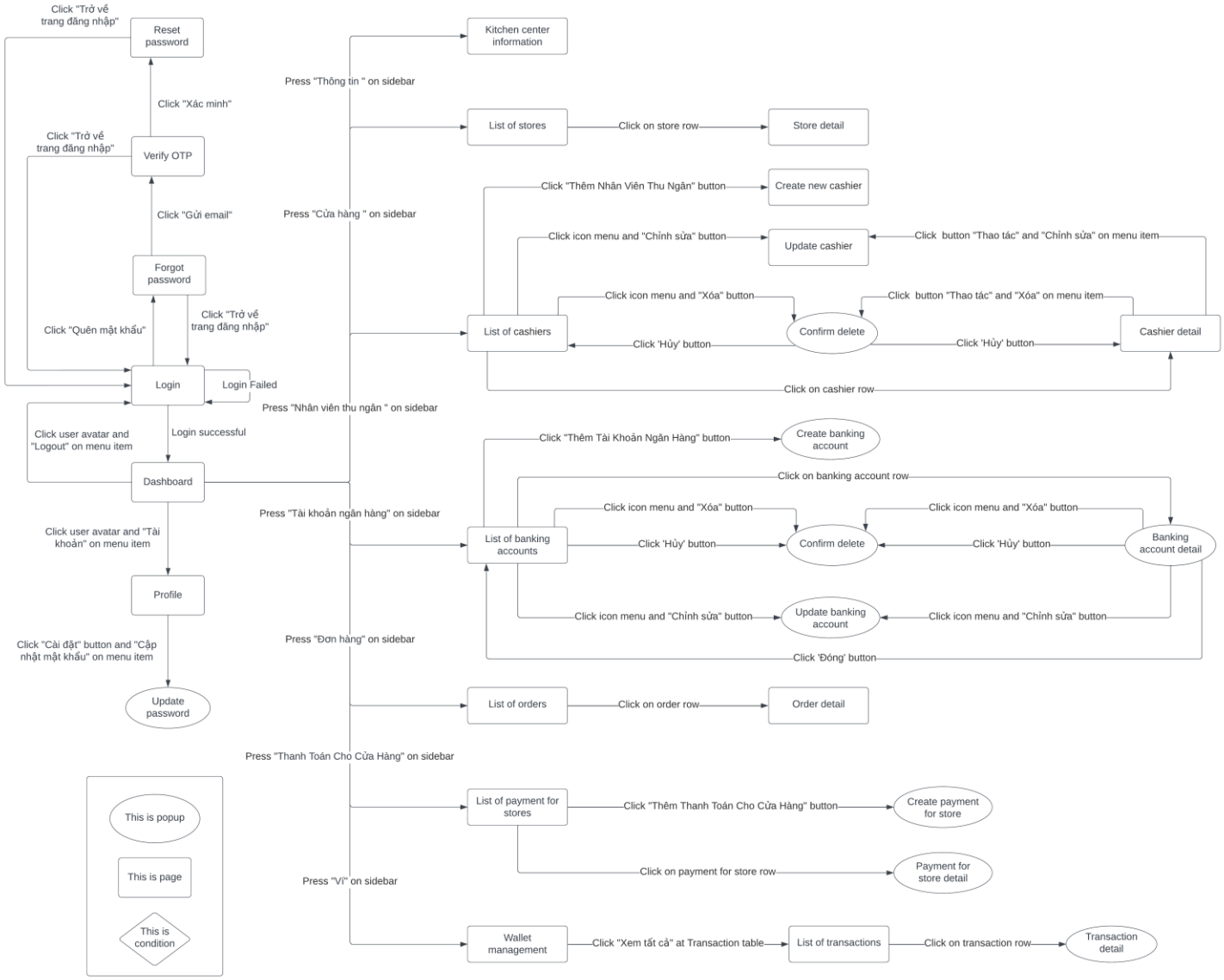


Figure 11: Kitchen Center Manager on Web Application Screen Flow

3.1.1.3. Brand Manager Screen Flow

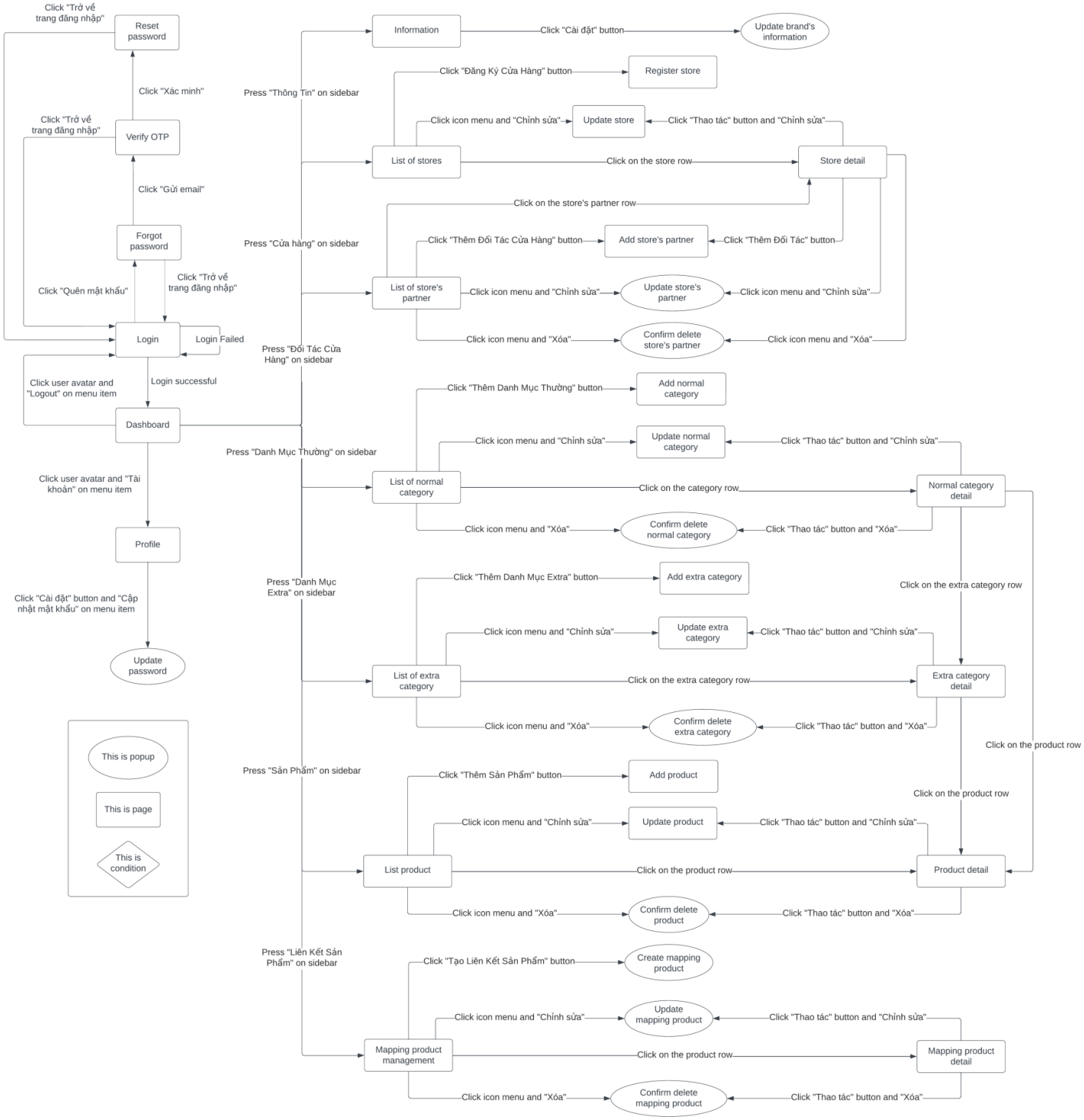


Figure 12: Brand Manager on Web Application Screen Flow

3.1.1.4. Cashier Screen Flow

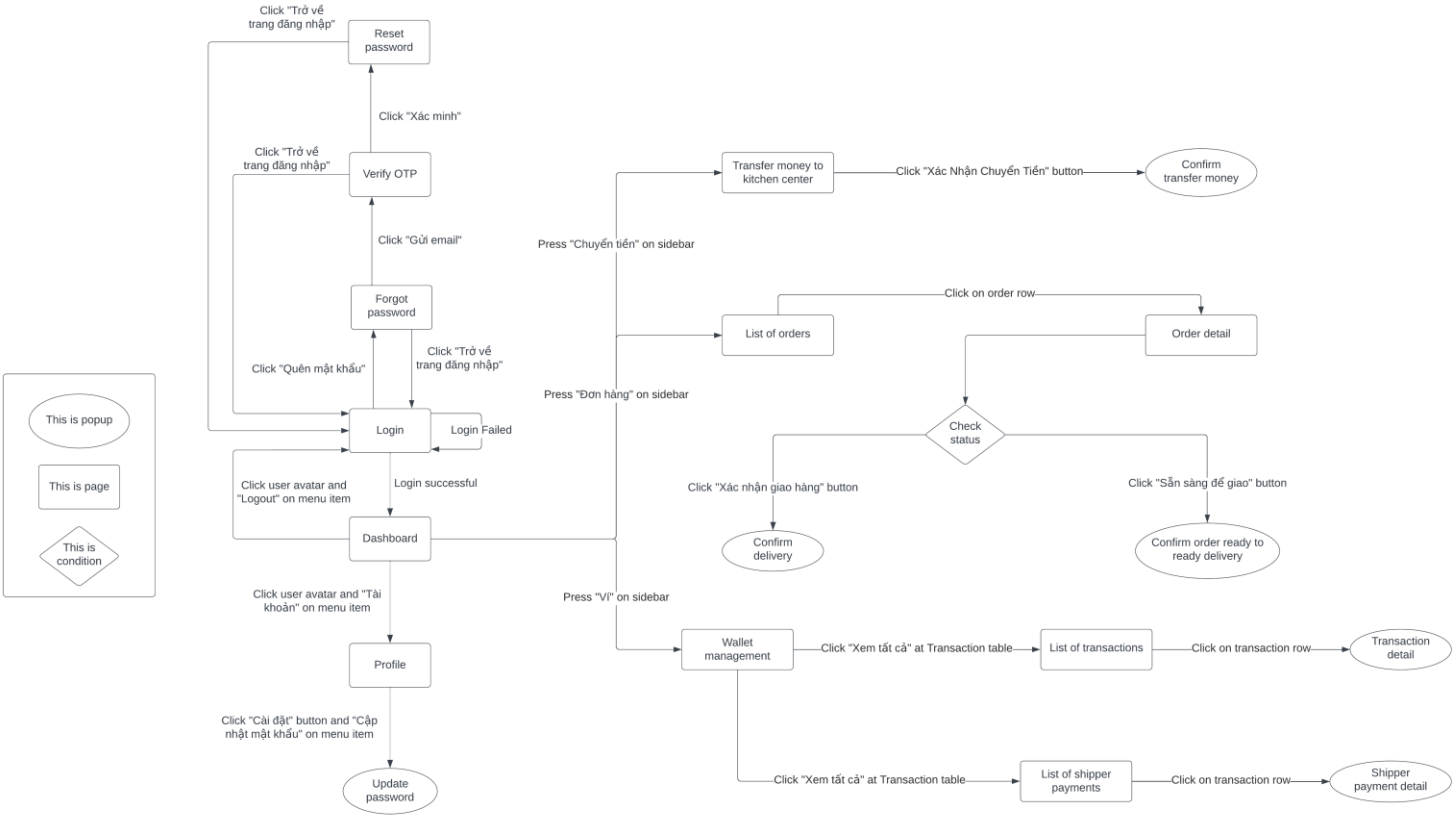


Figure 13: Cashier on Web Application Screen Flow

3.1.1.5. Store Manager Screen Flow

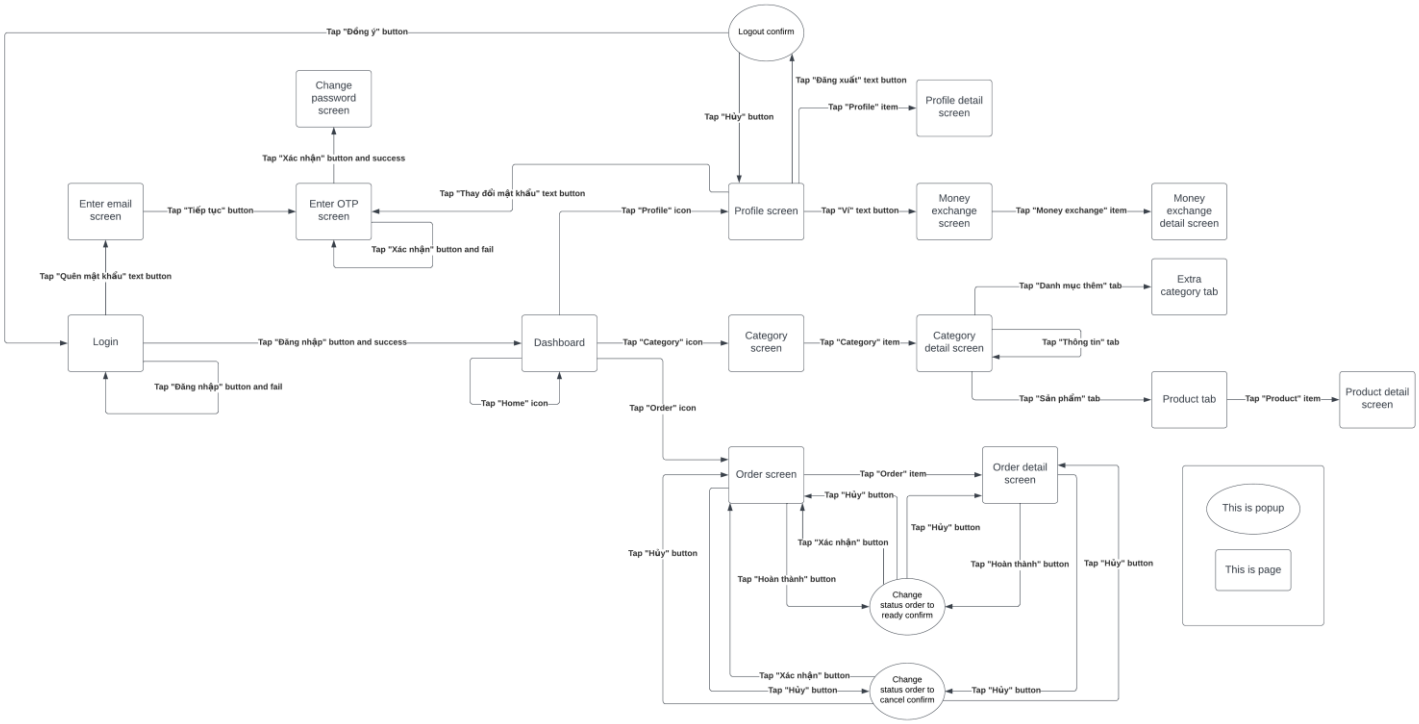


Figure 14: Store Manager on Mobile Application Screen Flow

3.1.2. Screen Descriptions

3.1.2.1. MBKC Admin Screen Description

No	Feature	Screen	Description
1	Login	Login	MBKC Admin can login into MBKC web application by username and password
2	Reset password	Forgot password	MBKC Admin can reset password.
3	View dashboard	Home	MBKC Admin can view a few information such as: number of kitchen centers, number of brands, number of stores, number of partners, list of new kitchen centers, list of new brands, list of active stores.
4	View list of kitchen centers	List of kitchen centers	MBKC Admin can view list of kitchen centers in the system, each row of table list has no, logo, name, address, and status.
5	View kitchen center detail	Kitchen center detail	MBKC Admin can view kitchen center details. Include kitchen center information details and kitchen center's store list.
6	Create new kitchen center	Create new kitchen center	MBKC Admin can create a new kitchen center for the system with input kitchen center information: name, logo, manager email, address.
7	Update kitchen center	Update kitchen center	MBKC Admin can update kitchen center with input kitchen center information: name, logo, manager email, address.
8	Change status of kitchen center	List of kitchen centers	MBKC Admin can change status of kitchen centers from active to inactive, it means kitchen center's from working to temporarily not working and vice versa.
9	Delete kitchen center	List of kitchen centers, Kitchen center detail	MBKC Admin can delete kitchen center.
10	View list of brands	List of brands	MBKC Admin can view list of brands in the system, each row of table list has no, logo, name, address, and status.
11	View brand detail	Brand detail	MBKC Admin can view brand details. Include brand information details and brand's store list.
12	Create new brand	Create new brand	MBKC Admin can create a new brand for the system with input brand information: name, logo, manager email, address.
13	Update brand	Update brand	MBKC Admin can update an existed brand with input brand information: name, logo, manager email, address.
14	Change status of brand	List of brands	MBKC Admin can change the status of brand from active to inactive, it means brand's from working to temporarily not working and vice versa.
15	Delete brand	List of brands, Brand detail	MBKC Admin can delete an existed brand.
16	View list of stores	List of stores	MBKC Admin can view list of stores in the system, each row of table list has no, logo, name, kitchen center, brand, and status.
17	View store detail	Store detail	MBKC Admin can view store details. Include all details information of store's

18	Confirm registration store	List of stores, Store detail	MBKC Admin can accept or reject the registration from the store.
19	Delete store	List of stores, Store detail	MBKC Admin can delete store.
20	View list of partners	List of partners	MBKC Admin can view list of partners in the system, each row of table list has no, logo, name, status.
21	View partner detail	List of partners	MBKC Admin can view partner details. Include all details information of partner's
22	Update partner	List of partners	MBKC Admin can update partner with input partner information: logo, web URL.
23	Update system configuration	System Configuration	MBKC Admin can update system configuration with three options such as: Order receiving time, Money transaction time to transfer money to kitchen center and Money transaction time to transfer money to store.

Table 20: MBKC Admin Screen Description

3.1.2.2. Kitchen Center Manager Screen Description

No	Feature	Screen	Description
1	Login	Login	Kitchen Center Manager can login into MBKC web application by username and password.
2	Reset password	Forgot password	Kitchen Center Manager can reset password.
3	View dashboard	Home	Kitchen Center Manager can view a few information such as: total balances in wallet daily, number of stores, number of cashiers, statistical chart of the kitchen center's income in the last 7 days, list of active stores, list of cashiers.
4	View list of stores	List of stores	Kitchen Center Manager can view a list stores of the kitchen center, each row of table list has no, logo, name, brand, and status.
5	View store detail	Store detail	Kitchen Center Manager can view store details. Include all details information of store's
6	View list of cashiers	List of cashiers	Kitchen Center Manager can view the list cashiers of the kitchen center, each row of table list has no, avatar, full name, email, gender, and status.
7	View cashier detail	Cashier detail	Kitchen Center Manager can view cashier details. Include all details information of cashier.
8	Create new cashier	Create new cashier	Kitchen Center Manager can create new cashier for kitchen center with input cashier information: image, email, full name, gender, date of birth, citizen number.
9	Update cashier	Update cashier	Kitchen Center Manager can update an existed cashier with input cashier information: image, full name, gender, date of birth, citizen number.

10	Change status of cashier	List of cashiers	Kitchen Center Manager can change status of cashier from active to inactive, it means cashier's from working to temporarily not working and vice versa.
11	Delete cashier	List of cashiers	Kitchen Center Manager can delete cashier.
12	View information of kitchen center	Information kitchen center	Kitchen Center Manager can view information about the kitchen center.
13	View account information	Account information	Kitchen Center Manager can view account information.
14	Update password	Account information	Kitchen Center Manager can update password with input update password information: new password and confirm password.
15	View list banking accounts	List of banking accounts	Kitchen Center Manager can view list of banking accounts of kitchen center, each row of table list has no, logo, name, and status.
16	View banking account detail	List of banking accounts	Kitchen Center Manager can view banking account details. Include all the details of the banking account.
17	Update banking account	List of banking accounts	Kitchen Center Manager can update an existed banking account with input banking account information: logo, web URL.
18	Delete banking account	List of banking accounts	Kitchen Center Manager can delete banking account.
19	Change status of banking account	List of banking accounts	Kitchen Center Manager can change the status of banking account from active to inactive, it means banking account's from working to temporarily not working and vice versa.
20	View list of orders	List of orders	Kitchen Center Manager can view a list of orders of stores in kitchen center, each row of table list has no, order id, order partner id, partner name, store name, collect, system status and partner order status.
21	View order detail	Order detail	Kitchen Center Manager can view order details. Include all details information of order.
22	View list of payment for stores	List of payment for stores	Kitchen Center Manager can view the list of payment for stores of kitchen center, each row of table list has no, sender, receiver, transaction time, amount, exchange type and status.
23	View payment for store detail	List of payment for stores	Kitchen Center Manager can view payment for store details. Include all the detailed information about payment for the store.
24	Create payment for store	List of payment for stores	Kitchen Center Manager can create payment for the store to record withdrawals from kitchen center to store.
25	View wallet	Kitchen center wallet	Kitchen Center Manager can view the wallet of the kitchen center with wallet balance, total daily receive and total daily

			send.
26	View list of transactions	List of transactions	Kitchen Center Manager can view list of transactions of kitchen center, each row of table list has no, sender, receiver, transaction time, amount, exchange type and status.
27	View transaction detail	List of transactions	Kitchen Center Manager can view transaction details. Include all details information of transaction.

Table 21: Kitchen Center Manager Screen Description

3.1.2.3. Brand Manager Screen Descriptions

No	Feature	Screen	Description
1	Login	Login	Brand Manager can login into MBKC web application by username and password.
2	Reset password	Forgot password	Brand Manager can reset password.
3	View dashboard	Home	Brand Manager can view a few information such as: number of stores, number of normal categories, number of extra categories, number of products, statistical chart of the store's income in the last 7 days, list of active stores, list of products width number of products sold.
4	View information of brand	Information brand	Brand Manager can view information about brands.
5	Update information of brand	Information brand	Brand Manager can update information of brand with input brand information: logo, name, and address.
6	View account information	Account information	Brand Manager can update account information.
7	Update password	Account information	Brand Manager can update password with input update password information: new password and confirm password.
8	View list of stores	List of stores	Brand Manager can view list stores of brand, each row of table list has no, logo, name, manager email, kitchen center and status.
9	View store detail	Store detail	Brand Manager can view store details. Include all the detailed information about the store and list of store partners' the store.
10	Change status of store	List of stores	Brand Manager can change the status of store from active to inactive, it means 's from working to temporarily not working and vice versa.
11	Update store	Update store	Brand Manager can update information of store with input store information: logo, name and manager email.

12	Register store	Register store	Brand Manager can register a store for brand with input store information: logo, name, manager email and kitchen center.
13	View list of store partner	List of store partner	Brand Manager can view list of store partner of store's, each row of table list has no, name, kitchen center, partners.
14	View store partner detail	List of store partner	Brand Manager can view store partner details. Include all details information of store partner.
15	Create new store partner	Create new store partner	Brand Manager can create new store partner for store with input store partner information: store, partner, username, password, and commission.
16	Update information partner	List of store partner	Brand Manager can update information of partner with input partner information: username, password, and commission.
17	Change status of partner	List of store partner	Brand Manager can change the status of partner from active to inactive, it means 's from working to temporarily not working and vice versa.
18	View list of normal categories	List of normal categories	Brand Manager can view list of normal categories of brand's, each row of table list has no, image, category name, category code, display order and status.
19	Create new normal category	Create new category	Brand Manager can create new normal category for brand with input normal category information: logo, category name, category code, display order and description.
20	View normal category detail	Normal category detail	Brand Manager can view normal category details. Include all details information of normal category, list of products using this normal category and list of extra categories go with this normal category.
21	Update normal category	Update category	Brand Manager can update information of normal category with input normal category information: logo, category name, display order and description.
22	Change status of normal category	List of normal categories	Brand Manager can change the status of normal category from active to inactive, it means 's from working to temporarily not working and vice versa.
23	Delete normal category	List of normal categories, Normal category detail	Brand Manager can delete normal category.
24	View list of extra categories	List of extra categories	Brand Manager can view a list extra categories of the brand, each row of the table list has a note, image, category name, category code, display order and status.
25	View extra	Extra category	Brand Manager can view extra category details. Include all details information of extra category, list of products using this

	category detail	detail	extra category.
26	Create new extra category	Create new extra category	Brand Manager can create new extra category for the brand with input of extra category information: logo, category name, category code, display order and description.
27	Change status of extra category	List of extra categories	Brand Manager can change the status of an extra category from active to inactive, it means from working to temporarily not working and vice versa.
28	Update extra category	Update extra category	Brand Manager can update information of extra category with input extra category information: logo, category name, display order and description.
29	Delete extra category	List of extra categories, Extra category detail	Brand Manager can delete extra an existed category.
30	View list of products	List of products	Brand Manager can view list of products of brand's include list of all products, parent product, child product, single product, and extra product, each row of table list has no, image, product name, product code, display order, selling price, category, type, and status.
31	View product detail	Product detail	Brand Manager can view product details. Include all details information of product and list of partner product mapping information.
32	Create new product	Create new product	Brand Manager can create a new product for the brand with input product information: image, product type, product name, product code, display order and description.
33	Change status of product	List of products	Brand Manager can change the status of a product from active to inactive, it means from working to temporarily not working and vice versa.
34	Update product	Update product	Brand Manager can update information of product with input product information: parent product, display order, selling price, discount price, original price, description.
35	Delete product	List of products	Brand Manager can delete product.
36	View list of partner product	List of partner product	Brand Manager can view the list partner products of brand, each row of table list has no, product name, product code, partner, store and status.
37	View partner product detail	Partner product detail	Brand Manager can view partner product details. Include all details information of partner product details.
38	Create partner product	List of partner product	Brand Manager can create a new partner product for brand in case brand does not want system automatically to map with input partner product information: product, store, partner,

			product code, status partner product and product price.
39	Update partner product	List of partner product	Brand Manager can update information of partner product with input partner product information: product code, status partner product and product price.
40	Delete partner product	List of partner product, partner product detail	Brand Manager can delete partner product.
41	Change status of partner product	List of partner product	Brand Manager can change the status of partner product with three status options: available, out of stock today and out of stock indefinitely.

Table 22: Brand Manager Screen Description

3.1.2.4. Cashier Screen Descriptions

No	Feature	Screen	Description
1	Login	Login	Cashier can login into MBKC web application by username and password.
2	Reset password	Forgot password	Cashier can reset password.
3	View dashboard	Home	Cashier can view a few information such as: total revenue of today, total orders of today, list of orders delivered successfully, list new shipper payments, list new transactions.
4	Transfer money	Transfer money to kitchen center	Cashier can transfer money to kitchen center when cashier end of shift.
5	View account information	Account information	Cashier can view account information.
6	Update password	Account information	Cashier can update password with input update password information: new password and confirm password.
7	View list orders of today	Transfer money to kitchen center	Cashier can check again list orders of today before sending money to kitchen center.
8	View list of orders	List of orders	Cashier can view list of orders of stores in kitchen center, each row of table list has no, order id, order partner id, partner name, store name, collect, system status and partner order status.
9	View order detail	Order detail	Cashier can view order details. Include all details information of order.
10	Change status of order	Order detail	Cashier can change status of order from In store to ready delivery to confirm that order is on shelf and ready to delivery.
11	Confirm Delivery	Order detail	Cashier can confirm delivery of the paid order to confirm that the order has been successfully delivered to the shipper.

12	Create shipper payment	Order detail	Cashier can create shipper payment for unpaid orders to confirm that the order has been successfully delivered to the shipper and record the shipper's payment method.
13	View wallet	Cashier wallet	Cashier can view wallet of cashier with wallet balance, total daily transactions, and total daily shipper payments.
14	View list of shipper payments	List of shipper payments	Cashier can view list of shipper payments of cashier, each row of table list has no, order partner id, cashier created, created date, collect, total, payment method, banking account and status.
15	View shipper payment detail	List of shipper payments	Cashier can view shipper payment details. Include all detailedailed information about shipper payment.
16	View list of transactions	List of transactions	Cashier can view list of transactions of cashier, each row of table list has no, sender, receiver, created date, collect, total, payment method, banking account and status.
17	View transaction detail	List of transactions	Cashier can view transaction details. Include all detailed information of transactions.

Table 23: Cashier Screen Description

3.1.2.5. Store Manager Descriptions

No	Feature	Screen	Description
1	Login	Login	Store Manager can login to MBKC mobile application by username and password.
2	Reset password	Forgot password	Store Manager can reset password.
3	View profile	Profile	Store Manager can view profile.
4	Change password	Change password	Store Manager can change password.
5	Logout	Account	Store Manager can logout MBKC mobile application.
6	View wallet	Wallet	Store Manager can view balance in store wallet.
7	View list of orders	Orders	Store Manager can view list of orders.
8	View order detail	Information of order	Store Manager can view order details. Include all details information of order.
9	View dashboard	Home	Store Manager can view a few information such as: total upcoming orders, total preparing orders, total ready orders, total completed orders, total revenues daily.
10	Confirm Order	Order, Order detail	Store Manager can confirm order to change partner order status from preparing to ready.

11	View list product categories	Product Category	Store Manager can view list of product categories with two type: normal category and extra category.
12	View product category detail	Information of category	Store Manager can view product category details. Include all details information of product category and list of products are using this category.

Table 24: Store Manager Screen Description

3.1.3. Screen Authorization

Screen	MBKC Admin	Kitchen Center Manager	Brand Manager	Cashier	Store Manager
Login	x	x	x	x	x
Forgot Password	x	x	x	x	x
Home	x	x	x	x	x
List of kitchen centers	x				
Kitchen center detail	x				
Create new kitchen center	x				
Update kitchen center	x				
List of brands	x				
Brand detail	x				
Create new brand	x				
Update brand	x				
List of stores	x	x	x		
Store detail	x	x	x		
Update store			x		
List of partners	x				
System configuration	x				
List of cashiers		x			
Cashier detail		x			
Create new cashier		x			
Update cashier		x			

Information kitchen center		x			
Account information		x	x	x	
List of banking accounts		x			
List of orders		x		x	
Order detail		x		x	x
List of payment for stores		x			
Kitchen center wallet		x			
List of transactions		x		x	
Information brand			x		
Register store			x		
List of store partner			x		
List of normal categories			x		
Create new category			x		
Normal category detail			x		
Update category			x		
List of extra categories			x		
Extra category detail			x		
Create new extra category			x		
List of products			x		
Product detail			x		
Create new product			x		
Update product			x		
List of partner product			x		
Partner product detail			x		
Transfer money to				x	

kitchen center					
Cashier wallet				x	
List of shipper payments				x	
Profile		x	x	x	x
Change password					x
Account					x
Orders					x
Information of order					x
Product Category					x
Information of category					x
Wallet					x

Table 25: Screen Authorization

3.1.4. Non-Screen Functions

No	Feature	System Function	Description
1	MBKC API	Data Processing API	The API provides the processing of data for web application and mobile application.
2	MBKC Private API	Data Processing API	The API provides data to support for getting and parsing order data.
3	MBKC Bot	Collect and parse order data	MBKC Bot is a worker service. It will collect order data from the partners and parse data for the MBKC System.

Table 26: Non-Screen Functions

3.1.5. Entity Relationship Diagram

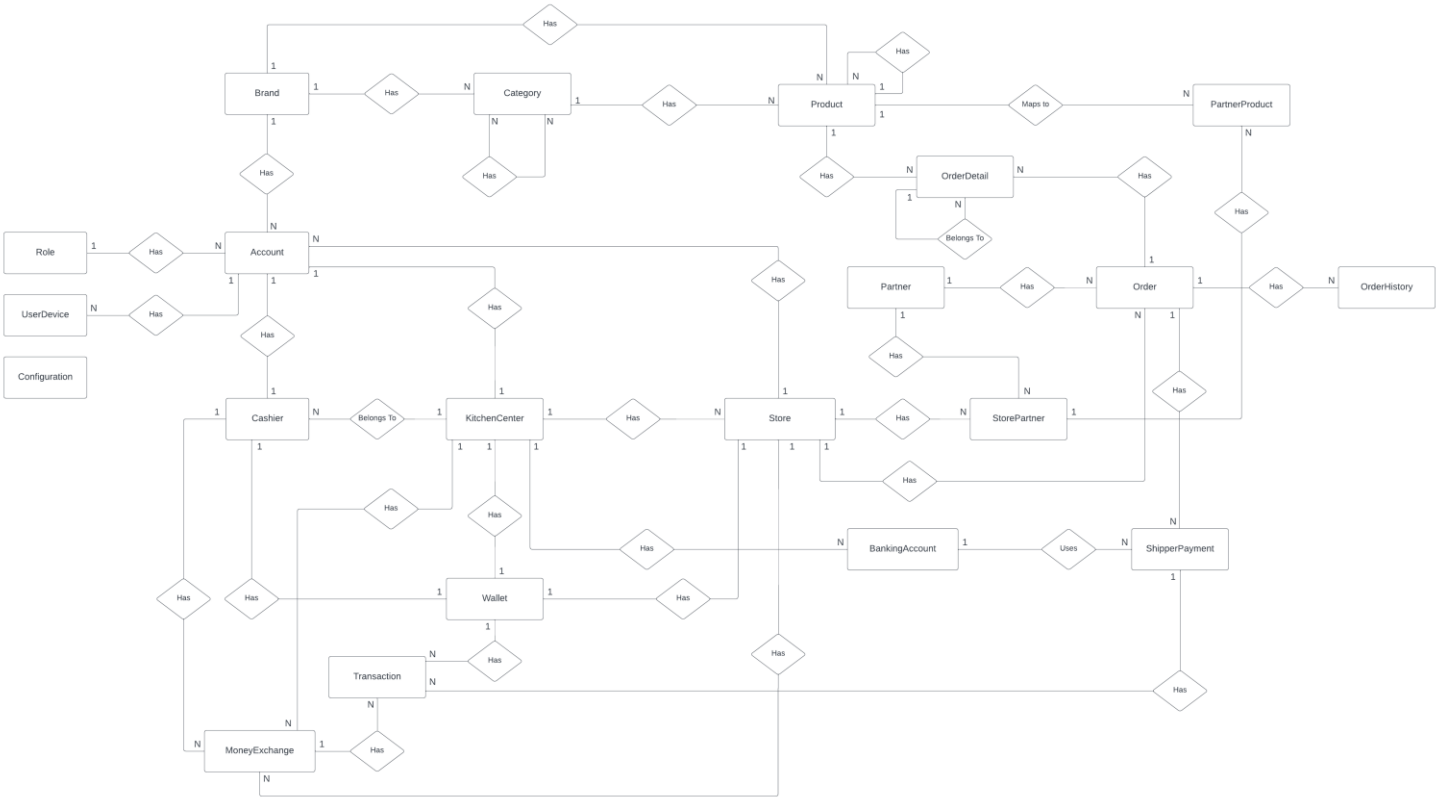


Figure 15 : Entity Relationship Diagram

3.1.5.1. Entities Description

No	Entity	Description
1	Role	Role of user account included: MBKC Admin, Kitchen Center Manager, Brand Manager, Store Manager, Cashier.
2	Account	Store information about user account.
3	KitchenCenter	Store all the information about kitchen center.
4	Brand	Store all the information about the brand.
5	Cashier	Store all the information about the cashier.
6	Category	Store all information about category that has 2 types of category: Normal category and Extra category.
7	Product	Store all information about product that has 4 types of product: Single, Parent, Child, Extra.
8	Store	Store all information about store.
9	Partner	Store all information about partner.
10	BankingAccount	Store all information about banking account.

11	Wallet	Store balance of wallet.
12	Order	Store all information about order.
13	OrderDetail	Store all information of order detail.
14	OrderHistory	Store all the information of order history.
15	PartnerProduct	Store all information about partner products; that information is provided to support mapping when the system collects orders from the partners the store is using.
16	StorePartner	Store all information about the store partner including username and password which is the account to access the partner's merchant system that the store uses.
17	ShipperPayment	Store all information about shipper payment.
18	Transaction	Store all information about transaction.
19	MoneyExchange	Store all information about money exchange.
20	UserDevice	Store all information about user device.
21	Configuration	Store all information about system configuration.

Table 27: Entities Description

3.2. Functional requirements

3.2.1. Accounts Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Kitchen center manager, Brand manager, Store manager, Cashier	View account information	View information of a specific account	Account id		
2	Kitchen center manager, Brand manager, Store manager, Cashier	Update account information	Update password for first time login	Password Confirm password	Password: + Type: text field (string) + Not empty Confirm password: + Type: text field (string) + Not empty + Confirmation password must match the password	BR-02

Table 28: Account Feature

3.2.2. Authentications Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	MBKC admin, Kitchen center manager, Brand manager, Store manager, Cashier	Login	Login to mbkc.vercel.app website and MBKC mobile app	Email Password	Email: + Type: text field (string) + Email address format + Not empty Password: + Type: text field (string) + Not empty	BR-01 BR-02 BR-03
2	Kitchen center manager, Brand manager, Store manager, Cashier	Update password	Reset password for account	Password Confirm password	Password: + Type: text field (string) + Not empty Confirm password: + Type: text field (string) + Not empty	BR-02

					+ Confirmation password must match the password	
3	MBKC admin, Kitchen center manager, Brand manager, Store manager, Cashier	Regeneration token	Re-generate pair token from the old pair token provided by the MBKC system before.	Access token Refresh token	AccessToken: + Type: text field (string) + Not empty RefreshToken: + Type: text field (string) + Not empty	BR-05 BR-06

Table 29: Authentications Feature

3.2.3. Banking Accounts Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Kitchen center manager, Cashier	View list banking account	Get a list of banking accounts from the system with condition paging, searchByName, sort	SearchValue CurrentPage ItemsPerPage SortBy IsGetAll Kitchen center id	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC"	
2	Kitchen center manager	Create banking account	Create new banking account	Bank name Bank logo Number account Kitchen center id	Bank name: + Type: text field (string) + Not empty + Max Length = 100 Bank logo: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg Number account: + Type: text field (string) + Not empty + Min Length = 10 + Max Length = 20	BR-11 BR-12 BR-71 BR-72

3	Kitchen center manager	Get specific banking account	Get specific banking account information of a kitchen center.	Banking account id Kitchen center id		
4	Kitchen center manager	Update banking account	Update banking account information	Bank name Bank logo	Bank name: + Type: text field (string) + Not empty + Max Length = 100 Bank logo: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	BR-11 BR-12 BR-71
5	Kitchen center manager	Update status banking account	Update banking account status	Banking account id Status	Status: + Must equal "INACTIVE" OR "ACTIVE"	
6	Kitchen center manager	Delete banking account	Delete a specific banking account	Banking account id		

Table 30: Banking Account Feature

3.2.4. Brands Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	MBKC admin	View list brand	Get a list of brands from the system	SearchValue ItemsPerPage CurrentPage SortBy	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC"	
2	MBKC admin	Create brand	Create new brand	Brand name Email brand manager Address Province District Ward Logo	Brand name: + Type: text field (string) + Not empty + Max Length = 120 Email brand manager: + Type: text field (string) + Email address format + Not empty	BR-04 BR-11 BR-12 BR-13 BR-20 BR-21

					+ Max Length = 100 Address: + Type: text field (string) + Not empty + Max Length = 255 Province + Type: dropdown + Not empty District + Type: dropdown + Not empty Ward + Type: dropdown + Not empty Logo + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	
3	MBKC admin, Brand manager	Get specific brand information	Get specific information of brand	Brand id		
4	MBKC admin	Update brand information	Update an existed brand information	Brand name Email brand manager Address Province District Ward Logo Brand id	Brand name: + Type: text field (string) + Not empty + Max Length = 120 Email brand manager: + Type: text field (string) + Email address format + Not empty + Max Length = 100 Address: + Type: text field (string) + Not empty + Max Length = 255 Province: + Type: dropdown + Not empty District:	BR-04 BR-11 BR-12 BR-13 BR-20 BR-21

					+ Type: dropdown + Not empty Ward: + Type: dropdown + Not empty Logo: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	
5	MBKC admin	Update status brand	Update an existed brand status	Brand id Status	Status: + Must equal “INACTIVE” OR “ACTIVE”	
6	MBKC admin	Delete brand	Delete existed brand	Brand id		
7	Brand manager	View brand profile	Get brand profile information	Brand id		
8	Brand manager	Update brand profile	Update an existed brand profile	Brand name Address Province District Ward Logo Brand id	Brand name: + Type: text field (string) + Not empty + Max Length = 120 Address: + Type: text field (string) + Not empty + Max Length = 255 Province: + Type: dropdown + Not empty District: + Type: dropdown + Not empty Ward: + Type: dropdown + Not empty Logo: + Type: file (image) + Not empty + Max size < 5MB	BR-04 BR-11 BR-12 BR-13 BR-20 BR-21

					+ Extension file: .png, .jpg, .webp, .jpeg	

Table 31: Brands Feature

3.2.5. Cashiers Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Kitchen center manager	View list cashiers	Get a list of cashiers from the system	Kitchen center id Cashier id SearchValue ItemsPerPage CurrentPage SortBy	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC"	
2	Kitchen center manager	Create cashier	Create new cashier	Email Full name Gender Date of birth Citizen number Logo Kitchen center id	Email: + Type: text field (string) + Not empty + Max Length = 100 + Email address format Full name: + Type: text field (string) + Not empty + Max Length = 180 Gender: + Type: dropdown + Not empty Date of birth: + Type: datetime picker + Not empty Citizen number: + Type: text field (string) + Not empty + Length = 12 Logo: + Type: file (image) + Not empty	BR-04 BR-11 BR-12 BR-73 BR-74 BR-75

					+ Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	
3	Kitchen center manager, Cashier	Get specific cashier information	Get specific information of cashier	Kitchen center id Cashier id		
4	Kitchen center manager, Cashier	Update cashier information	Update an existed cashier information	Full name Gender Date of birth Citizen number Logo	Full name: + Type: text field (string) + Not empty + Max Length = 180 Gender: + Type: dropdown + Not empty Date of birth: + Type: datetime picker + Not empty Citizen number: + Type: text field (string) + Not empty + Length = 12 Logo: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	BR-04 BR-11 BR-12 BR-73 BR-74 BR-75
5	Kitchen center manager	Update status cashier	Update an existed cashier status	Kitchen center id Cashier id Status	Status: + Must equal “INACTIVE” OR “ACTIVE”	
6	Kitchen center manager	Delete cashier	Delete existed cashier	Kitchen center id Cashier id		
7	Cashier	View cashier report	Get cashier report of a shift	Cashier id		

Table 32: Cashiers Feature

3.2.6. Categories Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Brand manager, Store manager	View list categories	Get a list of categories from the system	Brand id Type SearchValue ItemsPerPage CurrentPage SortBy IsGetAll	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC" Type: + Required: "Normal" or "Extra"	
2	Brand manager	Create category	Create new category	Category name Category code Display order Description Logo Brand id	Category name: + Type: text field (string) + Not empty + Max Length = 100 Category code: + Type: text field (string) + Not empty + Max Length = 20 Display order: + Type: text field(number) + Not empty Description: + Type: text field (string) + Not empty + Max Length = 200 Logo: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	BR-11 BR-12 BR-26 BR-27 BR-28 BR-29 BR-30 BR-31 BR-32 BR-33
3	Brand manager, Store manager	Get specific category	Get a specific category	Category id Brand id		

4	Brand manager	Update category information	Update an existed category information	Category name Display order Description Logo	Category name: + Type: text field (string) + Not empty + Max Length = 100 Display order: + Type: text field (number) + Not empty Description: + Type: text field (string) + Not empty + Max Length = 200 Logo: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	BR-11 BR-12 BR-26 BR-27 BR-28 BR-29 BR-30 BR-31 BR-32 BR-33
5	Brand manager	Delete category	Delete existed category	Category id Brand id		
6	Brand manager	Get extra categories by category	Get extra categories by specific category	Category id Brand id SearchValue ItemsPerPage CurrentPage SortBy IsGetAll	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC"	
7	Brand manager	Add extra categories to normal category	Add list of extra categories to specific normal category	List extra category	List extra category: + Type: check box	

Table 33: Categorie Feature

3.2.7. Configurations Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	MBKC admin	Get configuration	Get configuration of system			
2	MBKC admin	Update configuration	Update configuration of system	Order receiving start time. Order receiving end time. Transfer money to kitchen center time. Transfer money to kitchen store time.	Order receiving start time: + Type: time picker (string) + Not empty Order receiving end time: + Type: time picker (string) + Not empty Transfer money to kitchen center time: + Type: time picker (string) + Not empty Transfer money to kitchen store time: + Type: time picker (string) + Not empty	BR-56 BR-67 BR-68

Table 34: Configurations Feature

3.2.8. Dashboards Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	MBKC admin	View admin dashboard	Get information about total of kitchen center, total of brands, total of stores, total of partners, new kitchen centers, new brands and list active stores			
2	Kitchen center manager	View kitchen center dashboard	Get information about total balance in wallet daily, total of stores, total of cashiers, money exchange to kitchen center in last seven day, list of active stores list of cashiers	Kitchen center id		

3	Brand manager	View brand dashboard	Get information about total of store, total of normal categories, total of extra categories, total of products, revenue of store in last seven day, list of active stores list products with number of sold products	Brand id		
4	Store manager	View store dashboard	Get information about total of upcoming orders, total of preparing orders, total of ready orders, total of completed orders revenue of store in today	Store id		
5	Cashier	View cashier dashboard	Get information about total of revenue today, total of orders of today, list of orders delivery successfully, list new shipper payments, list new transactions	Cashier id		

Table 35: Dashboards Feature

3.2.9. Kitchen Centers Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	MBKC admin, Brand manager	View list kitchen centers	Get a list of kitchen centers from the system	SearchValue ItemsPerPage CurrentPage SortBy IsGetAll Brand id	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC"	
2	MBKC admin	Create kitchen center	Createw kitchen center	Kitchen center name Manage email Address Province District Ward Logo	Kitchen center name: + Type: text field (string) + Not empty + Max Length = 100 Manager email: + Type: text field (string) + Not empty + Max Length = 100 + Email address format Address: + Type: text field(string) + Not empty + Max Length = 255 Province: + Type: dropdown + Not empty District: + Type: dropdown + Not empty Ward: + Type: dropdown + Not empty Logo:	BR-09 BR-10 BR-11 BR-12 BR-13 BR-14 BR-15 BR-16

					+ Type: file (image) + Not empty + Max size < 5MB + Extension: .png, .jpg, .webp, .jpeg	
3	MBKC admin	Get a specific kitchen center	Get a specific kitchen center	Kitchen center id		
4	MBKC admin	Update kitchen center information	Update an existed kitchen center information	Kitchen center name Manage email Address Province District Ward Logo Kitchen center id	Kitchen center name: + Type: text field (string) + Not empty + Max Length = 100 Manager email: + Type: text field (string) + Not empty + Max Length = 100 + Email address format Address: + Type: text field(string) + Not empty + Max Length = 255 Province: + Type: dropdown + Not empty District: + Type: dropdown + Not empty Ward: + Type: dropdown + Not empty Logo: + Type: file (image) + Not empty + Max size < 5MB + Extension: .png, .jpg, .webp, .jpeg	BR-09 BR-10 BR-11 BR-12 BR-13 BR-14 BR-15 BR-16
5	MBKC admin	Delete kitchen center	Delete existed kitchen center	Kitchen center id		

6	Kitchen center manager	Get kitchen center profile	Get profile of kitchen center	Kitcen center id		
7	MBKC admin	Update kitchen center status	Update status of an existed kitchen center	Kitchen center id Status	Status: + Must equal “INACTIVE” OR “ACTIVE”	

Table 36: Kitchen Centers Feature

3.2.10. Money Exchanges Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Kitchen center manager, Store manager, Cashier	View list money exchange	Get a list of money exchanges from the system	ItemsPerPage CurrentPage ExchangeType SearchDateFrom SearchDateTo Status SortBy Kitchen center id Store id Cashier id	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format “PropertyName_ASC” or “PropertyName_DESC” ExchangeType: + Must equal “SEND”, “RECEIVE” or “WITHDRAW” SearchDateFrom: + Type: text field (string) + Format type dd/MM/yyyy + Must less than or equal SearchDateTo SearchDateTo: + Type: text field (string) + Format type dd/MM/yyyy Status: + Must equal “SUCCESS” or “FAIL”	
2	Kitchen center manager	View list money exchange for kitchen center	Get list money exchanges of kitchen center	Kitchen center id ItemsPerPage CurrentPage SearchDateFrom	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0	

				SearchDateTo Status SortBy	SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC" SearchDateFrom: + Type: text field (string) + Format type dd/MM/yyyy + Must less than or equal SearchDateTo SearchDateTo: + Type: text field (string) + Format type dd/MM/yyyy Status: + Must equal "SUCCESS" or "FAIL"	
3	Cashier	Money exchange to kitchen center	Send money from wallet of cashier to kitchen center	Cashier id		
4	Kitchen center manager	Withdraw money to store	Withdraw money in wallet of store	Store Amount Image	Store: + Type: dropdown + Not empty Amount: + Type: text field (number) + Not empty + Min = 1000 + Max = 999999999 Image: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	BR-11 BR-12

Table 37: Money Exchanges Feature

3.2.11. Orders Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Cashier	Confirm order to completed	Confirm order status to completed.	Order Partner Id Banking Account Id Image	Order Partner Id: + Type: text field (string) + Not empty + Max Length = 100 Banking Account Id: + Type: number + Not empty + Greater than 0 Image: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg	BR-11 BR-12
2	Kitchen center manager, Store manager, Cashier	Get orders	Get all orders for a specified store or kitchen center, cashier.	SearchValue ItemsPerPage CurrentPage SystemStatus PartnerOrderStatus SearchDateFrom SearchDateTo SortBy ConfirmedBy Kitchen center id Store id	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC" SearchDateFrom: + Type: text field (string) + Format type dd/MM/yyyy + Must less than or equal SearchDateTo SearchDateTo: + Type: text field (string) + Format type dd/MM/yyyy SystemStatus: + SystemStatus is required IN_STORE,	

					READY_DELIVERY, COMPLETED, CANCELLED PartnerOrderStatus: + SystemStatus is required PREPARING, READY, UPCOMING, COMPLETED, CANCELLED	
3	MBKC admin, Kitchen center manager, Store manager, Cashier	Get order	Get order by id.	Order id Kitchen center id Store id		
4	Store manager	Change order to ready	Change order status to ready.	Order id Store id		
5	Cashier	Change order to ready delivery	Change order status to ready delivery.	Order id Store id		
6	Store manager	Cancel order	Cancel order when order status is UPCOMING or PREPARING	Order id Store id		
7	MBKC bot	Get order	Get order detail for only MBKC bot role	Order id		
8	MBKC bot	Post order	Create order for only MBKC bot role	Order: OrderPartnerId ShipperName ShipperPhone CustomerName CustomerPhone Note PaymentMethod DeliveryFee SubTotalPrice TotalDiscount FinalTotalPrice	OrderPartnerId: + Type: text field (string) + Not empty DisplayId: + Type: text field (string) + Not empty Address: + Type: text field (string) + Not empty ShipperName: + Type: text field (string) + Not empty	

				Commission Tax PartnerOrderStatus PartnerId StoreId DisplayId Address Cutlery StorePartnerCommission Order details ProductId SellingPrice Quantity Note ExtraOrderDetails ProductId SellingPrice Quantity Note	ShipperPhone: + Type: text field (string) + Not empty CustomerName: + Type: text field (string) + Not empty CustomerPhone: + Type: text field (string) + Not empty Note (Order): + Type: text field (string) + Not empty PaymentMethod: + Type: text field (string) + Not empty + Must be “Cash”, “Cashless” DeliveryFee: + Type: text field (number) + Greater than or equal 0 SubTotalPrice: + Type: text field (number) + Greater than or equal 0 TotalDiscount: + Type: text field (number) + Greater than or equal 0 FinalTotalPrice: + Type: text field (number) + Greater than or equal 0 Commission: + Type: text field (number) + Greater than or equal 0 Tax: + Type: text field (number) + Greater than or equal 0 PartnerOrderStatus:	

					+ Type: text field (string) + Not empty + Must be “Upcoming”, “Preparing” PartnerId: + Type: text field (number) + Greater than or equal 0 StoreId: + Type: text field (number) + Greater than or equal 0 Cutlery: + Type: text field (number) + Greater than or equal 0 StorePartnerCommission: + Type: text field (number) + Between 0 and 100 ProductId: + Type: text field (number) + Greater than 0 SellingPrice: + Type: text field (number) + Greater than or equal 0 Quantity: + Type: text field (number) + Greater than 0 Note (Order Detail): + Type: text field (string)	
9	MBKC bot	Put order	Update order for only MBKC bot role	Order id Status	Status: + Type: text field (string) + Not empty + must be “Upcoming”	

Table 38: Orders Feature

3.2.12. Partner Products Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Brand manager	Get partner products	Get partner products in the system.	SearchValue ItemsPerPage CurrentPage SortBy Brand id	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC"	
2	Brand manager	Get partner product	Get a specific partner product by storeId, partnerId, productId.	Store id Brand id Partner id Product id		
3	Brand manager	Create partner product	Create new partner product.	Product Store Partner Product Code Status Partner Product Product Price	Product: + Type: dropdown + Not empty Store: + Type: dropdown + Not empty Partner: + Type: dropdown + Not empty Product code: + Type: text field (string) + Not empty + Max Length = 180 Product price + Type: text field (number) + Not empty + Greater than 0	
4	Brand manager	Update partner	Update an existed	Product Code	Product code:	

		product	partner product.	Status Partner Product Product Price Brand id Product id Partner id Store id	+ Type: text field (string) + Not empty + Max Length = 180 Status Partner Product + Type: dropdown + Not empty Product price + Type: text field (number) + Not empty + Greater than 0	
5	Brand manager	Update partner product status	Update status of an existed partner product.	Product id Parter id Store id Brand id Status Partner Product	Status Partner Product + Type: dropdown + Not empty	
6	Brand manager	Delete partner product	Delete an existed partner product by id.	Product id Parter id Store id Brand id		

Table 39: Partner Products Feature

3.2.13. Partners Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	MBKC admin, Brand manager	Get partners	Get a list of partners from the system with search, sort, paging.	SearchValue ItemsPerPage CurrentPage SortBy IsGetAll	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC"	
2	MBKC admin	Get partner	Get specific partner by partner id.	Partner id		
3	MBKC admin	Update partner	Update an existed	Logo	Logo:	

			partner information.	Web URL Partner id	+ Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg Web URL: + Type: text field (string) + Not empty + Max Length = 150	
4	MBKC admin	Update partner status	Update an existed partner's status.	Partner id Status	Status: + Required INACTIVE or ACTIVE	

Table 40: Partners Feature

3.2.14. Products Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	MBKC admin, Brand manager, Store manager, Kitchen center manager	Get products	Get Products in the system.	SearchValue ItemsPerPage CurrentPage SortBy ProductType IdCategory IdStore IsGetAll Brand id	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC" ProductType: + Type: string + Required some types such as: SINGLE, PARENT, CHILD, EXTRA.	
2	MBKC admin, Brand manager, Store manager, Kitchen center manager	Get product	Get a specific product by id.	Product id Brand id		
3	Brand manager	Get product sold	Get Products with	SearchValue	CurrentPage:	

			number of products sold in the system.	ItemsPerPage CurrentPage ProductSearchDateFrom ProductSearchDateTo SortBy ProductType IdCategory IsGetAll	+ Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC" ProductType: + Type: string + Required some types such as: SINGLE, PARENT, CHILD, EXTRA. ProductSearchDateFrom: + Type: string + Required format dd/MM/yyyy. ProductSearchDateTo: + Type: string + Required format dd/MM/yyyy.	
4	Brand manager	Create product	Create a new product.	Image Product Type Product name Parent product Product size Product code Display order Selling price Discount price Original price Category Description Brand id	Image: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg Product type: + Type: dropdown + Not empty Product name: + Type: text field (string) + Not empty + Max Length = 120 + Required for product type SINGLE, PARENT and EXTRA Parent product: + Type: dropdown + Not empty	BR-11 BR-12 BR-33 BR-34 BR-35 BR-36 BR-37 BR-38 BR-39 BR-40 BR-41 BR-42 BR-43 BR-44 BR-45 BR-46 BR-47 BR-48

					+ Required for product type CHILD Product size: + Type: dropdown + Not empty + Required for product type CHILD Product code: + Type: text field (string) + Not empty + Max Length = 20 Display order: + Type: text field (number) + Not empty + Greater than 0 Selling price: + Type: text field (number) + Not empty + Greater than 0 + Required for product type SINGLE, CHILD and EXTRA Discount price: + Type: text field (number) + Not empty + Greater than 0 + Required for product type SINGLE, CHILD and EXTRA. Less than or equal to the selling price. Original price: + Type: text field (number) + Not empty + Greater than 0 + Required for product type SINGLE, CHILD and EXTRA. Category: + Type: dropdown + Not empty	BR-49 BR-50 BR-51

					+ Required for product type PARENT, SINGLE and EXTRA. Description: + Type: text field (string) + Not empty + Max Length = 1000	
5	Brand manager	Update product	Update a specific product information.	Image Product name Parent product Display order Selling price Discount price Original price Category Description Brand id Product id	Image: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg Product name: + Type: text field (string) + Not empty + Max Length = 120 + Required for product type SINGLE, PARENT and EXTRA Parent product: + Type: dropdown + Not empty + Required for product type CHILD Display order: + Type: text field (number) + Not empty + Greater than 0 Selling price: + Type: number + Not empty + Greater than 0 + Required for product type SINGLE, CHILD and EXTRA Discount price: + Type: text field (number) + Not empty	BR-11 BR-12 BR-33 BR-34 BR-35 BR-36 BR-37 BR-38 BR-39 BR-40 BR-41 BR-42 BR-43 BR-44 BR-45 BR-46 BR-47 BR-48 BR-49 BR-50 BR-51

					+ Greater than 0 + Required for product type SINGLE, CHILD and EXTRA. Less than or equal to the selling price. Original price: + Type: text field (number) + Not empty + Greater than 0 + Required for product type SINGLE, CHILD and EXTRA. Category: + Type: dropdown + Not empty + Required for product type PARENT, SINGLE and EXTRA. Description: + Type: text field (string) + Not empty + Max Length = 1000	
6	Brand manager	Update product status	Update a specific product status.	Product id Brand id Status	Status: + Required Status is required some statuses such as: INACTIVE, ACTIVE.	
7	Brand manager	Delete product	Delete a specific product.	Product id Brand id		

Table 41: Products Feature

3.2.15. Shipper Payments Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Store manager, Kitchen center manager, Cashier	Get shipper payments	Get Products in the system.	ItemsPerPage CurrentPage SearchDateFrom SearchDateTo SortBy	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy:	

				Status PaymentMethod Store id Kitchen center id Cashier id	+ Match with format "PropertyName_ASC" or "PropertyName_DESC" Status: + Type: string + Required some types such as: SUCCESS or FAIL. ProductSearchDateFrom: + Type: string + Required format dd/MM/yyyy. ProductSearchDateTo: + Type: string + Required format dd/MM/yyyy. PaymentMethod: + Type: string + Required CASH, CASHLESS.	

Table 42: Shipper Payments Feature

3.2.16. Store Partners Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Brand manager	Get store partners	Get Products in the system.	SearchValue ItemsPerPage CurrentPage SortBy IsGetAll Brand id	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format "PropertyName_ASC" or "PropertyName_DESC"	
2	Brand manager	Get store partner	Get a store partner information by store Id	Store id Partner id Brand id		
3	Brand manager	Create store partners	Create new store partners.	Store Partner Username	Store: + Type: dropdown Partner:	BR-52

				Password Commission	+ Type: dropdown + Not empty Username: + Type: text field (string) + Not empty + Max Length = 100 Password: + Type: text field (string) + Not empty + Max Length = 50 Commission: + Type: text field (number) + Min = 0 + Max = 100	
4	Brand manager	Update store partner	Update a specific store partner information.	Username Password Commission Brand id Store id Partner id	Username: + Type: text field (string) + Not empty + Max Length = 100 Password: + Type: text field (string) + Not empty + Max Length = 50 Commission: + Type: number + Not empty + Min = 0 + Max = 100	BR-52
5	Brand manager	Update store partner status	Update stasus of store partner.	Brand id Store id Partner id Status	Status: + Required Status is required some statuses such as: INACTIVE, ACTIVE.	
6	Brand manager	Delete store partner	Delete a specific store partner.	Brand id Store id Partner id		

Table 43: Store Partners Feature

3.2.17. Stores Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	MBKC admin, Brand manager, Kitchen center manager	Get stores	Get stores in the system.	SearchValue ItemsPerPage CurrentPage SortBy IsGetAll Brand id Kitchen center id	CurrentPage: + Greater than 0 ItemsPerPage: + Greater than 0 SortBy: + Match with format “PropertyName_ASC” or “PropertyName_DESC”	
2	MBKC admin, Brand manager, Kitchen center manager	Get store	Get a specific store by store id.	Store id Brand id Kitchen center id		
3	Brand manager	Register new store	Register new store.	Logo Store name Manager email Kitchen center Brand id	Logo: + Type: file (image) + Not empty + Max size < 5MB + Extension file: .png, .jpg, .webp, .jpeg Store name: + Type: text field (string) + Not empty + Max Length = 80 Manager email: + Type: text field (string) + Not empty + Max Length = 100 + Required Email Format Kitchen center: + Type: dropdown + Not empty	BR-11 BR-12 BR-13 BR-22 BR-23
4	Brand manager	Update store	Update information of an existed store of a brand.	Store name Manager email Brand id	Store name: + Type: text field (string) + Not empty	BR-13

					+ Max Length = 80 Manager email: + Type: text field (string) + Not empty + Max Length = 100 + Required Email Format	
5	MBKC admin, Brand manager	Update store status	Update status of an existed store of a brand.	Store id Brand id Status	Status: + Required Status is required some statuses such as: INACTIVE, ACTIVE.	
6	MBKC admin	Delete store	Delete an existed store.	Store id		
7	MBKC admin	Confirm store registration	Confirm a store registration.	RejetedReason	RejetedReason: + Type: text field (string) + Not empty + Max Length = 250	
9	MBKC bot	Get list store	Get list store for only MBKC bot role			

Table 44: Stores Feature

3.2.18. User Devices Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Store manager	Create new user device	Get stores in the system.	FCM Token	FCM Token: + Not empty	
2	Store manager	Delete user device	Delete an existed user device	User device id		
3	MBKC bot	Delete user device	Delete an existed user device for only MBKC bot role	UserDeviceId		

Table 45: User Devices Feature

3.2.19. Verifications Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Store manager, Brand manager, Kitchen center manager, Cashier	Email verification	Verify email before resetting password.	Email	Email: + Type: text field (string) + Not empty + Max Length = 100 + Required Email Format	
2	Store manager, Brand manager, Kitchen center manager, Cashier	Otp verification	Compare sent OTP Code in the system with receiver's OTP Code.	Email OTP Code	Email: + Type: text field (string) + Not empty + Max Length = 100 + Required Email Format OTP Code: + Required 6 digits + Not empty	BR-07 BR-08

Table 46: Verifications Feature

3.2.20. Wallets Feature

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Store manager, Kitchen center manager, Cashier	Get wallet	Get wallet by cashier id, kitchencenter id or store id	SearchDate	SearchDate: + Type: text field (string) + Not empty + Required format dd/MM/yyyy	

Table 47: Wallets Feature

4. Non-Functional Requirements

4.1. External Interfaces

4.1.1. User Interfaces

- Vietnamese and English must be supported in the user interface (UI).
- Icons must be easy for users to identify what it's present for.
- Colors must have contrast.
- Font must be easy to read.

4.1.2. Communications Interfaces

- The system should transfer data using the HTTP request methods.
- HTTP Protocol is used for communication between Web Application and server.

4.2. Quality Attributes

4.2.1. Usability

- Vietnamese and English should be used to write all text, labels, alerts, and notifications for end users in User Interfaces.
- The application should be friendly and easy for users to use after training.

4.2.2. Reliability

- To be productive with the system, all users need to be trained.
- The system failure rate is less than 5%, while the system availability is greater than 80%.
- Fewer than ten critical bugs per year.

4.2.3. Performance

- All responses' time is less than 15 seconds.

4.2.4. Security

- All input data are validated before saving to the database.
- The system always checks authorization before executing a function.
- Password must be hashed by the MD5 algorithm on the client-side before being sent to the server-side.
- Each role can only access its own set of functions.

5. Requirement Appendix

5.1. Business Rules

No	Business Rule
BR-01	Username must be Email format.
BR-02	Password must be hashed by the MD5 algorithm on the client-side before being sent to the server-side.
BR-03	Password length of account must be 8-50 characters.

BR-04	Email must be unique.
BR-05	Access token will expire within 1 hour after the successful login.
BR-06	Refresh token will expire 5 days after the successful login.
BR-07	OTP code must include 6 digits.
BR-08	The OTP code will exist within 10 minutes after being sent to the Email of user.
BR-09	Kitchen center name is required less than or equal to 100 characters.
BR-10	Kitchen center address is required less than or equal to 255 characters.
BR-11	Image/Logo file is required file length greater than 0 and less than 5MB.
BR-12	Image/Logo file is required extension type “.png”, “.jpg”, “.jpeg”, “.webp”.
BR-13	Manager Email of the kitchen center is required in standard Email format and is required less than or equal to 100 characters.
BR-14	A kitchen center just has one manager, and a manager just manages one kitchen center.
BR-15	When creating a new kitchen center successfully, the account of the kitchen center manager will be sent to the email address of the kitchen center manager that was provided by MBKC Admin, and the kitchen will be assigned a wallet in the MBKC system.
BR-16	When the kitchen center manager, brand manager, store manager, cashier logs into the system for the first time, they are required to change their password to be able to continue using the system.
BR-17	The new brand will be created by MBKC admin in the system, which means that the brand has successfully signed a contract with MBKC admin outside the system before.
BR-18	A brand has only one manager, and a manager only manages one brand. Besides, the brand can replace the manager, but the new manager's email must not be the same as the email of the manager previously assigned to the brand.
BR-19	When creating a new brand successfully, the brand manager's account will be sent to the Email address provided by MBKC Admin.
BR-20	Brand name is required less than or equal to 120 chacracters.
BR-21	Brand address is required less than or equal to 255 characters.
BR-22	When the brand manager registers a new store in a specific kitchen center, the account of the store manager will be inactive status. After MBKC Admin accepts this store registration, the store account will be changed to active status, the store account will be sent to the store manager's email, and the store will be assigned a wallet in the MBKC system.
BR-23	The MBKC admin will receive a store registration from a brand manager. The MBKC admin must accept or reject this registration. If the MBKC admin rejects the store registration, the store account will not be sent to the email of the store manager, the status of the store is still inactive, and the wallet will not be assigned to the store.
BR-24	When the MBKC admin rejects the store registration, the MBKC admin must give the reason he or she does not accept it.
BR-25	The rejected reason is required less than or equal to 250 characters.
BR-26	Category code is required less than or equal to 20 characters.
BR-27	Category code is unique in a brand.
BR-28	Category name is required less than or equal to 100 characters.
BR-29	Category name is unique in a brand.
BR-30	The category name in the MBKC system must be given the same name as the category name in the partner system, which will help in mapping product data. If the name is not the same, the mapping product data will fail.
BR-31	Category type is 2 types that are “Normal” and “Extra”.

BR-32	Category description is required less than or equal to 100 characters.
BR-33	Display order is required an integer number that is greater than 0.
BR-34	Product code is required less than or equal to 20 characters.
BR-35	Product code is unique in a brand.
BR-36	Product name is required less than or equal to 120 characters.
BR-37	Product name is unique in a brand.
BR-38	The product name in the MBKC system must be given the same name as the product name in the partner system, which will help in mapping product data. If the name is not the same, the mapping product data will fail.
BR-39	The product type is 4 types that are “Single”, “Parent”, “Child”, “Extra”
BR-40	The product type is required less than or equal to 20 characters.
BR-41	If a product in the partner system is assigned an attribute group that relates to size, each size will be a “child” type, and the main product will be a “parent” type in the MBKC system.
BR-42	The rule name of the child product must be followed in the format "Parent product name - Size [S, M, L]".
BR-43	Selling price of product is required greater than or equal to 0 VNĐ.
BR-44	Discount price of product is required greater than or equal to 0 VNĐ.
BR-45	Historical price of product is required greater than or equal to 0 VNĐ.
BR-46	Selling price is required greater than or equal to historical price.
BR-47	Discount price is required less than or equal to selling price.
BR-48	Product size is 3 types that are “S”, “M”, “L”.
BR-49	The product size just has to be for the “Child” type of product.
BR-50	Selling price, historical price, and discount price are not required for the “Parent” type of product.
BR-51	When creating a new product that is “Child” type, the category is not required for this type of product.
BR-52	When the brand manager creates a new store partner for a specific store, he or she must provide the store's account with the partner system, which will help in mapping product data and collecting order data for this store. Besides, he or she must provide a commission percent between this store and the partner system that will help in calculating the collected price.
BR-53	When the MBKC system cannot map product data between the MBKC system and a partner system, it will create a new Excel file containing all non-mapping products from the partner side. The MBKC system will send an email that will attach a newly created Excel file.
BR-54	When the brand manager creates a new partner product or updates an existing partner product in the MBKC system, the brand manager must provide exactly the price and the status of the product in the partner system.
BR-55	MBKC Bot will collect order data following the schedule (from crawling order start time to crawling order end time) that is installed by MBKC admin.
BR-56	Crawling order start time is required less than or equal to crawling order end time.
BR-57	When the MBKC Bot parses order data failed, the failed orders will be sent to the store manager's email.
BR-58	Order has 2 types of status that are partner order status and system status.
BR-59	Partner order status is required for creating a new order that is “Upcoming” or “Preparing”.
BR-60	System status is required for creating a new order that is “In Store.”
BR-61	Partner order status will be changed from “Upcoming” to “Preparing” in the MBKC system. When the order that is pre-ordered on the partner system comes on time and changes status, the

	MBKC bot will collect order data and change the status of the order in the MBKC system.
BR-62	Store manager just be allowed to cancel order when the order status is “Upcoming” or “Preparing”.
BR-63	The collected price of the order will be calculated based on the provided commission percent in the store partner, the subtotal price of the order (not including delivery fee, promotion price), total store discount price, and tax partner commission.
BR-64	When an order is confirmed to have a “Ready” status, which is the status after finishing the preparation of the order, the store manager will bring the finished order to the cashier, and the cashier will change the system status of the order from “In Store” to “Ready Delivery".
BR-65	The creating shipper payment is required importing an image that is evidence for the delivered successfully.
BR-66	When the cashier transfers money to the kitchen center’s wallet, the cashier cannot confirm any orders and cannot create any shipper payments.
BR-67	The crawling order end time must be at least 10 minutes earlier than the time for automatically transferring money to the kitchen center’s wallet.
BR-68	The time of automatically transferring money to the kitchen center’s wallet must be at least 1 hour earlier than the time for automatically transferring money from kitchen center’s wallet to store’s wallet.
BR-69	When withdrawing money for a store, you must import an image that serves as proof that the money was successfully transferred.
BR-70	The balance of the store wallet is required to be more than or equal to the required amount when withdrawing money for a store.
BR-71	Banking account name is required less than or equal to 100 characters.
BR-72	Banking account number is required from 10 to 20 digits.
BR-73	Cashier fullname is required less than or equal to 180 characters.
BR-74	Cashier birthdate is required from 18 to 55 years old.
BR-75	Cashier’s citizen number is required exactly equal to 12 digits.

Table 48: Business Rules

5.2. Application Messages List (Partial)

No	Message Code	Message Type	Context	Content
1	MSG-01	Toast message	Login with wrong username or wrong password.	Email or Password is invalid.
2	MSG-02	Toast message	Login with disabled account.	Account has been disabled.
3	MSG-03	Toast message	Login with correct username and correct password.	Login Successfully.
4	MSG-04	Toast message	Change new password successfully.	Reset Password Successfully.
5	MSG-05	In line	Create / Update kitchen center with invalid name.	• Name is not empty. • Name is required less than or equal to 100 characters.
6	MSG-06	In line	Create / Update kitchen center with invalid address.	• Address is not empty. • Address is required less than or equal to 255 characters.

				• Please select province. • Please select district. • Please select ward.
7	MSG-07	In line	Create / Update kitchen center with invalid manager email.	• Manager Email is invalid Email format. • Manager Email is required less than or equal to 100 characters. • ManagerEmail is not empty. • Kitchen center manager email already existed in the system.
8	MSG-08	Toast message	Create / Update kitchen center with invalid image.	• Image is required file length greater than 0 and less than 5MB. • Image is required extension type .png, .jpg, .jpeg, .webp.
9	MSG-09	Toast message	Create kitchen center successfully.	Created New Kitchen Center Successfully.
10	MSG-10	Toast message	Update kitchen center successfully.	Updated Kitchen Center Information Successfully.
11	MSG-11	Toast message	Create / Update brand with invalid name.	• Name is not empty. • Name is required less then or equal to 120 characters.
12	MSG-12	Toast message	Create / Update brand with invalid address.	• Address is not empty. • Address is required less than or equal to 255 characters. • Please select province. • Please select district. • Please select ward.
13	MSG-13	Toast message	Create / Update brand with invalid manager email.	• Brand Manager Email is an invalid Email format. • Brand Manager Email is required less then or equal to 100 characters. • Brand Manager Email is not empty. • Brand manager email already existed in the system.
14	MSG-14	Toast message	Register / Update store with invalid name.	• Name is not empty. • Name is required less than or equal to 80 characters.
15	MSG-15	Toast message	Register / Update store with invalid manager email.	• Store Manager Email is not

				empty. • Store Manager Email is required less than or equal to 100 characters. • Store manager email already existed in the system. • Store Manager Email is invalid Email format.
16	MSG-16	Toast message	Register / Update store with invalid image.	• Image is required file length greater than 0 and less than 5MB. • Image is required extension type .png, .jpg, .jpeg, .webp.
17	MSG-17	Toast message	Confirm store registration with invalid rejected reason.	Rejected Reason is required less than or equal to 250 characters.
18	MSG-18	Toast message	Register store sussessfully.	Registered New Store Successfully.
19	MSG-19	Toast message	Confirm store registration successfully.	Confirmed Store Registration Successfully.
20	MSG-20	Toast message	Create Category with invalid category code.	• Category code already existed in brand. • Category code is not empty. • Category code is required less than or equal to 20 characters.
21	MSG-21	Toast message	Create category with invalid name.	• Category name is not empty. • Category name is is required less than or equal to 100 characters.
22	MSG-22	Toast message	Create category with invalid description.	• Description is not empty. • Description is required less than or equal to 100 characters.
23	MSG-23	Toast message	Create category with invalid image.	• Image is required file length greater than 0 and less than 5MB. • Image is required extension type .png, .jpg, .jpeg, .webp.
24	MSG-24	Toast message	Create category with invalid display order.	• Display order is not empty. • Display order must be greater than 0.
25	MSG-25	Toast message	Create category successfully.	Created New Category Successfully.
26	MSG-26	Toast message	Update category successfully.	Updated Category Successfully.

27	MSG-27	Toast message	Deleted Category Successfully.	Deleted Category Successfully.
28	MSG-28	Toast message	Create Extra-Categories To Normal Category Successfully.	Create Extra-Categories to Normal Category Successfully.
29	MSG-29	Toast message	Create product with invaid code.	• Code is required less than or equal to 20 characters. • Code is not empty.
30	MSG-30	Toast message	Create product with invalid name.	• Name is required less than or equal to 120 characters. • Name is not empty.
31	MSG-31	Toast message	Create product with invalid description.	• Description is required less than or equal to 1000 characters. • Description is not empty.
32	MSG-32	Toast message	Create product with invalid price.	• Selling price is required greater than or equal to 0. • Historical price is required greater than or equal to 0. • Selling price is required greater than or equal to Historical price. • Discount price is required less than or equal to Selling price.
33	MSG-33	Toast message	Created New Store Partner Successfully.	Created New Store Partner Successfully.
34	MSG-34	Toast message	Deleted Store Partner Successfully.	Deleted Store Partner Successfully.
35	MSG-35	Toast message	Updated Store Partner Status Successfully.	Updated Store Partner Status Successfully.
36	MSG-36	Toast message	Updated Store Partner Information Successfully.	Updated Store Partner Information Successfully.
37	MSG-37	Toast message	Change the status of partner product successfully.	Change the status of partner product successfully.
38	MSG-38	Toast message	Created New Partner Product Successfully.	Created New Partner Product Successfully.
39	MSG-39	Toast message	Created New Cashier Successfully.	Created New Cashier Successfully.
40	MSG-40	Toast message	Updated Cashier Successfully.	Updated Cashier Successfully.
41	MSG-41	Toast message	Updated Cashier Status Successfully.	Updated Cashier Status Successfully.
42	MSG-42	Toast message	Deleted Cashier Successfully.	Deleted Cashier Successfully.

43	MSG-43	Toast message	Update configuration with invalid ScrawlingOrderStartTime.	ScrawlingOrderStartTime is invalid time (HH:mm:ss).
44	MSG-44	Toast message	Update configuration with invalid ScrawlingOrderEndTime is.	ScrawlingOrderEndTime is invalid time (HH:mm:ss).
45	MSG-45	Toast message	Update configuration with ScrawlingOrderStartTime before ScrawlingOrderEndTime.	Scrawling order start time is required less than or equal to Scrawling order end time.

Table 49: Application Messages List

IV. Software Design Description

1. System Design

1.1. System Architecture Overview

1.1.1 Overall System Architecture

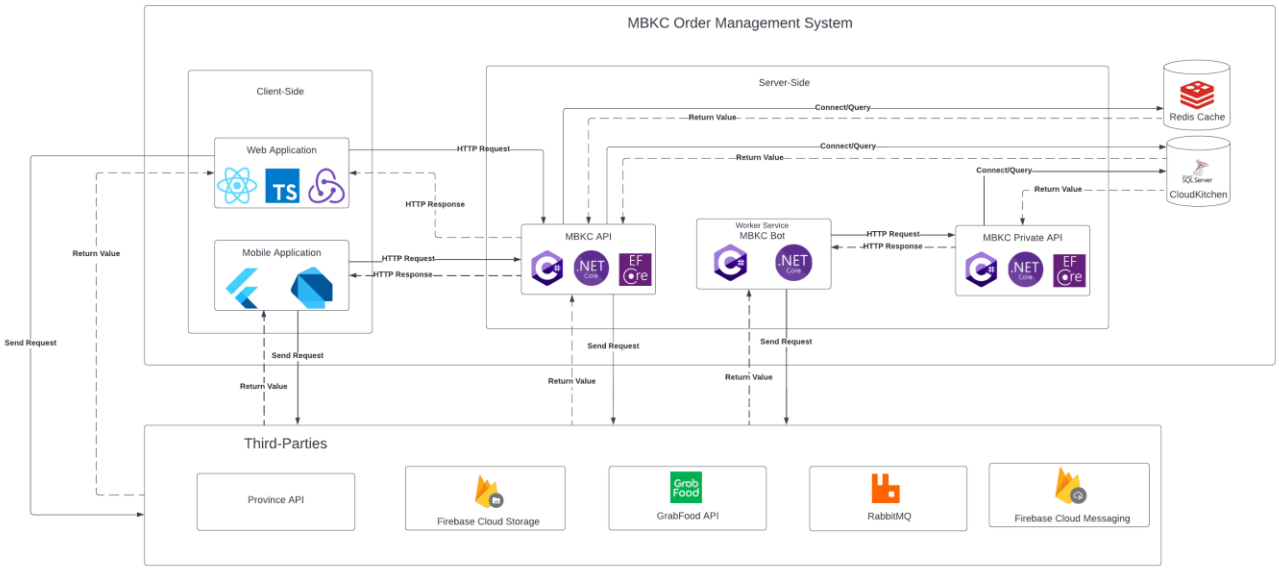


Figure 16: Overall System Architecture

1.1.2 Detail System Architecture

1.1.2.1. Web Application Sub-system

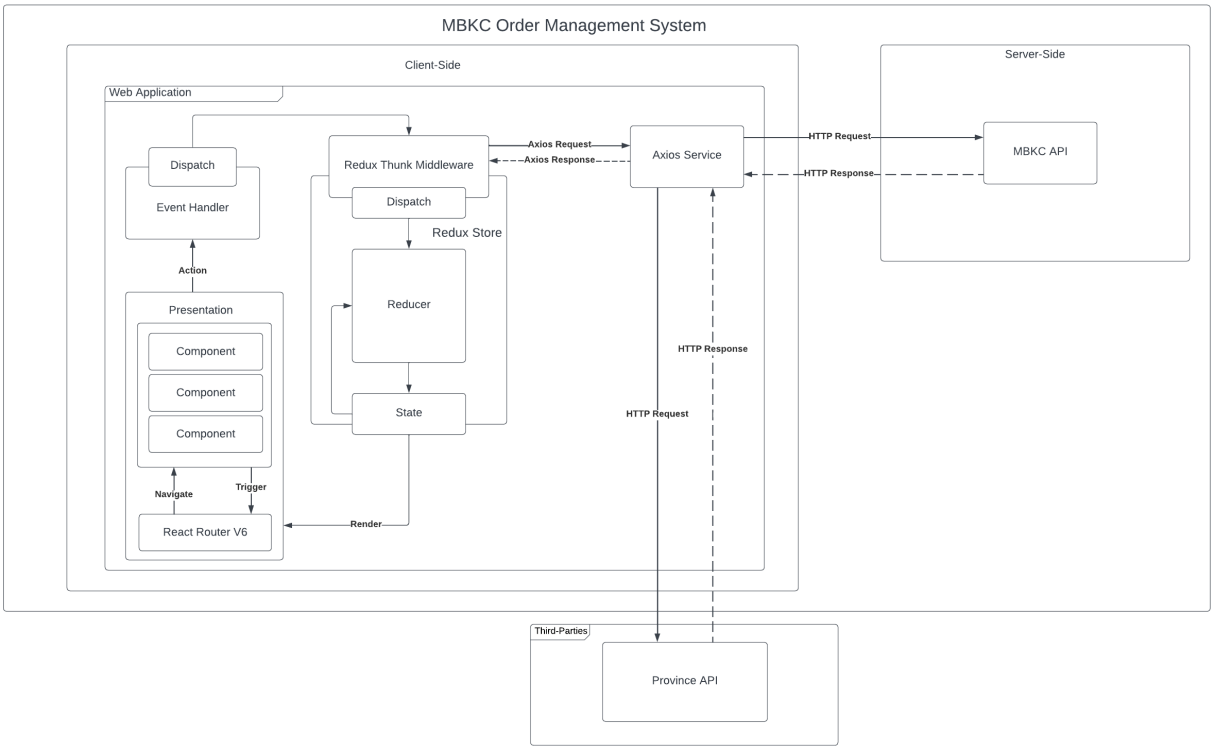


Figure 17: Web Application Sub-system

1.1.2.2. Mobile Application Sub-system

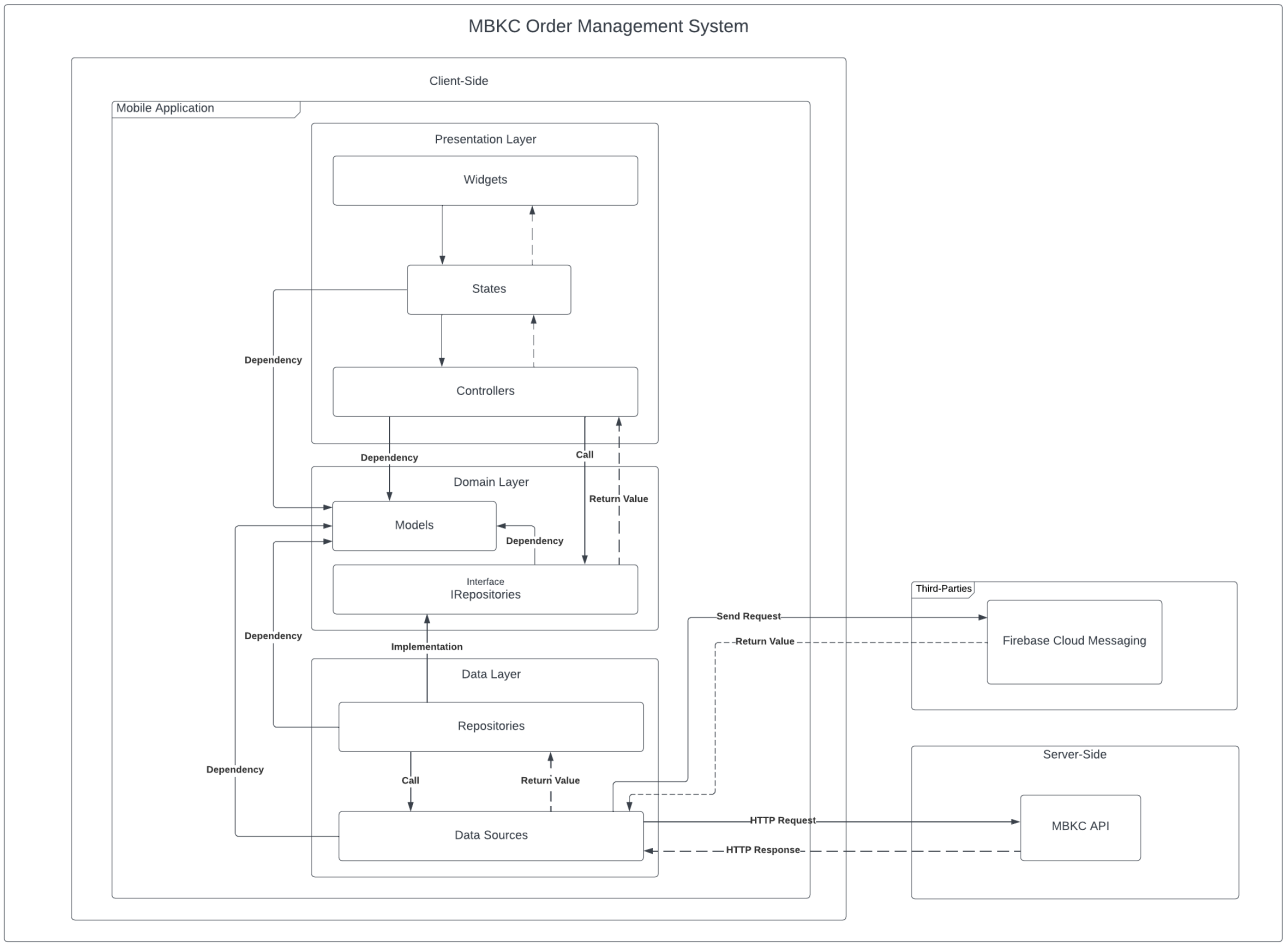


Figure 18: Mobile Application Sub-system

1.1.2.3. MBKC API Sub-system

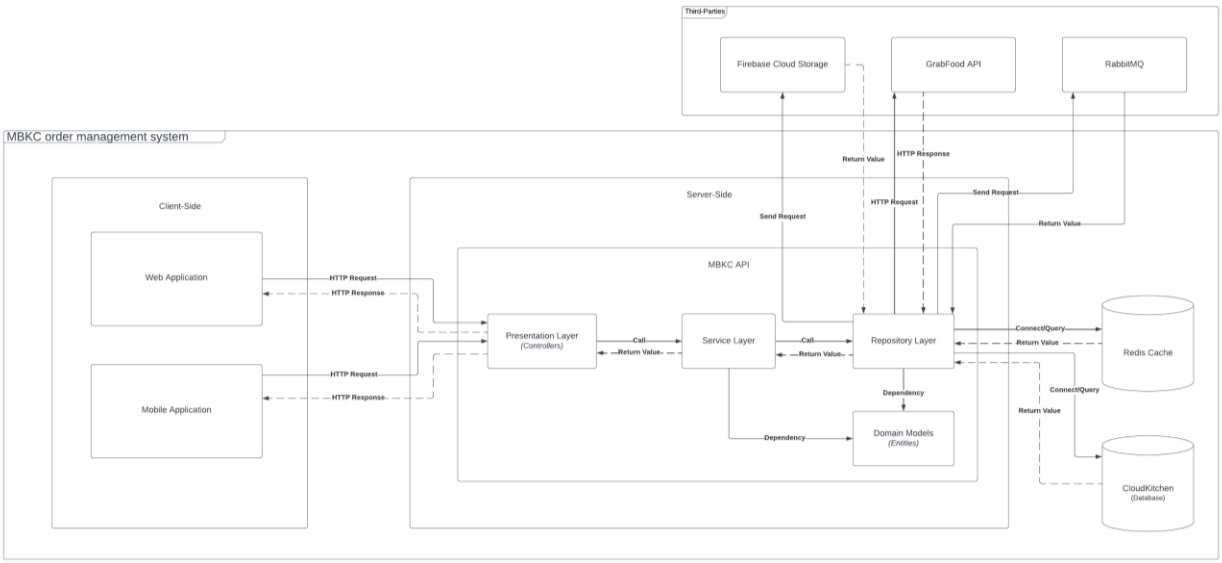


Figure 19: MBKC API Sub-system

1.1.2.4. MBKC Private API Sub-system

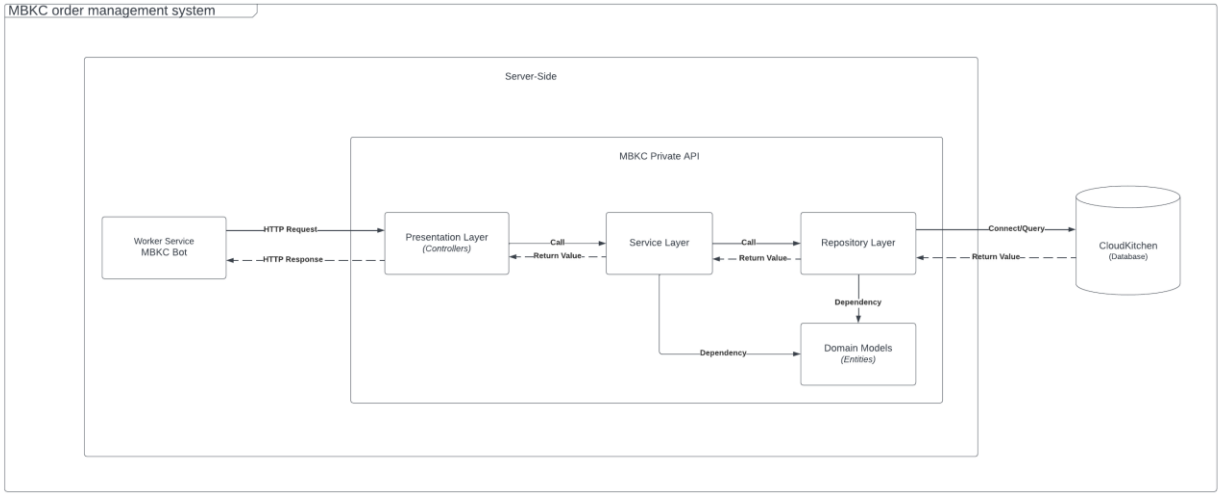


Figure 20: MBKC Private API Sub-system

1.1.2.5. MBKC Bot Sub-system

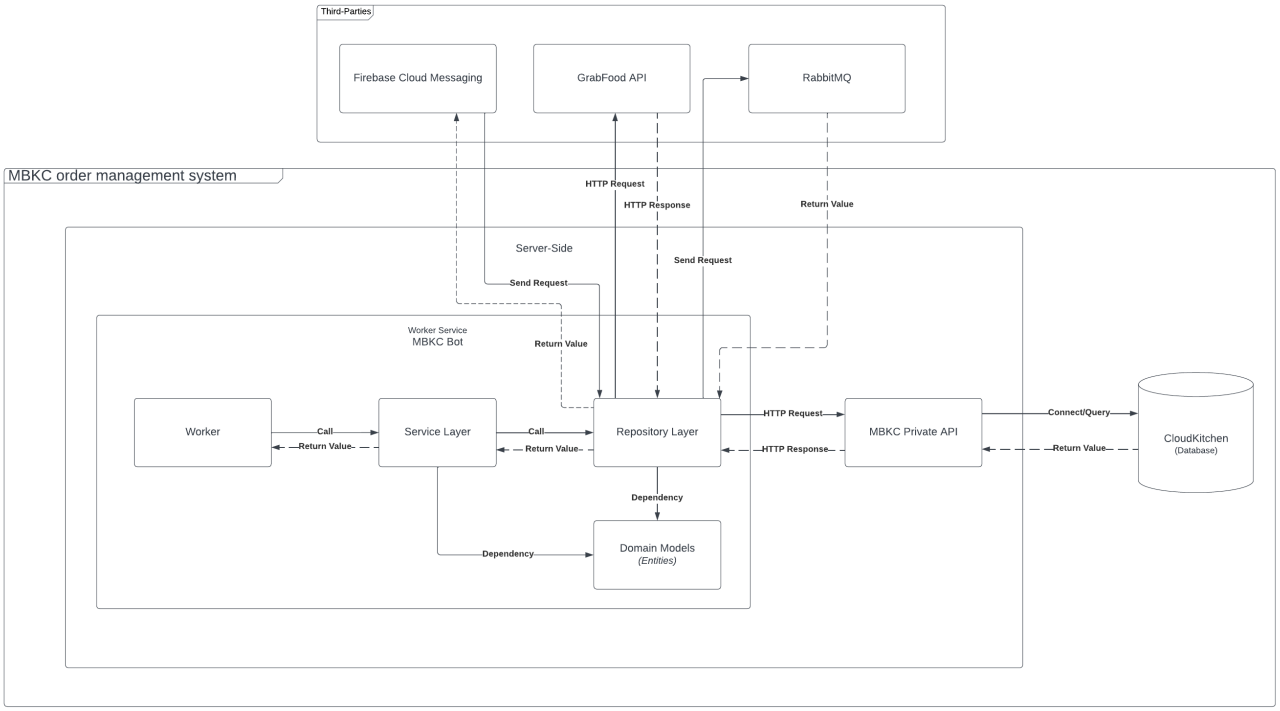


Figure 21: MBKC Bot Sub-system

1.2. Package Diagram

1.2.1. Web Application Package Diagram

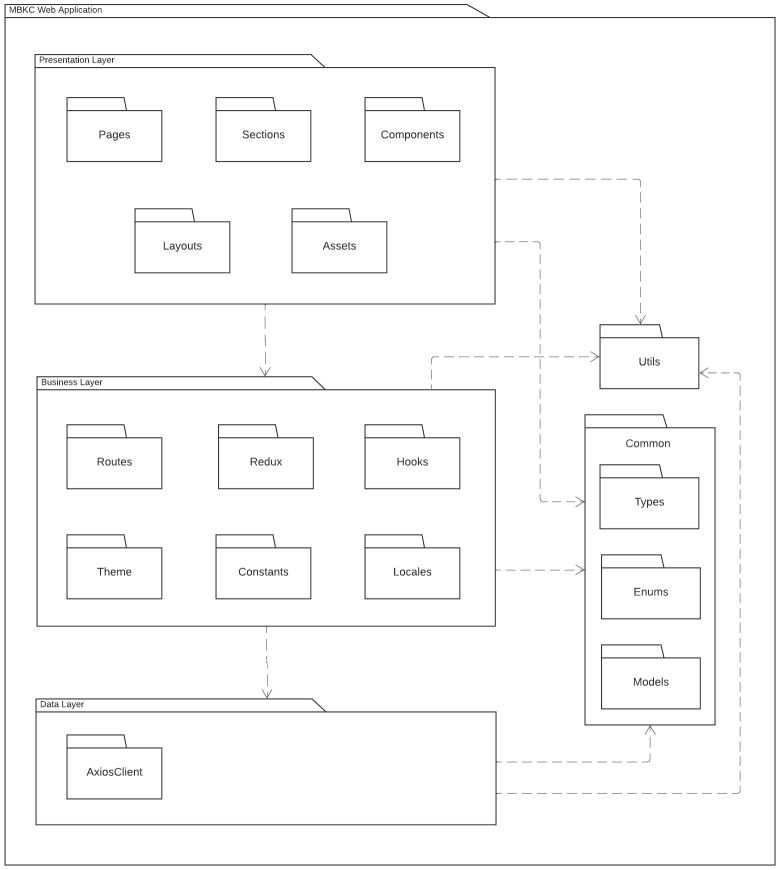


Figure 22: Web Application Package Diagram

Package Descriptions

No	Package	Description
1	Pages	The package includes page components for all pages of the entire application.
2	Sections	The package includes the function components for handling each section in the page interface.
3	Components	The package includes the detailed components of the page interface.
4	Layouts	This pack includes the general layouts of the interface.
5	Assets	The package includes static files, the components and pages used.
6	Routes	The package includes a config URL path for routing to the page component and the sitemap for the entire application.
7	Redux	The package includes a state management pattern for a web application and functions for sending requests to the API service and receiving responses from the API service.

8	Hooks	The package includes the custom hooks for components used to handle logic.
9	Theme	The package includes the custom css of components from the UI library for components of the application used to handle the UI.
10	Constants	The package includes defines constant for the entire application use.
11	Locales	The package includes configs and defines multiple languages for the application.
12	AxiosClient	The package includes API service for the Front-end.
13	Utils	The package includes the utilities function for the entire application use.
14	Common	The package includes three packages: types, enums, models. To define model interfaces, enums, and common interfaces to use for the entire application.
15	Types	The package includes common interfaces for common use across services for the entire application.
16	Enums	This package includes Typescript enums to define a set of named constants for common use for the entire application
17	Models	The package includes model interfaces to map models from Back-end

Table 50: Web Application Package Descriptions

1.2.2. Mobile Application Package Diagram

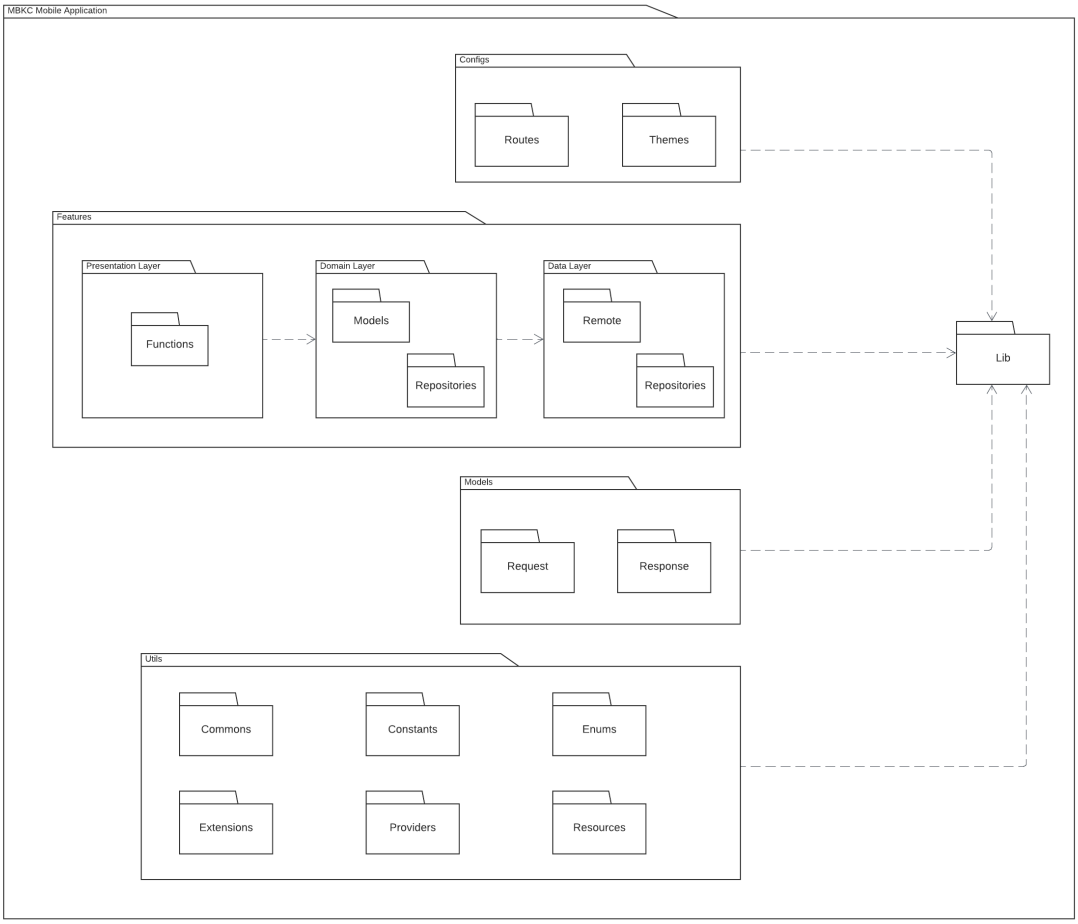


Figure 23: Mobile Application Package Diagram

Package Descriptions

No	Package	Description
1	Lib	This package contains the entire content of the source code.
2	Configs	This package contains all common settings for the application.
3	Features	This package includes all the features of the application.
4	Models	This package contains all the common models for the entire application.
5	Utils	This package is used to handle common cases, used in many files.
6	Routers	This package contains navigation configuration for each detailed screen.
7	Themes	This package is the place to configure the main color tones for the entire application.

8	Presentation Layer	This package is for handling the user interface and user interaction with the application. It is the layer closest to the user and is responsible for presenting data and receiving input.
9	Functions	This package contains a detailed screen of a function.
10	Domain Layer	This package contains the core business logic of the application. It is responsible for representing the concepts and entities of the business domain, and it defines the rules and behavior of the application.
11	Models in Domain Layer	This package includes application models.
12	Repositories in Domain Layer	This package includes repositories that have interface-level functions.
13	Data Layer	This package is responsible for managing the storage and retrieval of data used by the application. It interacts with the infrastructure of the application, such as the database, external APIs, or any other data source.
14	Remote	This package performs data processing tasks when communicating with the API.
15	Repositories in Data Layer	This package includes repositories that have implement-level functions.
16	Request	This package includes common classes throughout the system used to push information to the API.
17	Response	This package includes common classes throughout the system used to receive information from the API.
18	Commons	This package includes shared user interface components and other additional shared functions.
19	Constants	This package includes strings, variables, and user interface elements that do not change during application execution.
20	Enums	This package is responsible for converting the number or string type received from the API to the enum type in the application.
21	Extentions	This package contains extensions to handle errors from the API or user interactions on the screen.

22	Providers	This package is used to contain shared providers throughout the application process.
23	Resources	This package contains a shared class to handle responses when calling the API.

Table 51: Mobile Application Package Descriptions

1.2.3. MBKC API Package Diagram

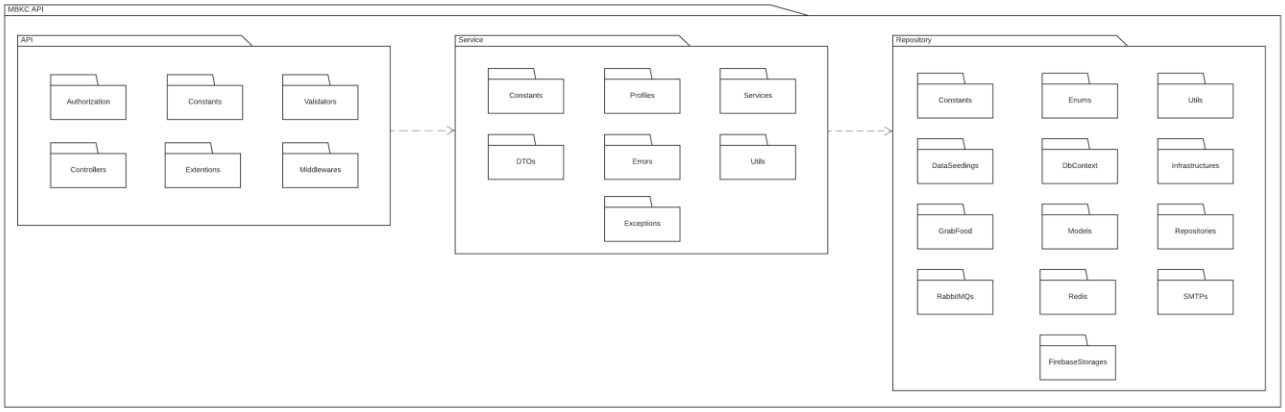


Figure 24: MBKC API Package Diagram

Package Descriptions

API		
No	**Package**	**Description**
1	Authorization	This package is responsible for permission authorization and checking whether those roles in the system can use that API or not.
2	Constant	This package includes constants about the API path and messages about the successful execution of the API.
3	Controllers	This package includes API Controllers.
4	Extentions	This package includes dependency injection, swagger configuration, and exception middleware.
5	Middlewares	This package includes exception middleware, used to throw exceptions to API.
6	Validator	This package is used to check the input of API.

Table 52: MBKC API Package Descriptions - API

Service		
No	**Package**	**Description**
1	Constants	This package includes exception messages during API execution.
2	DTOs	This package includes the API view models.
3	Errors	This package includes classes that define fields to throw exceptions.
4	Exceptions	This package includes exceptions that the api will throw when an error occurs.
5	Profiles	This package is used for mapping between models and view models.
6	Services	This package is used to handle project logic.
7	Utils	This package is used to handle common cases, used in many files.

Table 53: MBKC API Package Descriptions - Service

Repository		
No	**Package**	**Description**
1	Constants	This package includes partner constant.
2	DataSeedings	This package is used to automatically add data to the database when migration using code first.
3	DbContext	This package is used to define classes and their relationships when migration using code first.
4	Enums	These packages are predefined types and states, for clarity when coding.
5	FirebaseStorages	This package includes image processing logic using third party Firebase.
6	GrabFood	This package includes operations related to a third party GrabFood.
7	Infrastructure	This package includes settings for unit of work, factory, disposable.
8	Models	This package includes project models.
9	RabbitMQs	This package includes operations related to connect and send message.
10	Redis	This package includes operations related to connect and communication with redis cache.

11	Repositories	This package includes database processing and direct communication with the database.
12	SMTPs	This package includes handling of sending emails to users.
13	Utils	This package is used to handle common cases, used in many files.

Table 54: MBKC API Package Descriptions - Repository

1.2.4. MBKC Private API Package Diagram

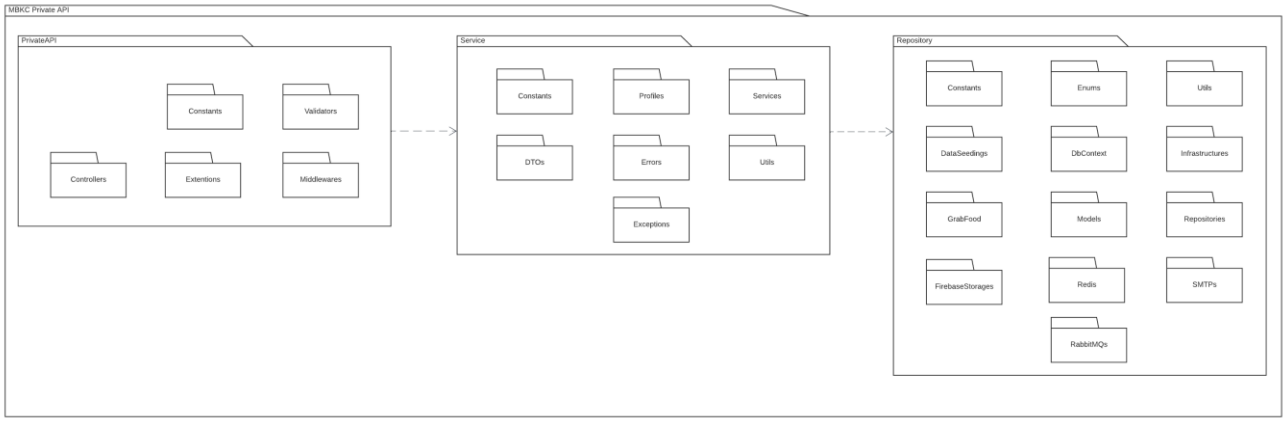


Figure 25: MBKC Private API Package Diagram

Package Descriptions

PrivateAPI		
No	**Package**	**Description**
1	Constant	This package includes constants about the API path and messages about the successful execution of the API.
2	Controllers	This package includes API Controllers.
3	Extentions	This package includes dependency injection, swagger configuration, and exception middleware.
4	Middlewares	This package includes exception middleware, used to throw exceptions to API.
5	Validator	This package is used to check the input of API.

Table 55: MBKC Private API Package Descriptions - PrivateAPI

Service		
No	**Package**	**Description**
1	Constants	This package includes exception messages during API execution.
2	DTOs	This package includes the API view models.
3	Errors	This package includes classes that define fields to throw exceptions.
4	Exceptions	This package includes exceptions that the APIwill throw when an error occurs.
5	Profiles	This package is used for mapping between models and view models.
6	Services	This package is used to handle project logic.
7	Utils	This package is used to handle common cases, used in many files.

Table 56: MBKC Private API Package Descriptions - Service

Repository		
No	**Package**	**Description**
1	Constants	This package includes partner constant.
2	DataSeedings	This package is used to automatically add data to the database when migration using code first.
3	DbContext	This package is used to define classes and their relationships when migration using code first.
4	Enums	These packages are predefined types and states, for clarity when coding.
5	FirebaseStorages	This package includes image processing logic using third party Firebase.
6	GrabFood	This package includes operations related to a third party GrabFood.
7	Infrastructure	This package includes settings for unit of work, factory, disposable.
8	Models	This package includes project models.
9	RabbitMQs	This package includes operations related to connect and send message.
10	Redis	This package includes operations related to connect and communication with redis cache.

11	Repositories	This package includes database processing and direct communication with the database.
12	SMTPs	This package includes handling of sending emails to users.
13	Utils	This package is used to handle common cases, used in many files.

Table 57: MBKC Private API Package Descriptions - Repository

1.2.2. MBKC Bot Package Diagram

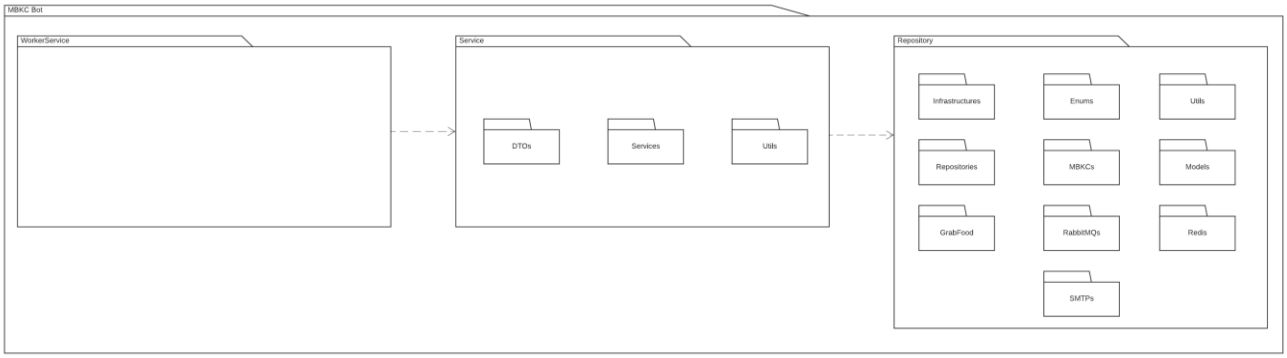


Figure 26: MBKC Bot Package Diagram

Package Descriptions

No	Package	Description
1	WokerService	This package contains worker class to execute background service.
2	DTOs	This package contains ViewModels.
3	Services	This package is used to handle project logic.
4	Utils	This package is used to handle common cases.
5	Infrastructure	This package includes UnitOfWork class.
6	Repositories	This package includes database processing and direct communication with the Private API.
7	MBKCs	This package includes PrivateAPI class about URIs to connect with Private API.
8	Models	This package includes domain models of Private API request and response.
9	GrabFood	This package includes operations related to a third party GrabFood.
10	RabbitMQs	This package includes operations related to connect and receive message.
11	Redis	This package includes operations related to connect and communication with redis

		cache.
12	SMTPs	This package includes handling of sending emails to users.

Table 58: MBKC Bot Package Descriptions

2. Database Design

2.1. Physical Database

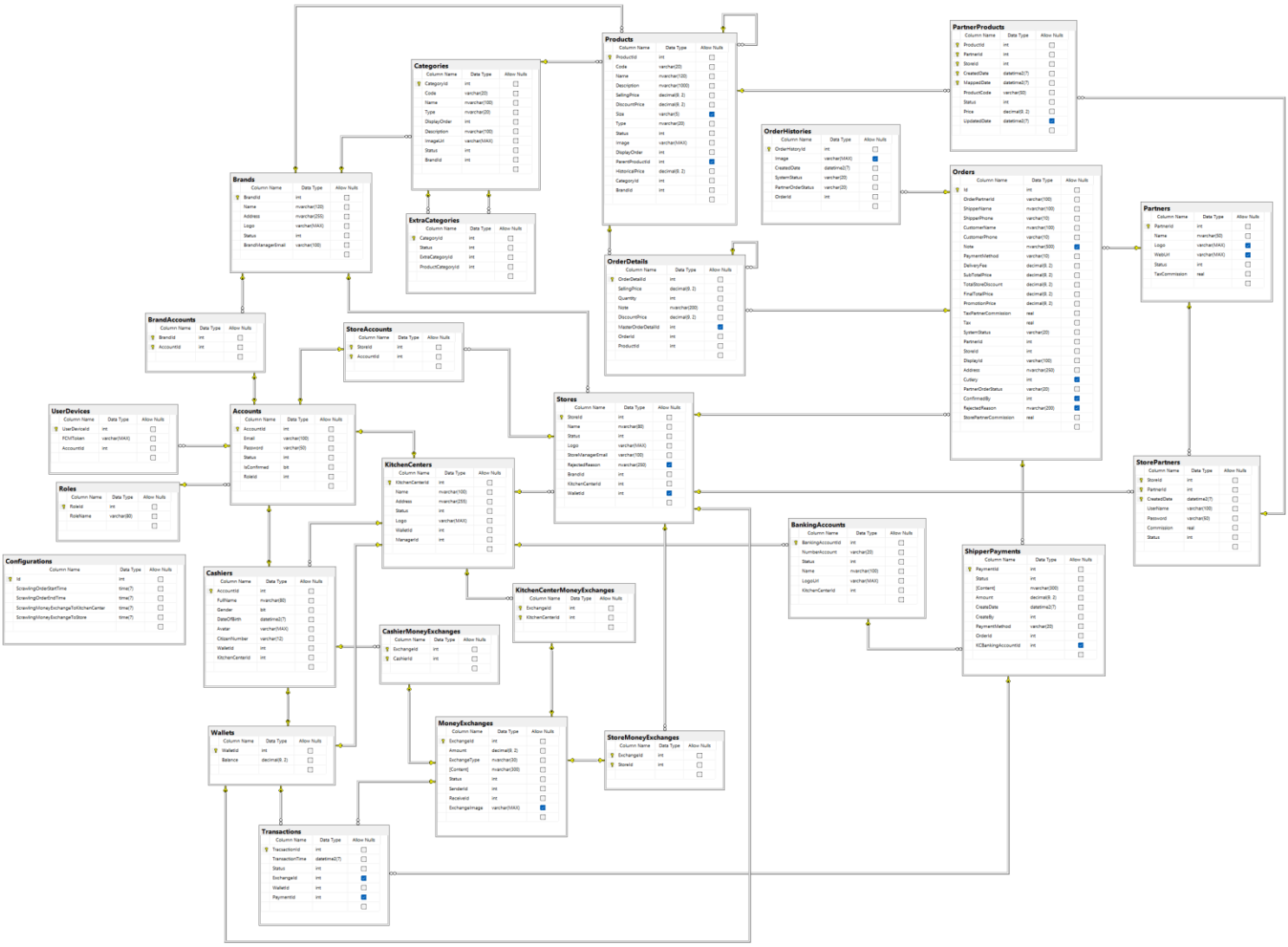


Figure 27: Physical Database

2.2. Data Dictionary

2.2.1. Accounts Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	AccountId	int	x	x	PK	
2	Email	varchar(100)		x		
3	Password	varchar(50)		x		
4	Status	int		x		Has value: • Inactive = 0 • Active 1 • Disable = 2
5	IsConfirmed	bit		x		
6	RoleId	int		x	FK	

Table 59: Accounts Table

2.2.2. Roles Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	RoleId	int	x	x	PK	
2	RoleName	varchar(80)		x		

Table 60: Roles Table

2.2.3. UserDevices Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	UserDeviceId	int	x	x	PK	
2	FCMToken	varchar(MAX)		x		
3	AccountId	int		x		

Table 61: UserDevices Table

2.2.4. BrandAccounts Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	BrandId	int	x	x	PK,FK	
2	AccountId	int	x	x	PK,FK	

Table 62: BrandAccounts Table

2.2.5. Brands Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	BrandId	int	x	x	PK	
2	Name	nvarchar(120)		x		
3	Address	nvarchar(255)		x		
4	Logo	varchar(MAX)		x		
5	Status	int		x		Has value: • Inactive = 0 • Active = 1 • Disable = 2
6	BrandManagerEmail	varchar(100)		x		

Table 63: Brands Table

2.2.6. Cashiers Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	AccountId	int	x	x	PK,FK	
2	FullName	nvarchar(80)		x		
3	Gender	bit		x		
4	DateOfBirth	datetime2(7)		x		
5	Avatar	varchar(MAX)		x		
6	CitizenNumber	varchar(12)		x		
7	WalletId	int		x	FK	
8	KitchenCenterId	int		x	FK	

Table 64: Cashiers Table

2.2.7. CashierMoneyExchanges Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	ExchangeId	int	x	x	PK,FK	
2	CashierId	int	x	x	PK,FK	

Table 65: CashierMoneyExchanges Table

2.2.8. Wallets Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	WalletId	int	x	x	PK	
2	Balance	decimal(9,2)		x		

Table 66: Wallets Table

2.2.9. Configurations Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	Id	int	x	x	PK	
2	ScrawlingOrderStartTime	time(7)		x		
3	ScrawlingOrderEndTime	time(7)		x		
4	ScrawlingMoneyExchangeToKitchenCenter	time(7)		x		
5	ScrawlingMoneyExchangeToStore	time(7)		x		

Table 67: Configurations Table

2.2.10. Categories Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	CategoryId	int	x	x	PK	
2	Code	varchar(20)		x		
3	Name	nvarchar(100)		x		
4	Type	nvarchar(20)		x		Has value: • Normal • Extra
5	DisplayOrder	int		x		
6	Description	nvarchar(100)		x		

7	ImageUrl	varchar(MAX)		x		
8	Status	int		x		Has value: • Inactive = 0 • Active = 1 • Disable = 2
9	BrandId	int		x	FK	

Table 68: Categories Table

2.2.11. ExtraCategories Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	CategoryId	int	x	x	PK	
2	Status	int		x		Has value: • Inactive = 0 • Active = 1 • Disable = 2
3	ExtraCategoryId	int		x	FK	
4	ProductCategoryId	int		x	FK	

Table 69: ExtraCategories Table

2.2.12. StoreAccounts Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	StoreId	int	x	x	PK,FK	
2	AccountId	int	x	x	PK,FK	

Table 70: StoreAccounts Table

2.2.13. KitchenCenters Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	KitchenCenterId	int	x	x	PK	
2	Name	nvarchar(100)		x		
3	Address	nvarchar(255)		x		
4	Status	int		x		Has value: • Inactive = 0 • Active = 1 • Disable = 2

5	Logo	varchar(MAX)		x		
6	WalletId	int		x	FK	
7	ManagerId	int		x	FK	

Table 71: KitchenCenters Table

2.2.14. KitchenCenterMoneyExchanges Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	ExchangeId	int	x	x	PK,FK	
2	KitchenCenterId	int	x	x	PK,FK	

Table 72: KitchenCenterMoneyExchanges Table

2.2.15. MoneyExchanges Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	ExchangeId	int	x	x	PK	
2	Amount	decimal(9,2)		x		
3	ExchangeType	nvarchar(30)		x		Has value: • Send • Receive • Withdraw
4	Content	nvarchar(300)		x		
5	Status	int		x		Has value: • Fail = 0 • Success = 1
6	SenderId	int		x		
7	ReceiveId	int		x		
8	ExchangeImage	varchar(MAX)				

Table 73: MoneyExchanges Table

2.2.16. Transactions Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	TransactionId	int	x	x	PK	
2	TransactionTime	datetime2(7)		x		
3	Status	int		x		Has value: • Fail = 0 • Success = 1
4	ExchangeId	int			FK	
5	WalletId	int		x	FK	
6	PaymentId	int			FK	

Table 74: Transactions Table

2.2.17. Products Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	ProductId	int	x	x	PK	
2	Code	varchar(20)		x		
3	Name	nvarchar(120)		x		
4	Description	nvarchar(1000)		x		
5	SellingPrice	decimal(9,2)		x		
6	DiscountPrice	decimal(9,2)		x		
7	Size	varchar(5)				Has value: • S • M • L
8	Type	nvarchar(20)		x		Has value: • Single • Parent • Child • Extra
9	Status	int		x		Has value:

						• Inactive = 0 • Inactive = 1 • Disable = 2
10	Image	varchar(MAX)		x		
11	DisplayOrder	int		x		
12	ParentProductId	int			FK	
13	HistoricalPrice	decimal(9,2)		x		
14	CategoryId	int		x	FK	
15	BrandId	int		x	FK	

Table 75: Products Table

2.2.18. Stores Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	StoreId	int	x	x	PK	
2	Name	nvarchar(80)		x		
3	Status	int		x		Has value: • Inactive = 0 • Active = 1 • Disable = 2 • Confirming = 3 • Rejected = 4
4	Logo	varchar(MAX)		x		
5	StoreManagerEmail	varchar(100)		x		
6	RejectReason	nvarchar(250)				
7	BrandId	int		x	FK	
8	KitchenCenterId	int		x	FK	
9	WalletId	int			FK	

Table 76: Stores Table

2.2.19. StoreMoneyExchanges Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	ExchangeId	int	x	x	PK,FK	
2	StoreId	int	x	x	PK,FK	

Table 77: StoreMoneyExchanges Table

2.2.20. Orders Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	Id	int	x	x	PK	
2	OrderPartnerId	varchar(100)		x		
3	ShipperName	nvarchar(100)		x		
4	ShipperPhone	varchar(10)		x		
5	CustomerName	nvarchar(100)		x		
6	CustomerPhone	varchar(10)		x		
7	Note	nvarchar(500)				
8	PaymentMethod	varchar(10)		x		
9	DeliveryFee	decimal(9,2)		x		
10	SubTotalPrice	decimal(9,2)		x		
11	TotalStoreDiscount	decimal(9,2)		x		
12	FinalTotalPrice	decimal(9,2)		x		
13	PromotionPrice	decimal(9,2)		x		
14	TaxPartnerCommission	real		x		
15	Tax	real		x		
16	SystemStatus	varchar(20)		x		Has value: • In_store • Ready_delivery • Completed • Cancelled
17	PartnerId	int		x	FK	

18	StoreId	int		x	FK	
19	DisplayId	varchar(100)		x		
20	Address	nvarchar(250)		x		
21	Cutlery	int				
22	PartnerOrderStatus	varchar(20)		x		Has value: • Preparing • Ready • Upcoming • Completed • Cancelled
23	ConfirmedBy	int				
24	RejectedReason	nvarchar(200)				
25	StorePartnerConmission	real		x		

Table 78: Orders Table

2.2.21. OrderDetails Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	OrderDetailId	int	x	x	PK	
2	SellingPrice	decimal(9,2)		x		
3	DiscountPrice	decimal(9,2)		x		
4	Quantity	int		x		
5	Note	nvarchar(200)		x		
6	MasterOrerDetailId	int			FK	
7	OrderId	int		x	FK	
8	ProductId	int		x	FK	

Table 79: OrderDetails Table

2.2.22. PartnerProducts Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	ProductId	int	x	x	PK,FK	

2	PartnerId	int	x	x	PK,FK	
3	StoreId	int	x	x	PK,FK	
4	CreatedDate	datetime2(7)	x	x	PK,FK	
5	MappedDate	datetime2(7)	x	x	PK	
6	ProductCode	varchar(50)		x		
7	Status	int		x		Has value: • Available = 1 • Out of stock today = 2 • Out of stock indefinitely = 3 • Disable = 4
8	Price	decimal(9,2)		x		
9	UpdateDate	datetime2(7)				

Table 80: PartnerProducts Table

2.2.23. OrderHistories Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	OrderHistoryId	int	x	x	PK	
2	Image	varchar(MAX)				
3	CreatedDate	datetime2(7)		x		
4	SystemStatus	varchar(20)		x		Has value: • In store • Ready delivery • Completed • Cancelled
5	PartnerOrderStatus	varchar(20)		x		Has value: • Preparing • Ready • Upcoming • Completed • Cancelled
6	OrderId	int		x	FK	

Table 81: OrderHistories Table

2.2.24. ShipperPayments Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	PaymentId	int	x	x	PK	
2	Status	int		x		Has value: • Inactive = 0 • Active = 1 • Disable = 2
3	Content	nvarchar(300)		x		
4	Amount	decimal(9,2)		x		
5	CreateDate	datime2(7)		x		
6	CreateBy	int		x		
7	PaymentMethod	varchar(20)		x		Has value: • Cash • Cashless
8	OrderId	int		x	FK	
9	KCBankingAccountId	int			FK	

Table 82: ShipperPayments Table

2.2.25. BankingAccounts Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	BankingAccountId	int	x	x	PK	
2	NumberAccountId	varchar(20)		x		
3	Status	int		x		Has value: • Inactive = 0 • Active 1 • Disable = 2
4	Name	nvarchar(100)		x		
5	LogoUrl	varchar(MAX)		x		
6	KitchenCenterId	int		x	FK	

Table 83: BankingAccounts Table

2.2.26. Partners Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	PartnerId	int	x	x	PK	
2	Name	nvarchar(50)		x		
3	Logo	varchar(MAX)				
4	WebUrl	varchar(MAX)				
5	TaxCommission	real		x		
6	Status	int		x		Has value: • Inactive = 0 • Active = 1 • Disable = 2

Table 84: Partners Table

2.2.27. StorePartners Table

No	Field name	Type	Unique	Not null	PK/FK	Notes
1	StoreId	int	x	x	PK,FK	
2	PartnerId	int	x	x	PK,FK	
3	CreatedDate	datetime2(7)	x	x	PK	
4	UserName	varchar(100)		x		
5	Password	varchar(50)		x		
6	Comission	real		x		
7	Status	int		x		Has value: • Inactive = 0 • Active = 1 • Disable = 2

Table 85: StorePartners Table

3. Detailed Design

3.1. Brand's Participation In MBKC System

3.1.1. Class Diagram

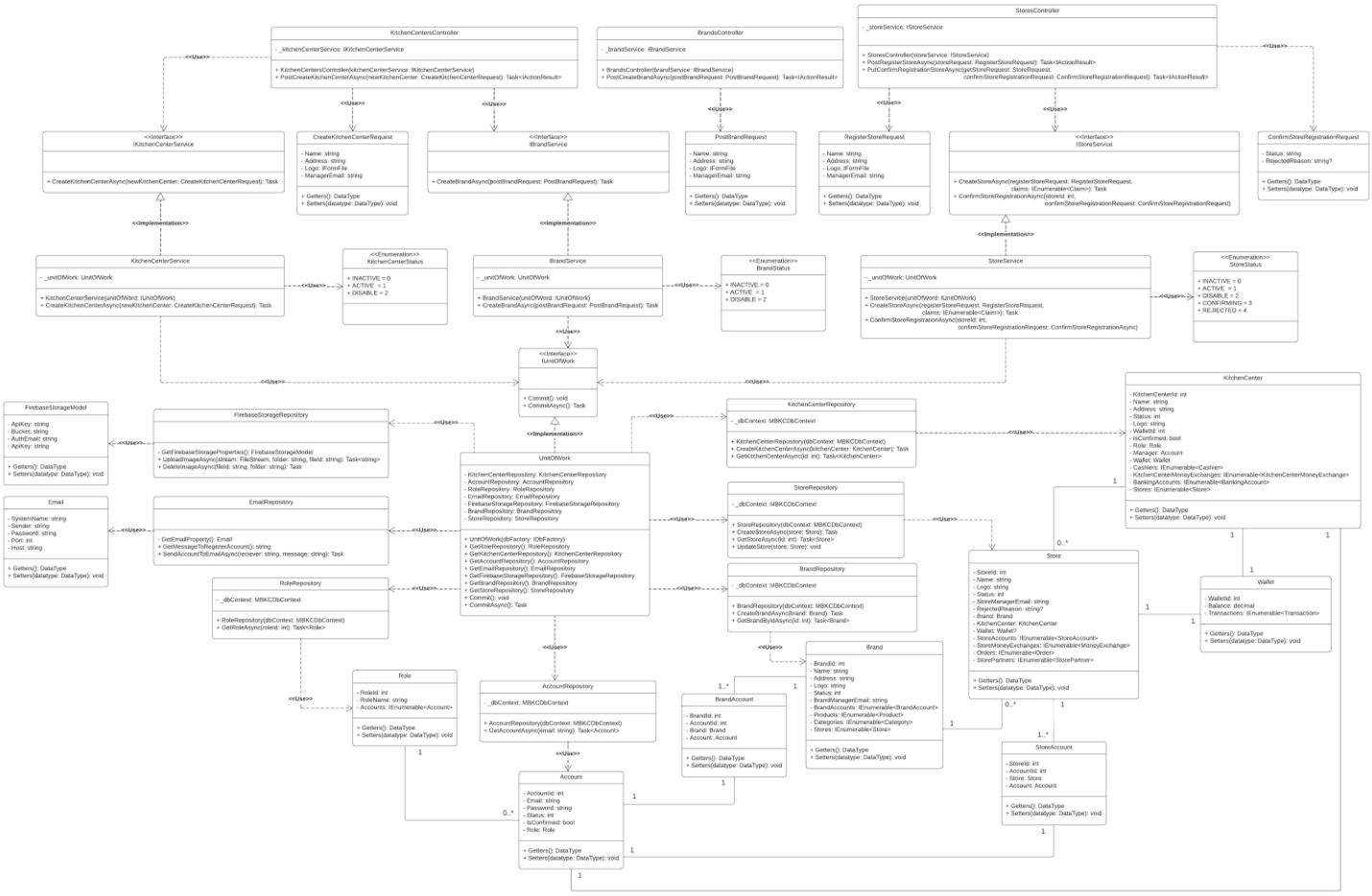


Figure 28: Brand's Participation Class Diagram

3.1.1.1. Class Specification

3.1.1.1.1. KitchenCentersController class

No	Attribute	Data Type	Visibility	Description
1	_kitchenCenterService	IKitchenCenterService	Private	This is a field that is used to store an instance of KitchenCenterService class.

Table 86: KitchenCentersController class - Attributes

No	Method	Data Type	Visibility	Description
1	KitchenCentersController		Public	This is a constructor of KitchenCentersController class.
2	PostCreateKitchenCenterAsync	Task<IActionResult>	Public	This is an action method that is used to create a new kitchen center in the MBKC system.

Table 87: KitchenCentersController class - Methods

3.1.1.1.2. BrandsController class

No	Attribute	Data Type	Visibility	Description
1	_brandService	IBrandService	Private	This is a field that is used to store an instance of BrandService class.

Table 88: BrandsController clas - Attributes

No	Method	Data Type	Visibility	Description
1	BrandsController		Public	This is a constructor of BrandsController class.
2	PostCreateBrandAsync	Task<IActionResult>	Public	This is an action method that is used to create a new brand in the MBKC system.

Table 89: BrandController class - Methods

3.1.1.1.3. StoresController class

No	Attribute	Data Type	Visibility	Description
1	_storeService	IStoreService	Private	This is a field that is used to store an instance of StoreService class.

Table 90: StoresController class - Attribute

No	Method	Data Type	Visibility	Description
1	StoresController		Public	This is a constructor of StoresController class.
2	PostRegisterStoreAsync	Task<IActionResult>	Public	This is an action method that is used to register a store for a brand.
3	PutConfirmRegistrationStoreAsync	Task<IActionResult>	Public	This is an action method that is used to confirm the brand's store registration.

Table 91: StoresController class - Methods

3.1.1.1.4. IKitchenCenterService interface

No	Method	Data Type	Visibility	Description
1	CreateKitchenCenterAsync	Task	Public	This is the abstract method that is used to create a new kitchen center.

Table 92: IKitchenCenterService interface

3.1.1.1.5. CreateKitchenCenterRequest class

No	Attribute	Data Type	Visibility	Description
1	Name	string	Private	This is a property that is used to store the name of the kitchen center.
2	Address	string	Private	This is a property that is used to store the address of the kitchen center.
3	Logo	IFormFile	Private	This is a property that is used to store the logo of the kitchen center.
4	ManagerEmail	string	Private	This is a property that is used to store the manager email of the kitchen center.

Table 93: CreateKitchenCenterRequest class

3.1.1.1.6. IBrandService interface

No	Method	Data Type	Visibility	Description
1	CreateBrandAsync	Task	Public	This is the abstract method that is used to create a new brand.

Table 94: IBrandService interface

3.1.1.1.7. PostBrandRequest class

No	Attribute	Data Type	Visibility	Description
1	Name	string	Private	This is a property that is used to store the name of the brand.
2	Address	string	Private	This is a property that is used to store the address of the brand.
3	Logo	IFormFile	Private	This is a property that is used to store the logo of the brand.
4	ManagerEmail	string	Private	This is a property that is used to store the manager email of the brand.

Table 95: PostBrandRequest class

3.1.1.1.8. RegisterStoreRequest class

No	Attribute	Data Type	Visibility	Description
1	Name	string	Private	This is a property that is used to store the name of the store.
2	Address	string	Private	This is a property that is used to store the address of the store.
3	Logo	IFormFile	Private	This is a property that is used to store the logo of the store.
4	ManagerEmail	string	Private	This is a property that is used to store the manager email of the store.

Table 96: RegisterStoreRequest class

3.1.1.1.9. IStoreService interface

No	Method	Data Type	Visibility	Description
1	CreateStoreAsync	Task	Public	This is the abstract method that is used to register a new store.
2	ConfirmStoreRegistrationAsync	Task	Public	This is the abstract method that is used to confirm a store registration.

Table 97: IStoreService interface

3.1.1.1.10. ConfirmStoreRegistrationRequest class

No	Attribute	Data Type	Visibility	Description
1	Status	string	Private	This is a property that is used to store the status of the store.
2	RejectedReason	string?	Private	This is a property that is used to store the rejected reason of the store.

Table 98: ConfirmStoreRegistrationRequest class

3.1.1.1.11. KitchenCenterService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 99: KitchenCenterService class - Attributes

No	Method	Data Type	Visibility	Description
1	KitchenCenterService		Public	This is a constructor of KitchenCenterService.
2	CreateKitchenCenterAsync	Task	Public	This is a method that is used to create a new kitchen center.

Table 100: KitchenCenterService class - Methods

3.1.1.1.12. KitchenCenterStatus enumeration

No	Enum	Visibility	Description
1	INACTIVE = 0	Public	
2	ACTIVE = 1	Public	
3	DISABLE = 2	Public	

Table 101: KitchenCenterStatus enumeration

3.1.1.1.13. KitchenCenterService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 102: KitchenCenterService class - Attributes

No	Method	Data Type	Visibility	Description
1	BrandService		Public	This is a constructor of BrandService class.
2	CreateBrandAsync	Task	Public	This is a method that is used to create a new brand.

Table 103: KitchenCenterService class - Methods

3.1.1.1.14. BrandStatus enumeration

No	Enum	Visibility	Description
1	INACTIVE = 0	Public	
2	ACTIVE = 1	Public	
3	DISABLE = 2	Public	

Table 104: BrandStatus enumeration

3.1.1.1.15. StoreService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 105: StoreService class - Attributes

No	Method	Data Type	Visibility	Description
1	StoreService		Public	This is a constructor of StoreService class.
2	CreateStoreAsync	Task	Public	This is a method that is used to register a new store.
3	ConfirmStoreRegistrationAsync	Task	Public	This is a method that is used to confirm a store registration.

Table 106: StoreService class - Methods

3.1.1.1.16. StoreStatus enumeration

No	Enum	Visibility	Description
1	INACTIVE = 0	Public	
2	ACTIVE = 1	Public	
3	DISABLE = 2	Public	
4	CONFIRMING = 3	Public	
5	REJECTED = 4	Public	

Table 107: StoreStatus enumeration

3.1.1.1.17. IUnitOfWork interface

No	Method	Data Type	Visibility	Description
1	Commit	void	Public	This is an abstract method that is used to execute non-query.
2	CommitAsync	Task	Public	This is an abstract method that is used to execute non-query with an asynchronous process.

Table 108: IUnitOfWork interface

3.1.1.1.18. FirebaseStorageModel class

No	Attribute	Data Type	Visibility	Description
1	ApiKey	string	Private	This is a property that is used to store the API key of the Firebase Storage.

2	Bucket	string	Private	This is a property that is used to store the bucket of the Firebase Storage.
3	AuthEmail	string	Private	This is a property that is used to store the authentication email of Firebase Storage.
4	AuthPassword	string	Private	This is a property that is used to store the authentication password of Firebase Storage.

Table 109: FirebaseStorageModel class

3.1.1.1.19. FirebaseStorageRepository class

No	Method	Data Type	Visibility	Description
1	GetFirebaseStorageProperties	FirebaseStorageModel	Private	This is a method that is used to get basic information to be able to connect to the Firebase Storage.
2	UploadImageAsync	Task<string>	Public	This is a method that is used to push an image to Firebase storage.
3	DeleteImageAsync	Task	Public	This is a method used to delete an image in Firebase storage.

Table 110: FirebaseStorageRepository class

3.1.1.1.20. UnitOfWork class

No	Attribute	Data Type	Visibility	Description
1	KitchenCenterRepository	KitchenCenterRepository	Private	This is a property that is used to store an instance of KitchenCenterRepository.
2	AccountRepository	AccountRepository	Private	This is a property that is used to store an instance of AccountRepository.
3	RoleRepository	RoleRepository	Private	This is a property that is used to store an instance of RoleRepository.
4	EmailRepository	EmailRepository	Private	This is a property that is used to store an instance of EmailRepository.
5	FirebaseStorageRepository	FirebaseStorageRepository	Private	This is a property that is used to store an instance of FirebaseStorageRepository.

6	BrandRepository	BrandRepository	Private	This is a property that is used to store an instance of BrandRepository.
7	StoreRepository	StoreRepository	Private	This is a property that is used to store an instance of StoreRepository.

Table 111: UnitOfWork class

No	Method	Data Type	Visibility	Description
1	UnitOfWork		Public	This is a constructor of UnitOfWork class.
2	GetRoleRepository	RoleRepository	Public	This is a method that is used to get an instance of RoleRepository class.
3	GetKitchenCenterRepository	KitchenCenterRepository	Public	This is a method that is used to get an instance of KitchenCenterRepository class.
4	GetAccountRepository	AccountRepository	Public	This is a method that is used to get an instance of AccountRepository class.
5	GetEmailRepository	EmailRepository	Public	This is a method that is used get an instance of EmailRepository class.
6	GetFirebaseStorageRepository	FirebaseStorageRepository	Public	This is a method that is used to get an instance of FirebaseStorageRepository class.
7	GetBrandRepository	BrandRepository	Public	This is a method that is used to get an instance of BrandRepository class.
8	GetStoreRepository	StoreRepository	Public	This is a method that is used to get an instance of StoreRepository class.
9	Commit	void	Public	This is a method that is used to execute non-query.
10	CommitAsync	Task	Public	This is a method that is used to execute non-query with an asynchronous process.

Table 112: UnitOfWork class

3.1.1.1.21. KitchenCenterRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 113: KitchenCenterRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	KitchenCenterRepository		Public	This is a constructor of KitchenCenterRepository class.
2	CreateKitchenCenterAsync	Task	Public	This is a method that is used to create a new kitchen center.
3	GetKitchenCenterAsync	Task<KitchenCenter>	Public	This is a method that is used to get a kitchen center.

Table 114: KitchenCenterRepository class - Methods

3.1.1.1.22. StoreRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 115: StoreRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	StoreRepository		Public	This is a constructor of StoreRepository class.
2	CreateStoreAsync	Task	Public	This is a method that is used to create a new store.
3	GetStoreAsync	Task<Store>	Public	This is a method that is used to get a store.
4	UpdateStore	void	Public	This is a method that is used to update information of a store.

Table 116: StoreRepository class - Methods

3.1.1.1.23. BrandRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 117: BrandRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	BrandRepository		Public	This is a constructor of BrandRepository class.
2	CreateBrandAsync	Task	Public	This is a method that is used to create a new brand.
3	GetBrandByIdAsync	Task<Brand>	Public	This is a method that is used to get a brand by Id.

Table 118: BrandRepository class - Methods

3.1.1.1.24. Email class

No	Attribute	Data Type	Visibility	Description
1	SystemName	string	Private	This is a property that is used to store the system name.
2	Sender	string	Private	This is a property that is used to store the sender email.
3	Password	string	Private	This is a property that is used to store the password email of sender.
4	Port	int	Private	This is a property that is used to store the port of the SMTP.
5	Host	string	Private	This is a property that is used to store the host of the SMTP.

Table 119: Email class

3.1.1.1.25. EmailRepository class

No	Method	Data Type	Visibility	Description
1	GetEmailProperty	Email	Private	This is a method that is used to get basic information to be able to send email.
2	GetMessageToRegisterAccount	string	Public	This is a method that is used to get the message format.
3	SendAccountToEmailAsync	Task	Public	This is a method that is used to send account information and a password to the registered email.

Table 120: EmailRepository class

3.1.1.1.26. RoleRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 121: StoreRepository class- Attributes

No	Method	Data Type	Visibility	Description
1	RoleRepository		Public	This is a constructor of RoleRepository class.
2	GetRoleAsync	Task<Role>	Public	This is a method that is used to get a role.

Table 122: StoreRepository class -Methods

3.1.1.1.27. Role class

No	Attribute	Data Type	Visibility	Description
1	RoleId	int	Private	This is a property that is used to store the role id of the role.
2	RoleName	string	Private	This is a property that is used to store the role name of the role.

3	Accounts	IEnumerable<Account>	Private	This is a property that is used to store the list accounts belong to the role.

Table 123: Role class

3.1.1.1.28. AccountRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 124: AccountRepository class- Attributes

No	Method	Data Type	Visibility	Description
1	AccountRepository		Public	This is a constructor of AccountRepository class.
2	GetAccountAsync	Task<Account>	Public	This is a method that is used to get a account.

Table 125: AccountRepository class – Methods

3.1.1.1.29. Account class

No	Attribute	Data Type	Visibility	Description
1	AccountId	int	Private	This is a property that is used to store the account id of the account.
2	Email	string	Private	This is a property that is used to store the email of the account.
3	Password	string	Private	This is a property that is used to store the password of the account
4	Status	int	Private	This is a property that is used to store the status of the account
5	IsConfirmed	bool	Private	This is a property that is used to store the is comfirm of the account
6	Role	Role	Private	This is a property that is used to store the role of the account

Table 126: Account class

3.1.1.1.30. BrandAccount class

No	Attribute	Data Type	Visibility	Description
1	BrandId	int	Private	This is a property that is used to store the brand id of the brand account.
2	AccountId	string	Private	This is a property that is used to store the account id of the brand account.
3	Brand	Brand	Private	This is a property that is used to store the brand of the brand account.
4	Account	Account	Private	This is a property that is used to store the account of the brand account

Table 127: BrandAccount class

3.1.1.1.31. Brand class

No	Attribute	Data Type	Visibility	Description
1	BrandId	int	Private	This is a property that is used to store the brand id of the brand.
2	Name	string	Private	This is a property that is used to store the name of the brand.
3	Address	string	Private	This is a property that is used to store the address of the brand.
4	Logo	string	Private	This is a property that is used to store the logo of the brand.
5	Status	int	Private	This is a property that is used to store the status of the brand.
6	BrandManagerEmail	string	Private	This is a property that is used to store the brand manager email of the brand.
7	BrandAccounts	IEnumerable<BrandAccount>	Private	This is a property that is used to store the list brand accounts of the brand.
8	Products	IEnumerable<Product>	Private	This is a property that is used to store the list products of the brand.
9	Categories	IEnumerable<Category>	Private	This is a property that is used to store the list categories of the brand.

10	Stores	IEnumerable<Store>	Private	This is a property that is used to store the list stores of the brand.

Table 128: Brand class

3.1.1.1.32. Store class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store.
2	Name	string	Private	This is a property that is used to store the name of the store.
3	Logo	string	Private	This is a property that is used to store the logo of the store.
4	Status	int	Private	This is a property that is used to store the status of the store.
5	StoreManagerEmail	string	Private	This is a property that is used to store the store manager email of the store.
6	RejectedReason	string?	Private	This is a property that is used to store the rejection reason of the store.
7	Brand	Brand	Private	This is a property that is used to store the brand of the store.
8	KitchenCenter	KitchenCenter	Private	This is a property that is used to store the kitchen center of the store.
9	Wallet	Wallet?	Private	This is a property that is used to store the wallet of the store.
10	StoreAccounts	IEnumerable<StoreAccount>	Private	This is a property that is used to store the list store accounts of the store.
11	StoreMoneyExchanges	IEnumerable<StoreAccount>	Private	This is a property that is used to store the list store money exchanges of the store.
12	Orders	IEnumerabe<Order>	Private	This is a property that is used to store the list orders of the store.
13	StorePartners	IEnumerable<StorePartner>	Private	This is a property that is used to store the list store partners of the store.

Table 129: Store class

3.1.1.1.33. KitchenCenter class

No	Attribute	Data Type	Visibility	Description
1	KitchenCenterId	int	Private	This is a property that is used to store the kitchen center id of the kitchen center.
2	Name	string	Private	This is a property that is used to store the name of the kitchen center.
3	Address	string	Private	This is a property that is used to store the address of the kitchen center.
4	Status	int	Private	This is a property that is used to store the status of the kitchen center.
5	Logo	string	Private	This is a property that is used to store the logo of the kitchen center.
6	WalletId	string	Private	This is a property that is used to store the wallet id of the kitchen center.
7	IsConfirmed	bool	Private	This is a property that is used to store the is confirm of the kitchen center.
8	Role	Role	Private	This is a property that is used to store the role of the kitchen center.
9	Manager	Account	Private	This is a property that is used to store the manager of the kitchen center.
10	Wallet	Wallet	Private	This is a property that is used to store the wallet of the kitchen center.
11	Cashiers	IEnumerable<Cashier>	Private	This is a property that is used to store the list cashiers of the kitchen center.
12	KitchenCenterMoney Exchanges	IEnumerabe<KitchenCenterMoney Exchanges>	Private	This is a property that is used to store the list money exchanges of the kitchen center.
13	BankingAccounts	IEnumerable<BankingAccount>	Private	This is a property that is a property that is used to store the list banking accounts of the kitchen center.

14	Stores	IEnumerable<Store>	Private	This is a property that is used to store the list stores of the kitchen center.

Table 130: KitchenCenter class

3.1.1.1.34. StoreAccount class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store account.
2	AccountId	int	Private	This is a property that is used to store the account id of the store account.
3	Store	Store	Private	This is a property that is used to store the store of the store account.
4	Account	Account	Private	This is a property that is used to store the account of the store account.

Table 131: StoreAccount class

3.1.1.1.35. Wallet class

No	Attribute	Data Type	Visibility	Description
1	WalletId	int	Private	This is a property that is used to store the wallet id of the wallet.
2	Balance	decimal	Private	This is a property that is used to store the balance of the wallet.
3	Transactions	IEnumerable<Transaction>	Private	This is a property that is used to store the list transactions of the wallet.

Table 132: Wallet class

3.1.2. Activity Diagram

3.1.2.1. Create Kitchen Center

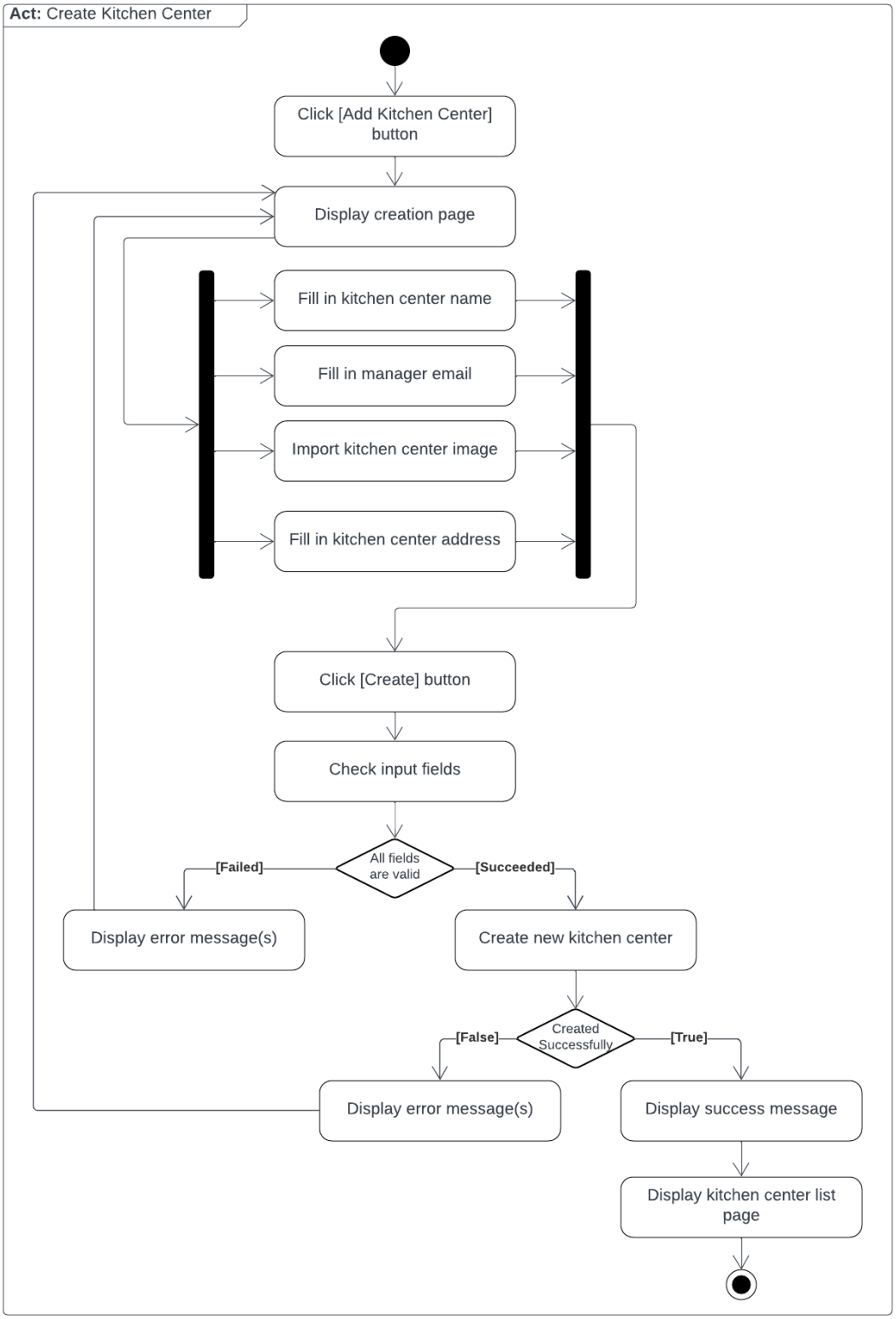


Figure 29: Create Kitchen Center

3.1.2.2. Create Brand

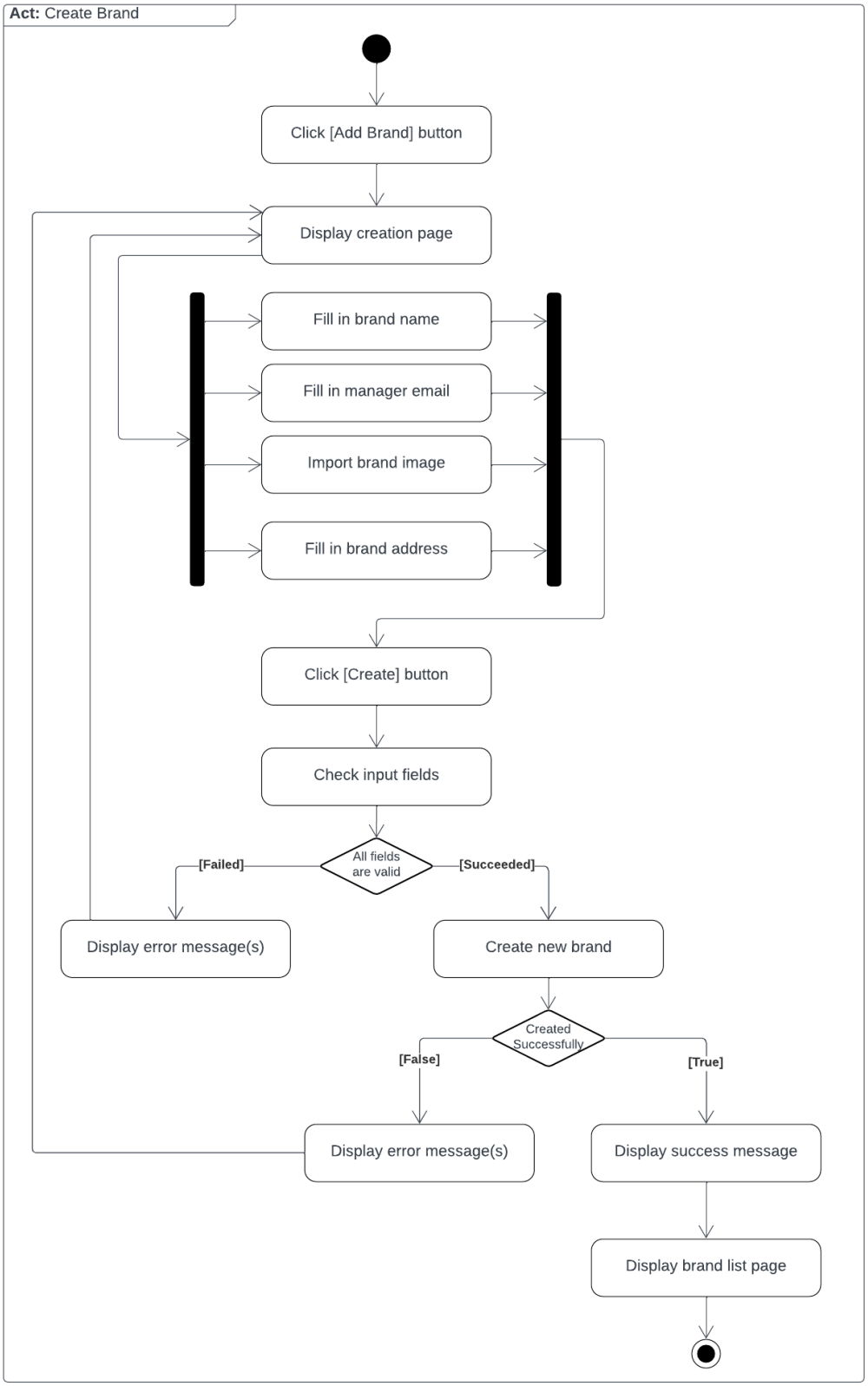


Figure 30: Create Brand

3.1.2.3. Register Store

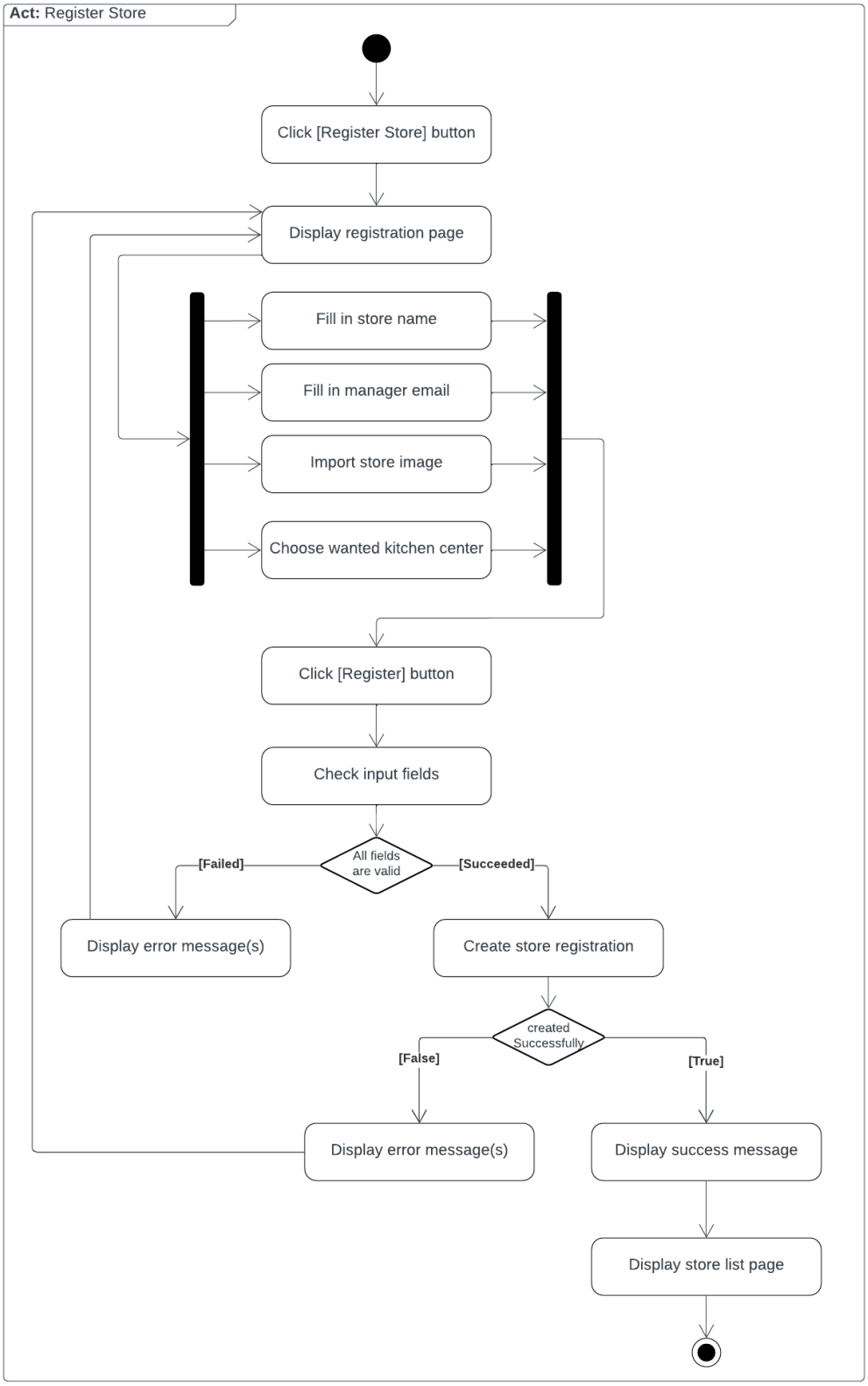


Figure 31: Register Store

3.1.2.4. Confirm Store Registration

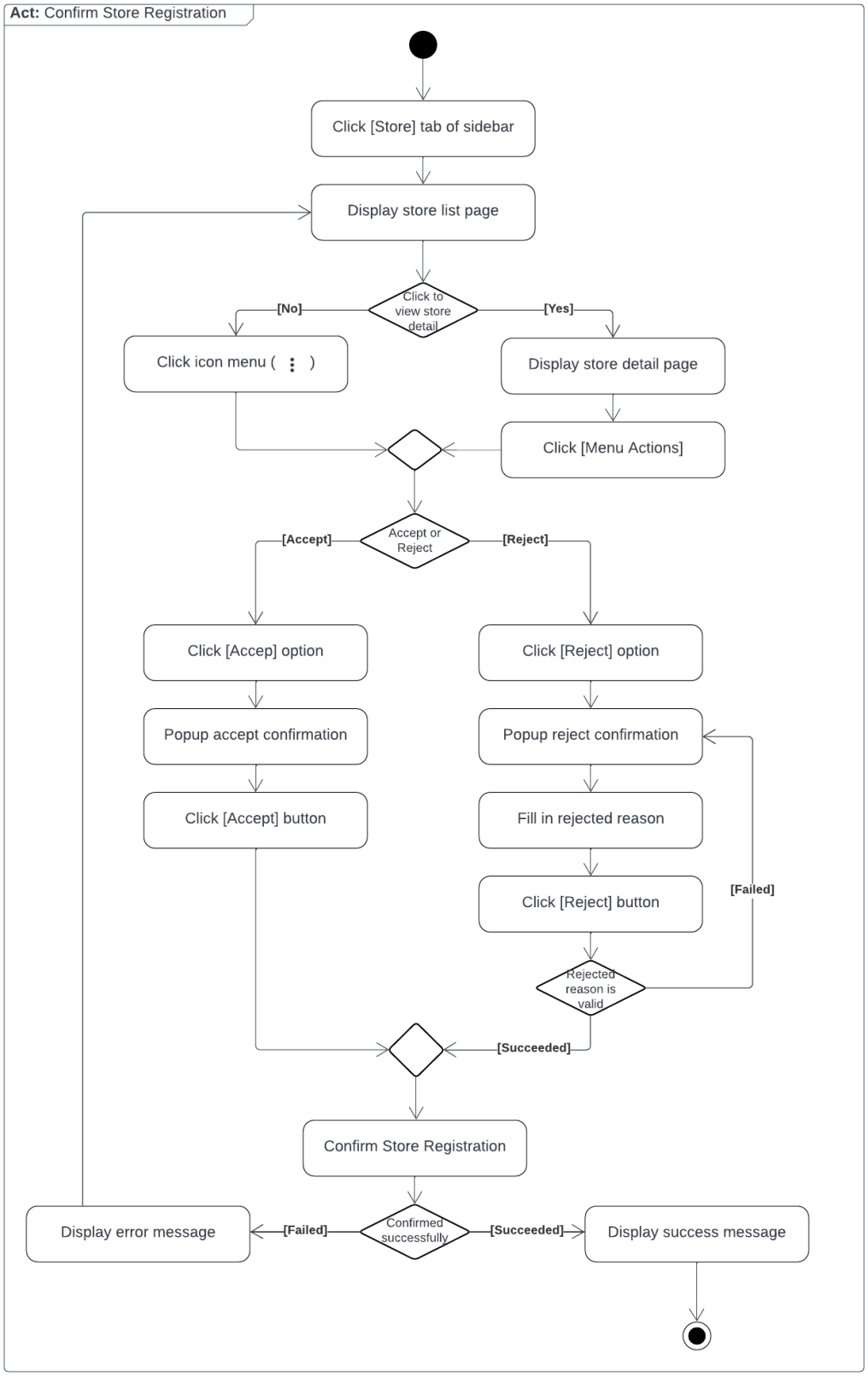


Figure 32: Confirm Store Registration

3.1.3. Sequence Diagram

3.1.3.1. Create Kitchen Center

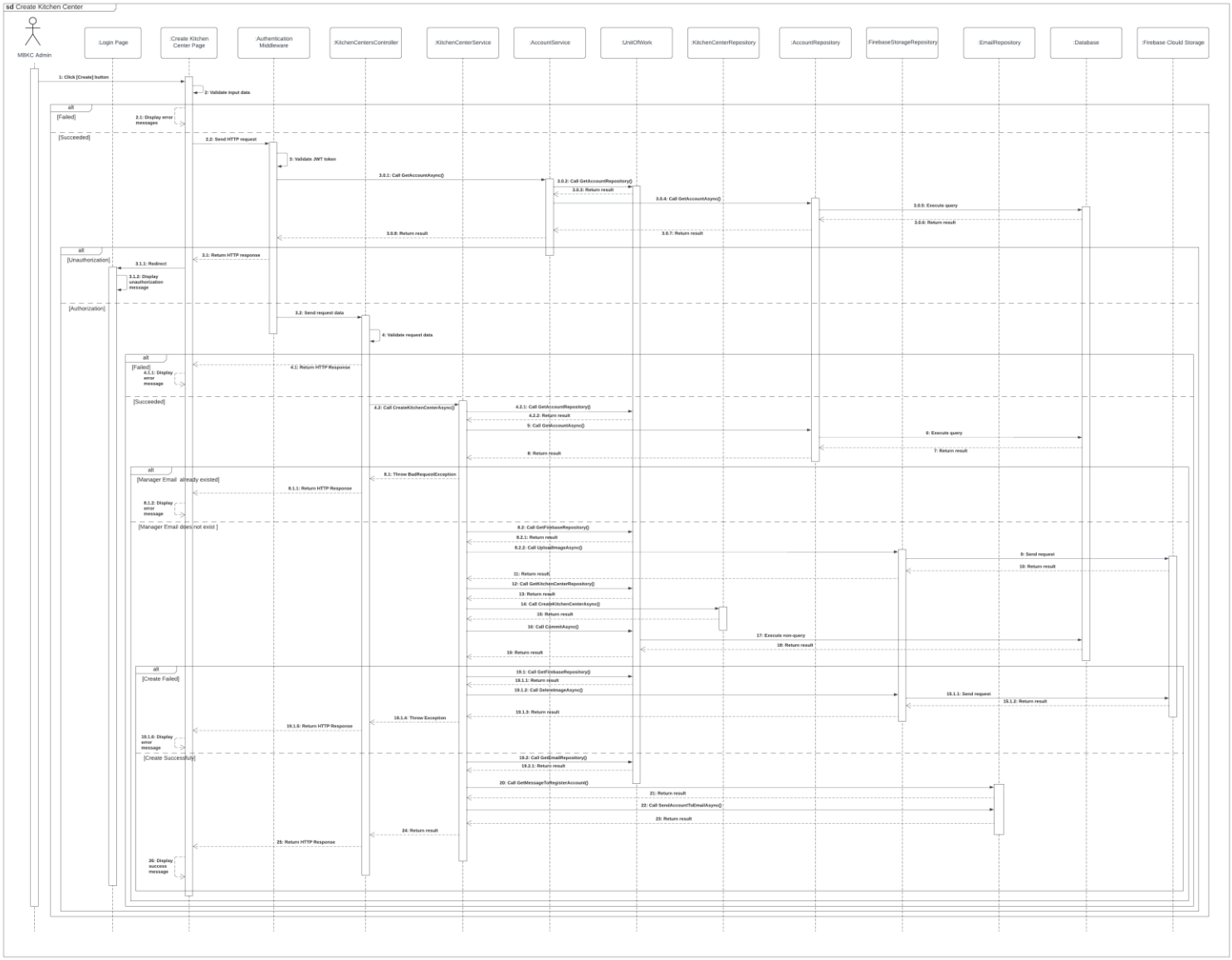


Figure 33: Create Kitchen Center

3.1.3.2. Create Brand

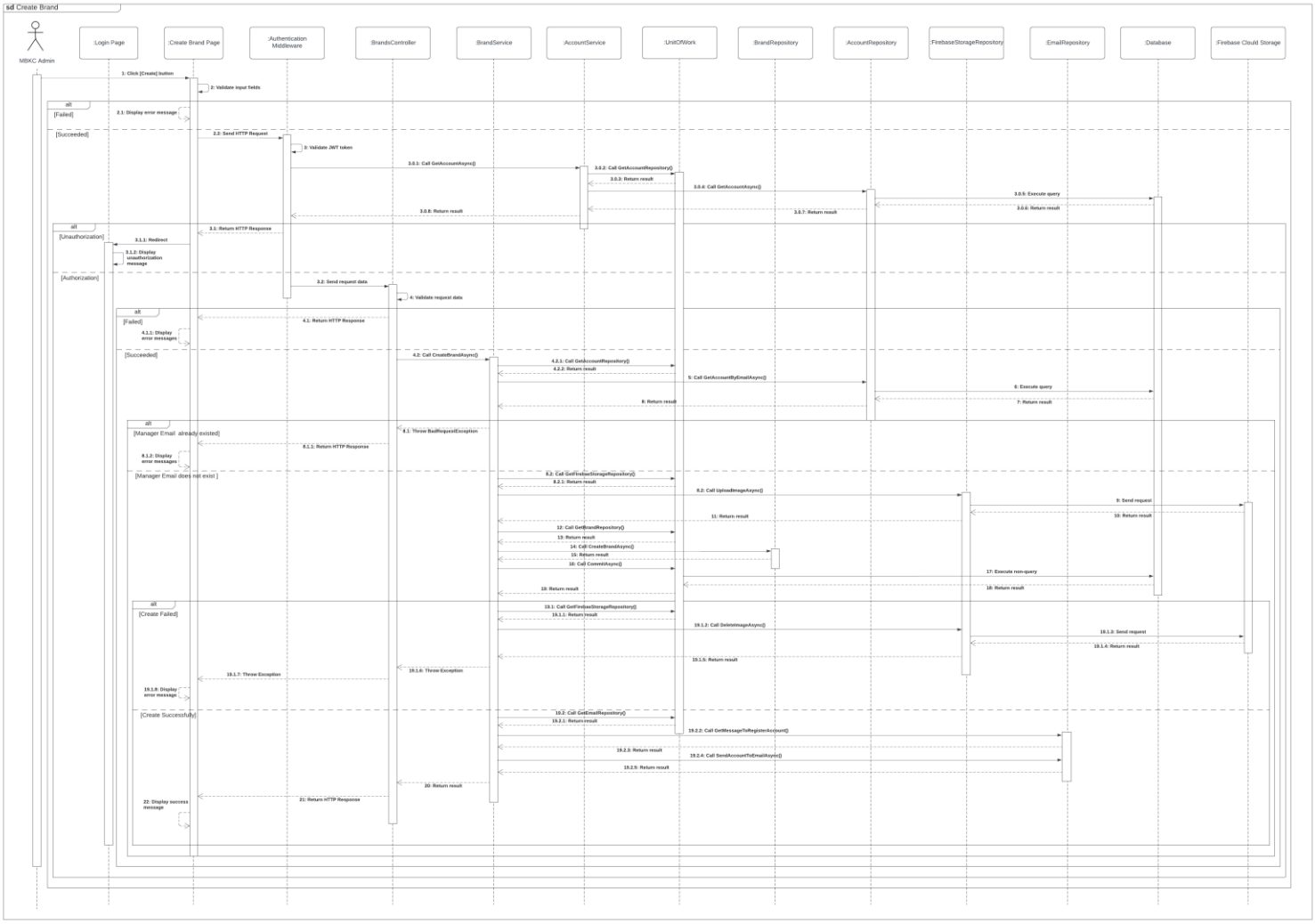


Figure 34: Create Brand

3.1.3.3. Register Store

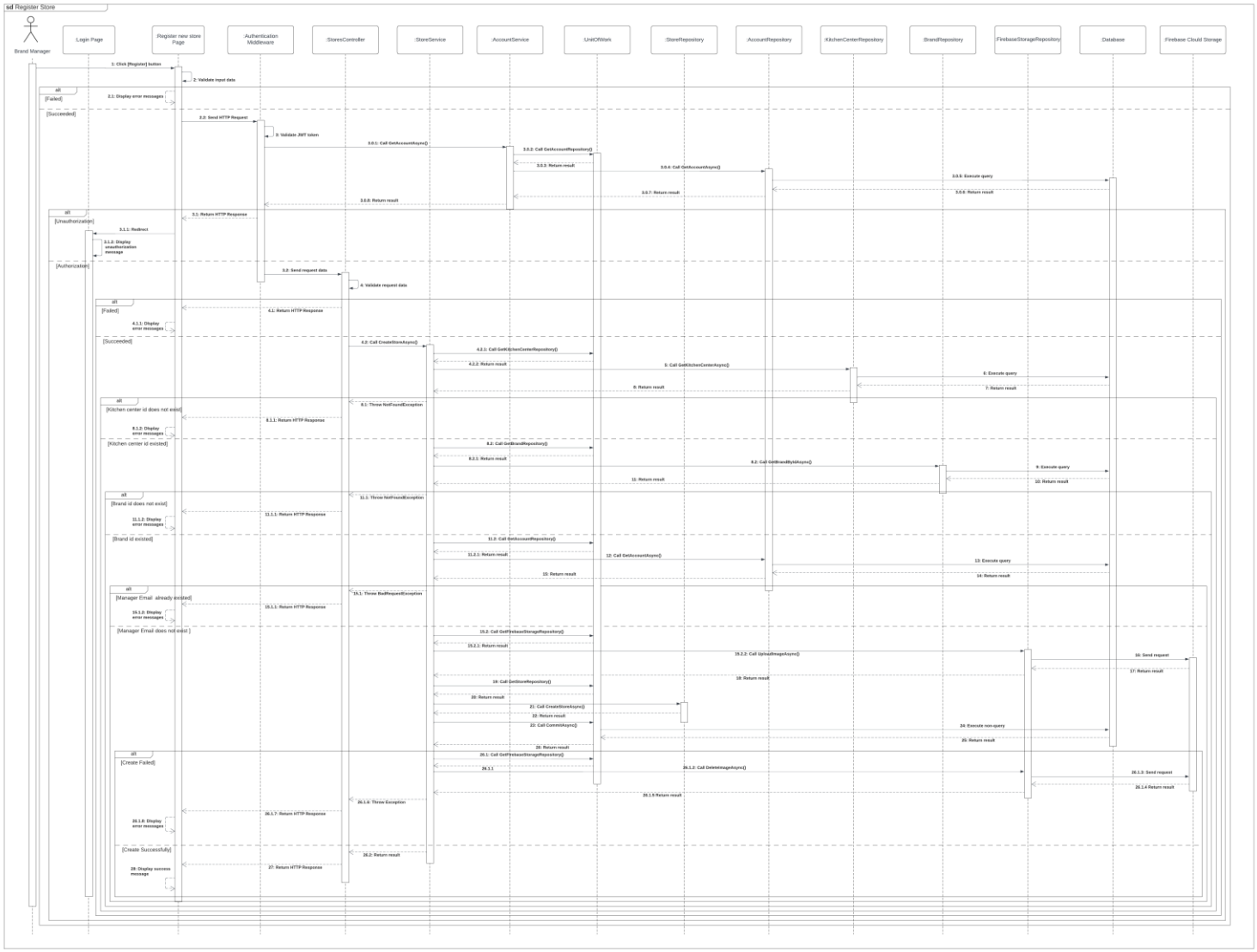


Figure 35: Register Store

3.1.3.4. Confirm Store Registration

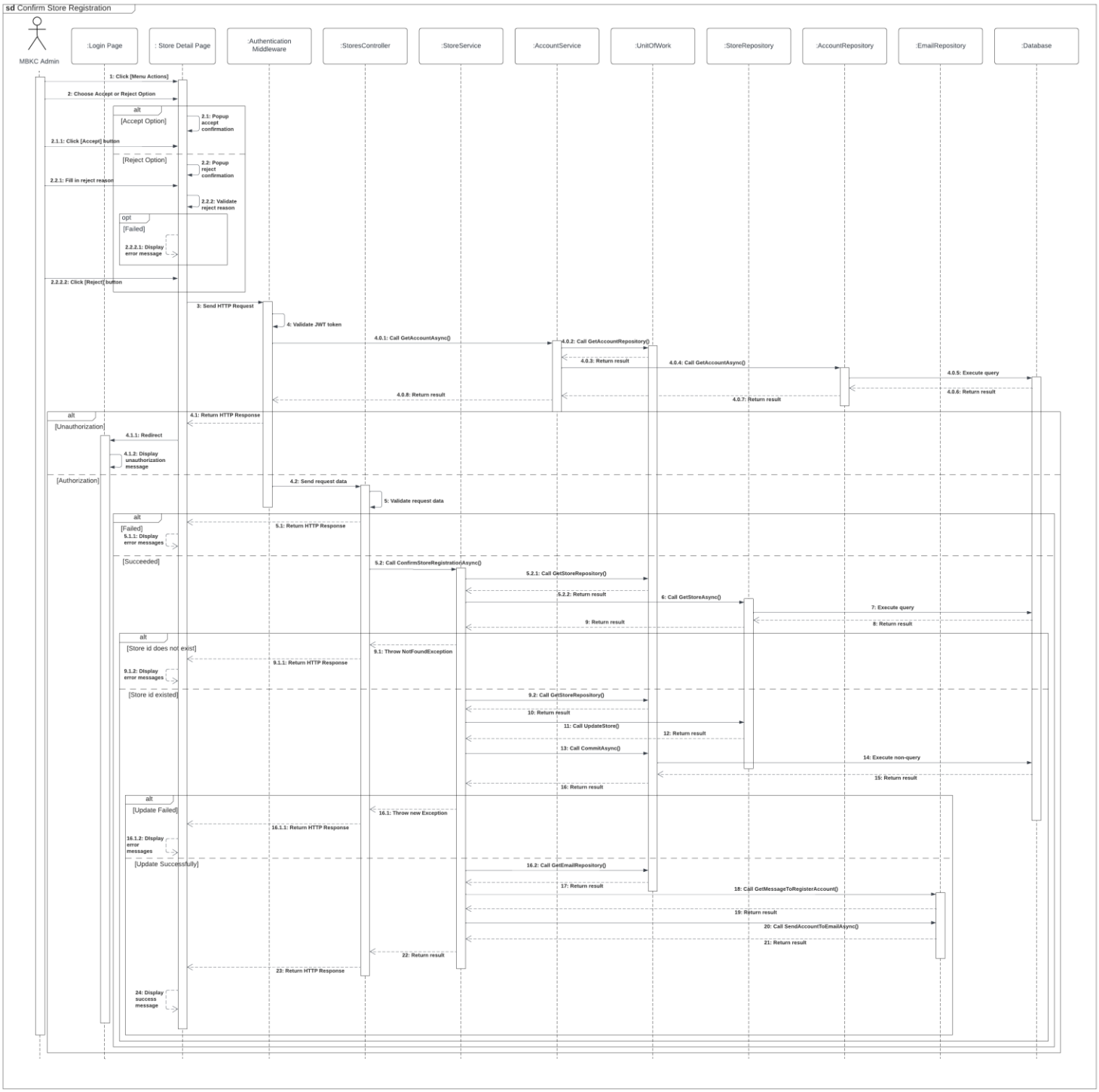


Figure 36: Confirm Store Registration

3.1.4. State Machine Diagram

3.1.4.1. Store State Machine Diagram

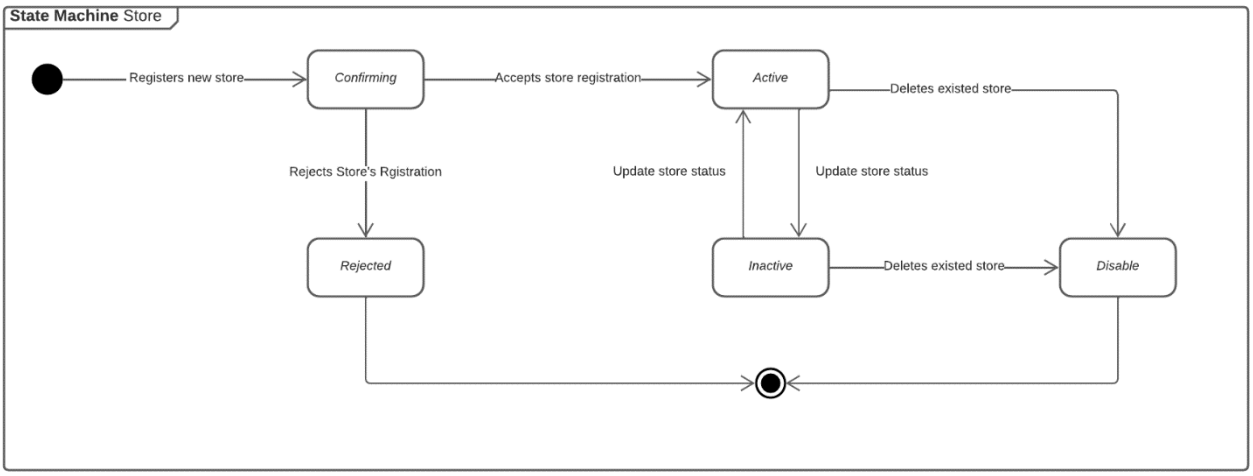


Figure 37: Store State Machine Diagram

3.2. Product Creation and Mapping

3.2.1. Class Diagram

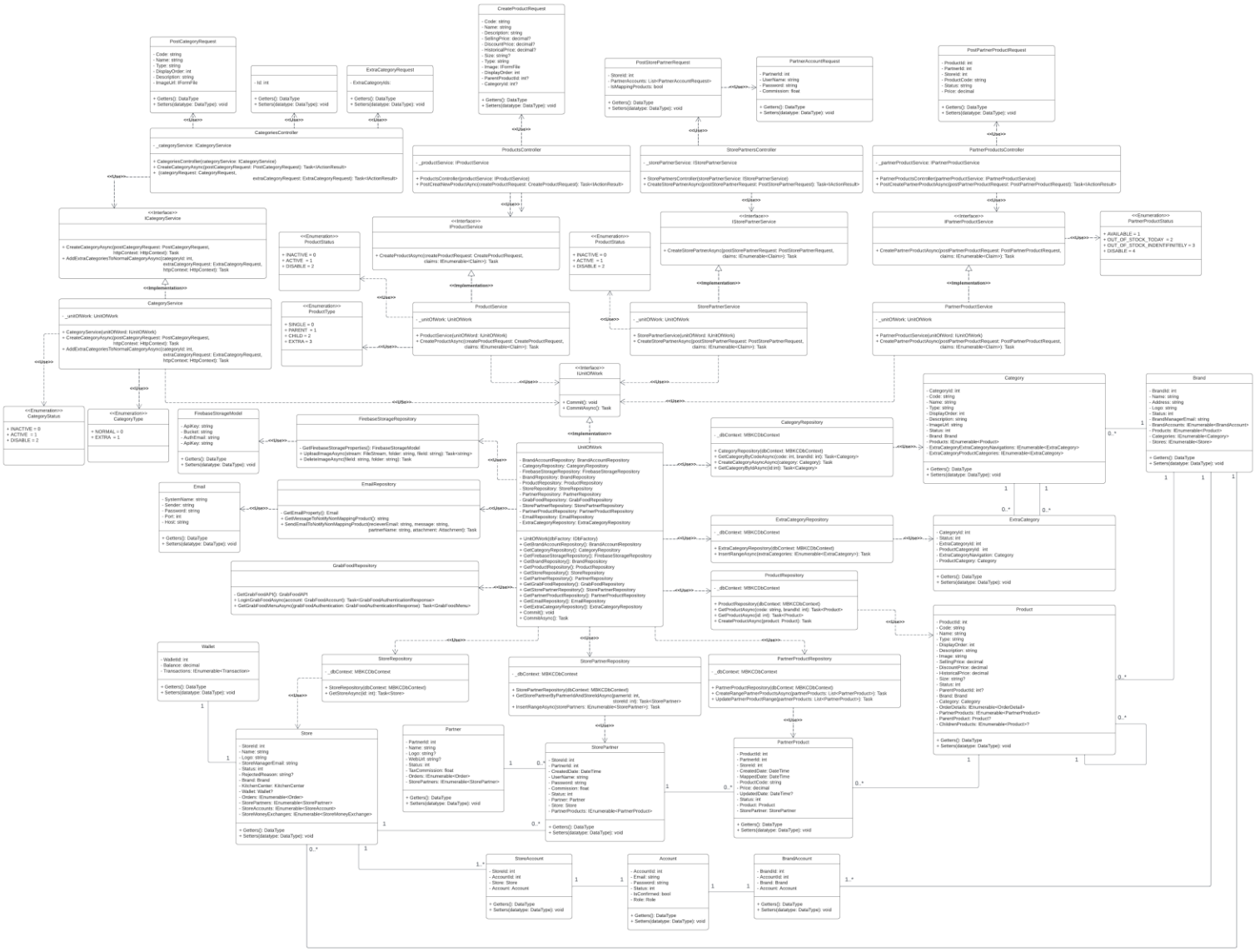


Figure 38: Product Creation and Mapping Class Diagram

3.2.1.1. Class Specifications

3.2.1.1.1. PostCategoryRequest class

No	Attribute	Data Type	Visibility	Description
1	Code	string	Private	This is a property that is used to store the code of the category.
2	Name	string	Private	This is a property that is used to store the name of the category.
3	Type	string	Private	This is a property that is used to store the type of the category.
4	DisplayOrder	int	Private	This is a property that is used to store the display order of the category.
5	Description	string	Private	This is a property that is used to store the description of the category.
6	ImageUrl	IFormFile	Private	This is a property that is used to store the image file of the category.

Table 133: PostCategoryRequest class

3.2.1.1.2. CategoryRequest class

No	Attribute	Data Type	Visibility	Description
1	Id	int	Private	This is a property that is used to store the id of the category.

Table 134: CategoryRequest class

3.2.1.1.3. ExtraCategoryRequest class

No	Attribute	Data Type	Visibility	Description
1	ExtraCategoryIds	List<int>	Private	This is a property that is used to store the list ids of the category.

Table 135: ExtraCategoryRequest class

3.2.1.1.4. CreateProductRequest class

No	Attribute	Data Type	Visibility	Description
1	Code	string	Private	This is a property that is used to store the code of the product.
2	Name	string	Private	This is a property that is used to store the name of the product.
3	Description	string	Private	This is a property that is used to store the description of the product.
4	SellingPrice	decimal?	Private	This is a property that is used to store the selling price of the product.

5	DiscountPrice	decimal?	Private	This is a property that is used to store the discount price of the product.
6	HistoricalPrice	decimal?	Private	This is a property that is used to store the history price of the product.
7	Size	string?	Private	This is a property that is used to store the size of the product.
8	Type	string	Private	This is a property that is used to store the type of product.
9	Image	IFormFile	Private	This is a property that is used to store the image of the product.
10	DisplayOrder	int	Private	This is a property that is used to store the display order of the product.
11	ParentProductId	int?	Private	This is a property that is used to store the partner product id of the product.
12	CategoryId	int?	Private	This is a property that is used to store the category of the product.

Table 136: CreateProductRequest class

3.2.1.1.5. PostStorePartnerRequest class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store partner.
2	PartnerAccounts	List<PartnerAccountRequest>	Private	This is a property that is used to store the list partner accouts of the store partner.
3	IsMappingProducts	bool	Private	This is a property that is used to store the is mapping product of the store partner.

Table 137: PostStorePartnerRequest class

3.2.1.1.6. PartnerAccountRequest class

No	Attribute	Data Type	Visibility	Description
1	PartnerId	int	Private	This is a property that is used to store the partner id of the partner accout.
2	UserName	string	Private	This is a property that is used to store the username of the partner accout.
3	Password	string	Private	This is a property that is used to store the password of the partner accout.

4	Commission	bool	Private	This is a property that is used to store the commission of the partner accout.

Table 138: PartnerAccountRequest class

3.2.1.1.7. PostPartnerProductRequest class

No	Attribute	Data Type	Visibility	Description
1	ProductId	int	Private	This is a property that is used to store the product id of the partner product.
2	PartnerId	int	Private	This is a property that is used to store the partner id of the partner product.
3	StoreId	int	Private	This is a property that is used to store the store id of the partner product.
4	ProductCode	string	Private	This is a property that is used to store the product code of the partner product.
5	Status	string	Private	This is a property that is used to store the status of the partner product.
6	Price	decimal	Private	This is a property that is used to store the price of the partner product.

Table 139: PostPartnerProductRequest class

3.2.1.1.8. CategoriesController class

No	Attribute	Data Type	Visibility	Description
1	_categoryService	ICategoryService	Private	This is a field that is used to store an instance of CategoryService class.

Table 140: CategoriesController class - Attributes

No	Method	Data Type	Visibility	Description
1	CategoriesController		Public	This is a constructor of class.
2	CreateCategoryAsync	Task<IActionResult>	Public	This is an action method that is used to create a new category.
3	AddExtraCategoriesToNormalCategoryAsync	Task<IActionResult>	Public	This is an action method that is used to add extra category to normal category.

Table 141: CategoriesController class - Methods

3.2.1.1.9. ProductsController class

No	Attribute	Data Type	Visibility	Description
1	_productService	IProductService	Private	This is a field that is used to store an instance of ProductService class.

Table 142: ProductsController class - Attributes

No	Method	Data Type	Visibility	Description
1	ProductsController		Public	This is a constructor of ProductsController class.
2	PostCreatNewProductAync	Task<IActionResult>	Public	This is an acttion method that is used to create a new product.

Table 143: ProductsController class - Methods

3.2.1.1.10. StorePartnersController class

No	Attribute	Data Type	Visibility	Description
1	_storePartnerService	IStorePartnerService	Private	This is a field that is used to store an instance of StorePartnerService class.

Table 144: ProductsController class - Attributes

No	Method	Data Type	Visibility	Description
1	StorePartnersController		Public	This is a constructor of StorePartnersController class.
2	CreateStorePartnerAsync	Task<IActionResult>	Public	This is an action method that is used to create a new store partner.

Table 145: ProductsController class - Methods

3.2.1.1.11. PartnerProductsController class

No	Attribute	Data Type	Visibility	Description
1	_partnerProductService	IPartnerProductService	Private	This is a field that is used to store an instance of PartnerProductService class.

Table 146: PartnerProductsController class - Attributes

No	Method	Data Type	Visibility	Description
1	PartnerProductsController	PartnerProductsController	Public	This is a constructor of PartnerProductsController class.
2	PostCreatePartnerProductAsync	Task<IActionResult>	Public	This is an action method that is used to create a new partner product.

Table 147: PartnerProductsController class - Methods

3.2.1.1.12. ICategoryService interface

No	Method	Data Type	Visibility	Description
1	CreateCategoryAsync	Task	Public	This is an abstract method that is used to create a new category.
2	AddExtraCategoriesToNormalCategoryAsync	Task	Public	This is an abstract method that is used to add a list of extra categories to the normal category.

Table 148: ICategoryService interface

3.2.1.1.13. ProductStatus enumeration

No	Enum	Visibility	Description
1	INACTIVE = 0	Public	
2	ACTIVE = 1	Public	
3	DISABLE = 2	Public	

Table 149: ProductStatus enumeration

3.2.1.1.14. IProductService interface

No	Method	Data Type	Visibility	Description
1	CreateProductAsync	Task	Public	This is an abstract method that is used to create a new product.

Table 150: IProductService interface

3.2.1.1.15. IStorePartnerService interface

No	Method	Data Type	Visibility	Description
1	CreateStorePartnerAsync	Task	Public	This is an abstract method that is used to create a new store partner.

Table 151: IStorePartnerService interface

3.2.1.1.16. IPartnerProductService interface

No	Method	Data Type	Visibility	Description
1	CreatePartnerProductAsync	Task	Public	This is an abstract method that is used to create a new partner product.

Table 152: IPartnerProductService interface

3.2.1.17. PartnerProductStatus enumeration

No	Enum	Visibility	Description
1	AVAILABLE = 1	Public	
2	OUT_OF_STOCK_TODAY = 2	Public	
3	OUT_OF_STOCK_INDENTIFINITELY = 3	Public	
4	DISABLE = 4	Public	

Table 153: PartnerProductStatus enumeration

3.2.1.1.18. StoreService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 154: StoreService class - Attributes

No	Method	Data Type	Visibility	Description
1	CategoryService		Public	This is a constructor of CategoryService class.
2	CreateCategoryAsync	Task	Public	This is a method that is used to create new a category.
3	AddExtraCategoriesToNormalCategoryAsync	Task	Public	This is a method that is to add extra category to normal category.

Table 155: StoreService class - Methods

3.2.1.1.19. ProductType enumeration

No	Enum	Visibility	Description
1	SINGLE = 0	Public	
2	PARENT = 1	Public	
3	CHILD = 2	Public	
4	EXTRA = 3	Public	

Table 156: ProductType enumeration

3.2.1.1.20. ProductService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 157: ProductService class - Attributes

No	Method	Data Type	Visibility	Description
1	ProductService		Public	This is a constructor of ProductService class.
2	CreateProductAsync	Task	Public	This is a method that is used to create new a product.

Table 158: ProductService class - Methods

3.2.1.1.21. StorePartnerService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 159: StorePartnerService class - Attributes

No	Method	Data Type	Visibility	Description
1	StorePartnerService		Public	This is a constructor of StorePartnerService class.
2	CreateStorePartnerAsync	Task	Public	This is a method that is used to create new store partner.

Table 160: StorePartnerService class - Methods

3.2.1.1.22. PartnerProductService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 161: PartnerProductService class - Attributes

No	Method	Data Type	Visibility	Description
1	PartnerProductService		Public	This is a constructor of PartnerProductService class.
2	CreatePartnerProductAsync	Task	Public	This is a method that is used to create a new partner product.

Table 162: PartnerProductService class - Methods

3.2.1.1.23. IUnitOfWork interface

No	Method	Data Type	Visibility	Description
1	Commit	void	Public	This is an abstract method that is used to execute non-query.
2	CommitAsync	Task	Public	This is an abstract method that is used to execute non-query with an asynchronous process.

Table 163: IUnitOfWork interface

3.2.1.1.24. CategoryStatus enumeration

No	Enum	Visibility	Description
1	INACTIVE = 0	Public	
2	ACTIVE = 1	Public	
3	DISABLE = 2	Public	

Table 164: CategoryStatus enumeration

3.2.1.1.25. CategoryType enumeration

No	Enum	Visibility	Description
1	NORMAL = 0	Public	
2	EXTRA = 1	Public	

Table 165: CategoryType enumeration

3.2.1.1.26. FirebaseStorageModel class

No	Attribute	Data Type	Visibility	Description
1	ApiKey	string	Private	This is a property that is used to store the API key of the Firebase Storage.
2	Bucket	string	Private	This is a property that is used to store the bucket of the Firebase Storage.
3	AuthEmail	string	Private	This is a property that is used to store the authentication email of Firebase Storage.
4	AuthPassword	string	Private	This is a property that is used to store the authentication password of the Firebase Storage.

Table 166: FirebaseStorageModel class

3.2.1.1.27. FirebaseStorageRepository class

No	Method	Data Type	Visibility	Description
1	GetFirebaseStorageProperties	FirebaseStorageModel	Private	This is a method that is used to get basic information to be able to connect to the Firebase Storage.
2	UploadImageAsync	Task<string>	Public	This is a method that is used to push an image to the Firebase storage.
3	DeleteImageAsync	Task	Public	This is a method used to delete an image in Firebase storage.

Table 167: FirebaseStorageRepository class

3.2.1.1.28. Email class

No	Attribute	Data Type	Visibility	Description
1	SystemName	string	Private	This is a property that is used to store the system name.
2	Sender	string	Private	This is a property that is used to store the sender email.
3	Password	string	Private	This is a property that is used to store the password email of sender.
4	Port	int	Private	This is a property that is used to store the port of the SMTP.
5	Host	string	Private	This is a property that is used to store the host of the SMTP.

Table 168: Email class

3.2.1.1.29. EmailRepository class

No	Method	Data Type	Visibility	Description
1	GetEmailProperty	Email	Private	This is a method that is used to get basic information to be able to send email.
2	GetMessageToNotifyNonMappingProduct	string	Public	This is a method that is used to get the message non-mapping product format.
3	SendEmailToNotifyNonMappingProduct	Task	Public	This is a method that is used to send the non-mapping product to the email of the store.

Table 169: EmailRepository class

3.2.1.1.30. UnitOfWork class

No	Attribute	Data Type	Visibility	Description
1	BrandAccountRepository	BrandAccountRepository	Private	This is a property that is used to store an instance of BrandAccountRepository class.
2	CategoryRepository	CategoryRepository	Private	This is a property that is used to store an instance of CategoryRepository class.
3	FirebaseStorageRepository	FirebaseStorageRepository	Private	This is a property that is used to store an instance of FirebaseStorageRepository class.
4	BrandRepository	BrandRepository	Private	This is a property that is used to store an instance of BrandRepository class.
5	ProductRepository	ProductRepository	Private	This is a property that is used to store an instance of BrandAccountRepository class.
6	StoreRepository	StoreRepository	Private	This is a property that is used to store an instance of StoreRepository class.
7	PartnerRepository	PartnerRepository	Private	This is a property that is used to store an instance of PartnerRepository class.
8	GrabFoodRepository	GrabFoodRepository	Private	This is a property that is used to store an instance of GrabFoodRepository class.
9	StorePartnerRepository	StorePartnerRepository	Private	This is a property that is used to store an instance of StorePartnerRepository class.
10	PartnerProductRepository	PartnerProductRepository	Private	This is a property that is used to store an instance of PartnerProductRepository class.
11	EmailRepository	EmailRepository	Private	This is a property that is used to store an instance of EmailRepository class.
12	ExtraCategoryRepository	ExtraCategoryRepository	Private	This is a property that is used to store an instance of ExtraCategoryRepository class.

Table 170: UnitOfWork class

No	Method	Data Type	Visibility	Description
1	UnitOfWork		Public	This is a constructor of UnitOfWork class.
2	GetBrandAccountRepository	BrandAccountRepository	Public	This is a method that is used to get an instance of BrandAccountRepository

				class.
3	GetCategoryRepository	CategoryRepository	Public	This is a method that is used to get an instance of CategoryRepository class.
4	GetFirebaseStorageRepository	FirebaseStorageRepository	Public	This is a method that is used to get an instance of FirebaseStorageRepository class.
5	GetBrandRepository	BrandRepository	Public	This is a method that is used to get an instance of BrandRepository class.
6	GetProductRepository	ProductRepository	Public	This is a method that is used to get an instance of ProductRepository class.
7	GetStoreRepository	StoreRepository	Public	This is a method that is used to get an instance of StoreRepository class.
8	GetPartnerRepository	PartnerRepository	Public	This is a method that is used to get an instance of PartnerRepository class.
9	GetGrabFoodRepository	GrabFoodRepository	Public	This is a method that is used to get an instance of GrabFoodRepository class.
10	GetStorePartnerRepository	StorePartnerRepository	Public	This is a method that is used to get an instance of StorePartnerRepository class.
11	GetPartnerProductRepository	PartnerProductRepository	Public	This is a method that is used to get an instance of PartnerProductRepository class.
12	GetEmailRepository	EmailRepository	Public	This is a method that is used to get an instance of EmailRepository class.
13	GetExtraCategoryRepository	ExtraCategoryRepository	Public	This is a method that is used to get an instance of ExtraCategoryRepository class.
14	Commit	void	Public	This is a method that is used to execute non-query.
15	CommitAsync	Task	Public	This is a method that is used to execute non-query with an asynchronous process.

Table 171: UnitOfWork class

3.2.1.1.31. CategoryRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 172: CategoryRepository class

No	Method	Data Type	Visibility	Description
1	CategoryRepository		Public	This is a constructor of CategoryRepository.
2	GetCategoryByCodeAsync	Task<Category>	Public	This is a method that is used to get a category by code.
3	CreateCategoryAsyncAsync	Task	Public	This is a method that is used to create a new category.
4	GetCategoryByIdAsync	Task<Category>	Public	This is a method that is used to get category by Id.

Table 173: CategoryRepository class

3.2.1.1.32. Category class

No	Attribute	Data Type	Visibility	Description
1	CategoryId	int	Private	This is a property that is used to store the category id of the category.
2	Code	string	Private	This is a property that is used to store the code of the category.
3	Name	string	Private	This is a property that is used to store the name of the category.
4	Type	string	Private	This is a property that is used to store the type of the category.
5	DisplayOrder	int	Private	This is a property that is used to store the display order of the category.
6	Description	string	Private	This is a property that is used to store the description of the store.

7	ImageUrl	string	Private	This is a property that is used to store the image url of the category.
8	Status	int	Private	This is a property that is used to store the status of the category.
9	Brand	Brand	Private	This is a property that is used to store the brand of the category.
10	Products	IEnumerable<Product>	Private	This is a property that is used to store the list product of the category.
11	ExtraCategoryExtraCategoryNavigations	IEnumerable<ExtraCategory>	Private	This is a property that is used to store the list normal categories of the extra category.
12	ExtraCategoryProductCategories	IEnumerabe<ExtraCategory>	Private	This is a property that is used to store the list extra categories of the normal category.

Table 174: Category class

3.2.1.1.33. ExtraCategoryRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 175: ExtraCategoryRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	ExtraCategoryRepository		Public	This is a constructor of ExtraCategoryRepository class.
2	InsertRangeAsync	Task	Public	This is a method that is used to add a list of extra categories.

Table 176: ExtraCategoryRepository class - Methods

3.2.1.1.34. Brand class

No	Attribute	Data Type	Visibility	Description
1	BrandId	int	Private	This is a property that is used to store the brand id of the brand.
2	Name	string	Private	This is a property that is used to store the name of the brand.
3	Address	string	Private	This is a property that is used to store the address of the brand.
4	Logo	string	Private	This is a property that is used to store the logo of the brand.
5	Status	int	Private	This is a property that is used to store the status of the brand.
6	BrandManagerEmail	string	Private	This is a property that is used to store the brand manager email of the brand.
7	BrandAccounts	IEnumerable<BrandAccount>	Private	This is a property that is used to store the list brand accounts of the brand.
8	Products	IEnumerable<Product>	Private	This is a property that is used to store the list products of the brand.
9	Categories	IEnumerable<Category>	Private	This is a property that is used to store the list categories of the brand.
10	Stores	IEnumerable<Store>	Private	This is a property that is used to store the list stores of the brand.

Table 177: Brand class

3.2.1.1.35. ExtraCategory class

No	Attribute	Data Type	Visibility	Description
1	CategoryId	int	Private	This is a property that is used to store the category id of the extra category.
2	Status	int	Private	This is a property that is used to store the status of the extra category.
3	ExtraCategoryId	int	Private	This is a property that is used to store the extra category id of the extra category.
4	ProductCategoryId	int	Private	This is a property that is used to store the product category id of the extra category.
5	ExtraCategoryNavigation	Category	Private	This is a property that is used to store the extra navigation of the extra category.

6	ProductCategory	Category	Private	This is a property that is used to store the product category of the extra category.

Table 178: ExtraCategory class

3.2.1.1.36. GrabFoodRepository class

No	Method	Data Type	Visibility	Description
1	GetGrabFoodAPI	ExtraCategoryRepository	Private	This is a method that is used to get basic information to be able to connect to grabfood api.
2	LoginGrabFoodAsync	Task<GrabFoodMenu>	Public	This is a method that is used to get authentication information for a partner account.
3	GetGrabFoodMenuAsync	Task<GrabFoodMenu>	Public	This is a method that is used to get list menus of grabfood.

Table 179: GrabFoodRepository class

3.2.1.1.37. ProductRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 180: ProductRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	ProductRepository		Public	This is a constructor of ProductRepository class.
2	GetProductAsync	Task<Product>	Public	This is a method that is used to get product by code and brand id.
3	GetProductAsync	Task<Product>	Public	This is a method that is used to get product.
4	CreateProductAsync	Task	Public	This is a method that is used to create product.

Table 181: ProductRepository class - Methods

3.2.1.1.38. Product class

No	Attribute	Data Type	Visibility	Description
1	ProductId	int	Private	This is a property that is used to store the product id of the product.
2	Code	string	Private	This is a property that is used to store the code of the product.

3	Name	string	Private	This is a property that is used to store the name of the product.
4	Type	string	Private	This is a property that is used to store the type of the product.
5	DisplayOrder	int	Private	This is a property that is used to store the display order of the product.
6	Description	string	Private	This is a property that is used to store the description of the product.
7	Image	string	Private	This is a property that is used to store the image of the product.
8	SellingPrice	decimal	Private	This is a property that is used to store the selling price of the product.
9	DiscountPrice	decimal	Private	This is a property that is used to store the discount price of the product.
10	HistoricalPrice	decimal	Private	This is a property that is used to store the historical price of the product.
11	Size	string	Private	This is a property that is used to store the size of the product.
12	Status	int	Private	This is a property that is used to store the status of the product.
13	ParentProductId	int	Private	This is a property that is used to store the parent product id of the product.
14	Brand	Brand	Private	This is a property that is used to store the brand of the product.
15	Category	Category	Private	This is a property that is used to store the category of the product.
16	OrderDetails	Enumerable<OrderDetail>	Private	This is a property that is used to store the list order details of the product.
17	PartnerProducts	IEnumerable<PartnerProduct>	Private	This is a property that is used to store the list partner products of the product.
18	ParentProduct	Product	Private	This is a property that is used to store the parent product of the product.

19	ChildrenProducts	IEnumerable<PartnerProduct>	Private	This is a property that is used to store the list children products of the product.

Table 182: Product class

3.2.1.1.39. Wallet class

No	Attribute	Data Type	Visibility	Description
1	WalletId	int	Private	This is a property that is used to store the wallet id of the wallet.
2	Balance	decimal	Private	This is a property that is used to store the balance of the wallet.
3	Transactions	IEnumerable<Transaction>	Private	This is a property that is used to store the list transactions of the wallet.

Table 183: Wallet class

3.2.1.1.40. StoreRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 184: StoreRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	StoreRepository		Public	This is a constructor of StoreRepository class.
2	GetStoreAsync	Task<Store>	Public	This is a method that is used to get a store.

Table 185: StoreRepository class - Methods

3.2.1.1.41. StorePartnerRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 186: StorePartnerRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	StorePartnerRepository		Public	This is a constructor of StorePartnerRepository class.

2	GetStorePartnerByPartnerIdAndStoreIdAsync	Task<StorePartner>	Public	This is a method thay is used to get a store partner by partner id, store id.
3	InsertRangeAsync	Task	Public	This is a method that is used to add list store partners..

Table 187: StorePartnerRepository class – Methods

3.2.1.1.42. PartnerProductRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 188: PartnerProductRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	PartnerProductRepository		Public	This is a constructor of PartnerProductRepository class.
2	CreateRangePartnerProductsAsync	Task	Public	This is a method that is used to add list partner products..
3	UpdatePartnerProductRange	Task	Public	This is a method that is used to update list partner products.

Table 189: PartnerProductRepository class - Methods

3.2.1.1.43. Store class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store.
2	Name	string	Private	This is a property that is used to store the name of the store.
3	Logo	string	Private	This is a property that is used to store the logo of the store.
4	Status	int	Private	This is a property that is used to store the status of the store.
5	StoreManagerEmail	string	Private	This is a property that is used to store the store manager email of the store.
6	RejectedReason	string?	Private	This is a property that is used to store the rejection reason of the store.
7	Brand	Brand	Private	This is a property that is used to store the brand of the store.
8	KitchenCenter	KitchenCenter	Private	This is a property that is used to store the kitchen center of the store.
9	Wallet	Wallet?	Private	This is a property that is used to

				store the wallet of the store.
10	StoreAccounts	IEnumerable<StoreAccount>	Private	This is a property that is used to store the list store accounts of the store.
11	StoreMoneyExchanges	IEnumerable<StoreAccount>	Private	This is a property that is used to store the list store money exchanges of the store.
12	Orders	IEnumerabe<Order>	Private	This is a property that is used to store the list orders of the store.
13	StorePartners	IEnumerable<StorePartner>	Private	This is a property that is used to store the list store partners of the store.

Table 190: Store class

3.2.1.1.44. Partner class

No	Attribute	Data Type	Visibility	Description
1	PartnerId	int	Private	This is a property that is used to store the partner id of the partner.
2	Name	string	Private	This is a property that is used to store the name of the partner.
3	Logo	string?	Private	This is a property that is used to store the logo of the partner.
4	WebUrl	string?	Private	This is a property that is used to store the web URL of the partner.
5	Status	int	Private	This is a property that is used to store the status of the partner.
6	TaxCommission	float	Private	This is a property that is used to store the tax of commision.
7	Orders	IEnumerable<Order>	Private	This is a property that is used to store the orders of the partner.
8	StorePartners	IEnumerable<StorePartner>	Private	This is a property that is used to store the list store partners of the partner.

Table 191: Partner class

3.2.1.1.45. StorePartner class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store partner.

2	PartnerId	string	Private	This is a property that is used to store the partner id of the store partner.
3	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the store partner.
4	UserName	string	Private	This is a property that is used to store the username of the store partner.
5	Password	string	Private	This is a property that is used to store the password of the store partner.
6	Commission	float	Private	This is a property that is used to store the commission of the store partner.
7	Status	int	Private	This is a property that is used to store the status of the store partner.
8	Partner	Partner	Private	This is a property that is used to store the partner of the store partner.
9	Store	Store	Private	This is a property that is used to store the store of the store partner.
10	PartnerProducts	IEnumerable<PartnerProduct>	Private	This is a property that is used to store the list partner products of the store partner.

Table 192: StorePartner class

3.2.1.1.46. PartnerProduct class

No	Attribute	Data Type	Visibility	Description
1	ProductId	int	Private	This is a property that is used to store the product id of the partner product.
2	PartnerId	int	Private	This is a property that is used to store the partner id of the partner product.
3	StoreId	int	Private	This is a property that is used to store the store id of the partner product.
4	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the partner product.
5	MappedDate	DateTime	Private	This is a property that is used to store the mapped date of the partner product.
6	ProductCode	string	Private	This is a property that is used to store the product code of the partner product.

7	Price	decimal	Private	This is a property that is used to store the price of the partner product.
8	UpdatedDate	DateTime	Private	This is a property that is used to store the updated date of the partner product.
9	Status	int	Private	This is a property that is used to store the status of the partner product.
10	Product	Product	Private	This is a property that is used to store the product of the partner product.
11	StorePartner	StorePartner	Private	This is a property that is used to store the store partner of the partner product.

Table 193: PartnerProduct class

3.2.1.1.47. StoreAccount class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store account.
2	AccountId	int	Private	This is a property that is used to store the account id of the store account.
3	Store	Store	Private	This is a property that is used to store the store of the store account.
4	Account	Account	Private	This is a property that is used to store the account of the store account.

Table 194: StoreAccount class

3.2.1.1.48. Account class

No	Attribute	Data Type	Visibility	Description
1	AccountId	int	Private	This is a property that is used to store the account id of the account.
2	Email	string	Private	This is a property that is used to store the email of the account.
3	Password	string	Private	This is a property that is used to store the password of the account
4	Status	int	Private	This is a property that is used to store the status of the account
5	IsConfirmed	bool	Private	This is a property that is used to store the is comfirm of the account

6	Role	Role	Private	This is a property that is used to store the role of the account

Table 195: Account class

3.2.1.1.49. BrandAccount class

No	Attribute	Data Type	Visibility	Description
1	BrandId	int	Private	This is a property that is used to store the brand id of the brand account.
2	AccountId	string	Private	This is a property that is used to store the account id of the brand account.
3	Brand	Brand	Private	This is a property that is used to store the brand of the brand account.
4	Account	Account	Private	This is a property that is used to store the account of the brand account

Table 196: BrandAccount class

3.2.2. Activity Diagram

3.2.2.1. Create Normal Category

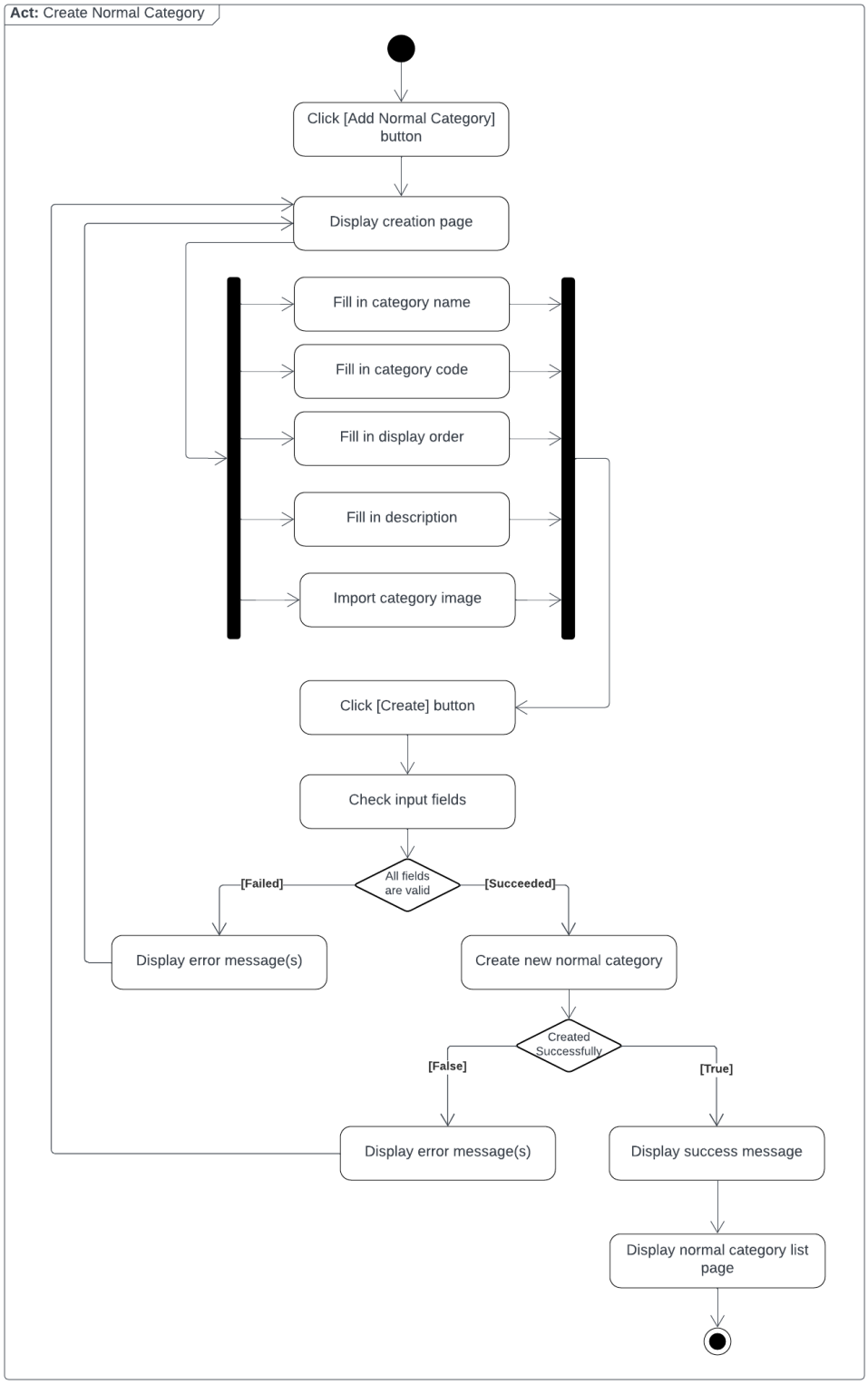


Figure 39: Create Normal Category

3.2.2.2. Create Extra Category

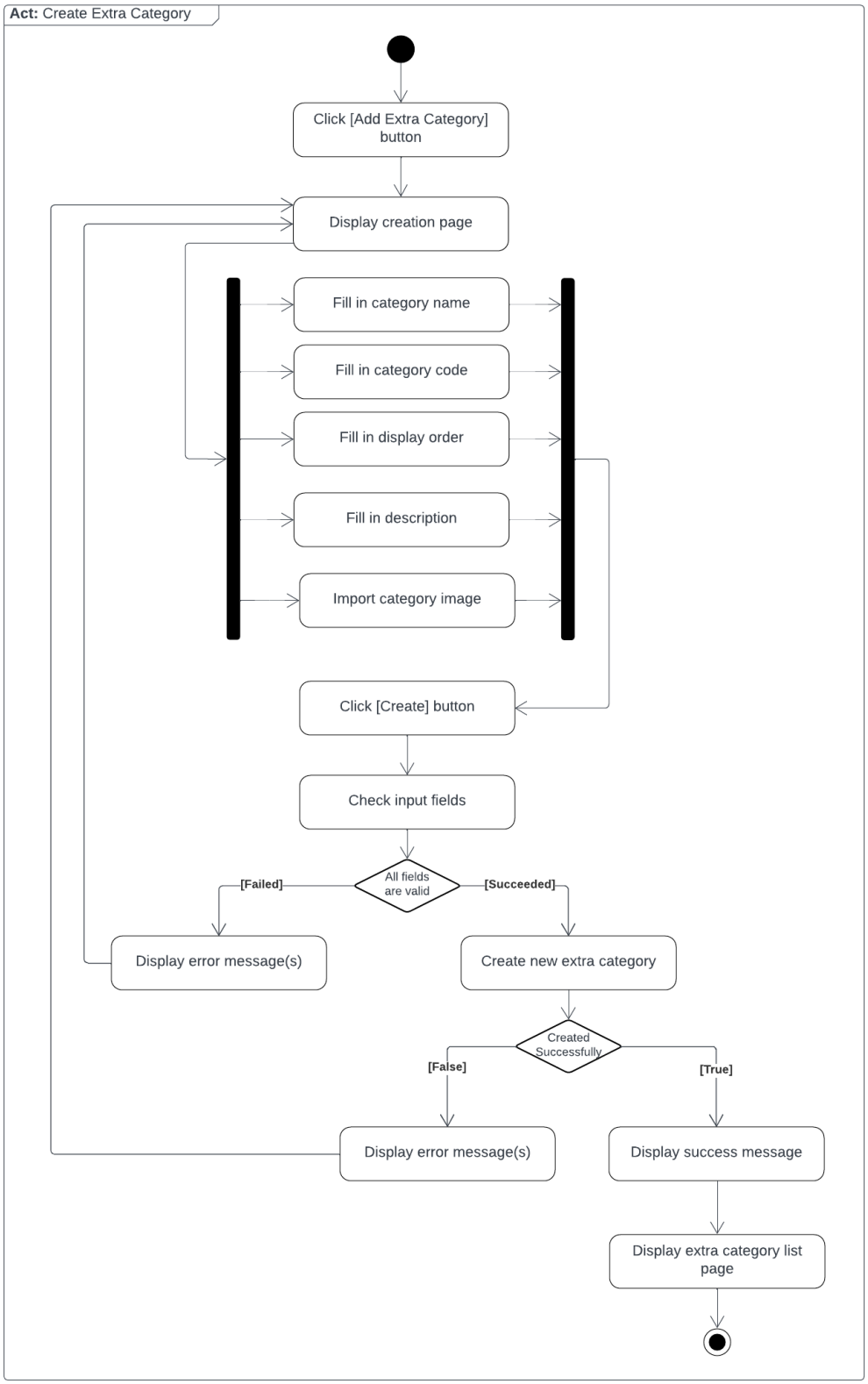


Figure 40: Create Extra Category

3.2.2.5. Add Extra Category to Normal Category

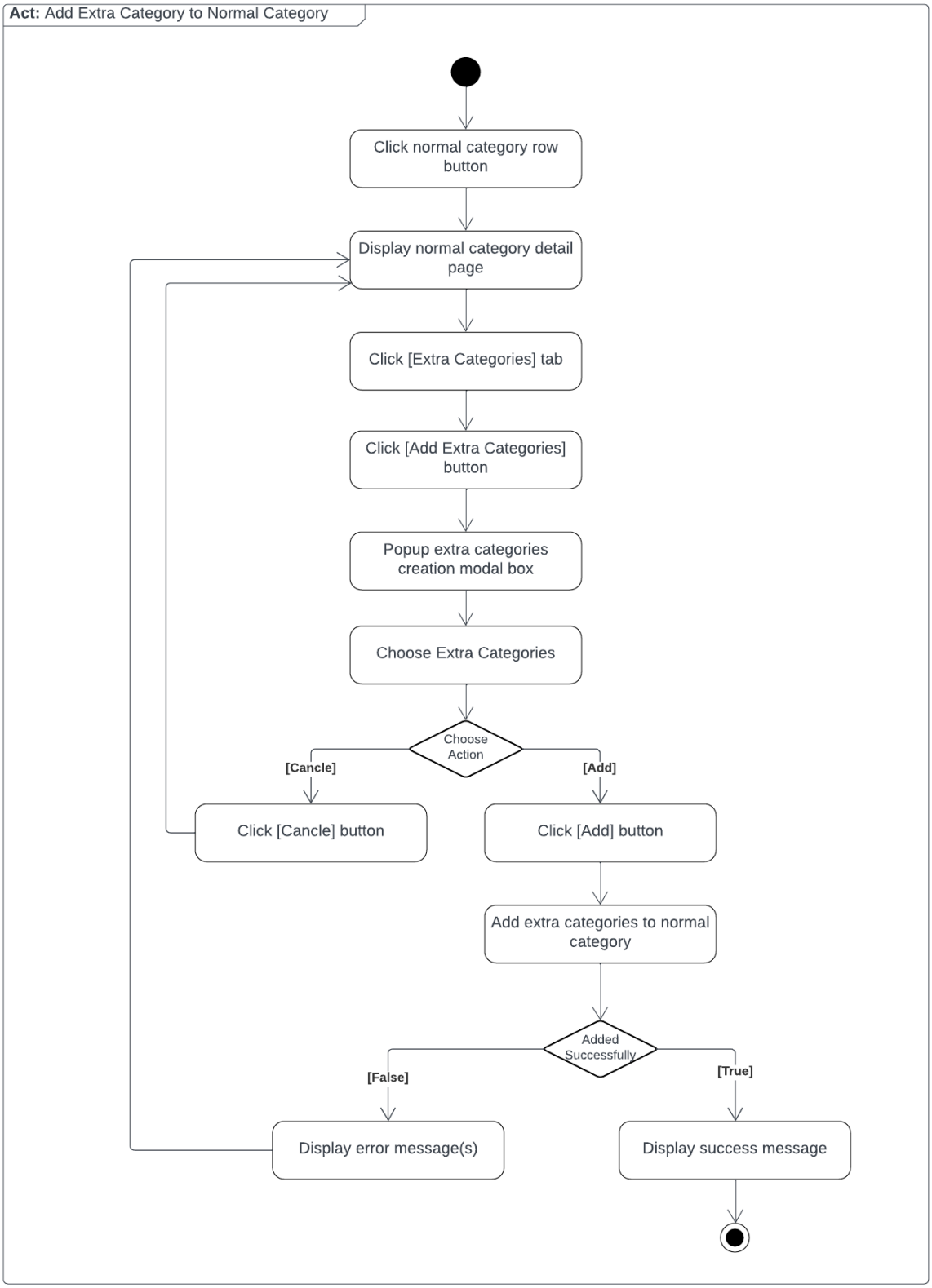


Figure 41: Add Extra Category to Normal Category

3.2.2.4. Create Product

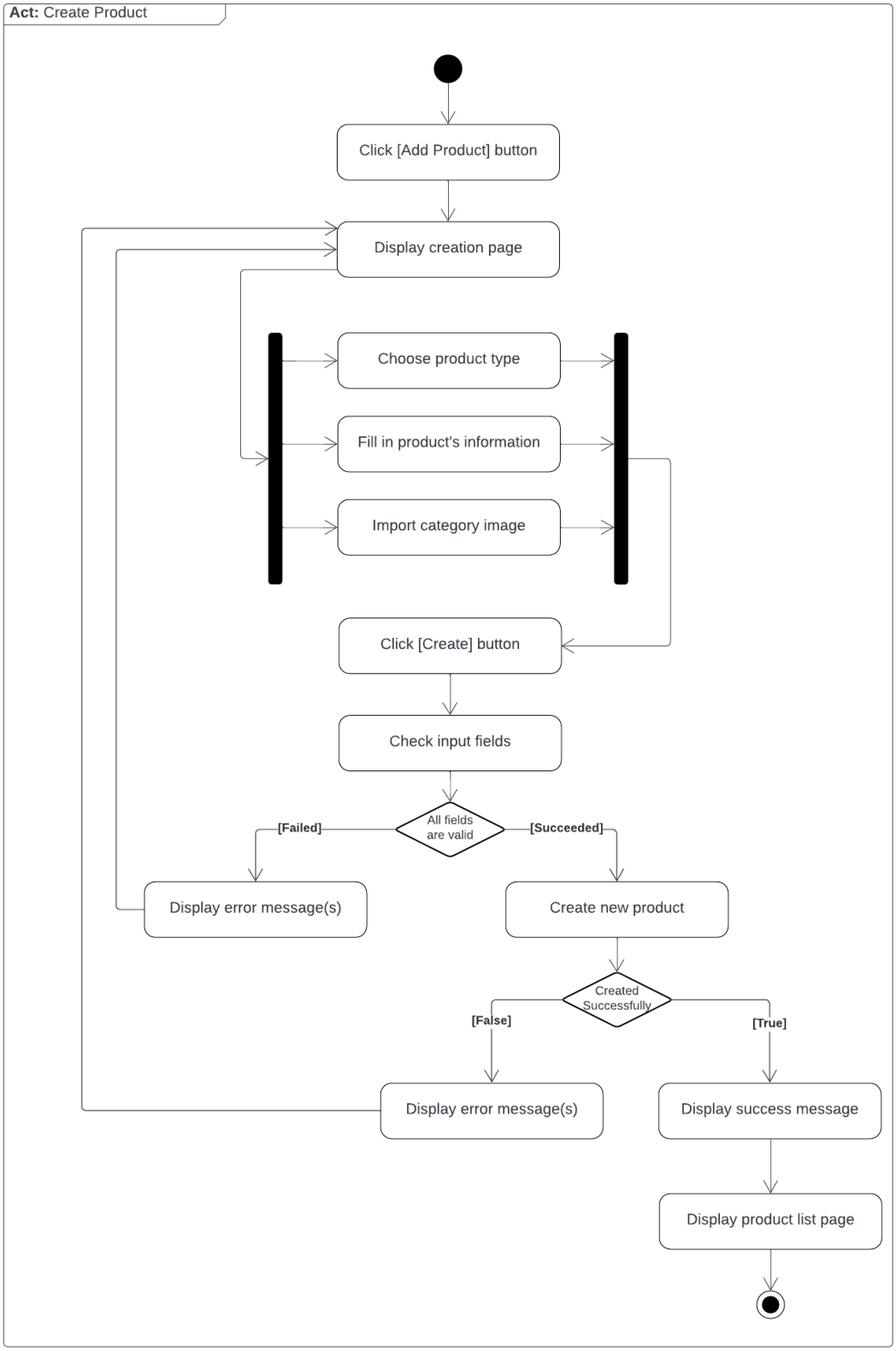


Figure 42: Create Product

3.2.2.5. Create Store Partner

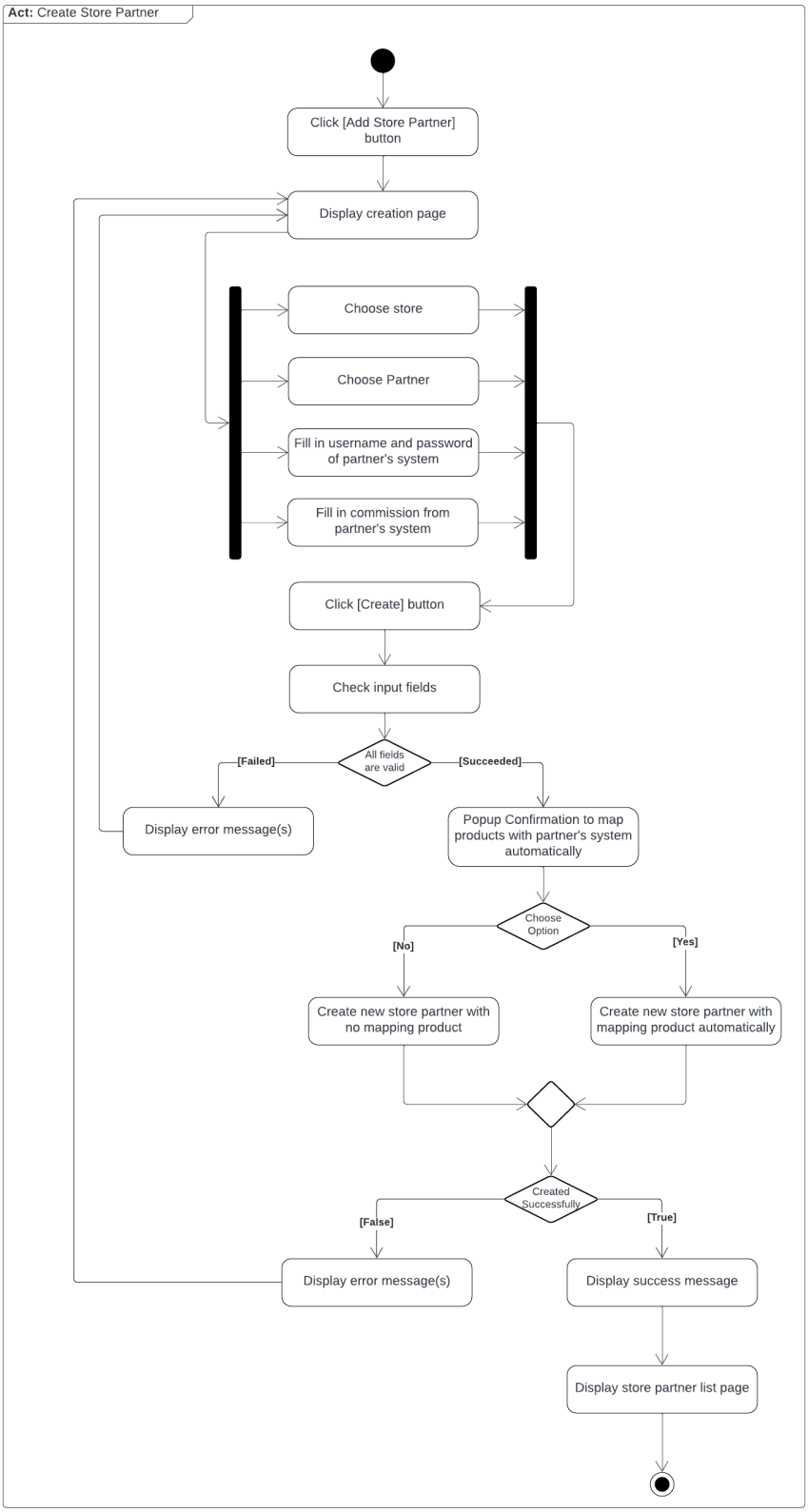


Figure 43: Create Store Partner

3.2.2.6. Create Partner Product

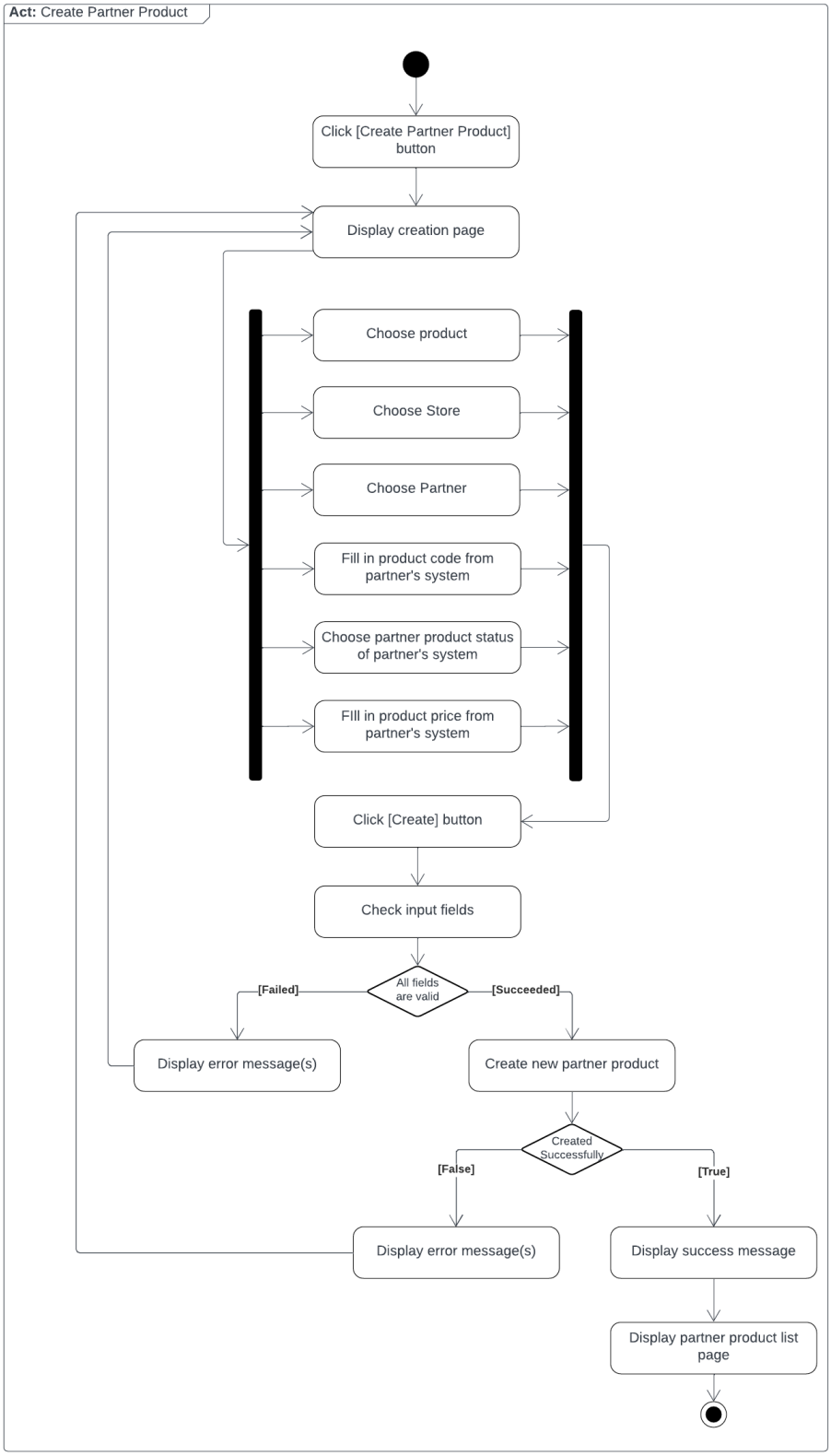


Figure 44: Create Partner Product

3.2.3. Sequence Diagram

3.2.3.1. Create Normal / Extra Category

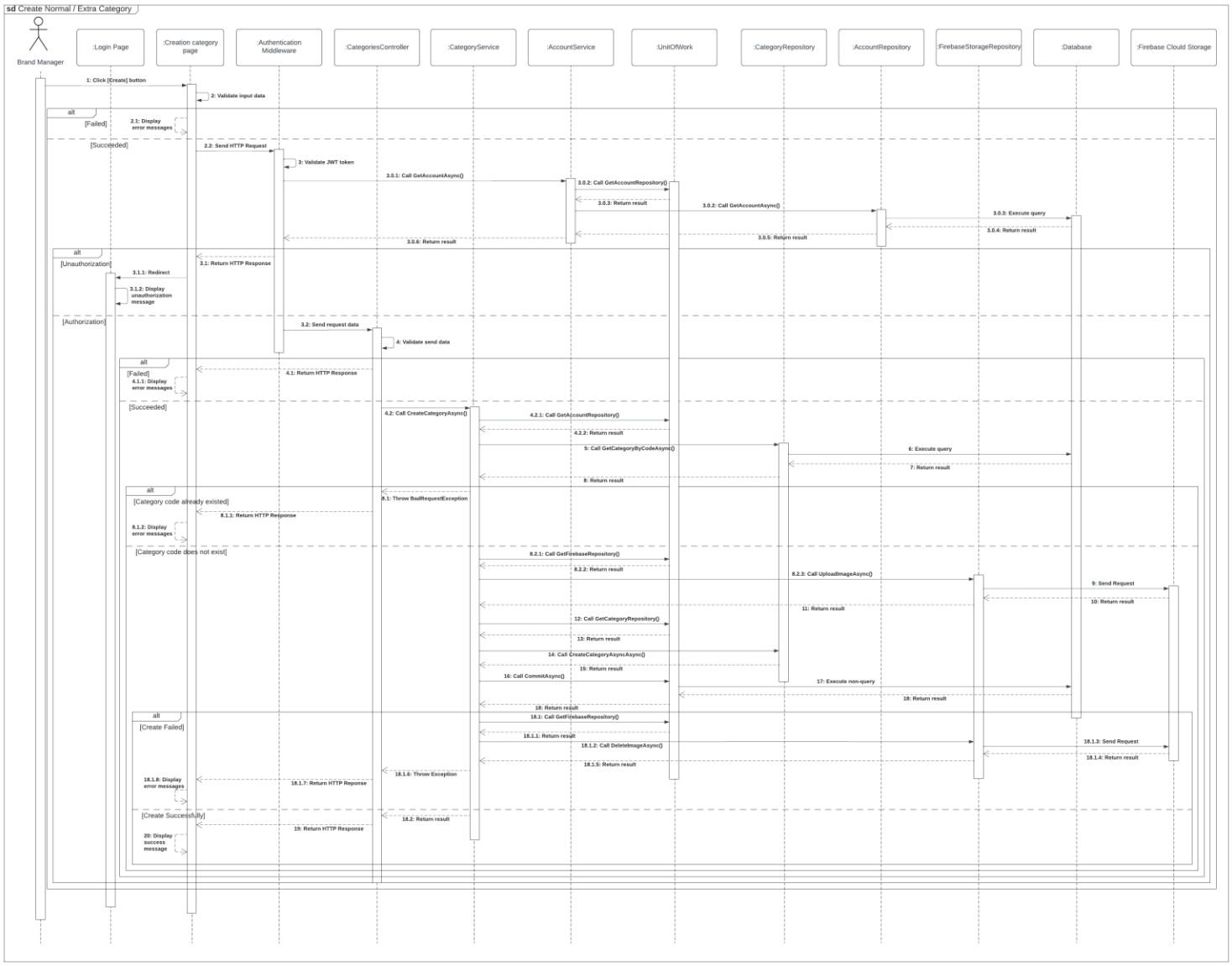


Figure 45: Create Normal / Extra Category

3.2.3.2. Add Extra Category to Normal Category

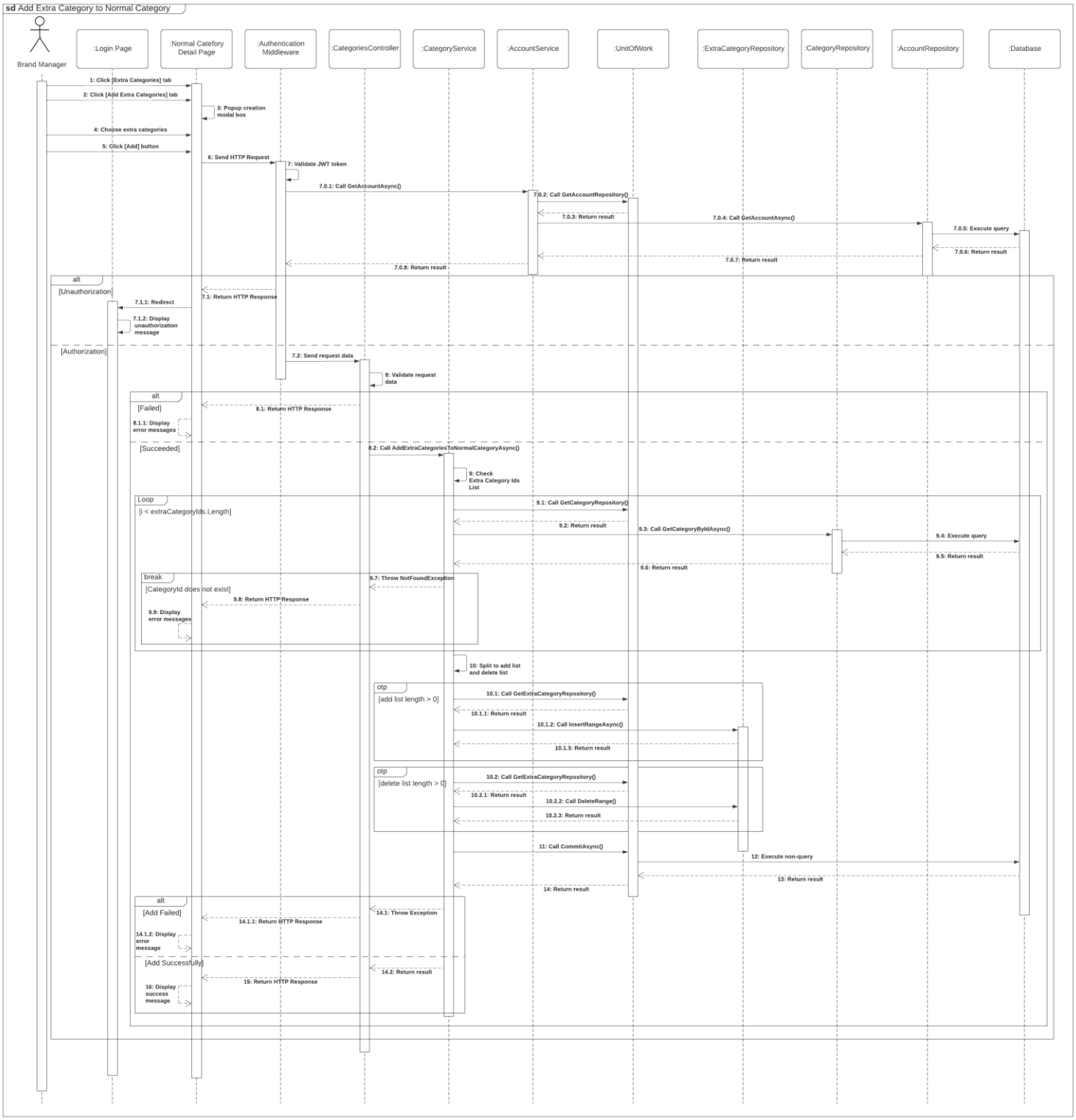


Figure 46: Add Extra Category to Normal Category

3.2.3.3. Create Product

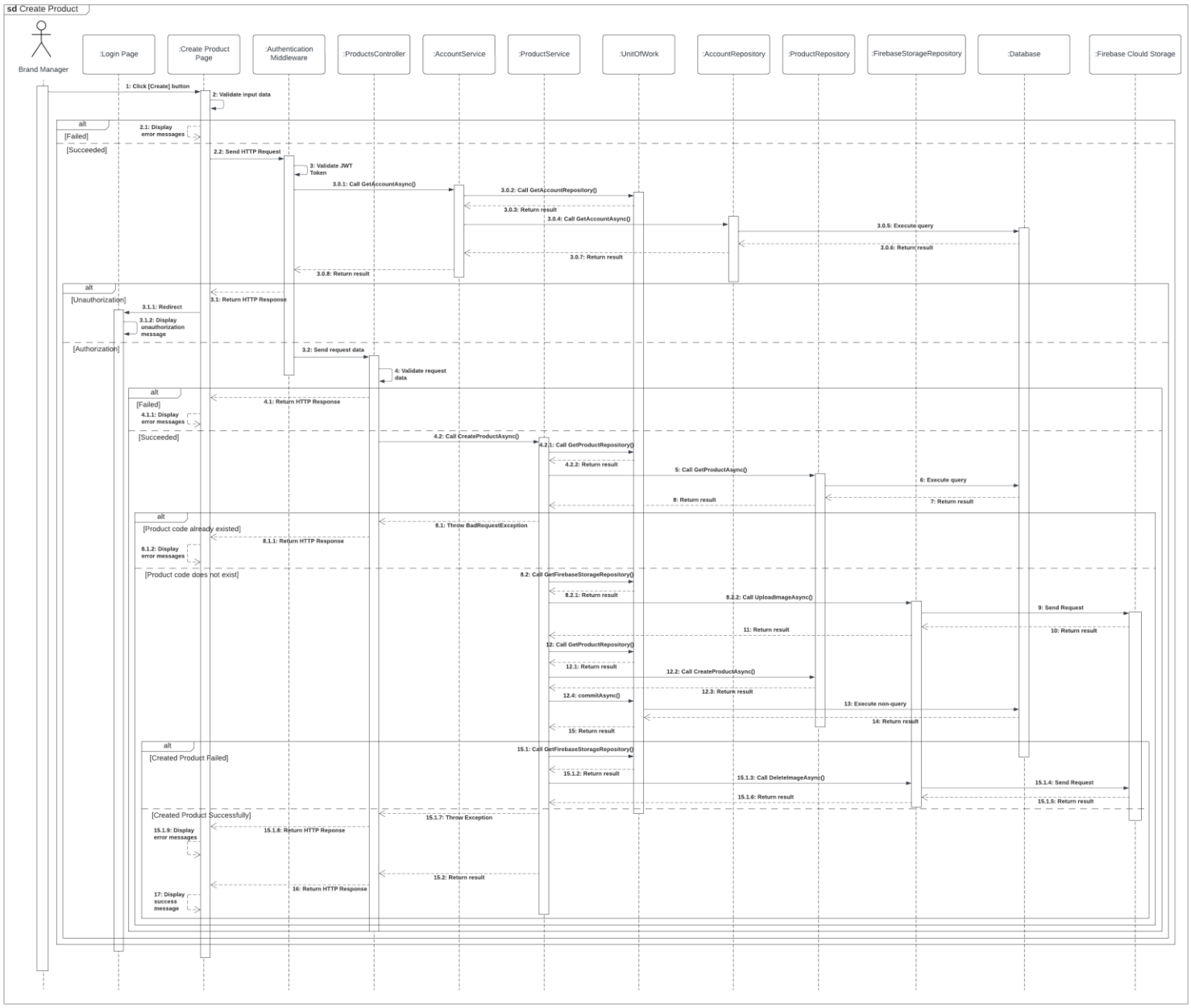


Figure 47: Create Product

3.2.3.4. Create Store Partner

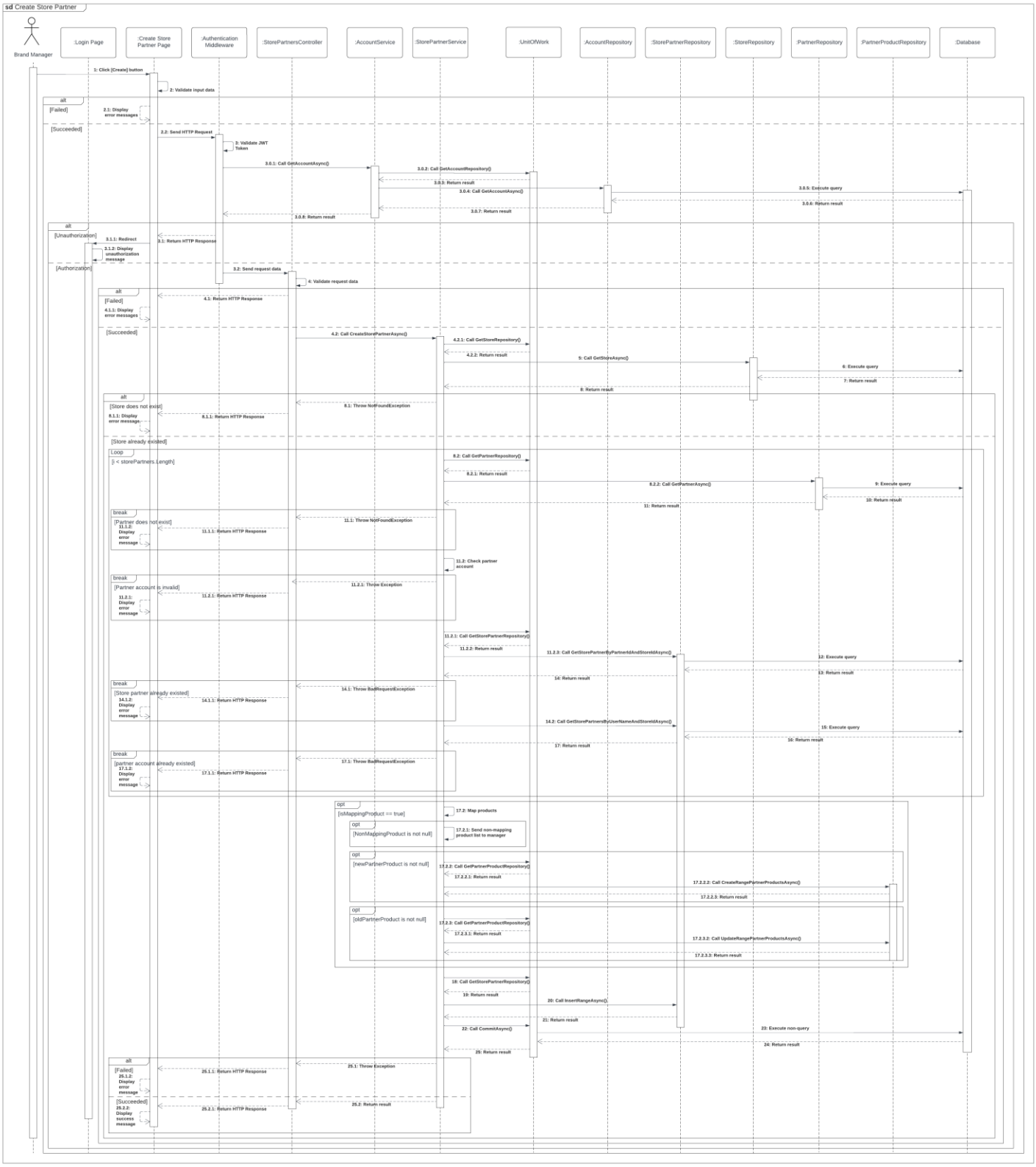


Figure 48: Create Store Partner

3.2.3.5. Create Partner Product

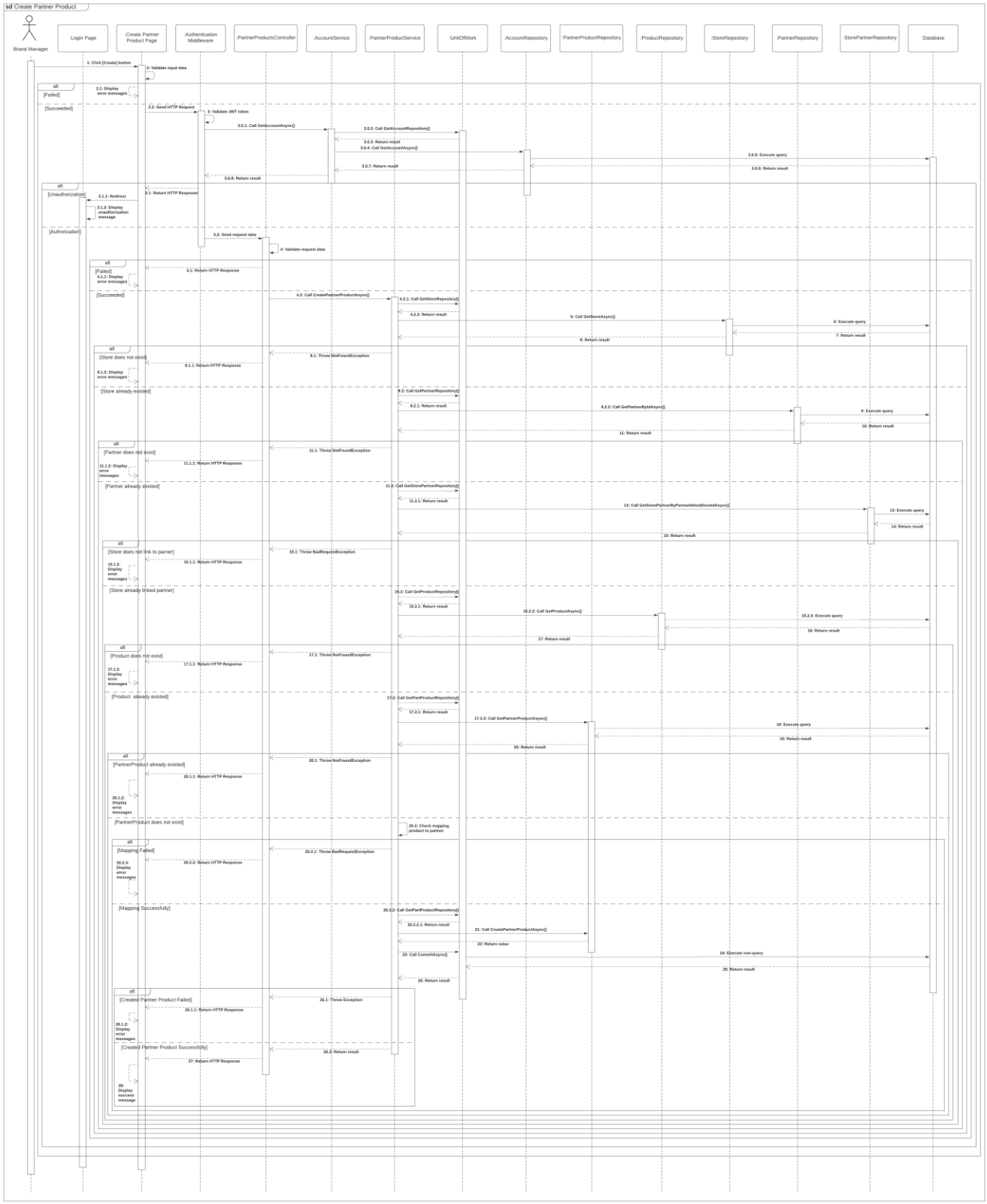


Figure 49: Create Partner Product

3.2.4. State Machine Diagram

3.2.4.1. Partner Product State Machine Diagram

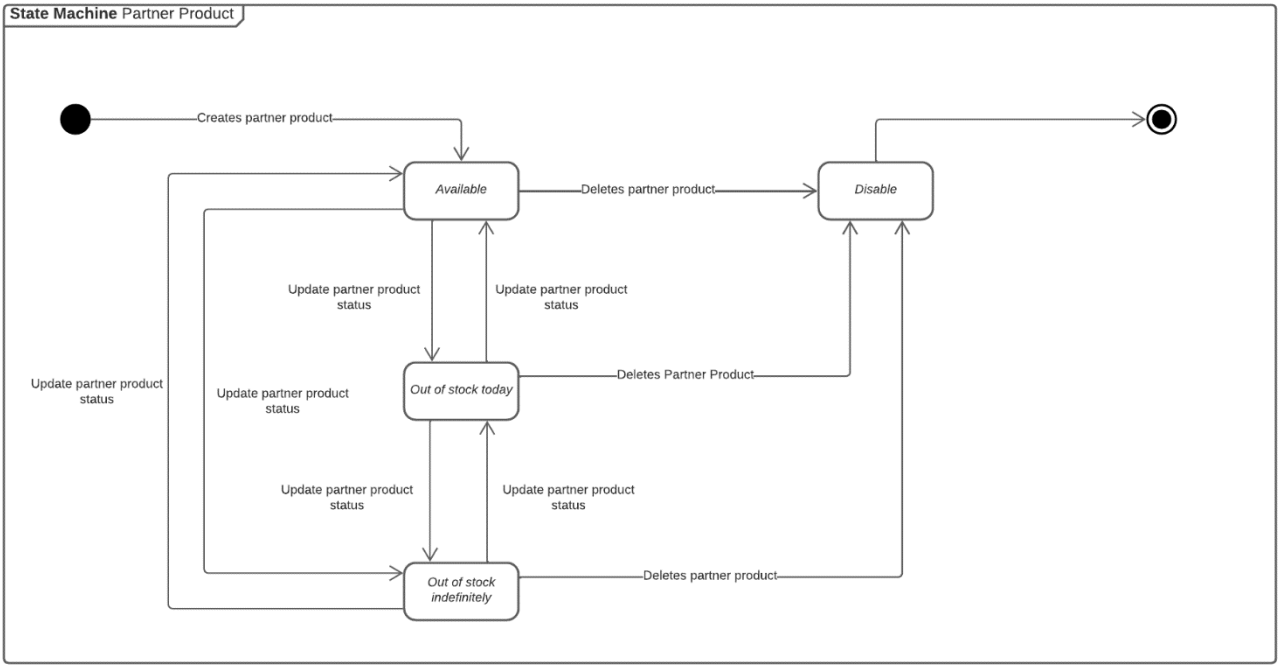


Figure 50: Partner Product State Machine Diagram

3.3. Getting, Parsing, and Confirming Orders from Partners

3.3.1. Class Diagram

3.3.1.1. MBKC Bot

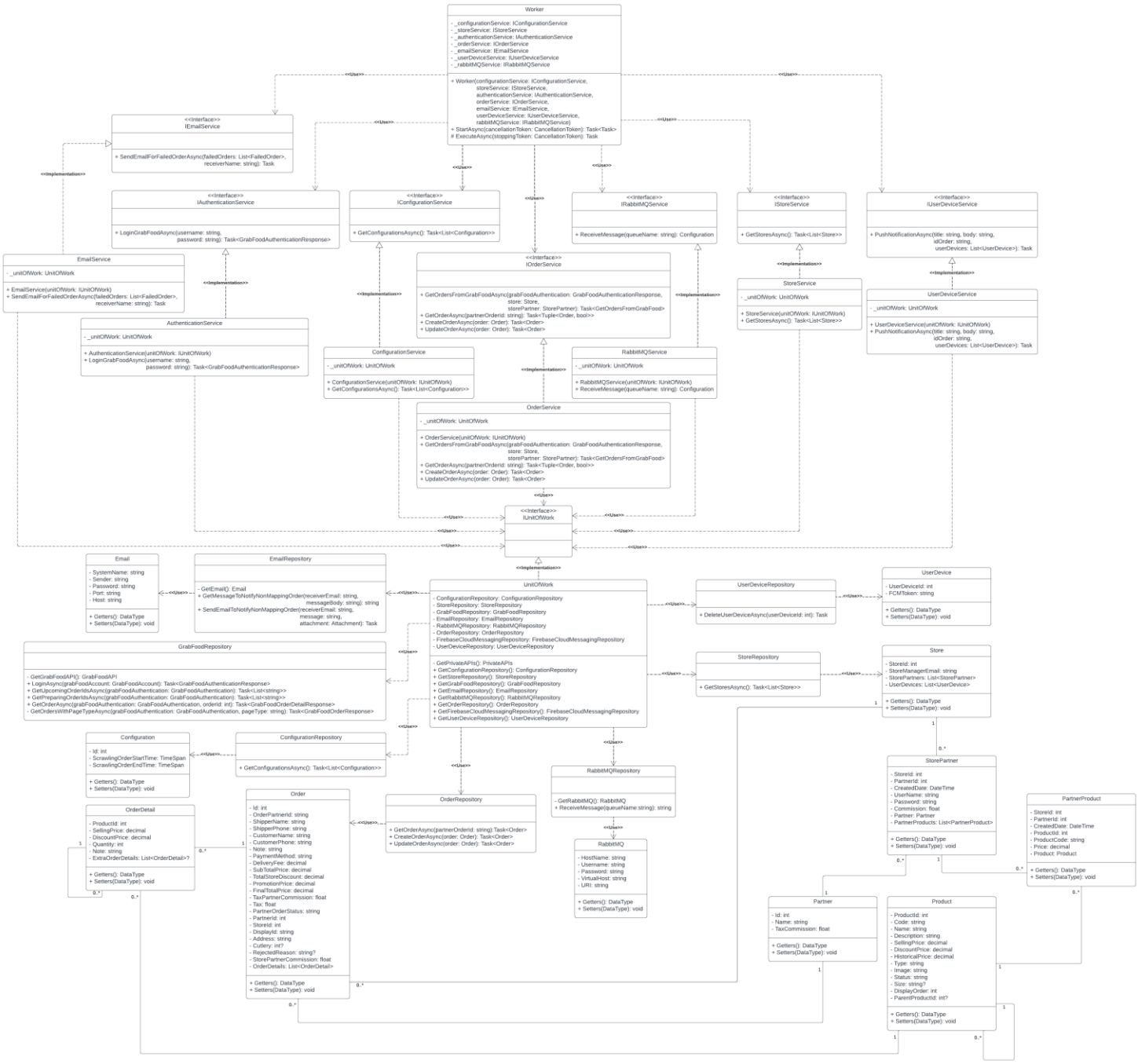


Figure 51: Getting and Parsing Orders From MBKC Bot

3.3.1.1.1. Class Specification

3.3.1.1.1.1. Worker class

No	Attribute	Data Type	Visibility	Description
1	_configurationService	IConfigurationService	Private	This is a field that is used to store an instance of ConfigurationService class.
2	_storeService	IStoreService	Private	This is a field that is used to store an instance of StoreServiceclass.
3	_authenticationService:	IAuthenticationService	Private	This is a field that is used to store an instance of AuthenticationServiceclass.
4	_orderService	IOrderService	Private	This is a field that is used to store an instance of OrderServiceclass.
5	_emailService	IEmailService	Private	This is a field that is used to store an instance of EmailServiceclass.
6	_userDeviceService	IUserDeviceService	Private	This is a field that is used to store an instance of UserDeviceServiceclass.
7	_rabbitMQService	IRabbitMQService	Private	This is a field that is used to store an instance of RabbitMQServiceclass.

Table 197: Worker class - Attributes

No	Method	Data Type	Visibility	Description
1	Worker		Public	This is a constructor of Worker class.
2	StartAsync	Task<Task>	Public	This is a method that will be called first when the application is run.
3	ExecuteAsync	Task	Protected	This is a method that will be called on regularly throughout the life of the application.

Table 198: Worker class – Methods

3.3.1.1.1.2. IconfigurationService interface

No	Method	Data Type	Visibility	Description
1	GetConfigurationsAsync	Task<List<Configuration>>	Public	This is an abstract method that is used to get list configurations.

Table 199: IconfigurationService interface

3.3.1.1.1.3. IRabbitMQService interface

No	Method	Data Type	Visibility	Description
1	ReceiveMessage	Configuration	Public	This is an abstract method that is used to reveive message from RabbitMQ.

Table 200: IRabbitMQService interface

3.3.1.1.1.4. IstoreService interface

No	Method	Data Type	Visibility	Description
1	GetStoresAsync	Task<List<Store>>	Public	This is an abstract method that is used to get list stores in the MBKC System.

Table 201: IstoreService interface

3.3.1.1.1.5. IAuthenticationService interface

No	Method	Data Type	Visibility	Description
1	LoginGrabFoodAsync	Task<GrabFoodAuthenticationResponse	Public	This is an abstract method that is used to login to the GrabFood API.

Table 202: IAuthenticationService interface

3.3.1.1.1.6. IOrderService interface

No	Method	Data Type	Visibility	Description
1	GetOrdersFromGrabFoodAsync	Task<GetOrdersFromGrabFood>	Public	This is an abstract method that is used to get ordres from the GrabFood System for stores.
2	GetOrderAsync	Task<Tuple<Order, bool>>	Public	This is an abstract method that is used to get order from MBKC system for stores.
3	CreateOrderAsync	Task<Order>	Public	This is an abstract method that is used to create new order to the MBKC System.
4	UpdateOrderAsync	Task<Order>	Public	This is an abstract method that is used to update existed order's status to the MBKC System.

Table 203: IOrderService interface

3.3.1.1.1.7. IEmailService interface

No	Method	Data Type	Visibility	Description
1	SendEmailForFailedOrderAsync	Task	Public	This is an abstract method that is used to send email to the store about the failed order from the GrabFood System.

Table 204: IEmailService interface

3.3.1.1.1.8. IuserDeviceService interface

No	Method	Data Type	Visibility	Description
1	PushNotificationAsync	Task	Public	This is an abstract method that is used to push notification when creating a new order or updating an existed order status successfully.

Table 205: IuserDeviceService interface

3.3.1.1.1.9. ConfigurationService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 206: ConfigurationService class - Attributes

No	Method	Data Type	Visibility	Description
1	ConfigurationService		Public	This is a constructor of ConfigurationService class.
2	GetConfigurationsAsync	Task<List<Configuration>>	Public	This is a method that is used to get list configurations.

Table 207: ConfigurationService class - Methods

3.3.1.1.1.10. RabbitMQService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 208: RabbitMQService class - Attributes

No	Method	Data Type	Visibility	Description
1	RabbitMQService		Public	This is a constructor of RabbitMQService class.
2	ReceiveMessage	Configuration	Public	This is a method that is used to reveive message from RabbitMQ.

Table 209: RabbitMQService class - Methods

3.3.1.1.1.11. AuthenticationService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 210: AuthenticationService class - Attributes

No	Method	Data Type	Visibility	Description
1	AuthenticationService		Public	This is a constructor of AuthenticationServic e class.
2	LoginGrabFoodAsync	Task<GrabFoodAuthenticationResponse>	Public	This is a method that is used to login to the GrabFood API.

Table 211: AuthenticationService class - Methods

3.3.1.1.1.12. OrderService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 212: OrderService class - Attributes

No	Method	Data Type	Visibility	Description
1	OrderService		Public	This is a constructor of OrderService class.
2	LoginGrabFoodAsync	Task<GrabFoodAuthenticationResponse>	Public	This is a method that is used to login to the GrabFood API.
3	GetOrderAsync	Task<Tuple<Order, bool>>	Public	This is a method that is used to get order

				from MBKC system for stores.
4	CreateOrderAsync	Task<Order>	Public	This is a method that is used to create new order to the MBKC System.
5	UpdateOrderAsync	Task<Order>	Public	This is a method that is used to update an existed order's status to the MBKC System.

Table 213: OrderService class - Methods

3.3.1.1.1.13. EmailService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 214: EmailService class - Attributes

No	Method	Data Type	Visibility	Description
1	EmailService		Public	This is a constructor of EmailService class.
2	SendEmailForFailedOrderAsync	Task	Public	This is a method that is used to send email to the store about the failed order from the GrabFood System.

Table 215: EmailService class - Methods

3.3.1.1.1.14. UserDeviceService Class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 216: UserDeviceService Class - Attributes

No	Method	Data Type	Visibility	Description
1	UserDeviceService		Public	This is a constructor of UserDeviceService class.
2	PushNotificationAsync	Task	Public	This is an abstract method that is used to push notification when creating a new order or updating an existed order status successfully.

Table 217: UserDeviceService Class - Methods

3.3.1.1.1.15. UnitOfWork class

No	Attribute	Data Type	Visibility	Description
1	ConfigurationRepository	ConfigurationRepository	Private	This is a field that is used to store an instance of class.
2	StoreRepository	StoreRepository	Private	This is a field that is used to store an instance of class.
3	GrabFoodRepository	GrabFoodRepository	Private	This is a field that is used to store an instance of class.
4	EmailRepository	EmailRepository	Private	This is a field that is used to store an instance of class.
5	RabbitMQRepository	RabbitMQRepository	Private	This is a field that is used to store an instance of class.
6	OrderRepository	OrderRepository	Private	This is a field that is used to store an instance of class.
7	FirebaseCloudMessaging Repository	FirebaseCloudMessaging Repository	Private	This is a field that is used to store an instance of class.
8	UserDeviceRepository	UserDeviceRepository	Private	This is a field that is used to store an instance of class.

Table 218: UnitOfWork Class - Attributes

No	Method	Data Type	Visibility	Description
1	GetPrivateAPIs	PrivateAPIs	Private	This is a method that is used to get information about URLs about MBKC Private API.
2	GetConfigurationRepository	ConfigurationRepository	Public	This is a method that is used to get an instance of class.
	GetStoreRepository	StoreRepository		This is a method that is used to get an instance of class.
	GetGrabFoodRepository	GrabFoodRepository		This is a method that is used to get an instance of class.
	GetEmailRepository	EmailRepository		This is a method that is used to get an instance of class.
	GetRabbitMQRepository	RabbitMQRepository		This is a method that is used to get an instance of class.
	GetOrderRepository	OrderRepository		This is a method that is used to get an instance of class.
	GetFirebaseCloudMessagingRepository	FirebaseCloudMessaging Repository		This is a method that is used to get an instance of class.
	GetUserDeviceRepository	UserDeviceRepository		This is a method that is used to get an instance of class.

Table 219: UnitOfWork class - Methods

3.3.1.1.1.16. ConfigurationRepository class

No	Method	Data Type	Visibility	Description
1	GetConfigurationsAsync	Task<List<Configuration>>	Public	This is a method that is used to get list configurations.

Table 220: ConfigurationRepository class

3.3.1.1.1.17. RabbitMQRepository class

No	Method	Data Type	Visibility	Description
1	GetRabbitMQ	RabbitMQ	Private	This is a method to get infotmation to connect to RabbitMQ.
2	ReceiveMessage	string	Public	This is a method that is used to reveive message from RabbitMQ.

Table 221: RabbitMQRepository class

3.3.1.1.1.18. StoreRepository class

No	Method	Data Type	Visibility	Description
1	GetStoresAsync	Task<List<Store>>	Public	This is a method that is used to get list stores in the MBKC System.

Table 222: StoreRepository class

3.3.1.1.1.19. GrabFoodRepository class

No	Method	Data Type	Visibility	Description
1	GetGrabFoodAPI	GrabFoodAPI	Private	This is a method to get infotmation to connect to GrabFood API.
2	LoginAsync	Task<GrabFoodAuthenticationResponse>	Public	This is a method that is used to login to the GrabFood API.
3	GetUpcomingOrderIdsAsync	Task<List<string>>	Public	This is a method that is used to get list upcoming orders from GrabFood API.
4	GetPreparingOrderIdsAsync	Task<List<string>>	Public	This is a method that is used to get

				list preparing orders from GrabFood API.
5	GetOrderAsync	Task<GrabFoodOrderDetailResponse>	Public	This is a method that is used to get an order from GrabFood API.
6	GetOrdersWithPageTypeAsync	Task<GrabFoodOrderResponse>	Private	This is a method that used to connect with GrabFood API to get orders following page type.

Table 223: GrabFoodRepository class

3.3.1.1.1.20. OrderRepository class

No	Method	Data Type	Visibility	Description
1	GetOrderAsync	Task<Order>	Public	This is a method that is used to get list orders from the MBKC System.
2	CreateOrderAsync	Task<Order>	Public	This is a method that is used to create a new order to the MBKC System.
3	UpdateOrderAsync	Task<Order>	Public	This is a method that is used to update an existing order to the MBKC System.

Table 224: OrderRepository class

3.3.1.1.1.21. EmailRepository class

No	Method	Data Type	Visibility	Description
1	GetEmail	Email	Priavte	This is a method that is used to get information to send email.
2	GetMessageToNotifyNonMappingOrder	string	Public	This is a method that is used to get a format for email about non-mapping order.
3	SendEmailToNotifyNonMappingOrder	Task	Public	This is a method that is used to send emails about non-mapping orders.

Table 225: EmailRepository class

3.3.1.1.1.22. UserDeviceRepository class

No	Method	Data Type	Visibility	Description
1	DeleteUserDeviceAsync	Task	Public	This is a method that is used to delete an old user device.

Table 226: UserDeviceRepository class

3.3.1.1.1.23. Configuration class

No	Attribute	Data Type	Visibility	Description
1	Id	int	Private	This is a property that is used to store id of configuration.
2	ScrawlingOrderStartTime	TimeSpan	Private	This is a property that is used to store the start time to get orders.
3	ScrawlingOrderEndTime	TimeSpan	Private	This is a property that is used to store the end time to get orders.

Table 227: Configuration class

3.3.1.1.1.24. RabbitMQ class

No	Attribute	Data Type	Visibility	Description
1	HostName	string	Private	This is a property that is used to store the host name of RabbitMQ.
2	Username	string	Private	This is a property that is used to store the username of RabbitMQ.
3	Password	string	Private	This is a property that is used to store the password of RabbitMQ.
4	VirtualHost	string	Private	This is a property that is used to store the virtual host of RabbitMQ.
5	URI	string	Private	This is a property that is used to store the uri of RabbitMQ.

Table 228: RabbitMQ class

3.3.1.1.1.25. Store Class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store.
2	StoreManagerEmail	string	Private	This is a property that is used to store the store manager email of the store.
3	StorePartners	List<StorePartner>	Private	This is a property that is used to store the list store partners of the store.
4	UserDevices	List<UserDevice>	Private	This is a property that is used to store the list user devices of the store.

Table 229: Store Class

3.3.1.1.1.26. StorePartner Class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store partner.
2	PartnerId	string	Private	This is a property that is used to store the partner id of the store partner.
3	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the store partner.
4	UserName	string	Private	This is a property that is used to store the username of the store partner.

5	Password	string	Private	This is a property that is used to store the password of the store partner.
6	Commission	float	Private	This is a property that is used to store the commission of the store partner.
7	Partner	Partner	Private	This is a property that is used to store the partner of the store partner.
8	PartnerProducts	IEnumerable<PartnerProduct>	Private	This is a property that is used to store the list partner products of the store partner.

Table 230: StorePartner Class

3.3.1.1.1.27. Partner class

No	Attribute	Data Type	Visibility	Description
1	Id	int	Private	This is a property that is used to store the partner id of the partner.
2	Name	string	Private	This is a property that is used to store the name of the partner.
3	TaxCommission	float	Private	This is a property that is used to store the tax of commission.

Table 231: Partner Class

3.3.1.1.1.28. PartnerProduct class

No	Attribute	Data Type	Visibility	Description
1	ProductId	int	Private	This is a property that is used to store the product id of the partner product.
2	PartnerId	int	Private	This is a property that is used to store the partner id of the partner product.
3	StoreId	int	Private	This is a property that is used to store the store id of the partner product.
4	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the partner product.
5	ProductCode	string	Private	This is a property that is used to store the product code of the partner product.
6	Price	decimal	Private	This is a property that is used to store the price of the partner product.
7	Product	Product	Private	This is a property that is used to store the product of the partner product.

Table 232: PartnerProduct class

3.3.1.1.1.29. Product Class

No	Attribute	Data Type	Visibility	Description
1	ProductId	int	Private	This is a property that is used to store the product id of the product.

2	Code	string	Private	This is a property that is used to store the code of the product.
3	Name	string	Private	This is a property that is used to store the name of the product.
4	Type	string	Private	This is a property that is used to store the type of product.
5	DisplayOrder	int	Private	This is a property that is used to store the display order of the product.
6	Description	string	Private	This is a property that is used to store the description of the product.
7	Image	string	Private	This is a property that is used to store the image of the product.
8	SellingPrice	decimal	Private	This is a property that is used to store the selling price of the product.
9	DiscountPrice	decimal	Private	This is a property that is used to store the discount price of the product.
10	HistoricalPrice	decimal	Private	This is a property that is used to store the historical price of the product.
11	Size	string?	Private	This is a property that is used to store the size of the product.
12	Status	string	Private	This is a property that is used to store the status of the product.
13	ParentProductId	int?	Private	This is a property that is used to store the parent product id of the product.

Table 233: Product Class

3.3.1.1.1.30. Order Class

No	Attribute	Data Type	Visibility	Description
1	Id	int	Private	This is a property that is used to store the id of the order.
2	OrderPartnerId	string	Private	This is a property that is used to store the order partner id of the order.
3	ShipperName	string	Private	This is a property that is used to store the shipper's name of the order.
4	ShipperPhone	string	Private	This is a property that is a property that is used to store the shipper phone of the order.

5	CustomerName	string	Private	This is a property that is used to store the customer's name of the order.
6	CustomerPhone	string	Private	This is a property that is used to store the customer phone of the order.
7	Note	string	Private	This is a property that is used to store the note of the order.
8	PaymentMethod	string	Private	This is a property that is used to store the payment method of the order.
9	DeliveryFee	decimal	Private	This is a property that is used to store the delivery fee of the order.
10	SubTotalPrice	decimal	Private	This is a property that is used to store the sub total price of the order.
11	TotalStoreDiscount	decimal	Private	This is a property that is used to store the total store discount of the order.
12	PromotionPrice	decimal	Private	This is a property that is used to store the total promotion price of the order.
13	FinalTotalPrice	decimal	Private	This is a property that is used to store the final total price of the order.
14	TaxPartnerCommission	float	Private	This is a property that is used to store the tax of partner's commission of the order.
15	Tax	float	Private	This is a property that is used to store the tax of the order.
16	SystemStatus	string	Private	This is a property that is used to store the system status of the order.
17	PartnerOrderStatus	string	Private	This is a property that is used to store the partner order status of the order.
18	PartnerId	int	Private	This is a property that is used to store the partner id of the order.
19	StoreId	int	Private	This is a property that is used to store the store id of the order.
20	DisplayId	string	Private	This is a property that is used to store the display id of the order.
21	Address	string	Private	This is a property that is used to store the address of the order.
22	Cutlery	int?	Private	This is a property that is used to store the cutlery of the order.
23	ConfirmedBy	int?	Private	This is a property that is used to store the confirm by of the order.
24	RejectedReason	string?	Private	This is a property that is used to store the rejected reason of the order.
25	StorePartnerCommission	float	Private	This is a property that is used to store the store partner commission of the order.
26	OrderDetails	List<OrderDetail>	Private	This is a property that is used to store the list order details of the order.

Table 234: Order Class

3.3.1.1.1.31. OrderDetail Class

No	Attribute	Data Type	Visibility	Description

1	ProductId	int	Private	This is a property that is used to store the order partner id of the order.
2	SellingPrice	decimal	Private	This is a property that is used to store the shipper's name of the order.
3				
4	Quantity	int	Private	This is a property that is used to store the shipper phone of the order.
5	Note	string	Private	This is a property that is used to store the customer's name of the order.
6	ExtraOrderDetails	List<OrderDetail>?	Private	This is a property that is used to store the customer phone of the order.

Table 235: OrderDetail Class

3.3.1.1.1.32. Email Class

No	Attribute	Data Type	Visibility	Description
1	SystemName	string	Private	This is a property that is used to store the system name.
2	Sender	string	Private	This is a property that is used to store the sender email.
3	Password	string	Private	This is a property that is used to store the password email of sender.
4	Port	int	Private	This is a property that is used to store the port of the SMTP.
5	Host	string	Private	This is a property that is used to store the host of the SMTP.

Table 236: Email Class

3.3.1.2. MBKC Private API

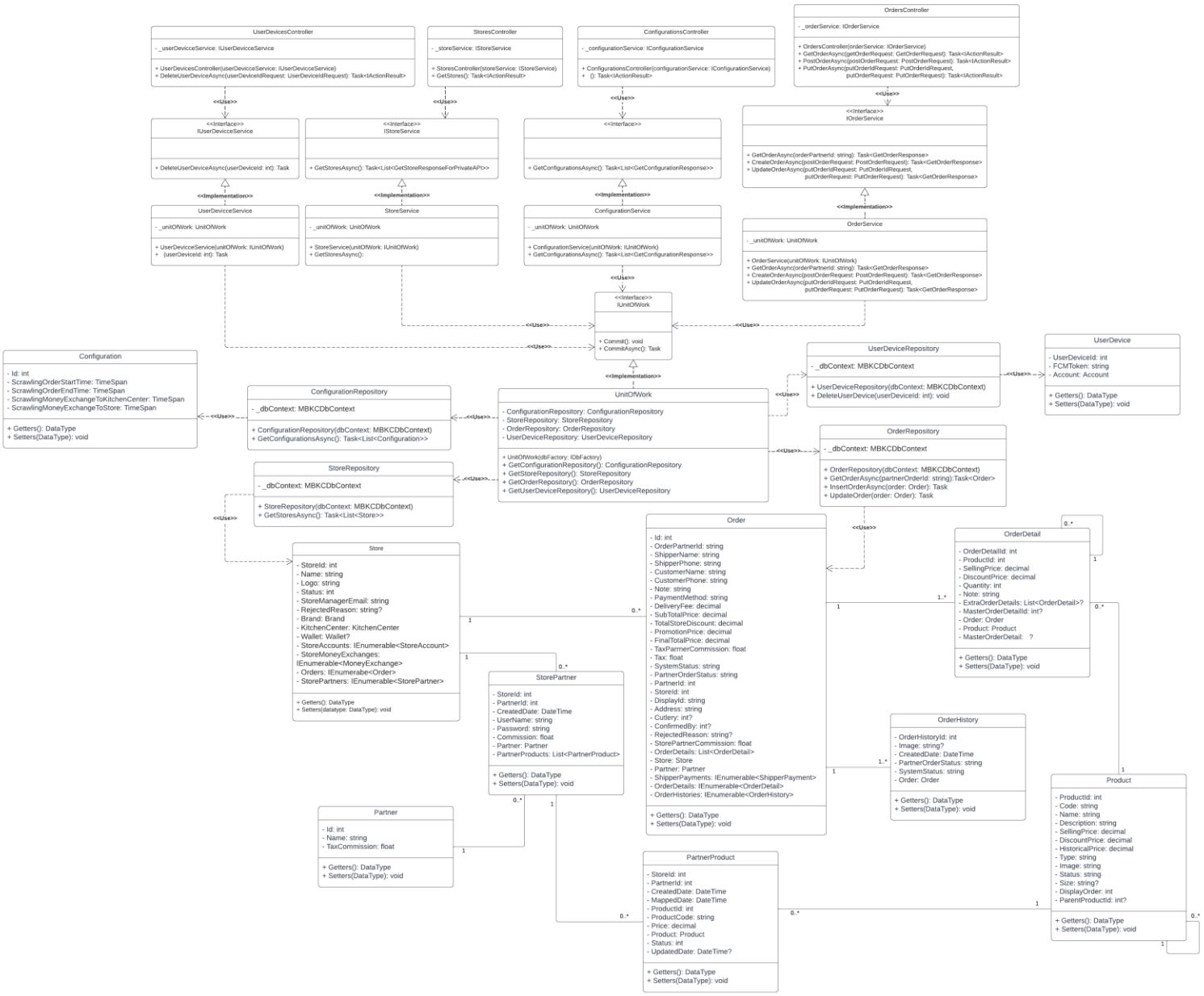


Figure 52: Creating Orders from MBKC Private API

3.3.1.2.1. Class Specification

3.3.1.2.1.1. UserDevicesController class

No	Attribute	Data Type	Visibility	Description
1	_userDevicceService	IUserDevicceService	Private	This is a field that is used to store an instance of UserDeviceService class.

Table 237: UserDevicesController class - Attributes

No	Method	Data Type	Visibility	Description
1	UserDevicesController		Public	This is a constructor of UserDevicesController class.
2	DeleteUserDeviceAsync	Task<IActionResult>	Public	This is an action method that is used to delete a user's device token.

Table 238: UserDevicesController class - Methods

3.3.1.2.1.2. StoresController class

No	Attribute	Data Type	Visibility	Description
1	_storeService	IStoreService	Private	This is a field that is used to store an instance of StoreService class.

Table 239: StoresController class - Attributes

No	Method	Data Type	Visibility	Description
1	StoresController		Public	This is a constructor of StoresController class.
2	GetStores	Task<IActionResult>	Public	This is an action method that is used to get list stores.

Table 240: StoresController class - Methods

3.3.1.2.1.3. ConfigurationsController class

No	Attribute	Data Type	Visibility	Description
1	_configurationService	IConfigurationService	Private	This is a field that is used to store an instance of ConfigurationService class.

Table 241: ConfigurationsController class - Attributes

No	Method	Data Type	Visibility	Description
1	ConfigurationsController		Public	This is a constructor of ConfigurationsController class.
2	GetConfigurationsAsync	Task<IActionResult>	Public	This is an action method that is used to get configurations.

Table 242: ConfigurationsController class - Methods

3.3.1.2.1.4. OrdersController class

No	Attribute	Data Type	Visibility	Description
1	_orderService	IOrderService	Private	This is a field that is used to store an instance of OrderService class.

Table 243: OrdersController class - Attributes

No	Method	Data Type	Visibility	Description
1	OrdersController		Public	This is a constructor of OrdersController class.
2	GetOrderAsync	Task<IActionResult>	Public	This is an action method that is used to get an order.
3	PostOrderAsync	Task<IActionResult>	Public	This is an action method that is used to create new order.
4	PutOrderAsync	Task<IActionResult>	Public	This is an action method that is used to update partner status of orde.

Table 244: OrdersController class - Methods

3.3.1.2.1.5. IUserDevicceService interface

No	Method	Data Type	Visibility	Description
1	DeleteUserDeviceAsync	Task	Public	This is an abstract method that is used to delete an existed user device.

Table 245: IUserDevicceService interface

3.3.1.2.1.6. IStoreService interface

No	Method	Data Type	Visibility	Description
1	GetStoresAsync	Task<List<GetStoreResponseForPrivateAPI>>	Public	This is an abstract method that is used to get list stores.

Table 246: IStoreService interface

3.3.1.2.1.7. IConfigurationService interface

No	Method	Data Type	Visibility	Description
1	GetConfigurationsAsync	Task<List<GetConfigurationResponse>>	Public	This is an abstract method that is used to get list configurations.

Table 247: IConfigurationService interface

3.3.1.2.1.8. IOrderService interface

No	Method	Data Type	Visibility	Description
1	GetOrderAsync	Task<GetOrderResponse>	Public	This is an abstract method that is used to get an order.
2	CreateOrderAsync	Task<GetOrderResponse>	Public	This is an abstract method that is used to create new order.
3	UpdateOrderAsync	Task<GetOrderResponse>	Public	This is an abstract method that is used to update partner status of order.

Table 248: IOrderService interface

3.3.1.2.1.9. UserDevicceService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 249: UserDevicceService class - Attributes

No	Method	Data Type	Visibility	Description
1	UserDevicceService		Public	This is a constructor of UserDevicceService class.
2	DeleteUserDeviceAsync	Task	Public	This is a method that is used to delete a user's device.

Table 250: UserDevicceService class - Methods

3.3.1.2.1.10. StoreService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 251: StoreService class - Attributes

No	Method	Data Type	Visibility	Description
1	StoreService		Public	This is a constructor of StoreService class.
2	GetStoresAsync	Task<List<GetStoreResponseForPrivateAPI>>	Public	This is a method that is used to get list stores.

Table 252: StoreService class - Methods

3.3.1.2.1.11. ConfigurationService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 253: ConfigurationService class - Attributes

No	Method	Data Type	Visibility	Description
1	ConfigurationService		Public	This is a constructor of ConfigurationService class.
2	GetConfigurationsAsync	Task<List<GetConfigurationResponse>>	Public	This is a method that is used to get list configurations.

Table 254: ConfigurationService class - Methods

3.3.1.2.1.12. OrderService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 255: OrderService class - Attributes

No	Method	Data Type	Visibility	Description
1	OrderService		Public	This is a constructor of OrderService class.
2	GetOrderAsync	Task<GetOrderResponse>	Public	This is a method that is used to get an order.
3	CreateOrderAsync	Task<GetOrderResponse>	Public	This is a method that is used to create new order.
4	UpdateOrderAsync	Task<GetOrderResponse>	Public	This is a method that is used to update partner status of order.

Table 256: OrderService class - Methods

3.3.1.2.1.13. IUnitOfWork interface

No	Method	Data Type	Visibility	Description
1	Commit	void	Public	This is an abstract method that is used to execute non-query.
2	CommitAsync	Task	Public	This is an abstract method that is used to execute non-query with an asynchronous process.

Table 257: IUnitOfWork interface

3.3.1.2.1.14. Configuration class

No	Attribute	Data Type	Visibility	Description
1	Id	int	Private	This is a property that is used to store the id of the configuration.
2	ScrawlingOrderStartTime	TimeSpan	Private	This is a property that is used store the crawling order start time.
3	ScrawlingOrderEndTime	TimeSpan	Private	This is a property that is used to store the crawling order end time.
4	ScrawlingMoneyExchangeToKitchenCenter	TimeSpan	Private	This is a property that is used to store the time about money exchange from cashier's wallet to kitchen center wallet.
5	ScrawlingMoneyExchangeToStore	TimeSpan	Private	This is a property that is used to store the timeabout money exchange from kitchen center's wallet to store's wallet.

Table 258: Configuration class

3.3.1.2.1.15. ConfigurationRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 259: ConfigurationRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	ConfigurationRepository		Public	This is a constructor of ConfigurationRepository class.
2	GetConfigurationsAsync	Task<List<Configuration>>	Public	This is a method that is used to get configuration.

Table 260: ConfigurationRepository class - Methods

3.3.1.2.1.16. UnitOfWork class

No	Attribute	Data Type	Visibility	Description
1	ConfigurationRepository	ConfigurationRepository	Private	This is a property that is used to store an instance of class.
2	StoreRepository	StoreRepository	Private	This is a property that is used to store an instance of class.
3	OrderRepository	OrderRepository	Private	This is a property that is used to store an instance of class.
4	UserDeviceRepository	UserDeviceRepository	Private	This is a property that is used to store an instance of class.

Table 261: UnitOfWork class - Attrubutes

No	Method	Data Type	Visibility	Description
1	UnitOfWork		Public	This is a constructor of UnitOfWork class.
2	GetConfigurationRepository	ConfigurationRepository	Public	This is a method that is used to store an instance of ConfigurationRepository class.
3	GetStoreRepository	StoreRepository	Public	This is a method that is used to store an instance of StoreRepository class.
4	GetOrderRepository	OrderRepository	Public	This is a method that is used to store an instance of OrderRepository class.
5	GetUserDeviceRepository	UserDeviceRepository	Public	This is a method that is used to store an instance of UserDeviceRepository class.

Table 262: UnitOfWork class - Methods

3.3.1.2.1.17. UserDeviceRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 263: UserDeviceRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	UserDeviceRepository		Public	This is a constructor of UserDeviceRepository class.
2	DeleteUserDevice	void	Public	Is the method used to delete a user's device token in the MBKC system.

Table 264: UserDeviceRepository class - Methods

3.3.1.2.1.18. UserDevice class

No	Attribute	Data Type	Visibility	Description
1	UserDeviceId	int	Private	This is a property that is used to store the user device id of the user device.
2	FCMToken	string	Private	This is a property that is used to store the FCM token of the user device.
3	Account	Account	Private	This is a property that is used to store the account of user device.

Table 265: UserDevice class

3.3.1.2.1.19. Store class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store.
2	Name	string	Private	This is a property that is used to store the name of the store.
3	Logo	string	Private	This is a property that is used to store the logo of the store.
4	Status	int	Private	This is a property that is used to store the status of the store.
5	StoreManagerEmail	string	Private	This is a property that is used to store the store manager email of the store.
6	RejectedReason	string?	Private	This is a property that is used to store the rejected reason of the store.
7	Brand	Brand	Private	This is a property that is used to store the brand of the store.
8	KitchenCenter	KitchenCenter	Private	This is a property that is used to store the kitchen center of the store.
9	Wallet	Wallet?	Private	This is a property that is used to store the wallet of the store.
10	StoreAccounts	IEnumerable<StoreAccount>	Private	This is a property that is used to store the list store accounts of the store.
11	StoreMoneyExchanges	IEnumerable<StoreAccount>	Private	This is a property that is used to store the list store money exchanges of the store.
12	Orders	IEnumerabe<Order>	Private	This is a property that is used to store the list orders of the store.
13	StorePartners	IEnumerable<StorePartner>	Private	This is a property that is used to store the list store partners of the store.

Table 266: Store class

3.3.1.2.1.20. OrderRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 267: OrderRepository class

No	Method	Data Type	Visibility	Description
1	OrderRepository		Public	This is a constructor of OrderRepository class.

2	GetOrderAsync	Task<Order>	Public	This is a method that is used to get a order.
3	InsertOrderAsync	Task	Public	This is a method that is used to create a new order.
4	UpdateOrder	Task	Public	This is a method that is used to update an order.

Table 268: OrderRepository class

3.3.1.2.1.21 Order class

No	Attribute	Data Type	Visibility	Description
1	Id	int	Private	This is a property that is used to store the id of the order.
2	OrderPartnerId	string	Private	This is a property that is used to store the order partner id of the order.
3	ShipperName	string	Private	This is a property that is used to store the shipper's name of the order.
4	ShipperPhone	string	Private	This is a property that is a property that is used to store the shipper phone of the order.
5	CustomerName	string	Private	This is a property that is used to store the customer's name of the order.
6	CustomerPhone	string	Private	This is a property that is used to store the customer phone of the order.
7	Note	string	Private	This is a property that is used to store the note of the order.
8	PaymentMethod	string	Private	This is a property that is used to store the payment method of the order.
9	DeliveryFee	decimal	Private	This is a property that is used to store the delivery fee of the order.
10	SubTotalPrice	decimal	Private	This is a property that is used to store the sub total price of the order.
11	TotalStoreDiscount	decimal	Private	This is a property that is used to store the total store discount of the order.
12	PromotionPrice	decimal	Private	This is a property that is used to store the promotion price of the order.
13	FinalTotalPrice	decimal	Private	This is a property that is used to store the final total price of the order.
14	TaxPartnerCommission	float	Private	This is a property that is used to

				store the tax of partner commission.
15	Tax	float	Private	This is a property that is used to store the tax of the order.
16	SystemStatus	string	Private	This is a property that is used to store the system status of the order.
17	PartnerOrderStatus	string	Private	This is a property that is used to store the partner order status of the order.
18	PartnerId	int	Private	This is a property that is used to store the partner id of the order.
19	StoreId	int	Private	This is a property that is used to store the store id of the order.
20	DisplayId	string	Private	This is a property that is used to store the display id of the order.
21	Address	string	Private	This is a property that is used to store the address of the order.
22	Cutlery	int?	Private	This is a property that is used to store the cutlery of the order.
23	ConfirmedBy	int?	Private	This is a property that is used to store the confirm by of the order.
24	RejectedReason	string?	Private	This is a property that is used to store the rejected reason of the order.
25	StorePartnerCommission	float	Private	This is a property that is used to store the store partner commission of the order.
26	OrderDetails	List<OrderDetail>	Private	This is a property that is used to store the list order details of the order.
27	Store	Store	Private	This is a property that is used to store the store of the order.
28	Partner	Partner	Private	This is a property that is used to store the parter of the order.
29	ShipperPayments	IEnumerable<ShipperPayment>	Private	This is a property that is used to store the list shipper payments of the order.
30	OrderDetails	IEnumerable<OrderDetail>	Private	This is a property that is used to store the list order details of the order.
31	OrderHistories	IEnumerable<OrderHistory>	Private	This is a property that is used to store the list order histories of the order.

Table 269: Order class

3.3.1.2.1.22. OrderDetail class

No	Attribute	Data Type	Visibility	Description
1	OrderDetailId	int	Private	This is a property that is used to store the id of the order detail.
2	ProductId	int	Private	This is a property that is used to store the product id of the order detail.
3	SellingPrice	decimal	Private	This is a property that is used to store the selling price of the order detail.
4	DiscountPrice	decimal	Private	This is a property that is used to store the discount price of the order detail.
5	Quantity	int	Private	This is a property that is used to store the quantity of the order detail.
6	Note	string	Private	This is a property that is used to store the note of the order detail.
7	ExtraOrderDetails	List<OrderDetail>	Private	This is a property that is used to store the list of extra order details of the master order detail.
8	MasterOrderDetailId	int	Private	This is a property that is used to store the id of master order detail of the order detail.
9	Order	Order	Private	This is a property that is used to store the order of the order detail.
10	Product	Product	Private	This is a property that is used to store the product of the order detail.
11	MasterOrderDetail	OrderDetail	Private	This is a property that is used to store the master order detail the order detail.

Table 270: OrderDetail class

3.3.1.2.1.23. Partner class

No	Attribute	Data Type	Visibility	Description
1	PartnerId	int	Private	This is a property that is used to store the partner id of the partner.
2	Name	string	Private	This is a property that is used to store the name of the partner.
3	Logo	string?	Private	This is a property that is used to store the logo of the partner.
4	WebUrl	string?	Private	This is a property that is used to store the web url of the partner.
5	TaxCommission	float	Private	This is a property that is used to store the tax of commission.
6	Status	int	Private	This is a property that is used to store the status of the partner.
7	Orders	IEnumerable<Order>	Private	This is a property that is used to store the this orders of the partner.
8	StorePartners	IEnumerable<StorePartner>	Private	This is a property that is used to store the list store partners of the partner.

Table 271: Partner class

3.3.1.2.1.24. StorePartner class

No	Attribute	Data Type	Visibility	Description
1	StoreId	int	Private	This is a property that is used to store the store id of the store partner.
2	PartnerId	string	Private	This is a property that is used to store the partner id of the store partner.
3	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the store partner.
4	UserName	string	Private	This is a property that is used to store the username of the store partner.
5	Password	string	Private	This is a property that is used to store the password of the store partner.
6	Commission	float	Private	This is a property that is used to store the commission of the store partner.
7	Status	int	Private	This is a property that is used to store the status of the store partner.
8	Partner	Partner	Private	This is a property that is used to store the partner of the store partner.
9	Store	Store	Private	This is a property that is used to store the store of the store partner.
10	PartnerProducts	IEnumerable<PartnerProduct>	Private	This is a property that is used to store the list partner products of the store partner.

Table 272: StorePartner class

3.3.1.2.1.25. OrderHistory class

No	Attribute	Data Type	Visibility	Description
1	OrderHistoryId	int	Private	This is a property that is used to store the order history id of the order history.
2	Image	string?	Private	This is a property that is used to store the image of the order history.
3	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the order history.
4	PartnerOrderStatus	string	Private	This is a property that is used to store the partner order status of the order history.
5	SystemStatus	string	Private	This is a property that is used to store the system status of the order history.
6	Order	Order	Private	This is a property that is used to store the order of the order history.

Table 273: OrderHistory class

3.3.1.2.1.26. PartnerProduct class

No	Attribute	Data Type	Visibility	Description
1	ProductId	int	Private	This is a property that is used to store the product id of the partner product.

2	PartnerId	int	Private	This is a property that is used to store the partner id of the partner product.
3	StoreId	int	Private	This is a property that is used to store the store id of the partner product.
4	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the partner product.
5	MappedDate	DateTime	Private	This is a property that is used to store the mapped date of the partner product.
6	ProductCode	string	Private	This is a property that is used to store the product code of the partner product.
7	Price	decimal	Private	This is a property that is used to store the price of the partner product.
8	UpdatedDate	DateTime	Private	This is a property that is used to store the updated date of the partner product.
9	Status	int	Private	This is a property that is used to store the status of the partner product.
10	Product	Product	Private	This is a property that is used to store the product of the partner product.
11	StorePartner	StorePartner	Private	This is a property that is used to store the store partner of the partner produc

Table 274: PartnerProduct class

3.3.1.2.27 Product class

No	Attribute	Data Type	Visibility	Description
1	ProductId	int	Private	This is a property that is used to store the product id of the product.
2	Code	string	Private	This is a property that is used to store the code of the product.
3	Name	string	Private	This is a property that is used to store the name of the product.
4	Type	string	Private	This is a property that is used to store the type of product.
5	DisplayOrder	int	Private	This is a property that is used to store the display order of the product.
6	Description	string	Private	This is a property that is used to store the description of the product.
7	Image	string	Private	This is a property that is used to store the image of the product.

8	SellingPrice	decimal	Private	This is a property that is used to store the selling price of the product.
9	DiscountPrice	decimal	Private	This is a property that is used to store the discount price of the product.
10	HistoricalPrice	decimal	Private	This is a property that is used to store the historical price of the product.
11	Size	string	Private	This is a property that is used to store the size of the product.
12	Status	int	Private	This is a property that is used to store the status of the product.
13	ParentProductId	int	Private	This is a property that is used to store the parent product id of the product.
14	Brand	Brand	Private	This is a property that is used to store the brand of the product.
15	Category	Category	Private	This is a property that is used to store the category of the product.
16	OrderDetails	Enumerable<OrderDetail>	Private	This is a property that is used to store the list order details of the product.
17	PartnerProducts	IEnumerable<PartnerProduct>	Private	This is a property that is used to store the list partner products of the product.
18	ParentProduct	Product	Private	This is a property that is used to store the parent product of the product.
19	ChildrenProducts	IEnumerable<PartnerProduct>	Private	This is a property that is used to store the list of children's products of the product.

Table 275: Product class

3.3.1.3. MBKC API

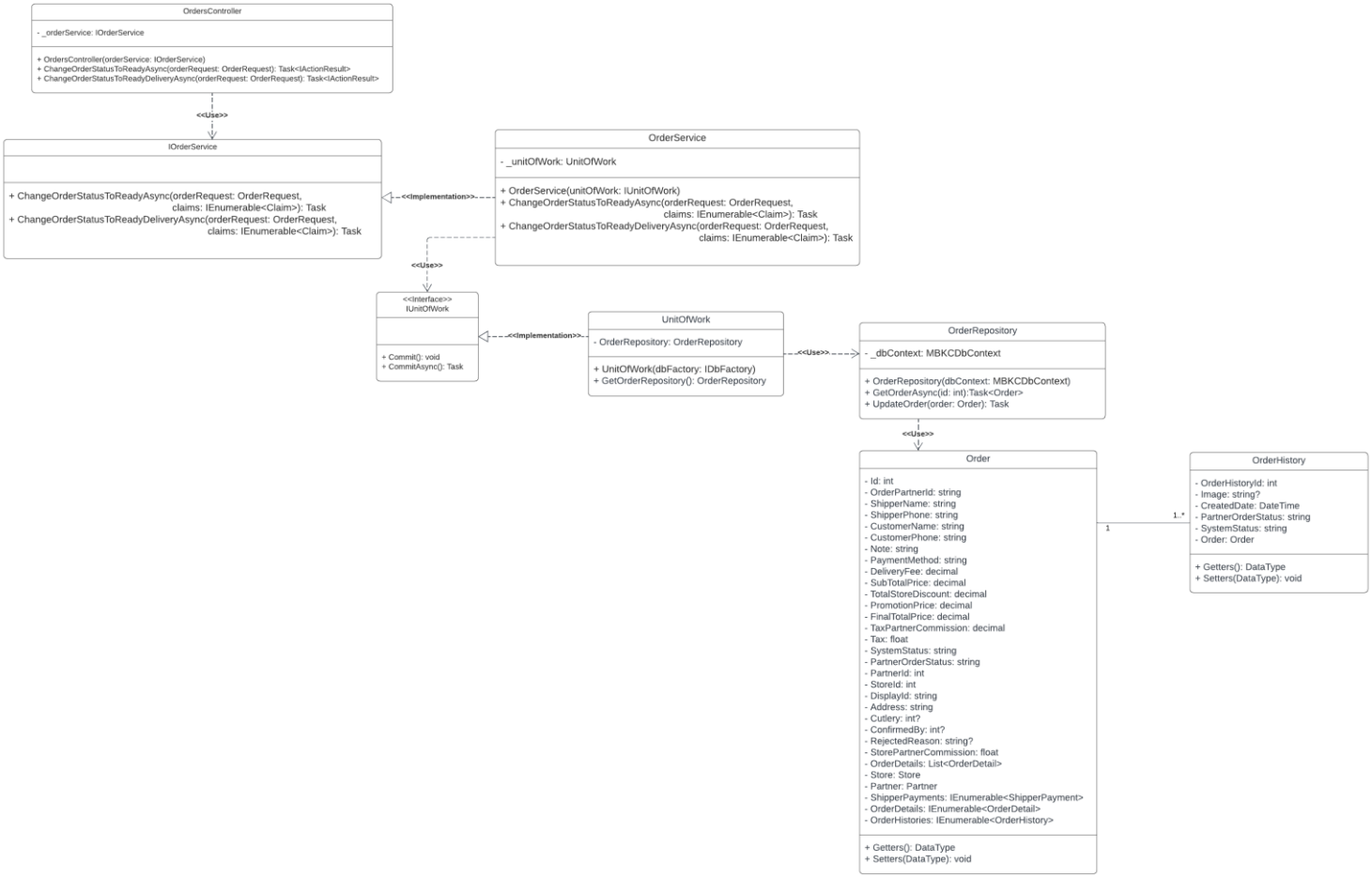


Figure 53: Confirming Orders From MBKC API

3.3.1.3.1. Class Specification

3.3.1.3.1.1. OrdersController class

No	Attribute	Data Type	Visibility	Description
1	_orderService	IOrderService	Private	This is a field that is used to store an instance of OrderService class.

Table 276: OrdersController class - Attributes

No	Method	Data Type	Visibility	Description
1	OrdersController		Public	This is a constructor of OrdersController class.
2	ChangeOrderStatusToReadyAsync	Task<IActionResult>	Public	This is a method that is used to update partner status of order from Preparing to Ready.
3	ChangeOrderStatusToReadyDeliveryAsync	Task<IActionResult>	Public	This is a method that is used to update the system status of order from In Store to Ready Delivery.

Table 277: OrdersController class - Methods

3.3.1.3.1.2 IOrderService interface

No	Method	Data Type	Visibility	Description
1	ChangeOrderStatusToReadyAsync	Task	Public	This is an abstract method that is used to update partner status of order from Preparing to Ready.
2	ChangeOrderStatusToReadyDeliveryAsync	Task	Public	This is an abstract method that is used to update the system status of order from In Store to Ready Delivery.

Table 278: IOrderService interface

3.3.1.3.1.3 OrderService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 279: OrderService class - Attributes

No	Method	Data Type	Visibility	Description
1	OrderService		Public	This is a constructor of OrderService class.
2	ChangeOrderStatusToReadyAsync	Task	Public	This is a method that is used to update partner status of order from Preapring to Ready.
3	ChangeOrderStatusToReadyDeliveryAsync	Task	Public	This is a method that is used to

				update system status of order from In Store to Ready Delivery.

Table 280: OrderService class - Methods

3.3.1.3.1.4. IUnitOfWork interface

No	Method	Data Type	Visibility	Description
1	Commit	void	Public	This is an abstract method that is used to execute non-query.
2	CommitAsync	Task	Public	This is an abstract method that is used to execute non-query with an asynchronous process.

Table 281: IUnitOfWork interface

3.3.1.3.1.5. UnitOfWork class

No	Attribute	Data Type	Visibility	Description
1	OrderRepository	OrderRepository	Private	This is a field that is used to store an instance of OrderRepository class.

Table 282: UnitOfWork class - Attributes

No	Method	Data Type	Visibility	Description
1	UnitOfWork		Public	This is a constructor of UnitOfWork Class.
1	GetOrderRepository	OrderRepository	Public	This is a method that is used to get an instance of OrderRepository class.

Table 283: UnitOfWork class - Methods

3.3.1.3.1.6. OrderRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 284: OrderRepository class

No	Method	Data Type	Visibility	Description
1	OrderRepository		Public	This is a constructor of OrderRepository class.
1	GetOrderAsync	Task<Order>	Public	This is a method that is used to get an order.
2	UpdateOrder	void	Public	This is a method that is used to update an order.

Table 285: OrderRepository class

3.3.1.3.1.7. Order class

No	Attribute	Data Type	Visibility	Description
1	Id	int	Private	This is a property that is used to store the id of the order.
2	OrderPartnerId	string	Private	This is a property that is used to store the order partner id of the order.
3	ShipperName	string	Private	This is a property that is used to store the shipper's name of the order.
4	ShipperPhone	string	Private	This is a property that is a property that is used to store the shipper phone of the order.
5	CustomerName	string	Private	This is a property that is used to store the customer's name of the order.
6	CustomerPhone	string	Private	This is a property that is used to store the customer phone of the order.
7	Note	string	Private	This is a property that is used to store the note of the order.
8	PaymentMethod	string	Private	This is a property that is used to store the payment method of the order.
9	DeliveryFee	decimal	Private	This is a property that is used to store the delivery fee of the order.
10	SubTotalPrice	decimal	Private	This is a property that is used to store the sub total price of the order.
11	TotalStoreDiscount	decimal	Private	This is a property that is used to store the total store discount of the order.
12	PromotionPrice	decimal	Private	This is a property that is used to store the promotion price of the order.
13	FinalTotalPrice	decimal	Private	This is a property that is used to store the final total price of the order.
14	TaxPartnerCommission	decimal	Private	This is a property that is used to store the tax partner commission of the order.
15	Tax	float	Private	This is a property that is used to store the tax of the order.
16	SystemStatus	string	Private	This is a property that is used to store the system status of the order.
17	PartnerOrderStatus	string	Private	This is a property that is used to

				store the partner order status of the order.
18	PartnerId	int	Private	This is a property that is used to store the partner id of the order.
19	StoreId	int	Private	This is a property that is used to store the store id of the order.
20	DisplayId	string	Private	This is a property that is used to store the display id of the order.
21	Address	string	Private	This is a property that is used to store the address of the order.
22	Cutlery	int?	Private	This is a property that is used to store the cutlery of the order.
23	ConfirmedBy	int?	Private	This is a property that is used to store the confirm by of the order.
24	RejectedReason	string?	Private	This is a property that is used to store the rejected reason of the order.
25	StorePartnerCommission	float	Private	This is a property that is used to store the store partner commission of the order.
26	OrderDetails	List<OrderDetail>	Private	This is a property that is used to store the list order details of the order.
27	Store	Store	Private	This is a property that is used to store the store of the order.
28	Partner	Partner	Private	This is a property that is used to store the parter of the order.
29	ShipperPayments	IEnumerable<ShipperPayment>	Private	This is a property that is used to store the list shipper payments of the order.
30	OrderDetails	IEnumerable<OrderDetail>	Private	This is a property that is used to store the list order details of the order.
31	OrderHistories	IEnumerable<OrderHistory>	Private	This is a property that is used to store the list order histories of the order.

Table 286: Order class

3.3.1.3.1.8. OrderHistory class

No	Attribute	Data Type	Visibility	Description
1	OrderHistoryId	int	Private	This is a property that is used to store the order history id of the order history.
2	Image	string?	Private	This is a property that is used to store the image of the order history.
3	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the order

				history.
4	PartnerOrderStatus	string	Private	This is a property that is used to store the partner order status of the order history.
5	SystemStatus	string	Private	This is a property that is used to store the system status of the order history.
6	Order	Order	Private	This is a property that is used to store the order of the order history.

Table 287: OrderHistory class

3.3.2. Activity Diagram

3.3.2.1. Get and Parse Order Data from Partner

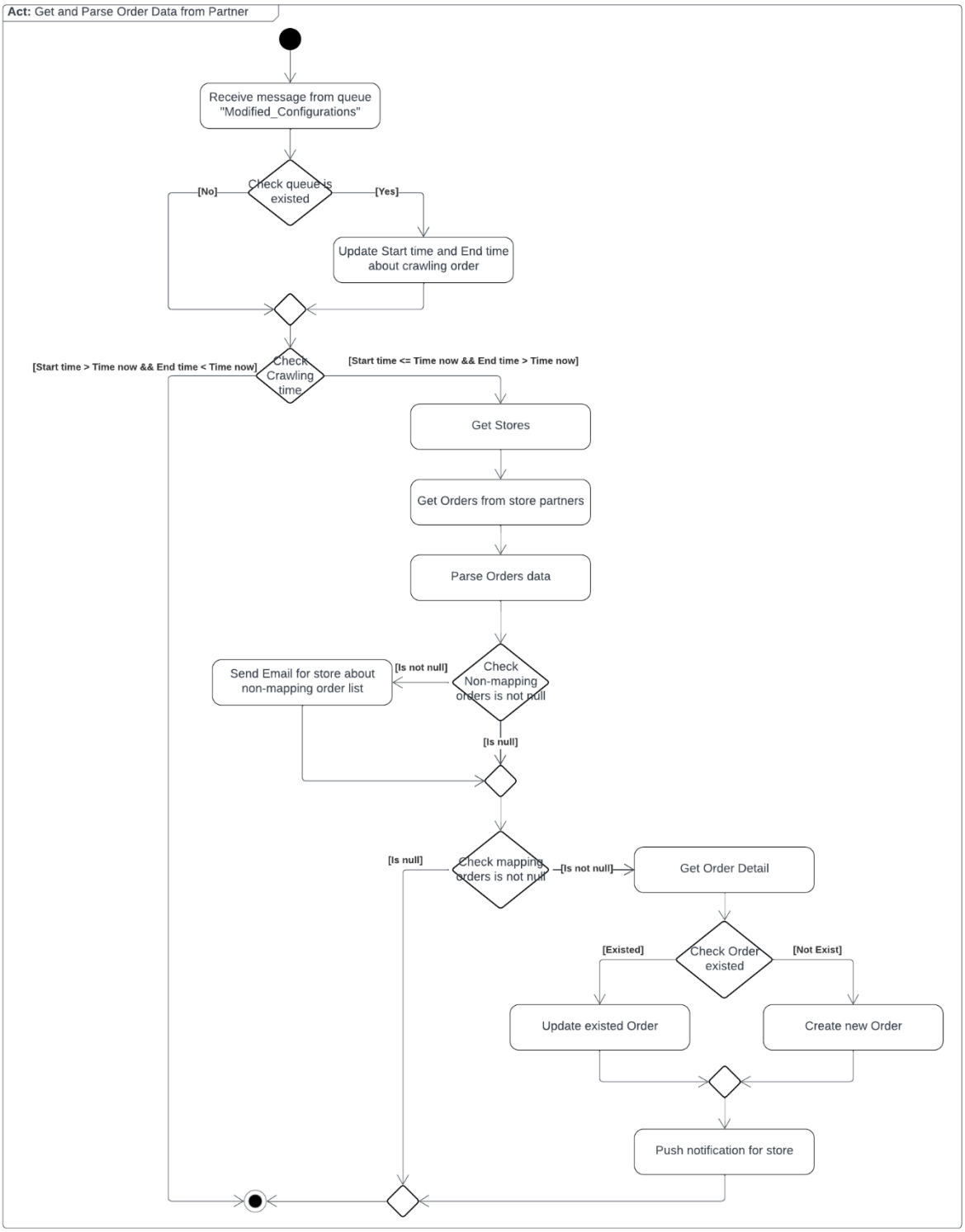


Figure 54: Get and Parse Order Data from Partner

3.3.2.2. Confirm Finished of Preparing Orders

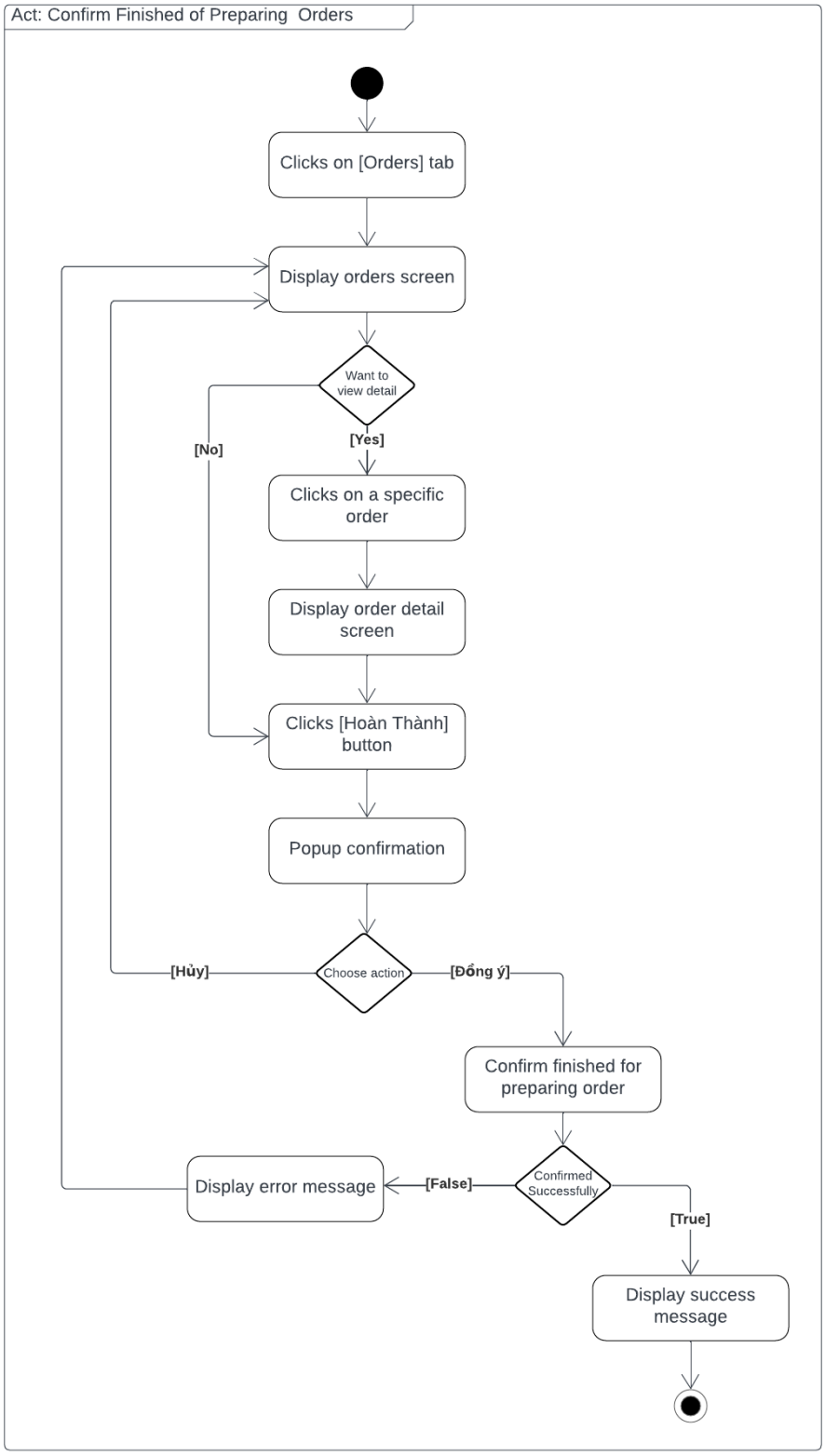


Figure 55: Confirm Finished of Preparing Orders

3.3.2.3. Confirm Status Order to Ready Delivery

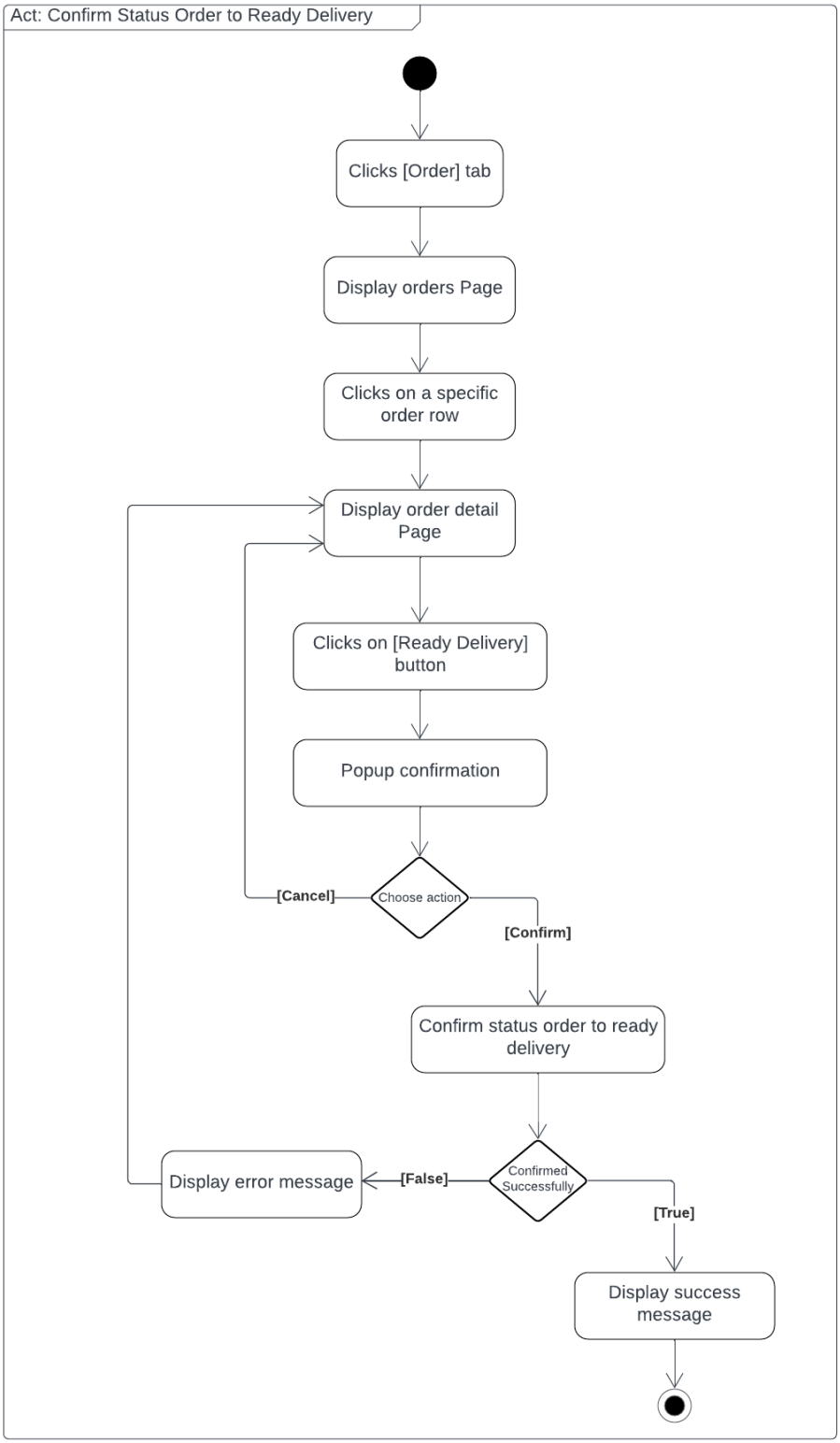


Figure 56: Confirm Status Order to Ready Delivery

3.3.3. Sequence Diagram

3.3.3.1. Get and Parse Order Data from Partner

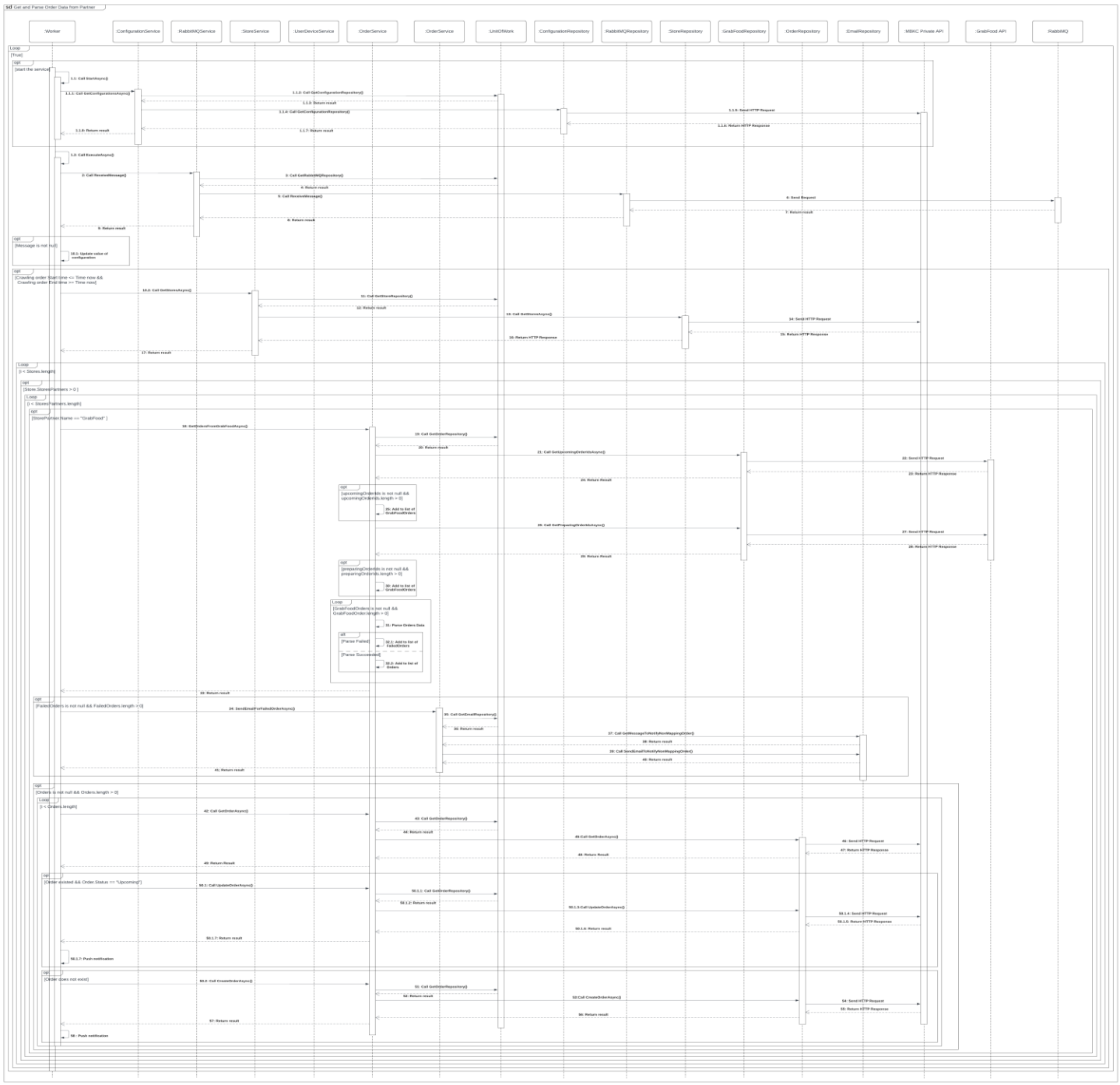


Figure 57: Get and Parse Order Data from Partner

3.3.3.2. Create New Order

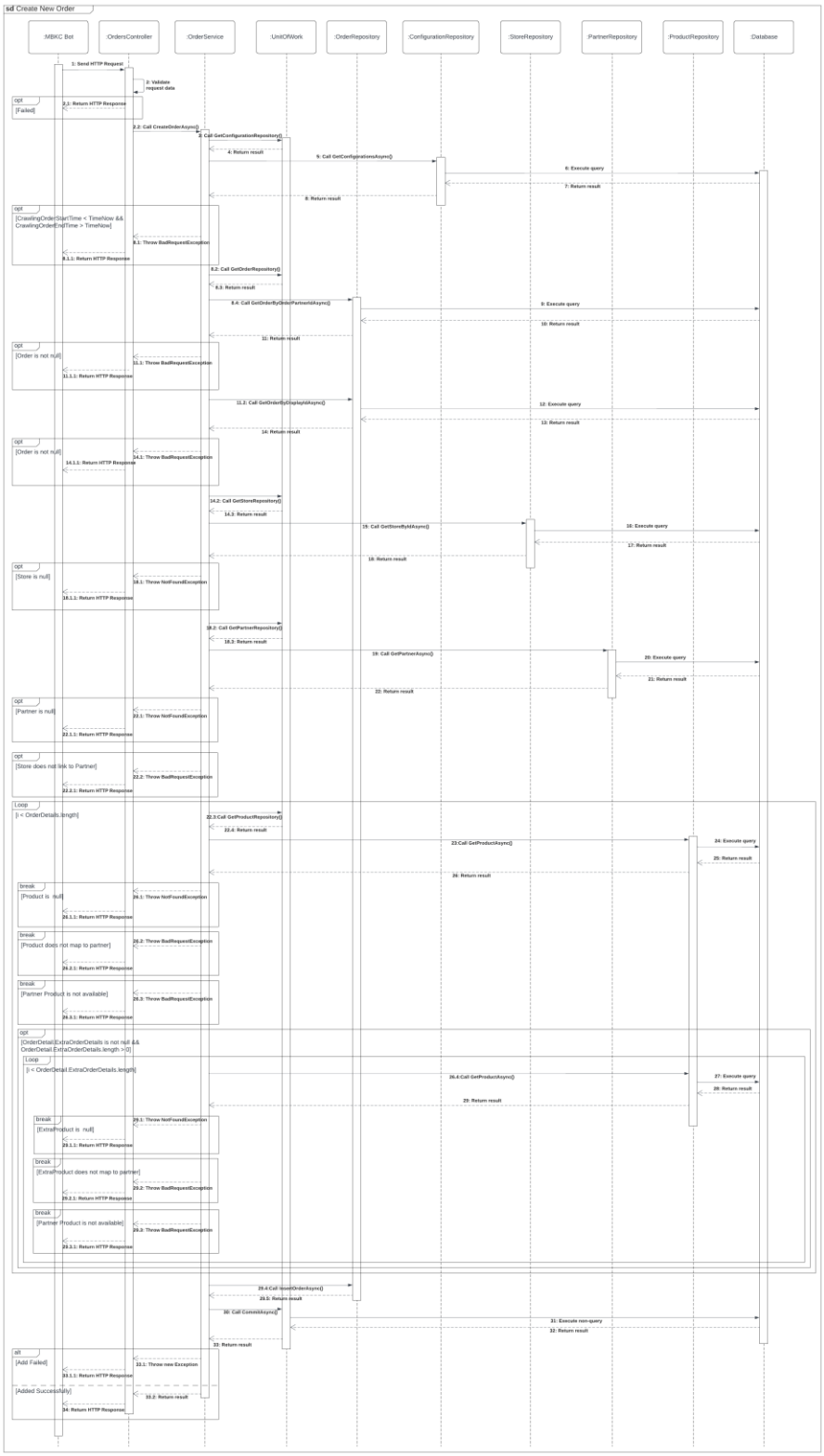


Figure 58: Create New Order

3.3.2.3. Update Existed Order

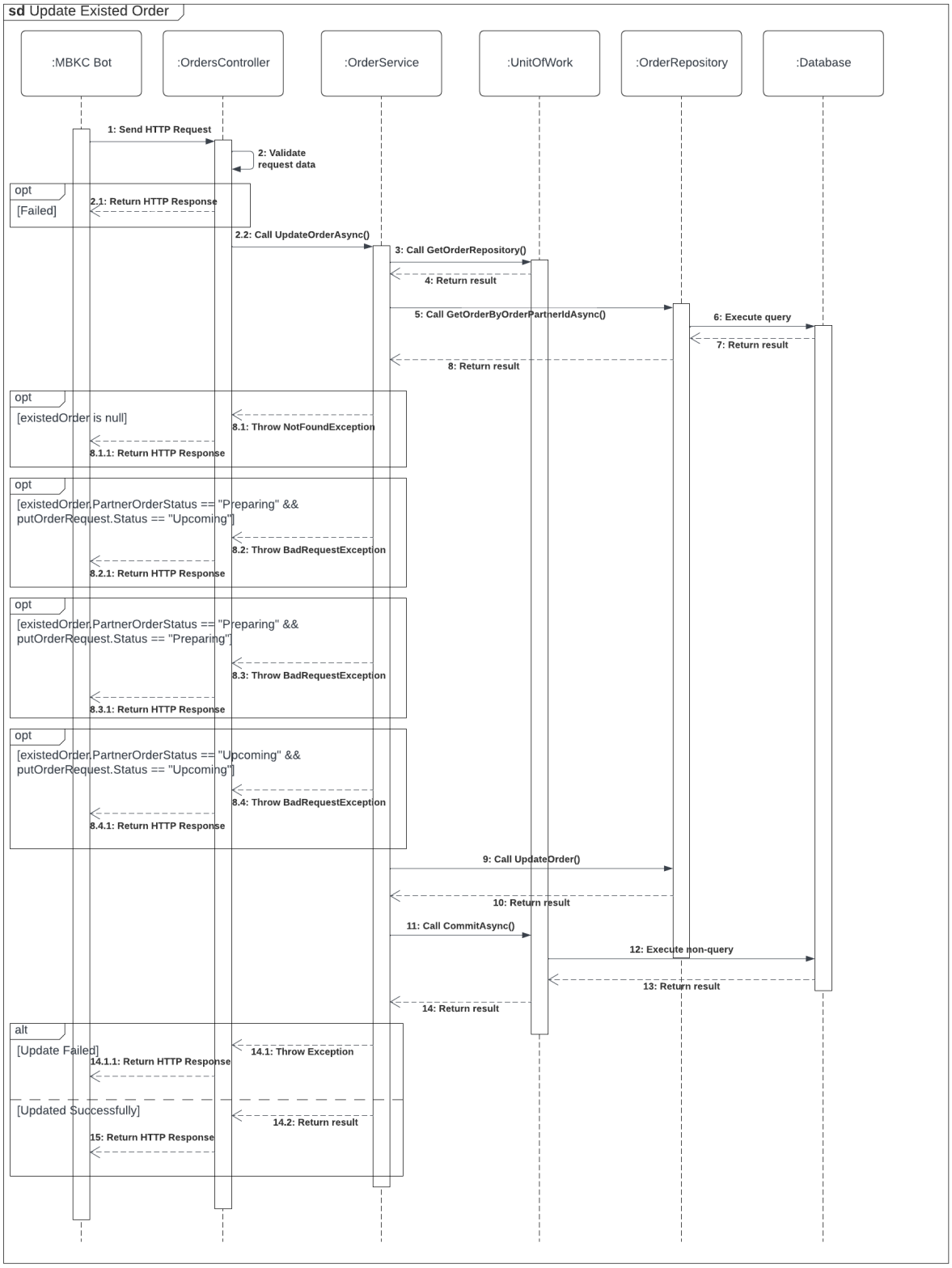


Figure 59: Update Existed Order

3.3.3.4. Confirm Finished of Preparing Orders

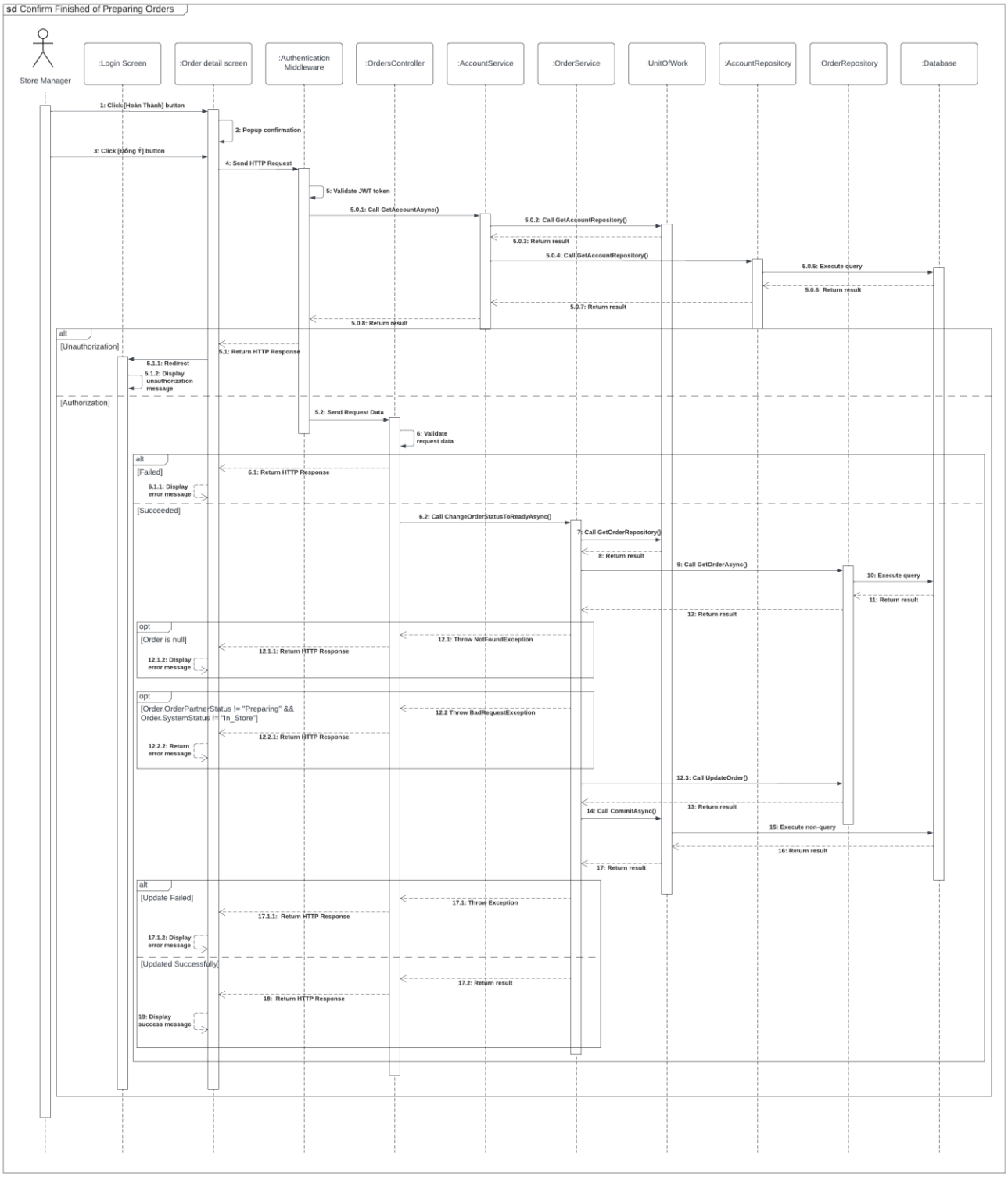


Figure 60: Confirm Finished of Preparing Orders

3.3.3.5. Confirm Status Order to Ready Delivery

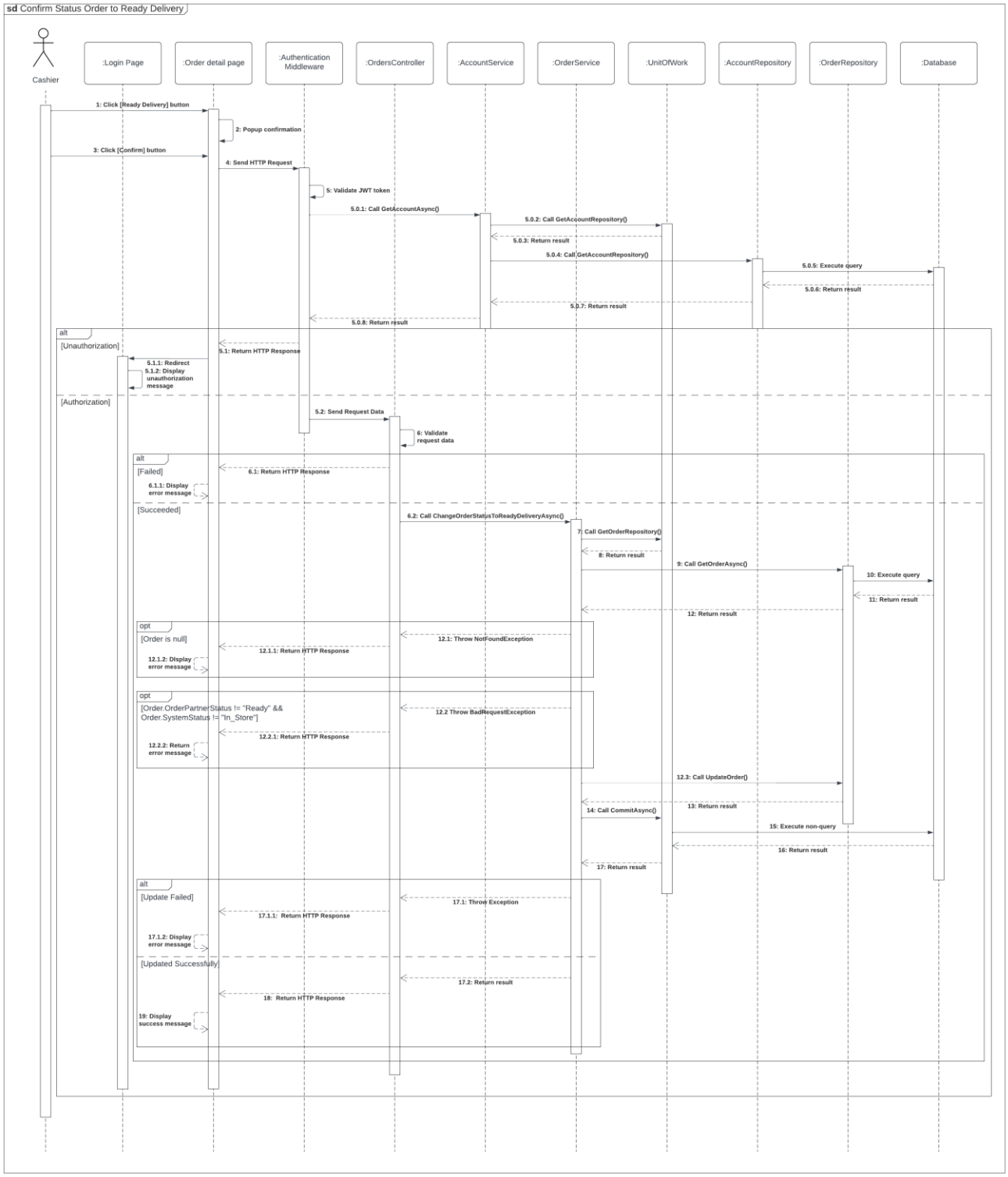


Figure 61: Confirm Status Order to Ready Delivery

3.3.4. State Machine Diagram

3.3.4.1. Order State Machine Diagram

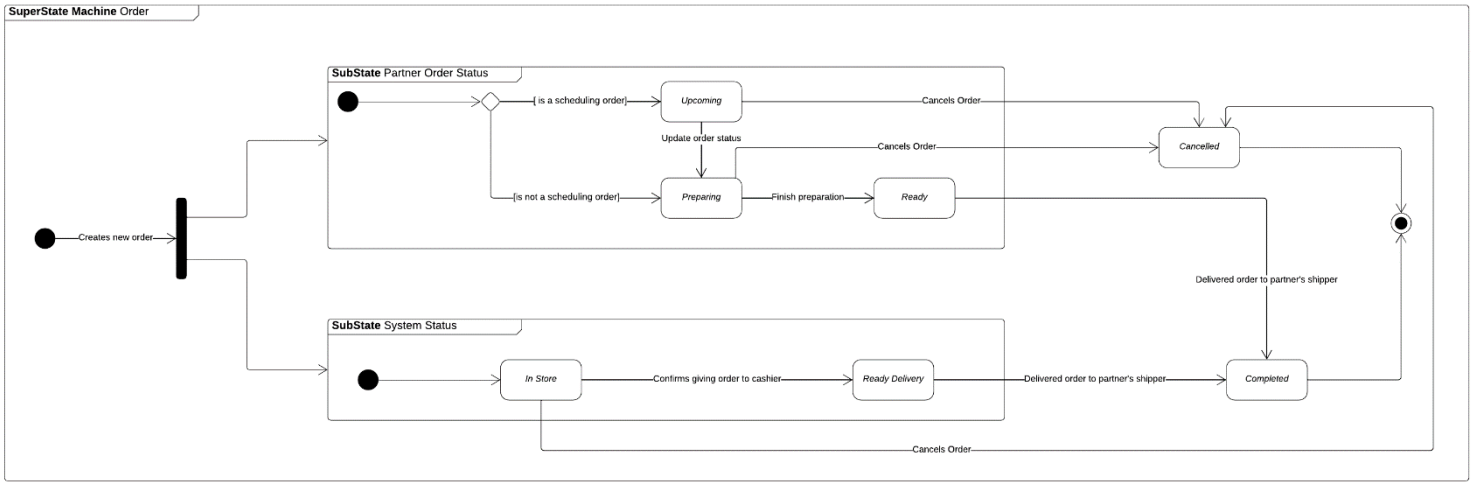


Figure 62: Order State Machine Diagram

3.4. Transferring and Withdrawing Money from a Wallet

3.4.1. Class Diagram

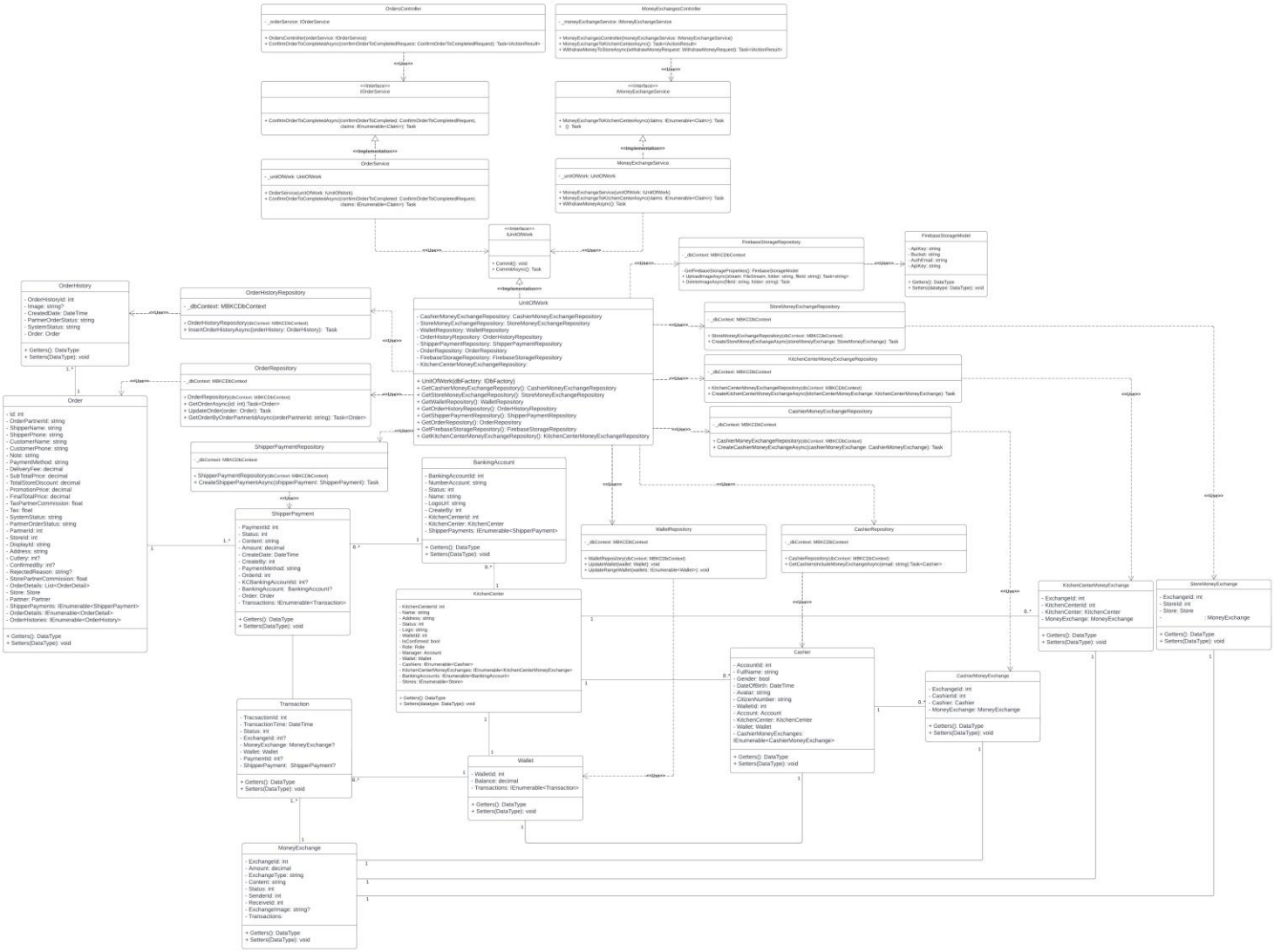


Figure 63: Transferring and Withdrawing Money

3.4.1.1. Class Specifications

3.4.1.1.1. OrdersController class

No	Attribute	Data Type	Visibility	Description
1	_orderService	IOrderService	Private	This is a field that is used to store an instance of OrderService class.

Table 288: OrdersController class - Attributes

No	Method	Data Type	Visibility	Description
1	OrdersController		Public	This is a constructor of OrdersController class.
2	ConfirmOrderToCompletedAsync	Task<IActionResult>	Public	This is an action method that is used to update status of order to completed.

Table 289: OrdersController class - Methods

3.4.1.1.2. MoneyExchangesController class

No	Attribute	Data Type	Visibility	Description
1	_moneyExchangeService	IMoneyExchangeService	Private	This is a field that is used to store an instance of MoneyExchangeService class.

Table 290: MoneyExchangesController class - Attributes

No	Method	Data Type	Visibility	Description
1	MoneyExchangesController		Public	This is a constructor of MoneyExchangesControl ler class.
2	MoneyExchangeToKitchenCenterAsync	Task<IActionResult>	Public	This is an action method that is used to transfer money from the cashier's wallet to the kitchen centen wallet.
3	WithdrawMoneyToStoreAsync	Task<IActionResult>	Public	This is an action method that is used to withdraw money from the store's wallet.

Table 291: MoneyExchangesController class - Methods

3.4.1.1.3. IOrderService interface

No	Method	Data Type	Visibility	Description
1	ConfirmOrderToCompletedAsync	Task	Public	This is an abstract method that is used to update status of order to completed.

Table 292: IOrderService interface

3.4.1.1.4. IMoneyExchangeService interface

No	Method	Data Type	Visibility	Description
1	MoneyExchangeToKitchenCenterAsync	Task	Public	This is an abstract method that is used to transfer money from the cashier's wallet to the kitchen centen wallet.
2	WithdrawMoneyAsync	Task	Public	This is an abstract method that is used to withdraw money from the store's wallet.

Table 293: IMoneyExchangeService interface

3.4.1.1.5. OrderService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 294: OrderService class - Attributes

No	Method	Data Type	Visibility	Description
1	OrderService		Public	This is a constructor of OrderService class.
2	ConfirmOrderToCompletedAsync	Task	Public	This is a method that is used to update status of order to completed.

Table 295: OrderService class - Methods

3.4.1.1.6. MoneyExchangeService class

No	Attribute	Data Type	Visibility	Description
1	_unitOfWork	UnitOfWork	Private	This is a field that is used to store an instance of UnitOfWork class.

Table 296: MoneyExchangeService class - Attributes

No	Method	Data Type	Visibility	Description
1	MoneyExchangeService		Public	This is a constructor of MoneyExchangeService class.
2	MoneyExchangeToKitchenCenterAsync	Task	Public	This is a method that is used to transfer money from the cashier's wallet to the kitchen centen wallet.
3	WithdrawMoneyAsync	Task	Public	This is a method that is used to withdraw money from the store's wallet.

Table 297: MoneyExchangeService class - Methods

3.4.1.1.7. IUnitOfWork interface

No	Method	Data Type	Visibility	Description
1	Commit	void	Public	This is an abstract method that is used to execute non-query.
2	CommitAsync	Task	Public	This is an abstract method that is used to execute non-query with an asynchronous process.

Table 298: IUnitOfWork interface

3.4.1.1.8. UnitOfWork class

No	Attribute	Data Type	Visibility	Description
1	CashierMoneyExchangeRepository	CashierMoneyExchangeRepository	Private	This is a property that is used to store instance of class.
2	StoreMoneyExchangeRepository	StoreMoneyExchangeRepository	Private	This is a property that is used to store instance of class.
3	WalletRepository	WalletRepository	Private	This is a property that is used to store instance of class.
4	OrderHistoryRepository	OrderHistoryRepository	Private	This is a property that is used to store instance of class.
5	ShipperPaymentRepository	ShipperPaymentRepository	Private	This is a property that is used to store instance of class.
6	OrderRepository	OrderRepository	Private	This is a property that is used to store instance of class.
7	FirebaseStorageRepository	FirebaseStorageRepository	Private	This is a property that is used to store instance of class.

8	KitchenCenterMoneyExchangeRepository	KitchenCenterMoneyExchangeRepository	Private	This is a property that is used to store instance of class.

Table 299: UnitOfWork class - Attributes

No	Method	Data Type	Visibility	Description
1	UnitOfWork		Public	This is a constructor of class.
2	GetCashierMoneyExchangeRepository	CashierMoneyExchangeRepository	Public	This is a method that is used to get an instance of class.
3	GetStoreMoneyExchangeRepository	StoreMoneyExchangeRepository	Public	This is a method that is used to get an instance of class.
4	GetWalletRepository	WalletRepository	Public	This is a method that is used to get an instance of class.
5	GetOrderHistoryRepository	OrderHistoryRepository	Public	This is a method that is used to get an instance of class.
6	GetShipperPaymentRepository	ShipperPaymentRepository	Public	This is a method that is used to get an instance of class.
7	GetOrderRepository	OrderRepository	Public	This is a method that is used to get an instance of class.

8	GetFirebaseStorageRepository	FirebaseStorageRepository	Public	This is a method that is used to get an instance of class.
9	GetKitchenCenterMoneyExchangeRepository	KitchenCenterMoneyExchangeRepository	Public	This is a method that is used to get an instance of class.

Table 300: UnitOfWork class - Methods

3.4.1.1.9. OrderHistoryRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 301: OrderHistoryRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	OrderHistoryRepository			This is a constructor of OrderHistoryRepository class.
2	InsertOrderHistoryAsync	Task	Public	This is a method that is used to add an order history.

Table 302: OrderHistoryRepository class - Methods

3.4.1.1.10. OrderRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 303: OrderRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	OrderRepository		Public	This is a constructor of OrderRepository class.
2	GetOrderAsync	Task<Order>	Public	This is a method that is used to get an order.
3	UpdateOrder	Task	Public	This is a method that is used to update an order.
4	GetOrderByOrderPartnerIdAsync	Task<Order>	Public	This is a method that is used to get an order by order partner id.

Table 304: OrderRepository class - Methods

3.4.1.1.11. FirebaseStorageModel class

No	Attribute	Data Type	Visibility	Description
1	ApiKey	string	Private	This is a property that is used to store the API key of the Firebase Storage.
2	Bucket	string	Private	This is a property that is used to store the bucket of the Firebase Storage.
3	AuthEmail	string	Private	This is a property that is used to store the authentication email of the Firebase Storage.
4	AuthPassword	string	Private	This is a property that is used to store the authentication password of the Firebase Storage.

Table 305: FirebaseStorageModel class

3.4.1.1.12. FirebaseStorageRepository class

No	Method	Data Type	Visibility	Description
1	GetFirebaseStorageProperties	FirebaseStorageModel	Private	This is a method that is used to get basic information to be able to connect to the Firebase Storage.
2	UploadImageAsync	Task<string>	Public	This is a method that is used to push an image to the Firebase storage.
3	DeleteImageAsync	Task	Public	This is a method used to delete an image in the Firebase storage.

Table 306: FirebaseStorageRepository class

3.4.1.1.13. StoreMoneyExchangeRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 307: StoreMoneyExchangeRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	StoreMoneyExchangeRepository		Public	This is a constructor of StoreMoneyExchangeRepository class.
2	CreateStoreMoneyExchangeAsync	Task	Public	This is a method that is used to create money exchange for store.

Table 308: StoreMoneyExchangeRepository class - Methods

3.4.1.1.14. KitchenCenterMoneyExchangeRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 309: KitchenCenterMoneyExchangeRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	KitchenCenterMoneyExchangeRepository		Public	This is a constructor of KitchenCenterMoneyExchange Repository class.
2	CreateKitchenCenterMoneyExchangeAsync	Task	Public	This is a method that is used to create money exchange for kitchen center.

Table 310: KitchenCenterMoneyExchangeRepository class - Methods

3.4.1.1.15 CashierMoneyExchangeRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 311: CashierMoneyExchangeRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	CashierMoneyExchangeRepository		Public	This is a constructor of CashierMoneyExchangeRepository class.
2	CreateCashierMoneyExchangeAsync	Task	Public	This is a method that is used to create money exchange for cashier.

Table 312: CashierMoneyExchangeRepository class - Methods

3.4.1.1.16. KitchenCenterMoneyExchange class

No	Attribute	Data Type	Visibility	Description
1	ExchangeId	int	Private	This is a property that is used to store the exchange id of the kitchen center money exchange.
2	KitchenCenterId	int	Private	This is a property that is used to store the kitchen center id of the kitchen center money exchange.
3	KitchenCenter	KitchenCenter	Private	This is a property that is used to store the kitchen center of the kitchen center money exchange.
4	MoneyExchange	MoneyExchange	Private	This is a property that is used to store the money exchange of the kitchen center money exchange.

Table 313: KitchenCenterMoneyExchange class

3.4.1.1.17. StoreMoneyExchange class

No	Attribute	Data Type	Visibility	Description
1	ExchangeId	int	Private	This is a property that ld is used to store the exchange id of the store money exchange.
2	StoreId	int	Private	This is a property that is used to store the store id of the store money exchange.
3	Store	Store	Private	This is a property that is used to store the store of the store money exchange.
4	MoneyExchange	MoneyExchange	Private	This is a property that is used to store the money exchange of the store money exchange.

Table 314: StoreMoneyExchange class

3.4.1.1.18. WalletRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 315: WalletRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	WalletRepository		Public	This is a constructor of WalletRepository class.
2	UpdateWallet	void	Public	This is a method that is used to update a wallet.
3	UpdateRangeWallet	void	Public	This is a method that is used to update list wallets.

Table 316: WalletRepository class - Methods

3.4.1.1.19. CashierRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 317: CashierRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	CashierRepository		Public	This is a constructor of CashierRepository class.
2	GetCashiersIncludeMoneyExchangeAsync	Task<Cashier>	Public	This is a method that is used to get a cashier.

Table 318: CashierRepository class - Methods

3.4.1.1.20. CashierMoneyExchange class

No	Attribute	Data Type	Visibility	Description
1	ExchangeId	int	Private	This is a property that is used to store the exchange id of the cashier money exchange.
2	CashierId	int	Private	This is a property that is used to store the cashier id of the cashier money exchange.
3	Cashier	Cashier	Private	This is a property that is used to store the cashier of the cashier money exchange.
4	MoneyExchange	MoneyExchange	Private	This is a property that is used to store the money exchange of the cashier money exchange.

Table 319: CashierMoneyExchange class

3.4.1.1.21. Cashier class

No	Attribute	Data Type	Visibility	Description
1	AccountId	int	Private	This is a property that is used to store the account id of the cashier.
2	FullName	int	Private	This is a property that is used to store the full name of the cashier.
3	Gender	bool	Private	This is a property that is used to store the gender of the cashier.
4	DateOfBirth	DateTime	Private	This is a property that is used to store the date of birth of the cashier.
5	Avatar	string	Private	This is a property that is used to store the avatar of the cashier.
6	CitizenNumber	string	Private	This is a property that is used to store the citizen number of the cashier.
7	WalletId	int	Private	This is a property that is used to store the wallet id of the cashier.

8	Account	Account	Private	This is a property that is a property that is used to store the account of the cashier.
9	KitchenCenter	KitchenCenter	Private	This is a property that is used to store the kitchen center of the cashier.
10	Wallet	Wallet	Private	This is a property that is used to store the wallet of the cashier.
11	CashierMoneyExchanges	IEnumerable<CashierMoneyExchange>	Private	This is a property that is used to store the cashier money exchange of the cashier.

Table 320: Cashier class

3.4.1.1.22. ShipperPaymentRepository class

No	Attribute	Data Type	Visibility	Description
1	_dbContext	MBKCDbContext	Private	This is a field that is used to store an instance of MBKCDbContext class.

Table 321: ShipperPaymentRepository class - Attributes

No	Method	Data Type	Visibility	Description
1	ShipperPaymentRepository		Public	This is a constructor of ShipperPaymentRepository class.
2	CreateShipperPaymentAsync	Task	Public	This is a method that is used to create shipper payment for order.

Table 322: ShipperPaymentRepository class – Methods

3.4.1.1.23. BankingAccount class

No	Attribute	Data Type	Visibility	Description
1	BankingAccountId	int	Private	This is a property that is used to store the banking account id of the banking account.
2	NumberAccount	string	Private	This is a property that is used to store the number account of the banking account.
3	Status	int	Private	This is a property that is used to store the status of the banking account.
4	Name	string	Private	This is a property that is used to store the name of the banking account.

5	LogoUrl	string	Private	This is a property that is used to store the logo url of the banking account.
6	CreateBy	int	Private	This is a property that is used to store the created by of the banking account.
7	KitchenCenterId	int	Private	This is a property that is used to store the kitchen center id of the banking account.
8	KitchenCenter	KitchenCenter	Private	This is a property that is used to store the kitchen center of the banking account.
9	ShipperPayments	IEnumerable<ShipperPayment>	Private	This is a property that is used to store the list shipper payments of the banking account.

Table 323: BankingAccount class

3.4.1.1.24. Order class

No	Attribute	Data Type	Visibility	Description
1	Id	int	Private	This is a property that is used to store the id of the order.
2	OrderPartnerId	string	Private	This is a property that is used to store the order partner id of the order.
3	ShipperName	string	Private	This is a property that is used to store the shipper's name of the order.
4	ShipperPhone	string	Private	This is a property that is a property that is used to store the shipper phone of the order.
5	CustomerName	string	Private	This is a property that is used to store the customer's name of the order.
6	CustomerPhone	string	Private	This is a property that is used to store the customer phone of the order.
7	Note	string	Private	This is a property that is used to store the note of the order.
8	PaymentMethod	string	Private	This is a property that is used to store the payment method of the order.
9	DeliveryFee	decimal	Private	This is a property that is used to store the delivery fee of the order.
10	SubTotalPrice	decimal	Private	This is a property that is used to store the sub total

				price of the order.
11	TotalStoreDiscount	decimal	Private	This is a property that is used to store the total store discount of the order.
12	PromotionPrice	decimal	Private	This is a property that is used to store the promotion price of the order.
13	FinalTotalPrice	decimal	Private	This is a property that is used to store the final total price of the order.
14	TaxPartnerCommission	decimal	Private	This is a property that is used to store the tax partner commission of the order.
15	Tax	float	Private	This is a property that is used to store the tax of the order.
16	SystemStatus	string	Private	This is a property that is used to store the system status of the order.
17	PartnerOrderStatus	string	Private	This is a property that is used to store the partner order status of the order.
18	PartnerId	int	Private	This is a property that is used to store the partner id of the order.
19	StoreId	int	Private	This is a property that is used to store the store id of the order.
20	DisplayId	string	Private	This is a property that is used to store the display id of the order.
21	Address	string	Private	This is a property that is used to store the address of the order.
22	Cutlery	int?	Private	This is a property that is used to store the cutlery of the order.
23	ConfirmedBy	int?	Private	This is a property that is used to store the confirm by of the order.
24	RejectedReason	string?	Private	This is a property that is used to store the rejected reason of the order.
25	StorePartnerCommission	float	Private	This is a property that is used to store the store partner commission of the order.
26	OrderDetails	List<OrderDetail>	Private	This is a property that is used to store the list order

				details of the order.
27	Store	Store	Private	This is a property that is used to store the store of the order.
28	Partner	Partner	Private	This is a property that is used to store the parter of the order.
29	ShipperPayments	IEnumerable<ShipperPayment>	Private	This is a property that is used to store the list shipper payments of the order.
30	OrderDetails	IEnumerable<OrderDetail>	Private	This is a property that is used to store the list order details of the order.
31	OrderHistories	IEnumerable<OrderHistory>	Private	This is a property that is used to store the list order histories of the order.

Table 324: Order class

3.4.1.1.25. OrderHistory class

No	Attribute	Data Type	Visibility	Description
1	OrderHistoryId	int	Private	This is a property that is used to store the order history id of the order history.
2	Image	string?	Private	This is a property that is used to store the image of the order history.
3	CreatedDate	DateTime	Private	This is a property that is used to store the created date of the order history.
4	PartnerOrderStatus	string	Private	This is a property that is used to store the partner order status of the order history.
5	SystemStatus	string	Private	This is a property that is used to store the system status of the order history.
6	Order	Order	Private	This is a property that is used to store the order of the order history.

Table 325: OrderHistory class

3.4.1.1.26. Wallet class

No	Attribute	Data Type	Visibility	Description
1	WalletId	int	Private	This is a property that is used to store the wallet id of the wallet.
2	Balance	decimal	Private	This is a property that is used to store the balance of the wallet.
3	Transactions	IEnumerable<Transaction>	Private	This is a property that is used to store the list transactions of the

				wallet.

Table 326: Wallet class

3.4.1.1.27. KitchenCenter class

No	Attribute	Data Type	Visibility	Description
1	KitchenCenterId	int	Private	This is a property that is used to store the kitchen center id of the kitchen center.
2	Name	string	Private	This is a property that is used to store the name of the kitchen center.
3	Address	string	Private	This is a property that is used to store the address of the kitchen center.
4	Status	int	Private	This is a property that is used to store the status of the kitchen center.
5	Logo	string	Private	This is a property that is used to store the logo of the kitchen center.
6	WalletId	string	Private	This is a property that is used to store the wallet id of the kitchen center.
7	IsConfirmed	bool	Private	This is a property that is used to store the is confirm of the kitchen center.

8	Role	Role	Private	This is a property that is used to store the role of the kitchen center.
9	Manager	Account	Private	This is a property that is used to store the manager of the kitchen center.
10	Wallet	Wallet	Private	This is a property that is used to store the wallet of the kitchen center.
11	Cashiers	IEnumerable<Cashier>	Private	This is a property that is used to store the list cashiers of the kitchen center.
12	KitchenCenterMoneyExchanges	IEnumerabe<KitchenCenterMoneyExchanges>	Private	This is a property that is used to store the list money exchanges of the kitchen center.
13	BankingAccounts	IEnumerable<BankingAccount>	Private	This is a property that is a property that is used to store the list banking accounts of the kitchen center.
14	Stores	IEnumerable<Store>	Private	This is a property that is used to store the list stores of the kitchen center.

Table 327: KitchenCenter class

3.4.1.1.28. ShipperPayment class

No	Attribute	Data Type	Visibility	Description
1	PaymentId	int	Private	This is a property that is used to

				store the payment id of the shipper payment.
2	Status	int	Private	This is a property that is used to store the status of the shipper payment.
3	Content	string	Private	This is a property that is used to store the content of the shipper payment.
4	Amount	decimal	Private	This is a property that is used to store the amount of the shipper payment.
5	CreateDate	Datetime	Private	This is a property that is used to store the create date of the shipper payment.
6	CreateBy	int	Private	This is a property that is used to store the create by of the shipper payment.
7	PaymentMethod	string	Private	This is a property that is used to store the payment method of the shipper payment.
8	OrderId	int	Private	This is a property that is used to store the order id of the shipper payment.
9	KCBankingAccountId	int?	Private	This is a property that is used to store the kc banking account id of the shipper payment.
10	BankingAccount	BankingAccount	Private	This is a property that is used to store the banking account of the shipper payment.
11	Order	Order	Private	This is a property that is used to store the order of the shipper payment.
12	Transactions	IEnumerable<Transaction>	Private	This is a property that is used to store the list transactions of the shipper payment.

Table 328: ShipperPayment class

3.4.1.1.29. Transaction class

No	Attribute	Data Type	Visibility	Description
1	TracsactionId	int	Private	This is a property that is used to store the transaction id of the transaction.
2	TransactionTime	DateTime	Private	This is a property that is used to store the transaction time of the transaction.
3	Status	int	Private	This is a property that is used to store the status of the transaction.
4	ExchangeId	int?	Private	This is a property that is used to

				store the exchange id of the transaction.
5	MoneyExchange	MoneyExchange	Private	This is a property that is used to store the money exchange of the transaction.
6	Wallet	Wallet	Private	This is a property that is used to store the wallet of the transaction.
9	PaymentId	int?	Private	This is a property that is used to store the payment id of the transaction.
10	ShipperPayment	ShipperPayment	Private	This is a property that is used to store the shipper payment of the transaction.

Table 329: Transaction class

3.4.1.1.30. MoneyExchange class

No	Attribute	Data Type	Visibility	Description
1	ExchangeId	int	Private	This is a property that is used to store the exchange id of the money exchange.
2	Amount	decimal	Private	This is a property that is used to store the amount of the money exchange.
3	ExchangeType	string	Private	This is a property that is used to store the exchange type of the money exchange.
4	Content	string	Private	This is a property that is used to store the content of the money exchange.
5	Status	int	Private	This is a property that is used to store the status of the money exchange.
6	SenderId	int	Private	This is a property that is used to store the sender id of the money exchange.
7	ReceiveId	int	Private	This is a property that is used to store the receive id of the money exchange.
8	ExchangeImage	string?	Private	This is a property that is used to store the exchange image of the money exchange.
9	Transactions	IEnumerable<Transaction>	Private	This is a property that is used to store the list transactions of the money exchange.

Table 330: MoneyExchange class

3.4.2. Activity Diagram

3.4.2.1. Create Shipper Payment

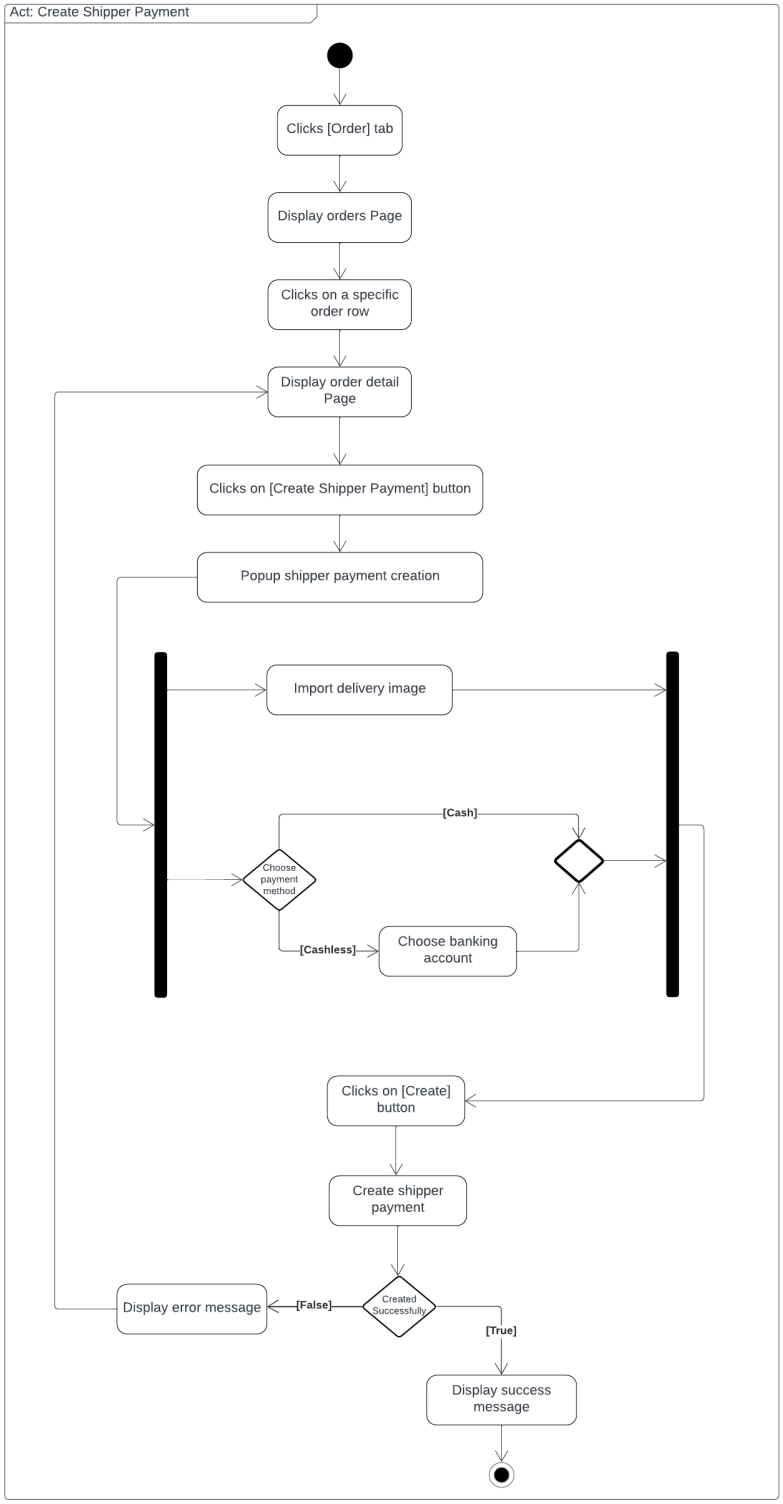


Figure 64: Create Shipper Payment

3.4.2.2. Transfer Money from Cashier's Wallet to Kitchen Center

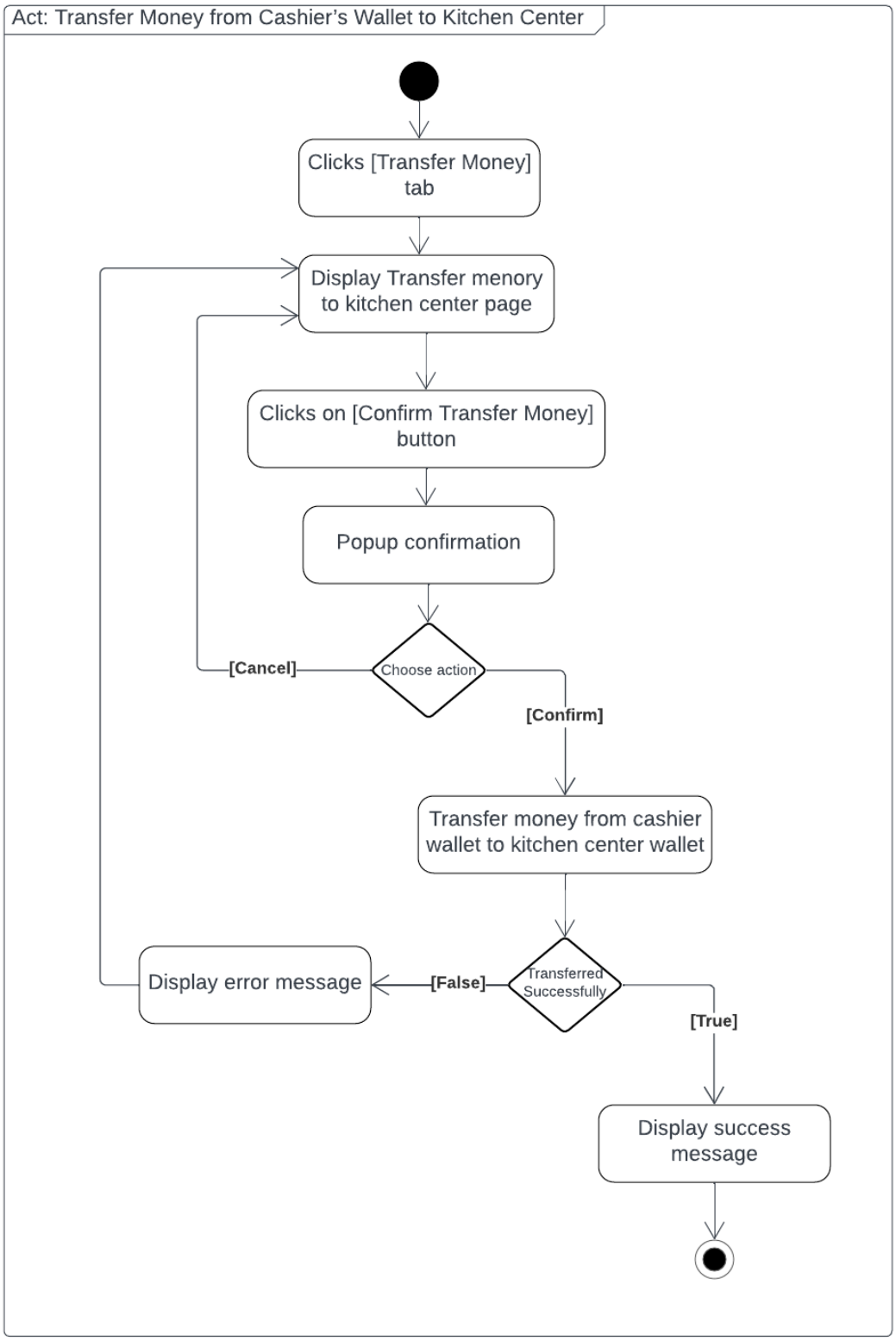


Figure 65: Transfer Money from Cashier's Wallet to Kitchen Center

3.4.2.3. Withdraw money for Store

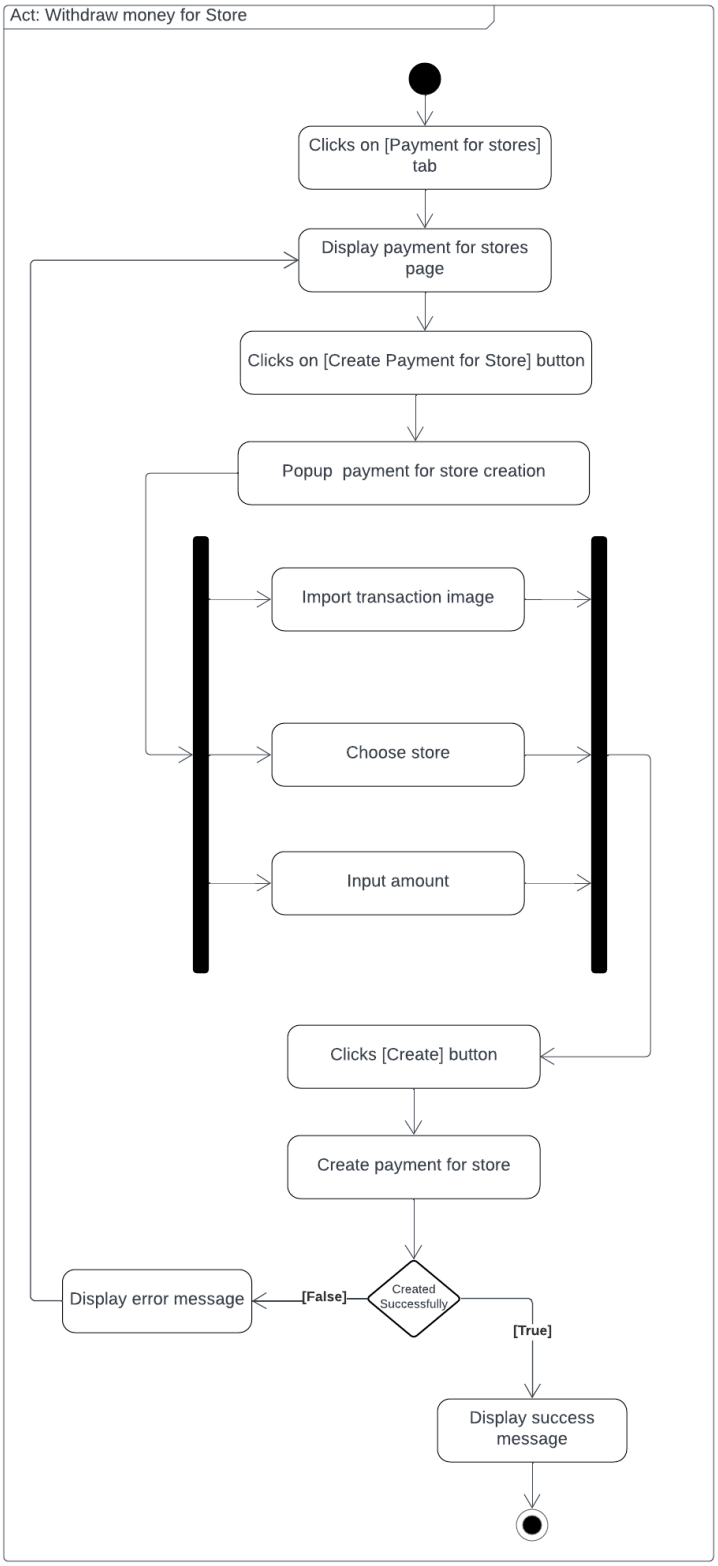


Figure 66: Withdraw money for Store

3.4.3. Sequence Diagram

3.4.3.1. Create Shipper Payment

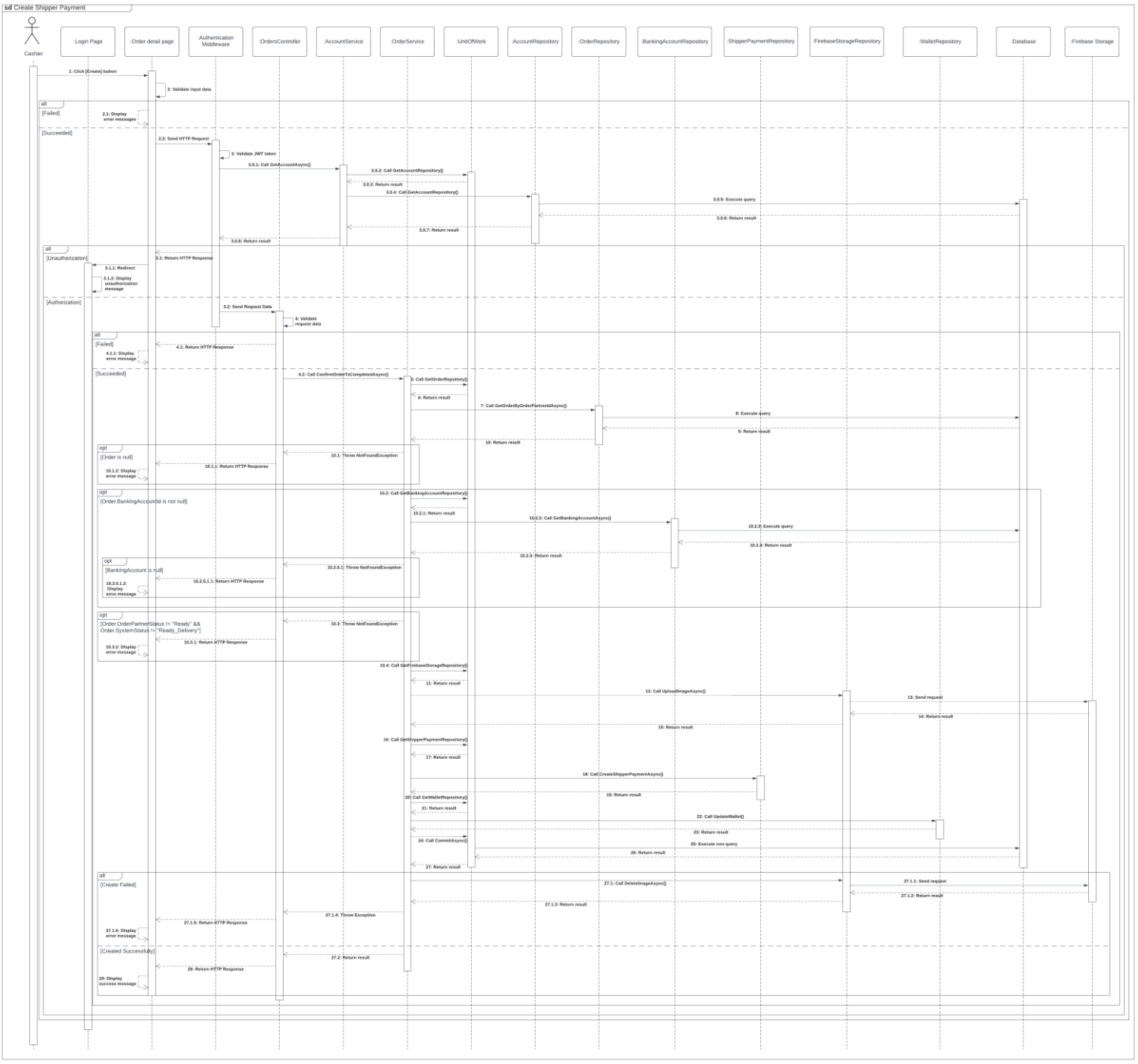


Figure 67: Create Shipper Payment

3.4.3.2. Transfer Money from Cashier's wallet to Kitchen Center

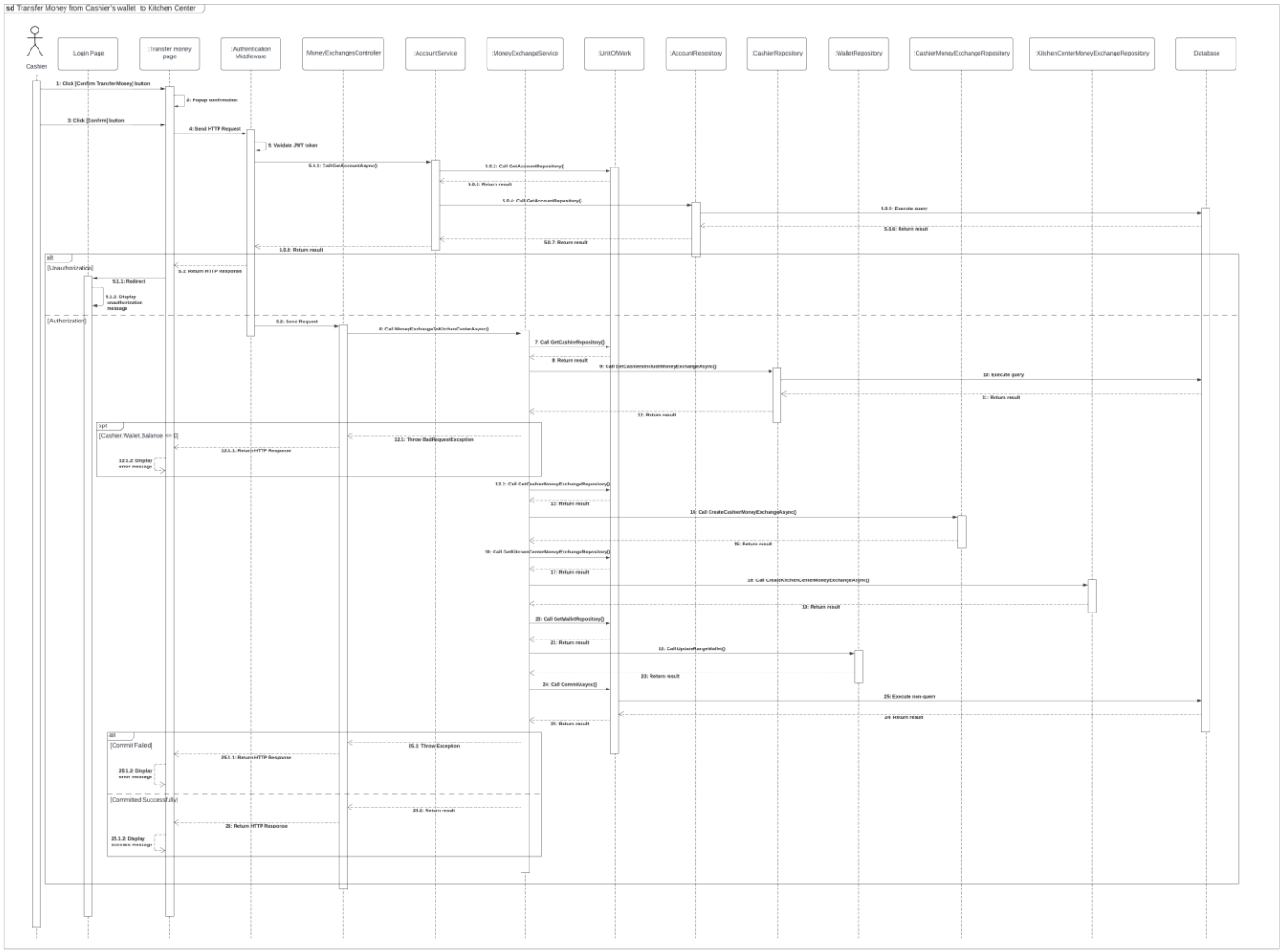


Figure 68: Transfer Money from Cashier's wallet to Kitchen Center

3.4.3.3. Withdraw money for Store

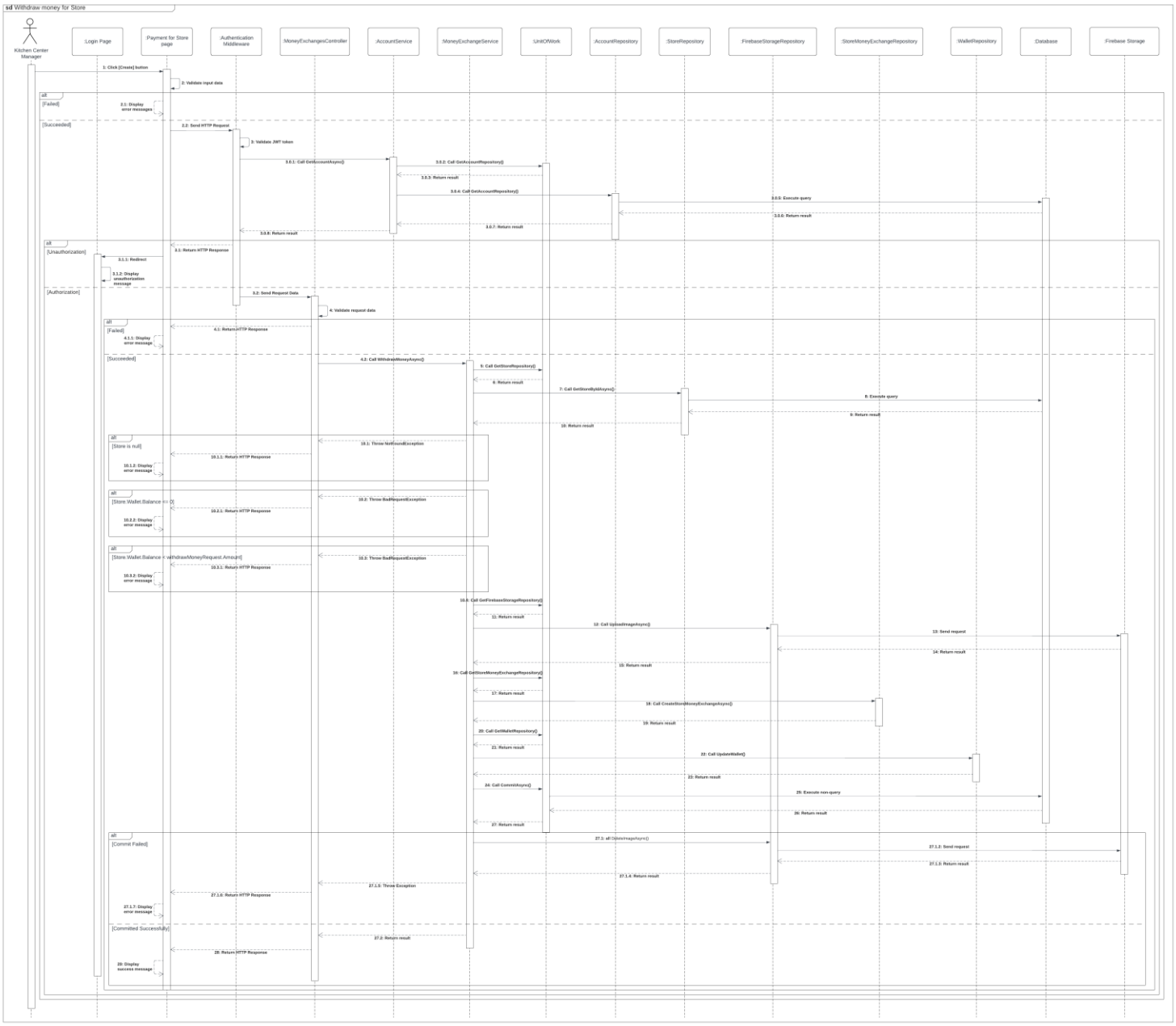


Figure 69: Withdraw money for Store

V. Software Testing Documentation

1. Overall Description

1.1. Test Model

In the MBKC Project, we apply Agile testing for the following purposes:

- Quickly adapt to projects that changed rapidly and continuously.
- Test cases are ready for execution when finishing the implementation.
- Detecting errors in the system early allows for quick correction.

1.2. Test Levels

- Integration testing: Ensure every module works appropriately together in the system.
- System testing: Ensure that the MBKC system meets the design specification and other requirements.

1.3. Testing Types

- Functional test: Ensure that all functions in the system meet its requirements.
- API test: Ensure that all APIs in the system process and return required data correctly.
- User Interface test: Ensure that all elements of the UI are correctly presented.

2. Test Plan

2.1. Test Stages

Type of Test	Stage of Test		
	Integration	System	Acceptance
Functional test	x	x	x
API test	x	x	x
User interface test		x	x

Table 331: Test Stages

2.2. Test Environments

Purpose	Tool	Provider	Version
User interface testing on web application	Google Chrome	Google Chrome	119.0.6045.200
User interface testing on mobile application	Android emulator Physical mobile device	LDPlayer Android Device	9.0
API test, Integration test	Postman	Postman	10.20.0
API test	Swagger	SmartBear Software	3.1.0

Table 332: Test Environments

2.2. Human Resources

Worker/Doer	Role	Specific Responsibilities/Comments
Thái Quốc Toàn	Leader	Planning, assigning test functions to members and verifying test results.
Trần Phú Sơn	Member	Execute test and report.
Lê Hồng Thành	Member	Execute test and report.
Võ Vỹ Khang	Member	Execute test and report.
Lê Xuân Bách	Member	Execute test and report.

Table 333: Human Resources

2.3. Test Milestones

Milestone Task	Due Date
Create test plan	15/10/2023
Create test cases for Web application	21/10/2023
Create test cases for APIs	20/11/2023
Create test cases for Mobile application	29/11/2023
Create report for testing	05/12/2023

Table 334: Test Milestones

3. Test Cases and Test Reports

Test Cases & Test Report: GFA23SE12-Test-Report.xlsx

VI. Release Package & User Guides

1. Deliverable Package

No	Deliverable Item	Sub-Items	Type	Version
Code package				

1	Source code	• MBKC web application
		• MBKC Back-end Server
		  ○ MBKC API
		  ○ MBKC Private API
		• MBKC Bot
		• MBKC mobile application
			New	1.0

2	Database Script(s)	• CloudKitchen-Script-Schema.sql • CloudKitchen-Script-Data.sql	New	1.0
Documents				
5	File Report Document	Report7_Final Project Report.docx	New	1.0
6	Test Cases Document	GFA23SE12-Test-Report.xlsx	New	1.0
7	Slide	GFA23SE12- Capstone Project Fall 2023.pptx	New	1.0

Table 335: Deliverable Package

2. Installation Guides

2.1. System Requirements

Component	Minimum	Recommended
CPU	At least 2 (GHz) x64-bit	Core i5 or i7
Memory	At least 2GB RAM	At least 4GB RAM
Storage Space	10GB HDD	20GB HDD
Network	Internet Connection	Internet Connection

Table 336: System Requirements

2.2. Installation Instruction

2.2.1. Environment Setup

2.2.1.1 Mobile application

- Download and install Visual Studio Code 1.84: Link
- Install extensions “Dart” and “Flutter”:

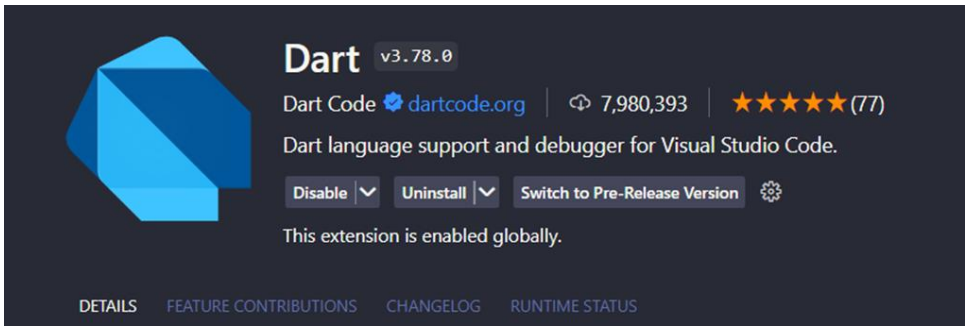


Figure 70: Install Dart Extension

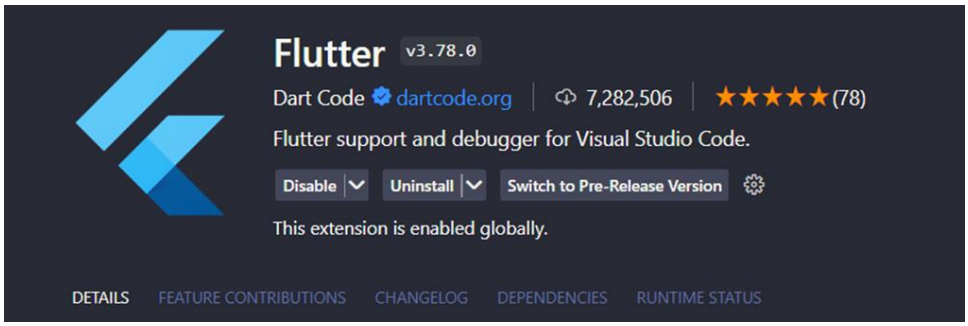


Figure 71: Install Fullter Extension

- Download and install Flutter SDK 3.16.2: Link
- Download and install LD Player 9: Link

2.2.1.2 Web application

Step 1: Install Visual Studio Code:

- Download the Visual Studio Code setup file.
- After the download is complete, run the VSCodeUserSetup.exe file.
- Click Next to proceed with the installation. Then, agree to the terms of use.
- Choose the installation location (it's recommended to keep the default) and click Next.
- In the subsequent steps, continue clicking Next until the installation is complete.

Step 2: Install npm through Visual Studio Code or you can install npm via NodeJS: Link.

2.2.1.3 Backend server

- Download and install Visual Studio 2022: Link
- Check “ASP.NET and web development”, “.NET desktop and development” and “.NET Core cross-platform development” options like the image below when install:

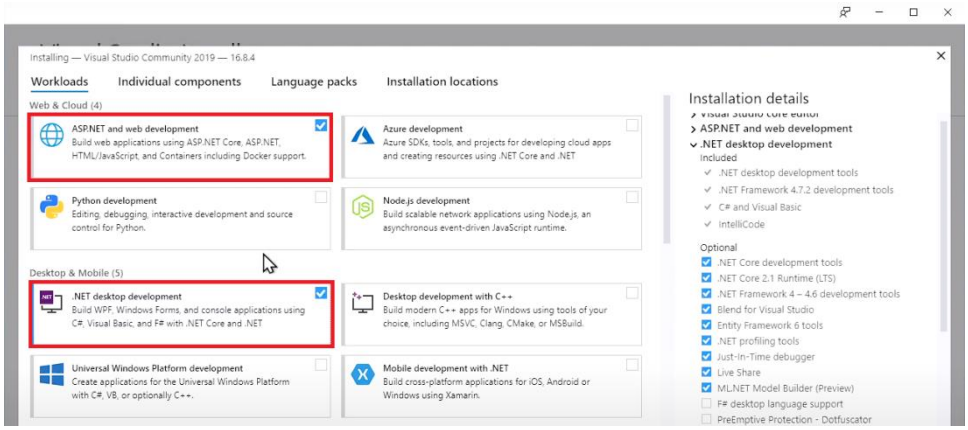


Figure 72: Check “ASP.NET and web development”, “.NET desktop and development”

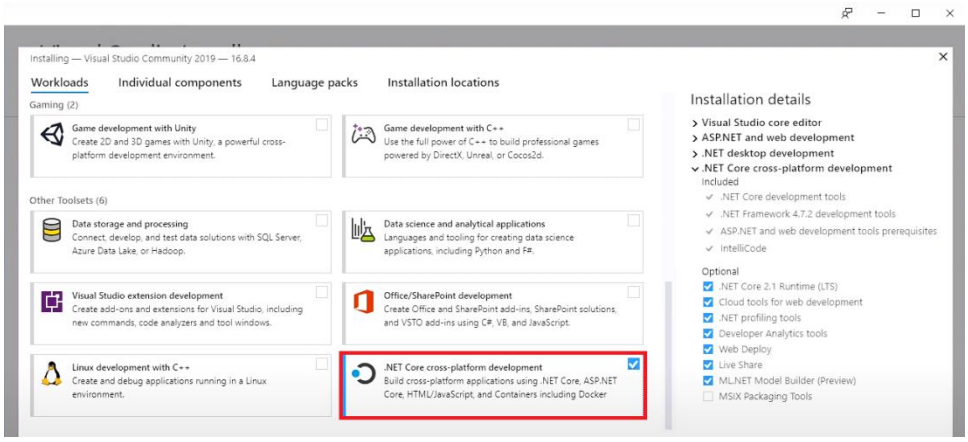


Figure 73: Check “.NET Core cross-platform development”

2.2.2. Run Backend Server

Step 1: Clone the repository or extract “MBKC-System-V2-main.zip” and open folder with Visual Studio 2022.

Step 2: Press "F5" or button as shown below to launch project "MBKC.API".

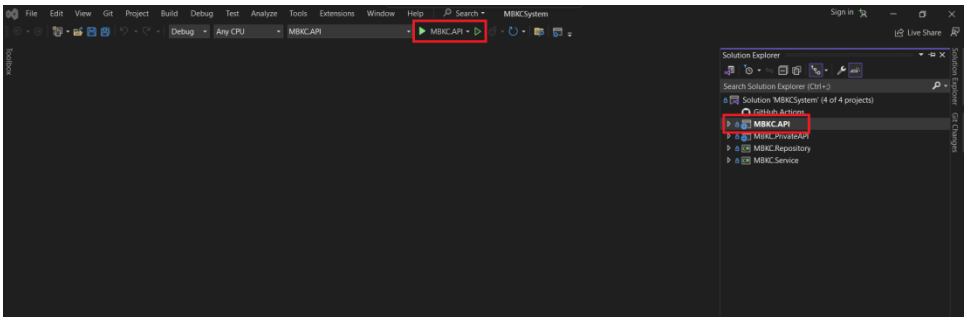


Figure 74: Launch project "MBKC.API"

If you want to launch MBKC.Private API:

Step 3: Select "MBKC.PrivateAPI" and right click, then select "Set as Startup Project" as shown below.

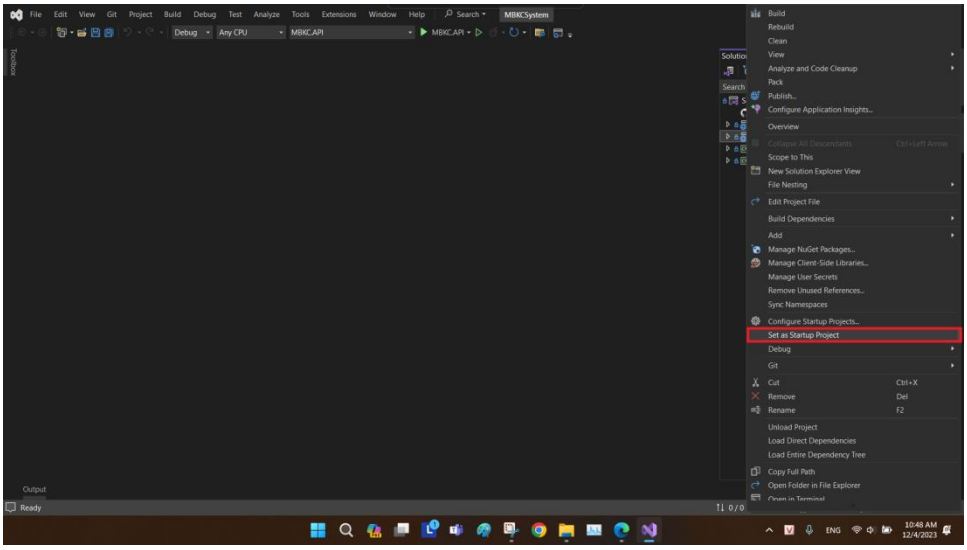


Figure 75: Select "MBKC.PrivateAPI"

Step 4: Press "F5" or button as shown below to launch project "MBKC.PrivateAPI".

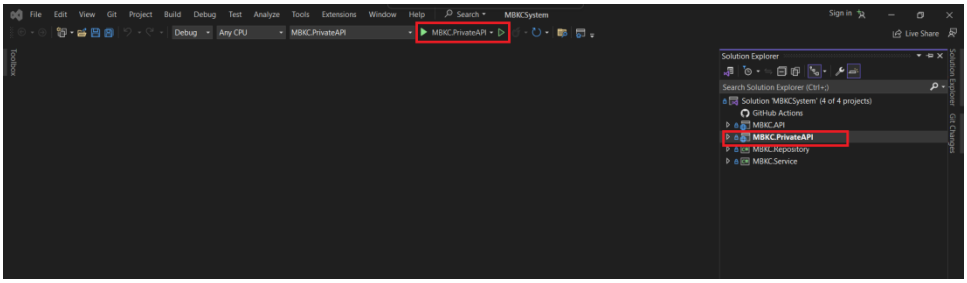


Figure 76: Launch project "MBKC.PrivateAPI"

2.2.3. Run Frontend web application

Step 1: Clone the repository or extract “MBKC_WebApplication.zip” and open folder with Visual Studio Code.

Step 2: Create a “.env” file in root folder and fill up all required keys:

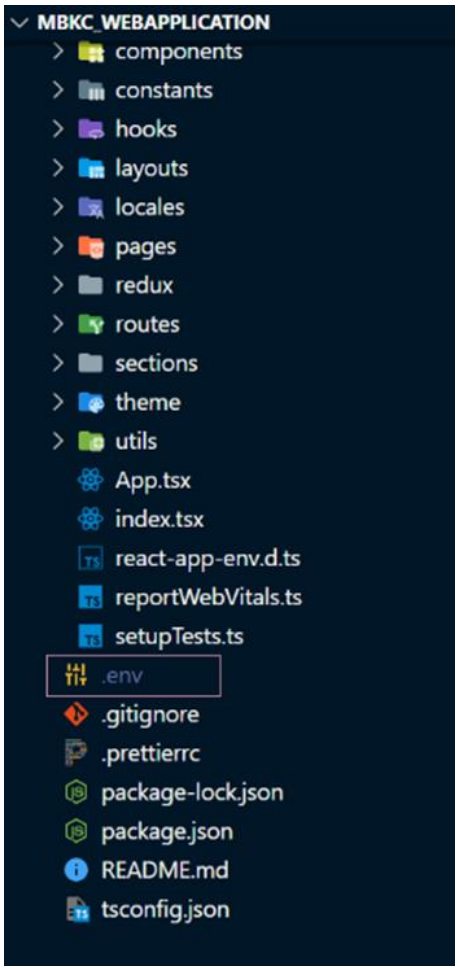


Figure 77: Create a “.env” file

REACT_APP_TITLE= MBKC Cloud Kitchen

REACT_APP_ENVIROMENT=development

REACT_APP_BASE_URL=https://mbkc.azurewebsites.net/api/v1

Step 3: Open terminal and enter npm install:

```
PS D:\Capstone Project\source-code\MBKC_WebApplication> npm install
```

Figure 78: Enter npm install

Step 4: In terminal, enter npm start then your browser will open and redirect to localhost:<your-port>

```
PS D:\Capstone Project\source-code\MBKC_WebApplication> npm start
```

Figure 79: Enter npm start

2.2.4. Run Frontend mobile application

Step 1: Clone the repository (private) or extract “MBKC_MobileApplication-V2-main.zip”.

Step 2: Open the code folder with Visual Studio Code.

Step 3: Press “Ctrl + ~” and enter command: “flutter pub get”.

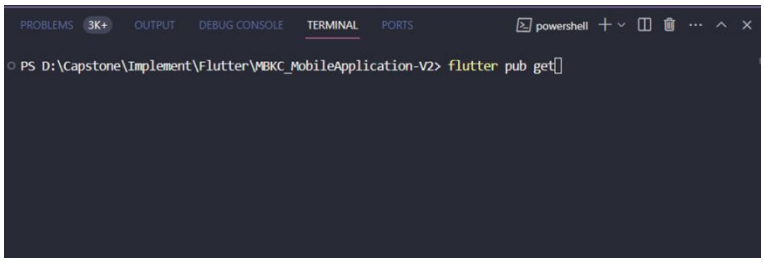


Figure 80: Enter command “Flutter pib get”

Step 4: Open LDPlayer 9.

Step 5: Click the "Setting" icon, in the "Advanced" section, click "Mobile mode" and then select the option "540 x 960(dpi 240)".

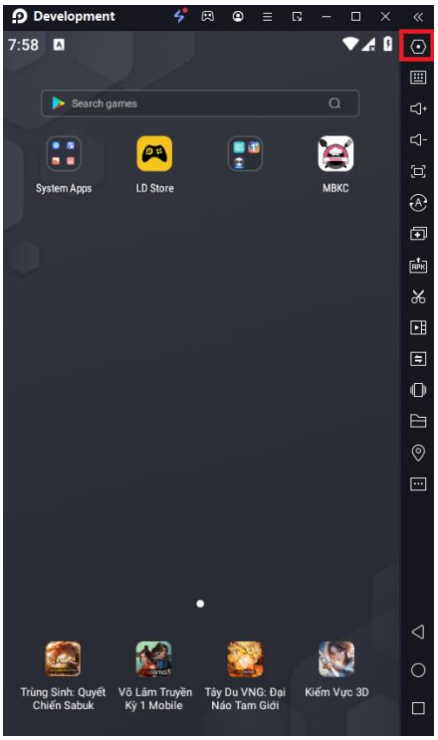


Figure 81: Setiing Mobile Device

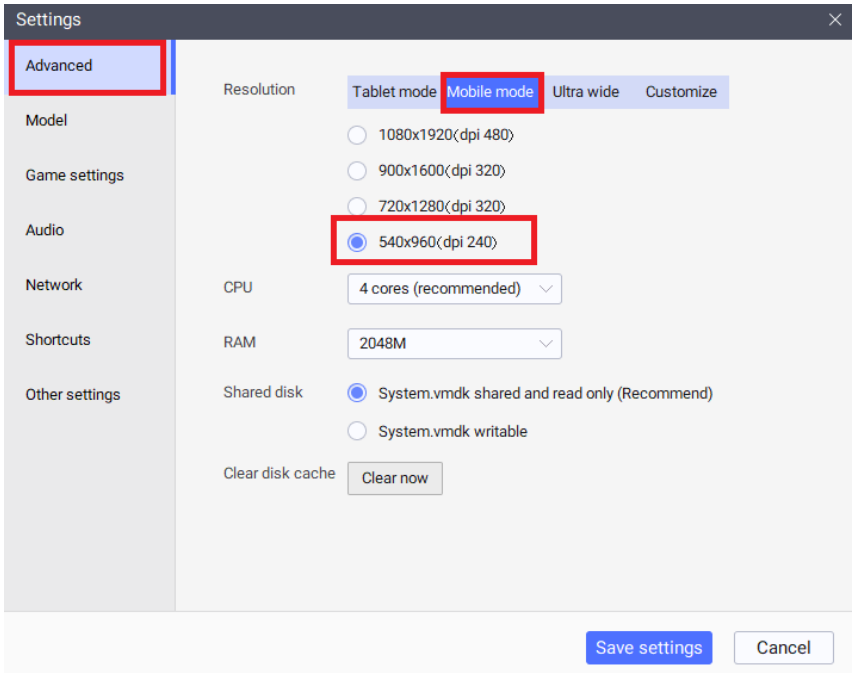


Figure 82: Change mobile mode

Step 6: Click the “Other settings”, in the “ADB debugging” section, select the option in the dropdown as: "open local connection" and click Save settings (and immediately click the "Restart now" button).

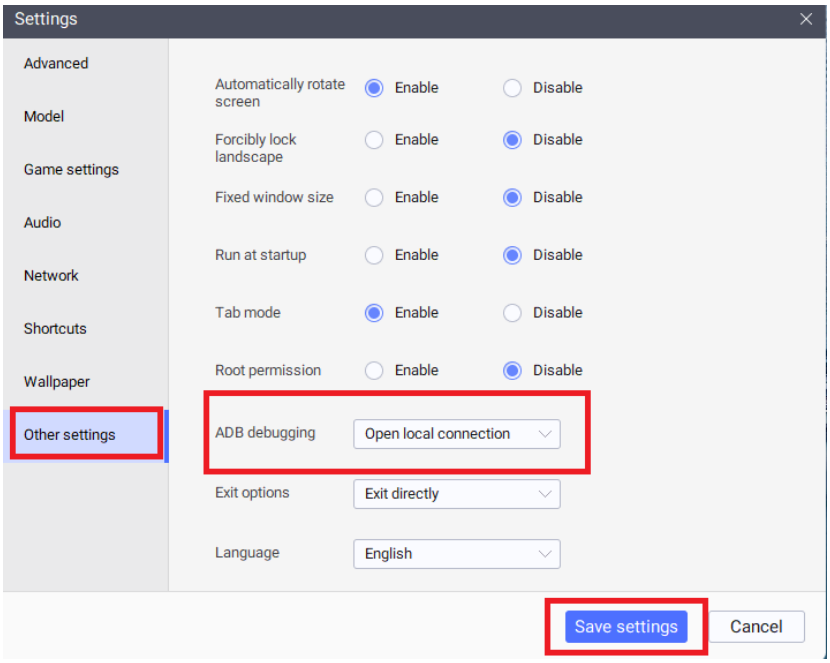


Figure 83: Save settings

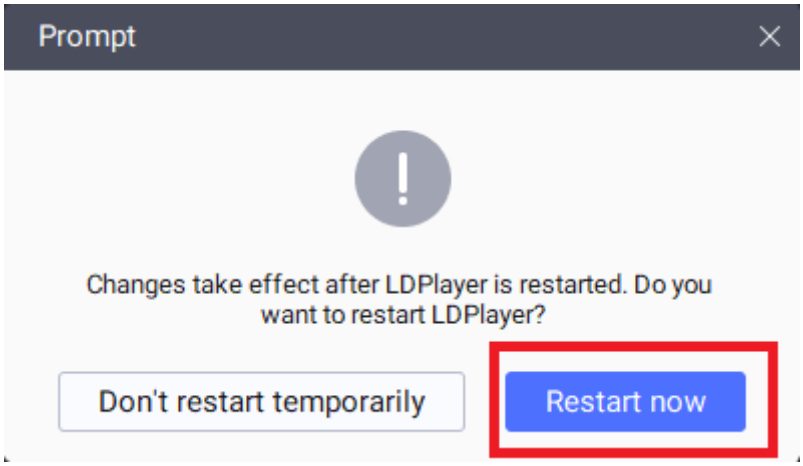


Figure 84: Restart Device

Step 7: Wait for the visual studio code to connect to LDPlayer or restart both. After connecting as shown below, press the key combination "Ctrl + F5" to launch the project.

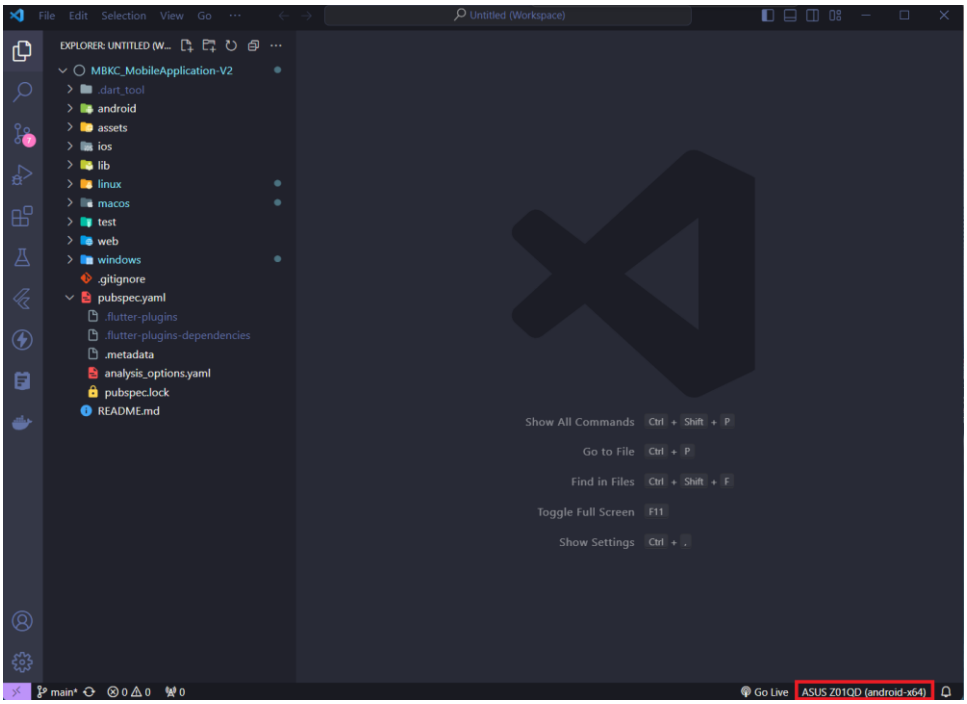


Figure 85: Launch project

2.2.5. Run MBKC Bot

Step 1: Clone the repository or extract “MBKC-Bot-main.zip” and open folder with Visual Studio 2022.

Step 2: Select " MBKCBot.Tool" and right click, then select "Set as Startup Project" as shown below.

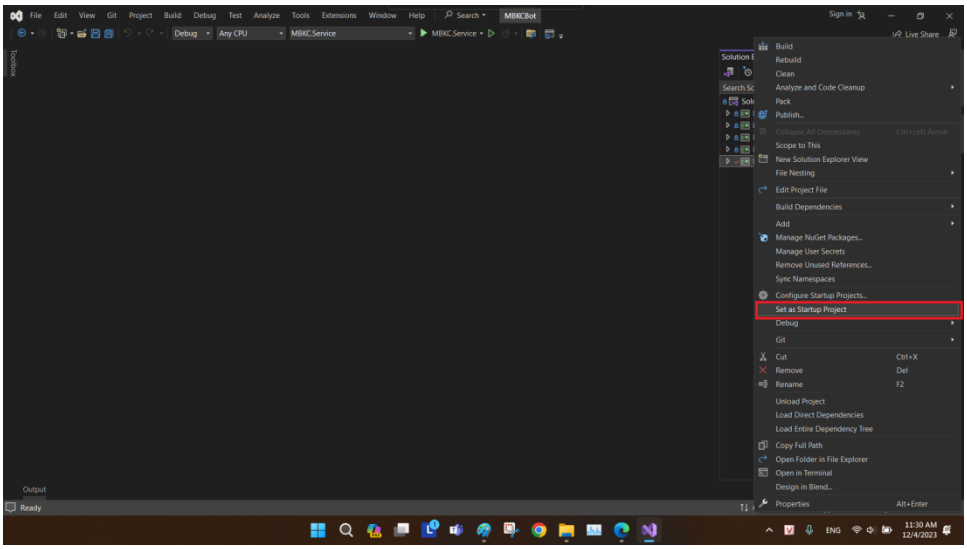


Figure 86: Select MBKCBot.Tool project

Step 3: Press "F5" or button as shown below to launch project "MBKCBot.Tool ".

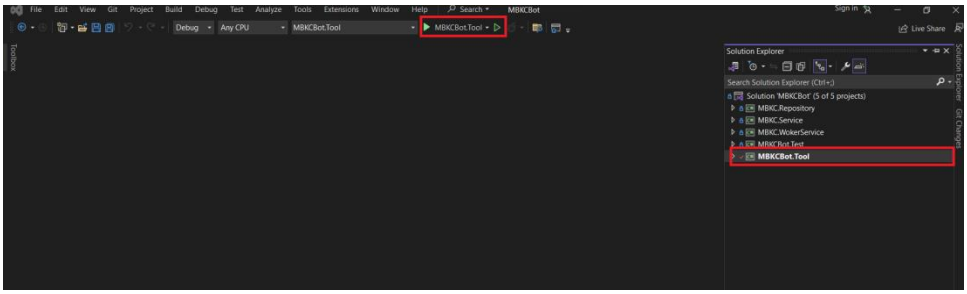


Figure 87: Launch project "MBKCBot.Tool"

3. Uer Manual

3.1. Overview

3.2. Web Application

3.2.1. Authentication

3.2.1.1 Login

3.2.1.1.1 Description
Use this guideline to login into MBKC with an email account.

This web application supports login with email account that has been added as a MBKC Admin, Kitchen Center Manager, Brand Manager or Cashier.

3.2.1.1.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password”.

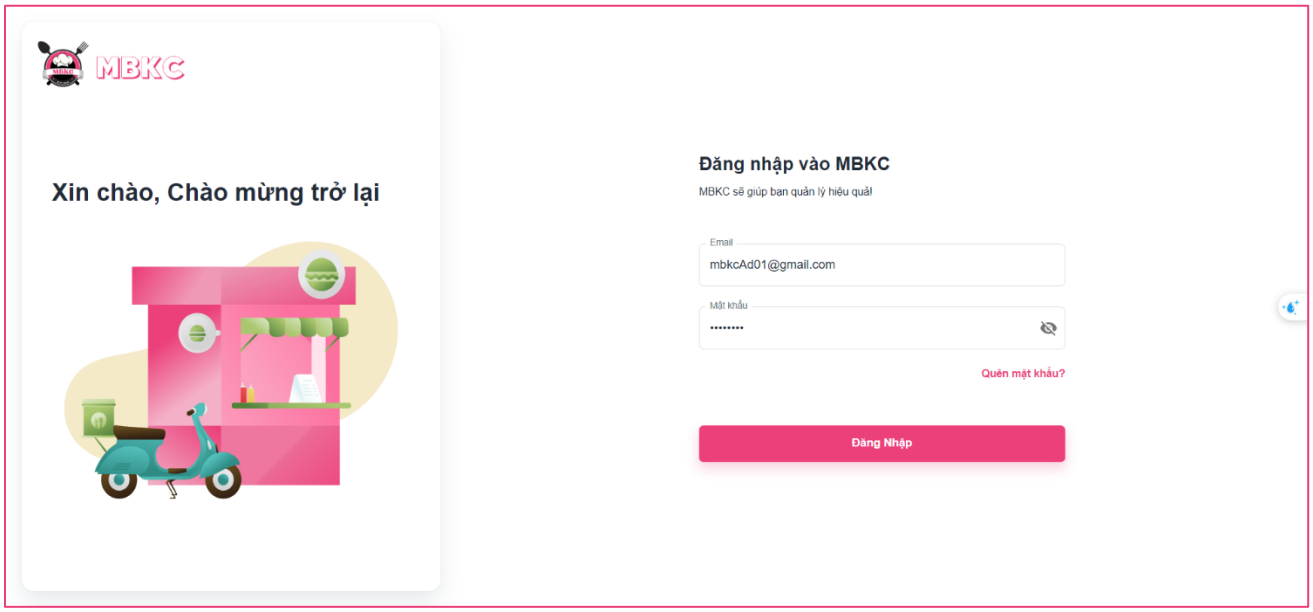


Figure 88: User Manual – Login – Step 1

Step 2: Click “Đăng Nhập” button.

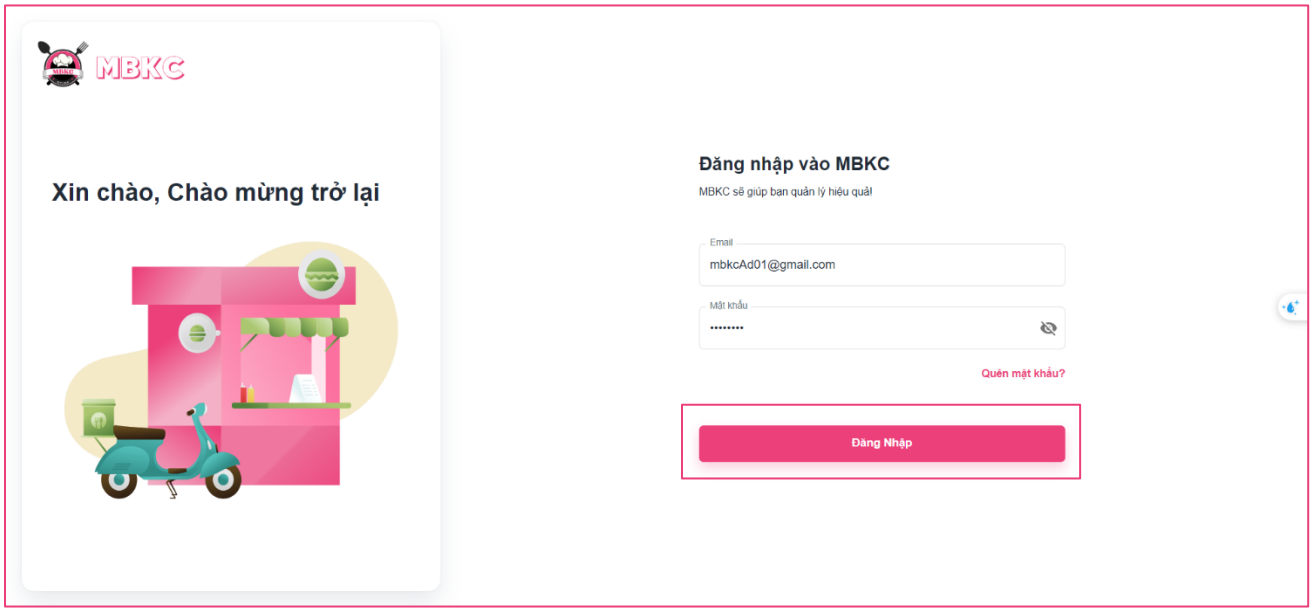


Figure 89: User Manual – Login – Step 2 - 1

If an error appears, check "Email" and "Password" again. Then click "Đăng Nhập" button again.

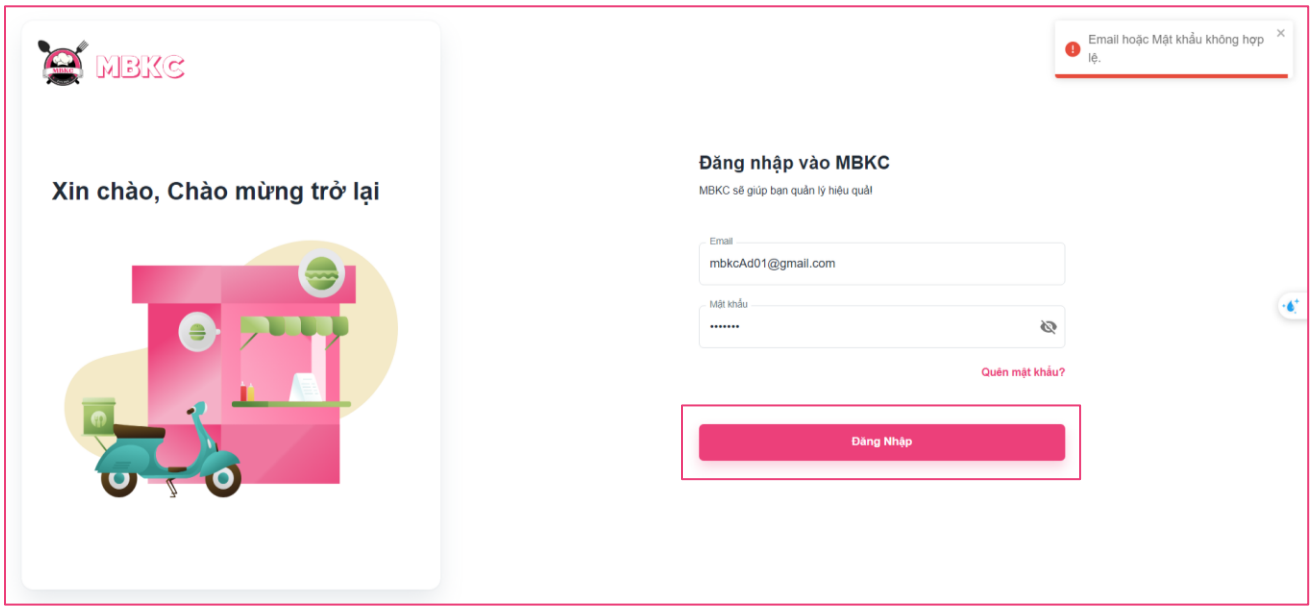


Figure 90: User Manual – Login – Step 2 - 2

After logging in successfully, user is directed to the Dashboard Page with account authenticated.

3.2.1.2. Reset Password

3.2.1.2.1. Description

Use this guide so users can reset their email account password when they forget their password.

3.2.1.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then press “Quên mật khẩu”.

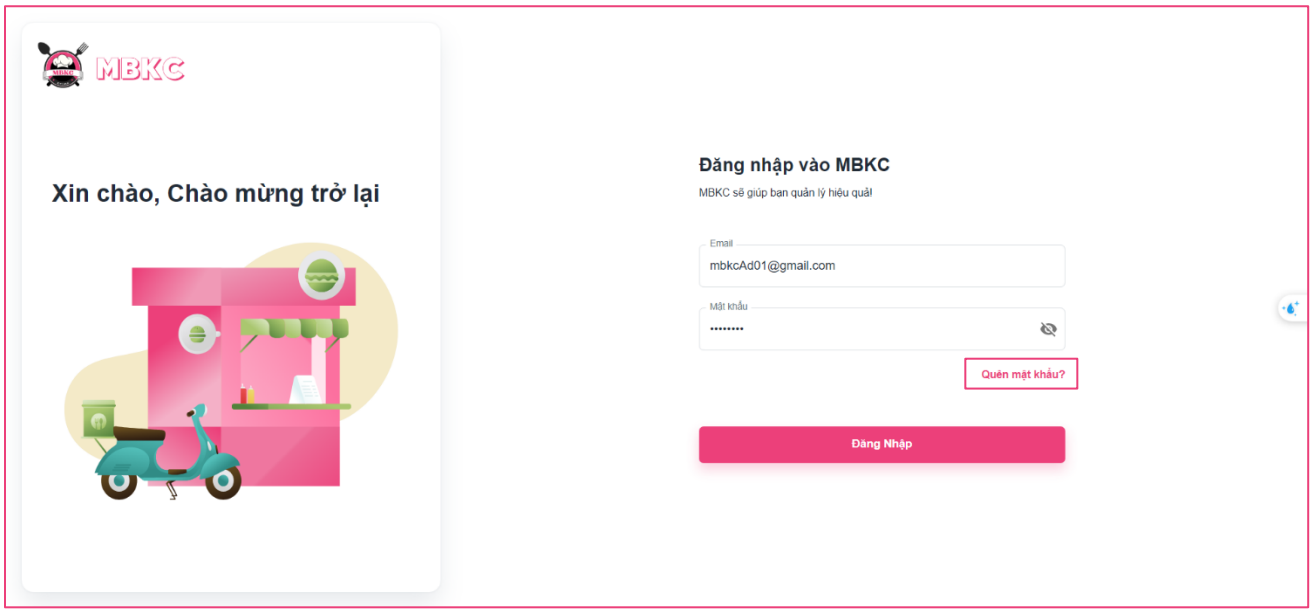


Figure 91: User Manual – Reset Password – Step 1

Step 2: Then fill in “Email”.

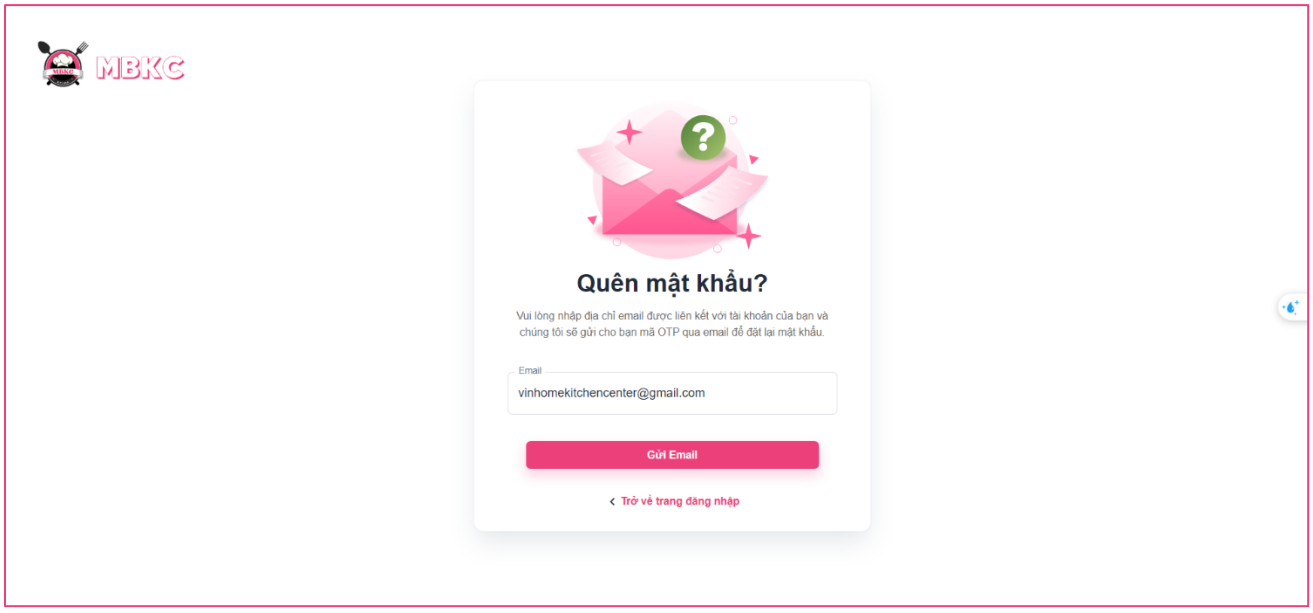


Figure 92: User Manual – Reset Password – Step 2

Step 3: Click “Gửi Email” button.

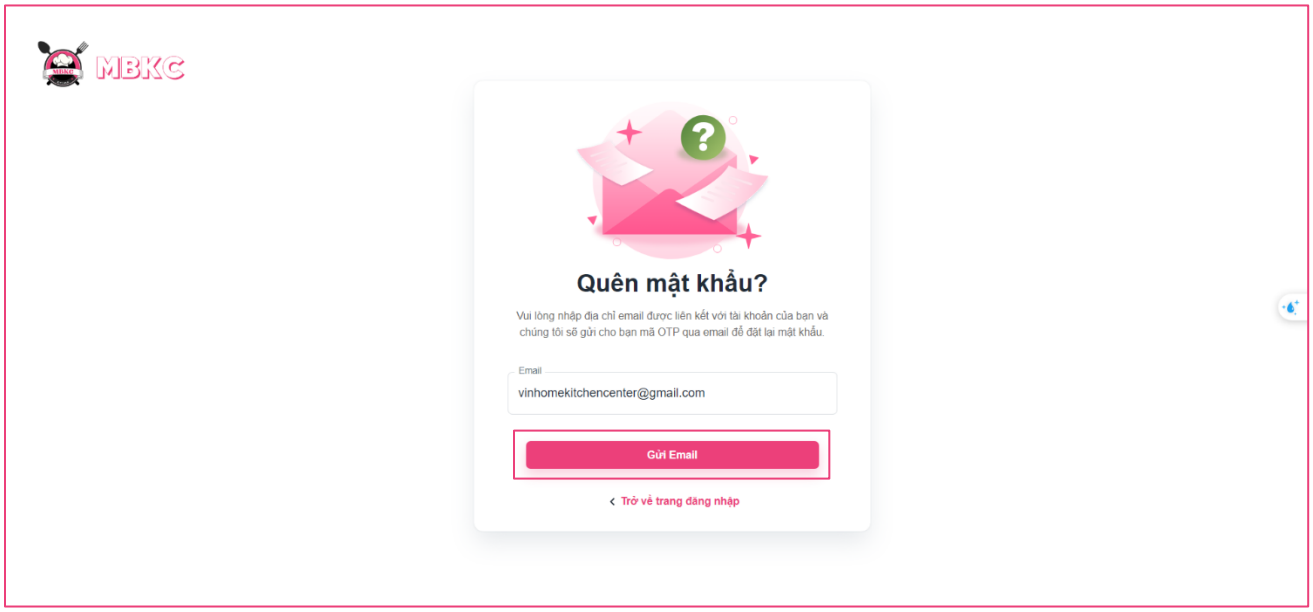


Figure 93: User Manual – Reset Password – Step 3

After sending the email successfully, the user is redirected to the Verify OTP Page with the email that sent the OTP authentication code.

Step 4: Check email in field “Email” and fill in “Mã OTP”.

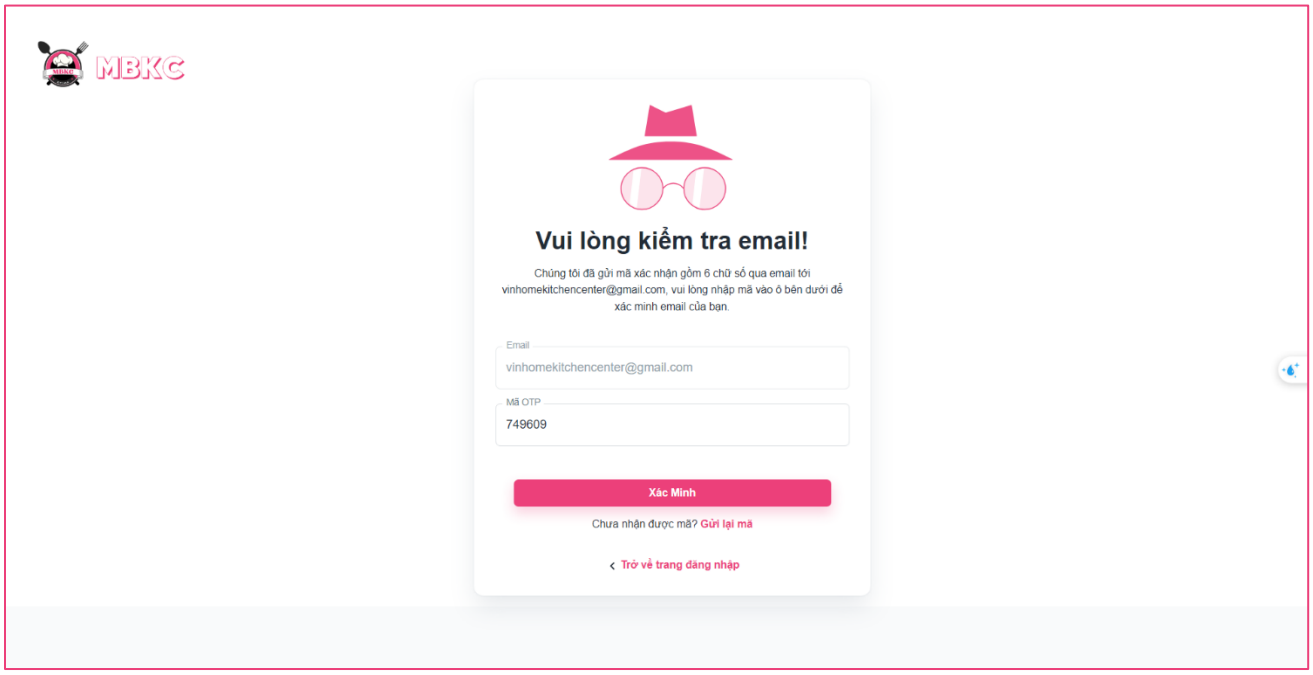


Figure 94: User Manual – Reset Password – Step 4 – 1

If you check your email and still have not received the OTP code. Please press "Gửi lại mã" and to receive the OTP code again, then fill in “Mã OTP”.

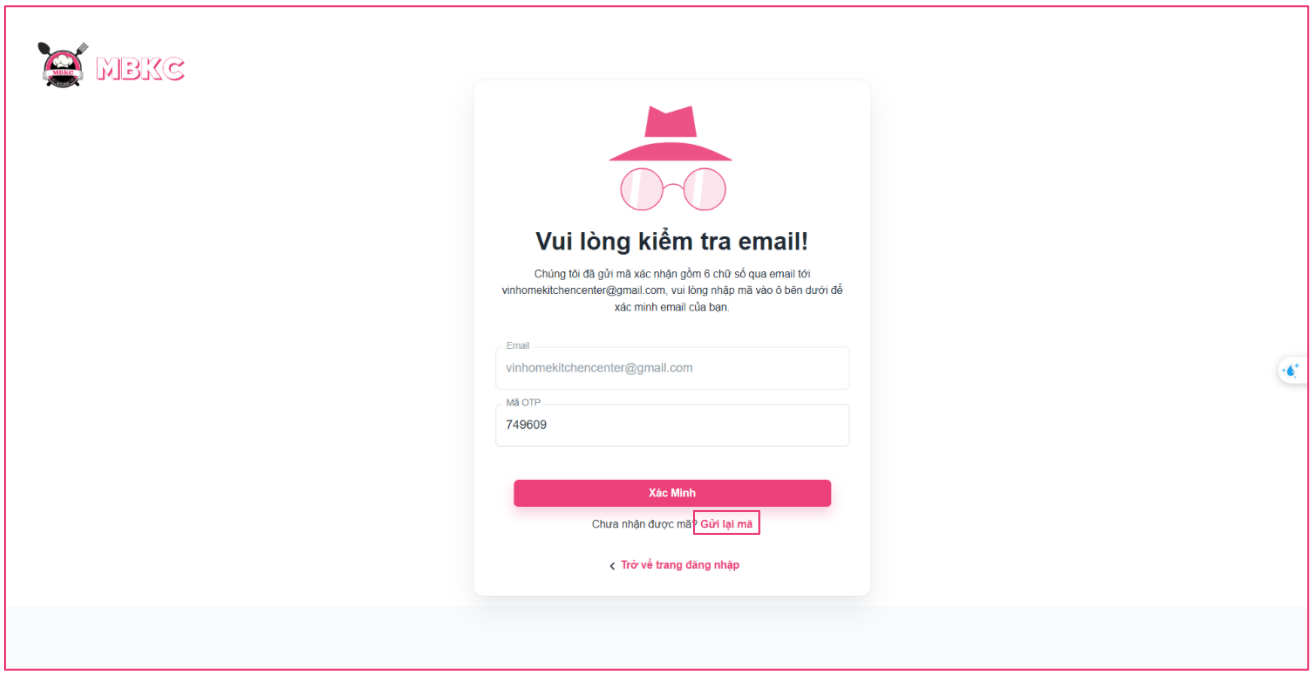


Figure 95: User Manual – Reset Password – Step 4 – 2

Step 5: Click “Xác Minh” button.

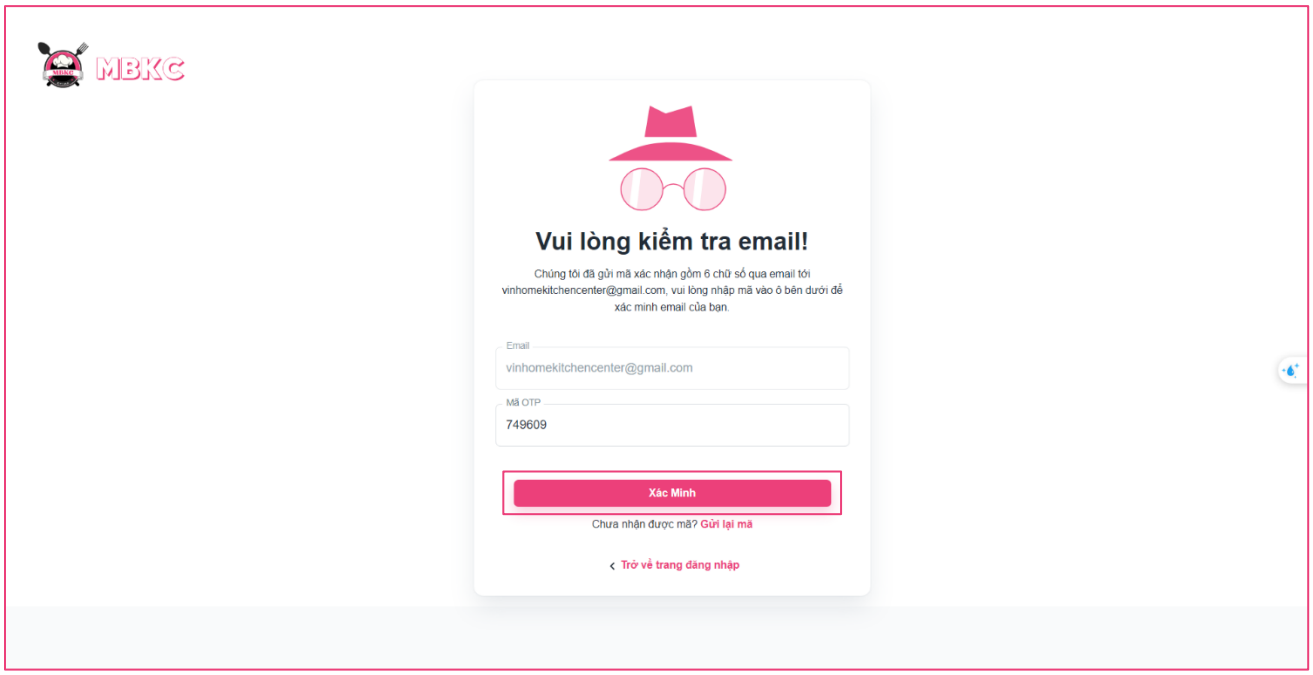


Figure 96: User Manual – Reset Password – Step 5

After successfully verifying the OTP code, the user is redirected to the Reset Password Page to reset the password.

Step 6: Then fill in “Mật khẩu mới” and “Xác nhận mật khẩu”.

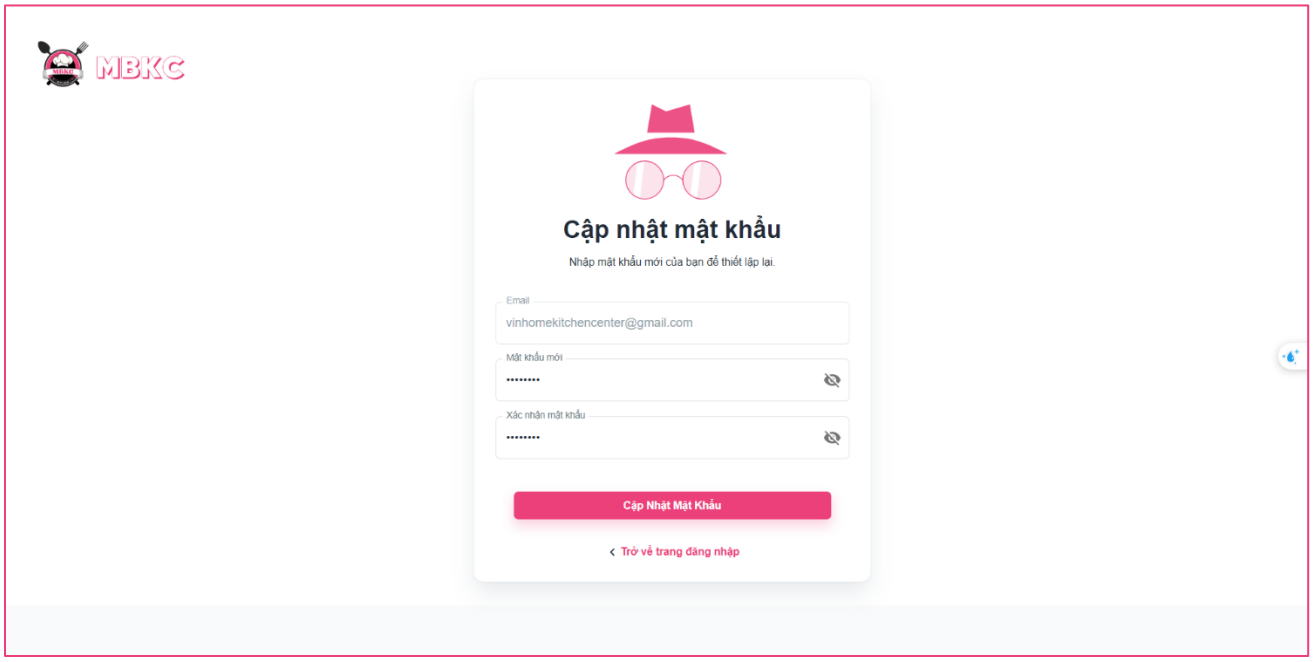


Figure 97: User Manual – Reset Password – Step 6

Step 7: Click “Cập Nhật Mật Khẩu” button.

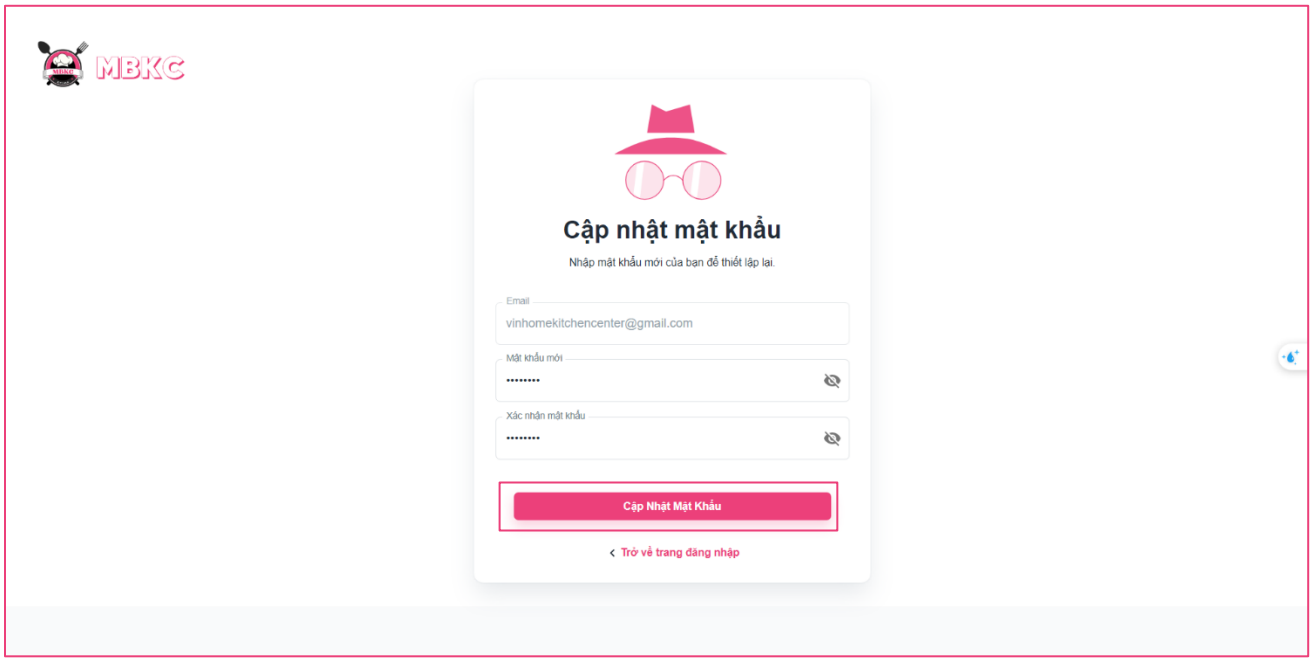


Figure 98: User Manual – Reset Password – Step 7 - 1

If an error appears, check “Mật khẩu mới” and “Xác nhận mật khẩu” again. Then click " Cập Nhật Mật Khẩu" button again.

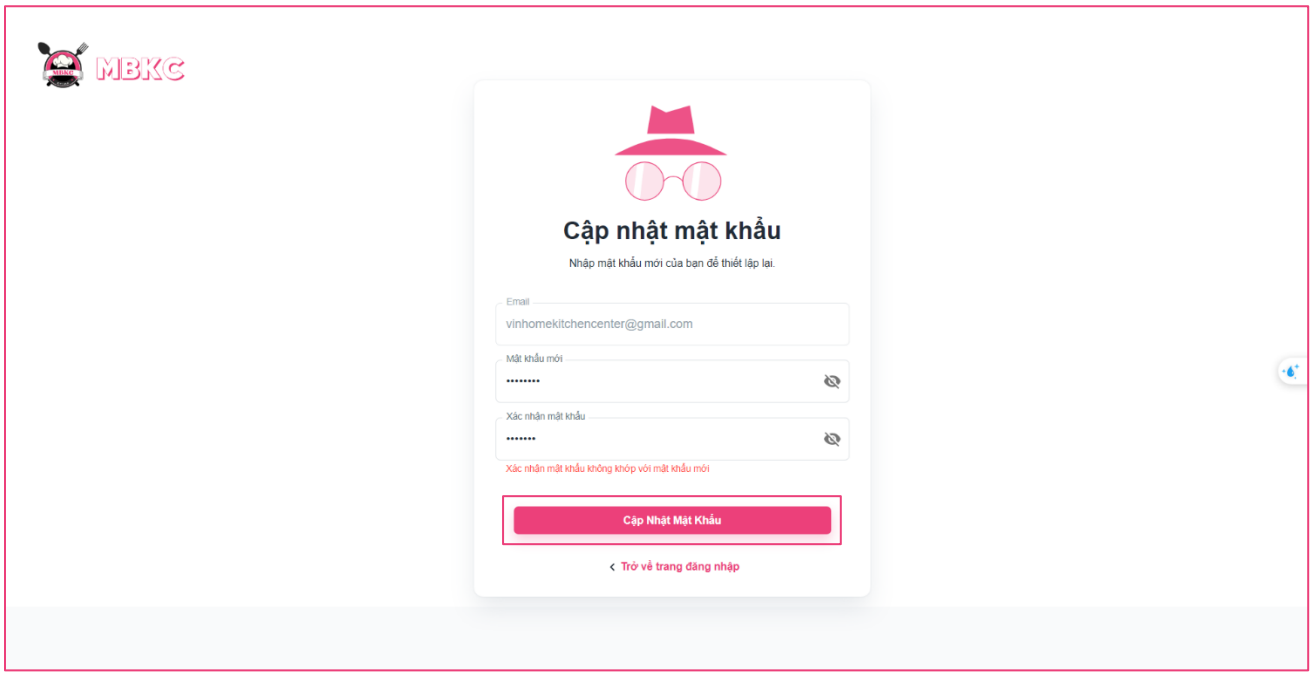


Figure 99: User Manual – Reset Password – Step 7 - 2

After successfully reset password, the user is redirected to the Login Page to login again.

Case: During the reset password process when you no longer want to continue. You can press "Trở về trang đăng nhập" to return to the Login Page.

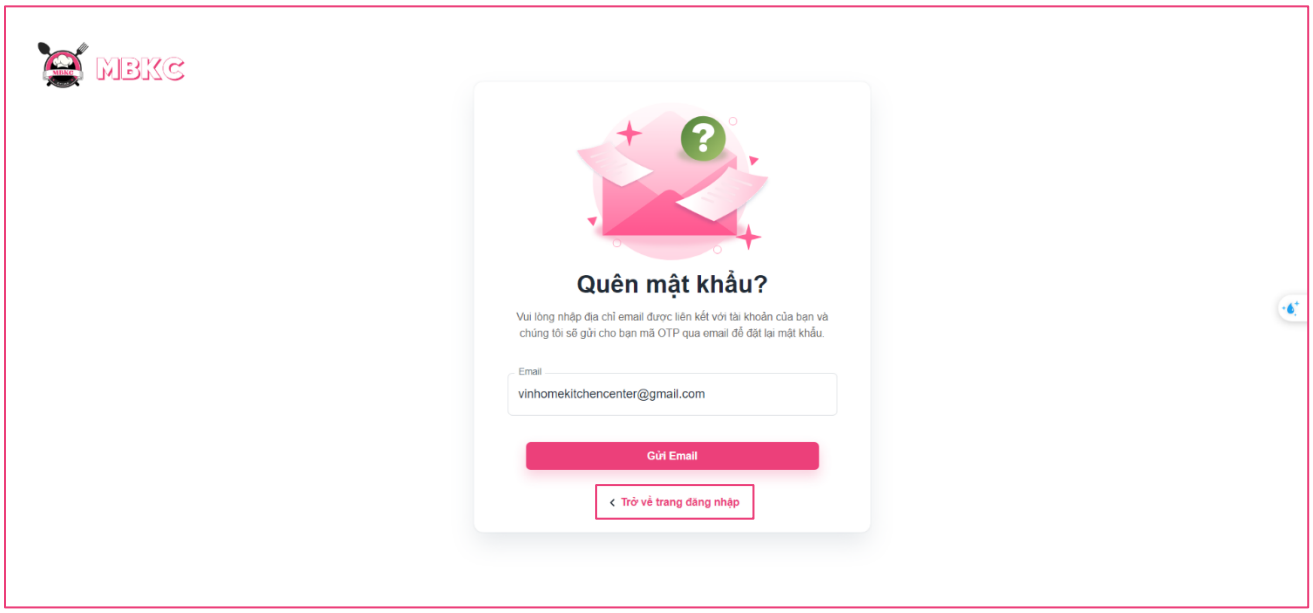


Figure 100: User Manual – Reset Password – Case – 1

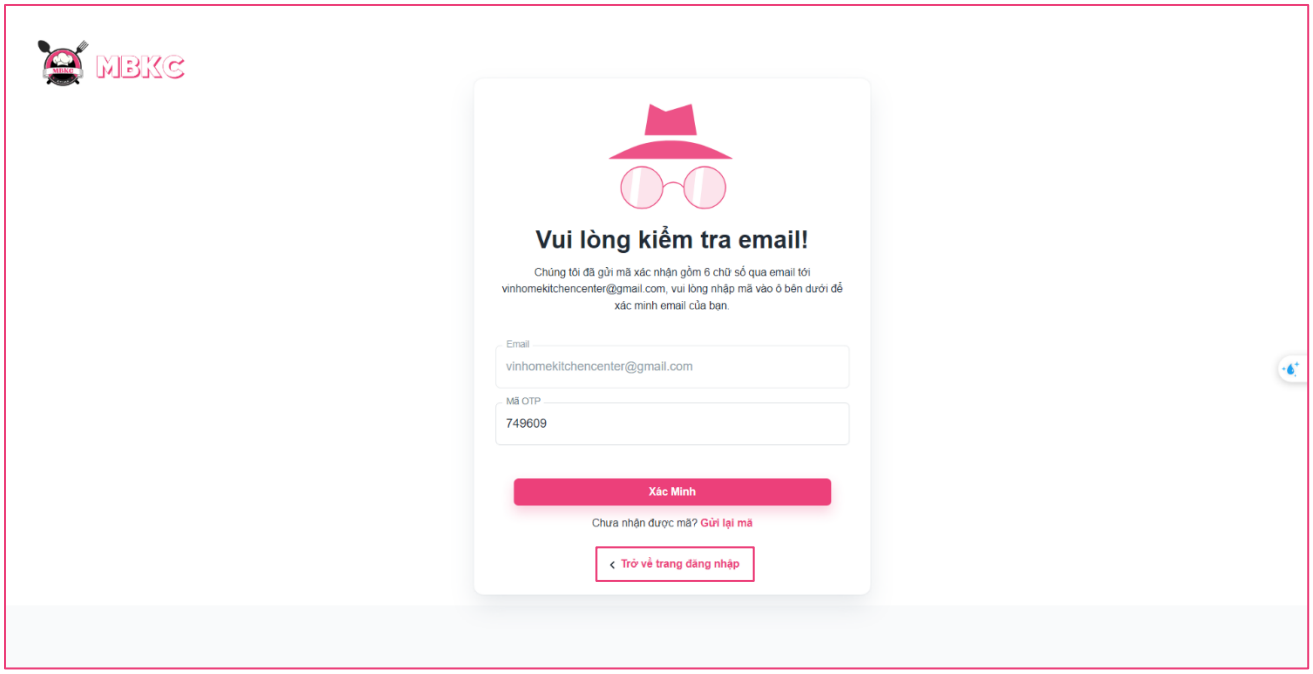


Figure 101: User Manual – Reset Password – Case - 2

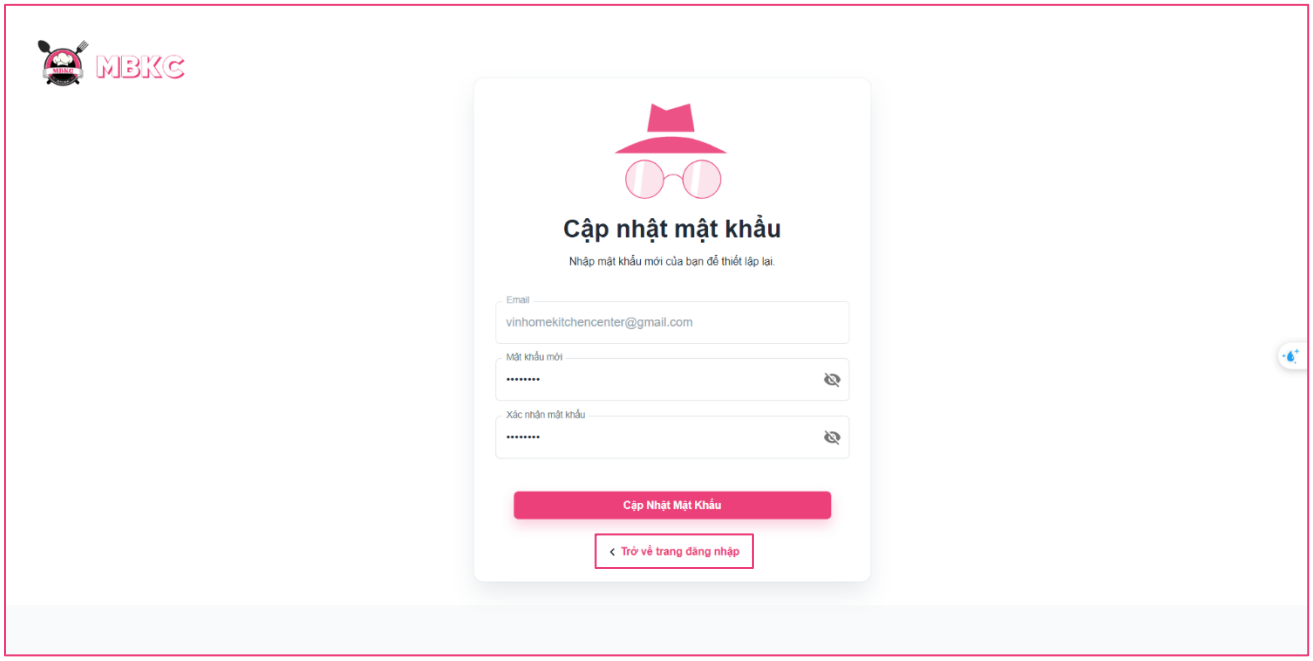


Figure 102: User Manual – Reset Password – Case - 3

3.2.1.3. Update Password in First Time Login

3.2.1.3.1. Description

Use this guide so users will update their password when they login for the first time with a new account.

This web application supports updating passwords for actors: Kitchen Center Manager, Brand Manager, and Cashier.

3.2.1.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. After login successfully.

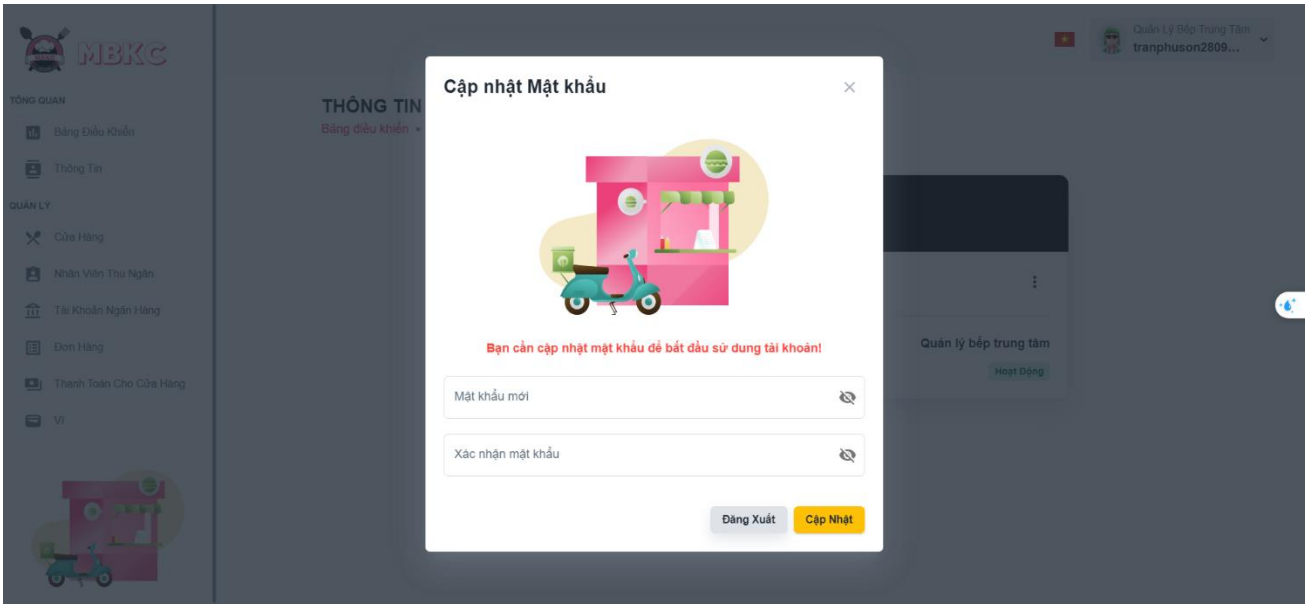


Figure 103: User Manual – Update Password in First Time Login – Step 1

Step 2: Fill in "Mật khẩu mới" and "Xác nhận mật khẩu", then click "Cập Nhật" button.

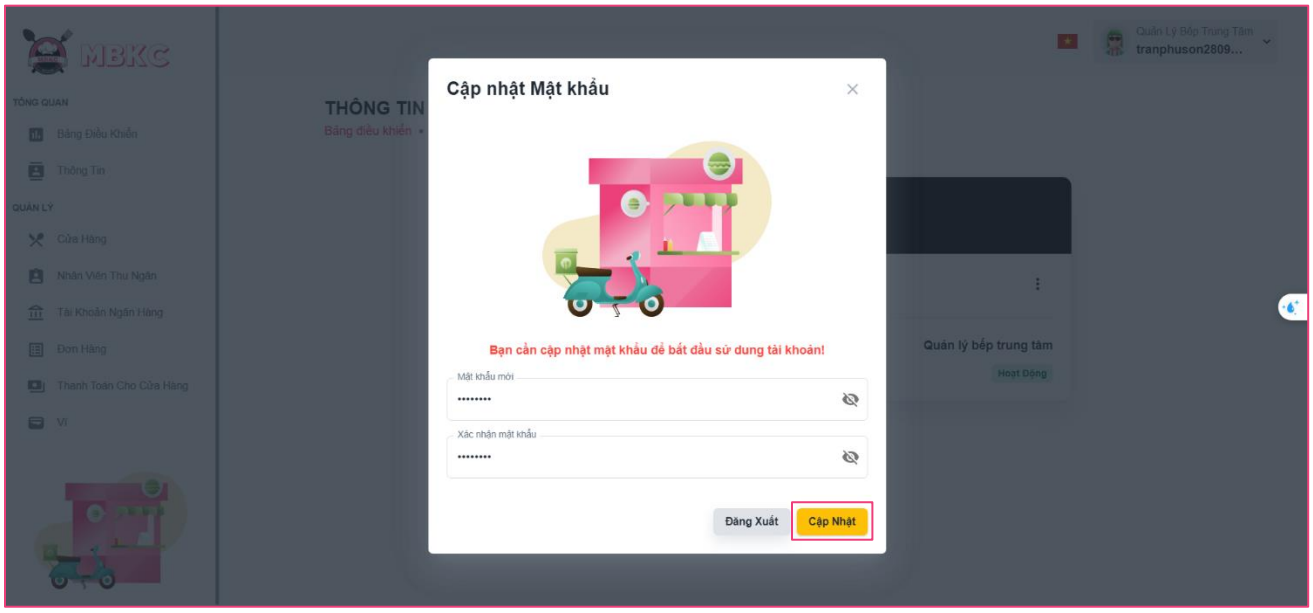


Figure 104: User Manual – Update Password in First Time Login – Step 2

After successfully update password, Update Password Modal will be reloading.

Step 3: Click “Tiếp Tục” button or click the Close icon button to continue using the web application.

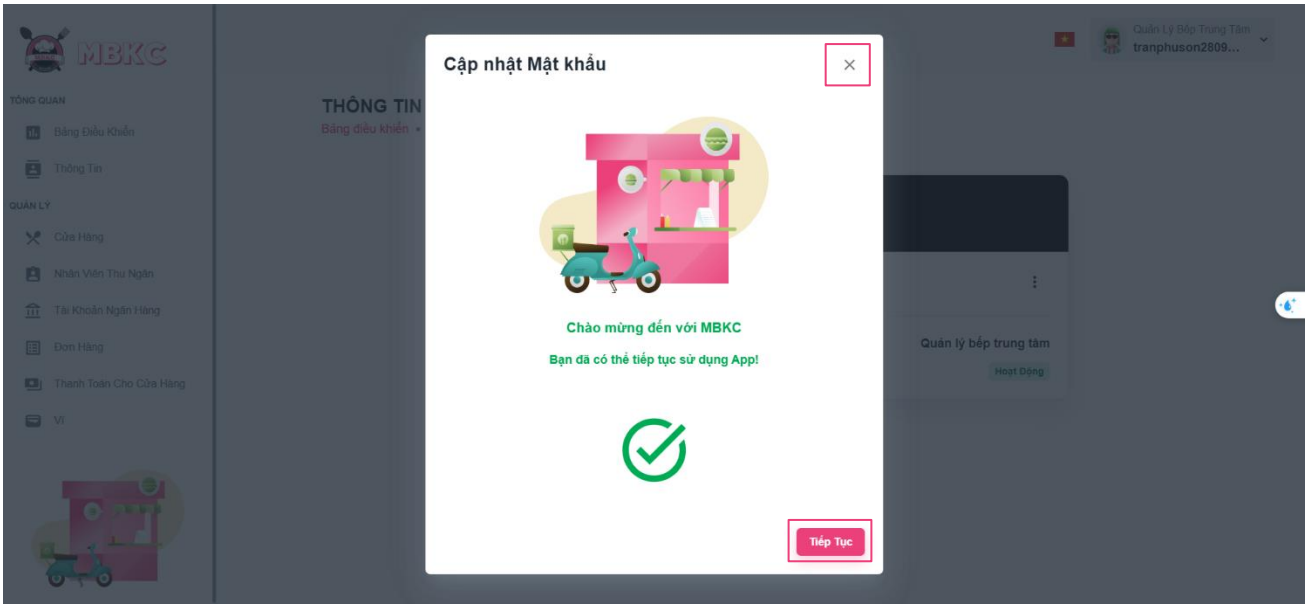


Figure 105: User Manual – Update Password in First Time Login – Step 3

3.2.1.4. Update Password

3.2.1.4.1. Description

Use this guide so users can update their email account password when they want to change their password.

This web application supports updating passwords for actors: Kitchen Center Manager, Brand Manager, and Cashier.

3.2.1.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. After logging in successfully, click the box avatar on the header and click “Tài khoản” menu item.

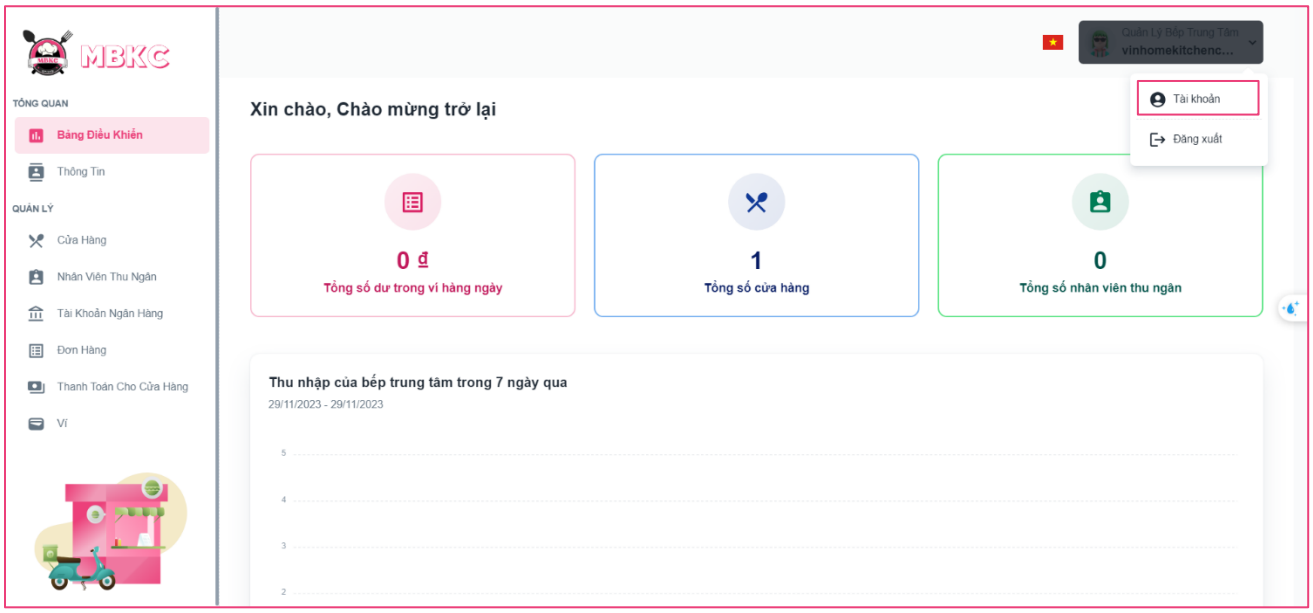


Figure 106: User Manual – Update Password – Step 1

After click “Tài khoản” menu item, the user is redirected to the Profile Page.

Step 2: Click menu icon button and click “Cập nhật mật khẩu” menu item.

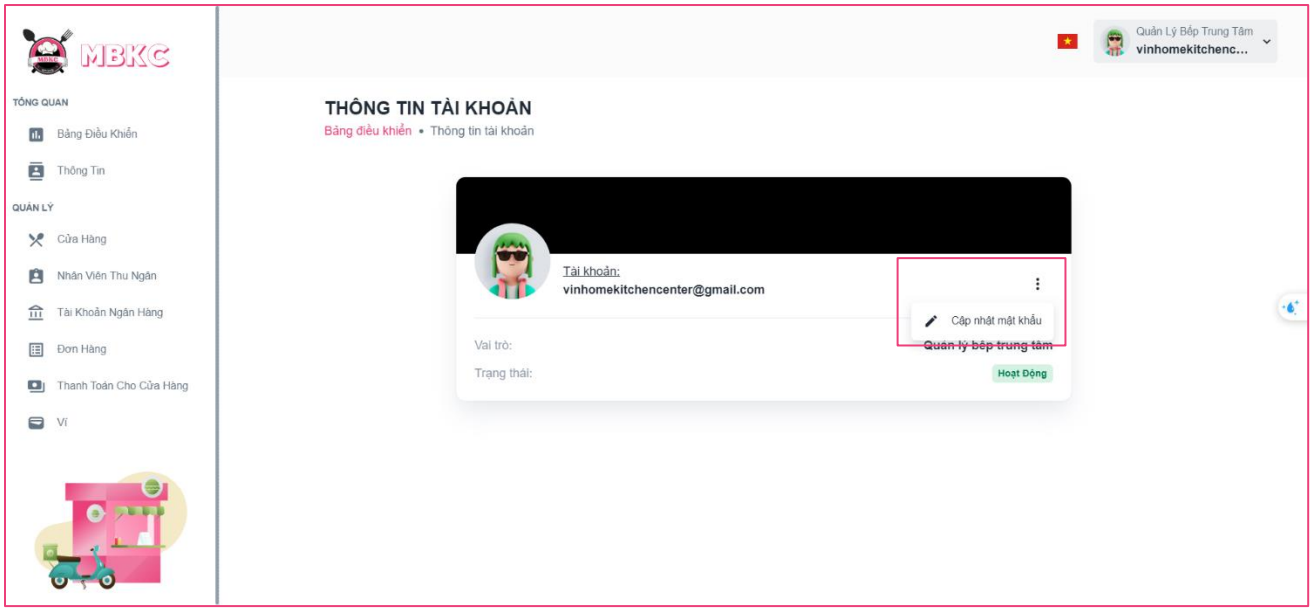


Figure 107: User Manual – Update Password – Step 2

Step 3: Fill in “Mật khẩu mới” and “Xác nhận mật khẩu”, then click “Cập Nhật” button.

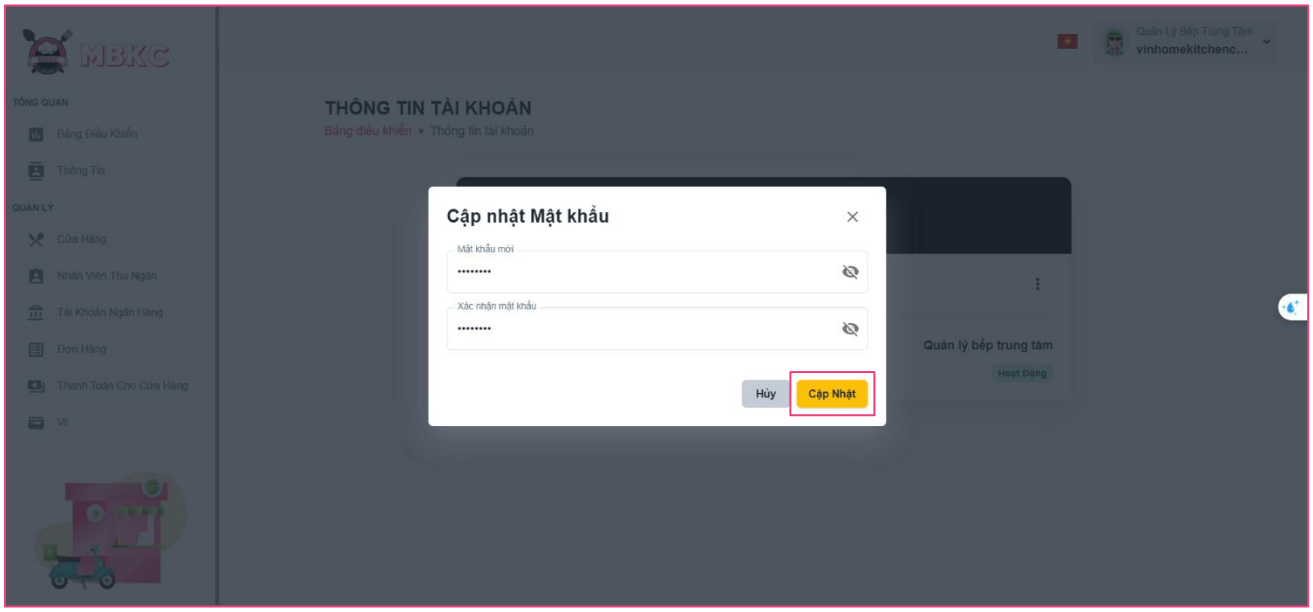


Figure 108: User Manual – Update Password – Step 3

After successfully update password, Update Password Page will be reloading.

3.2.2. Kitchen center management

3.2.2.1. Create new Kitchen Center

3.2.2.1.1. Description
Use this guideline to login into MBKC account to create a new kitchen center.

3.2.2.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

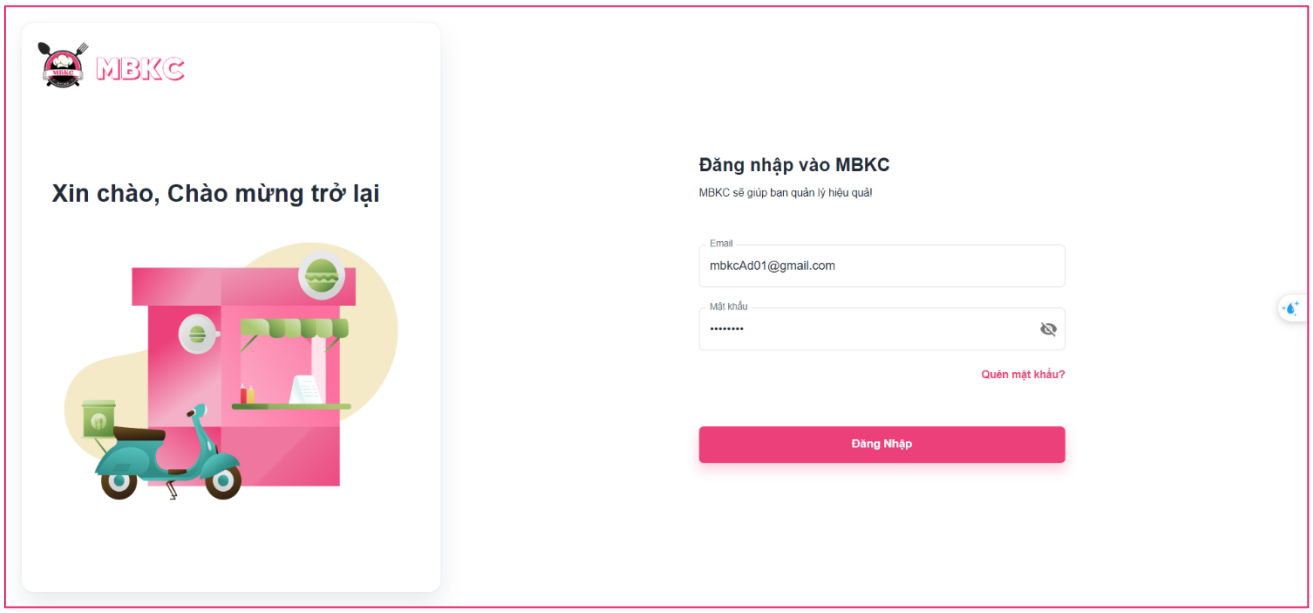


Figure 109: User Manual - Create new Kitchen Center – Step 1

Step 2: Click “Đăng Nhập” button.

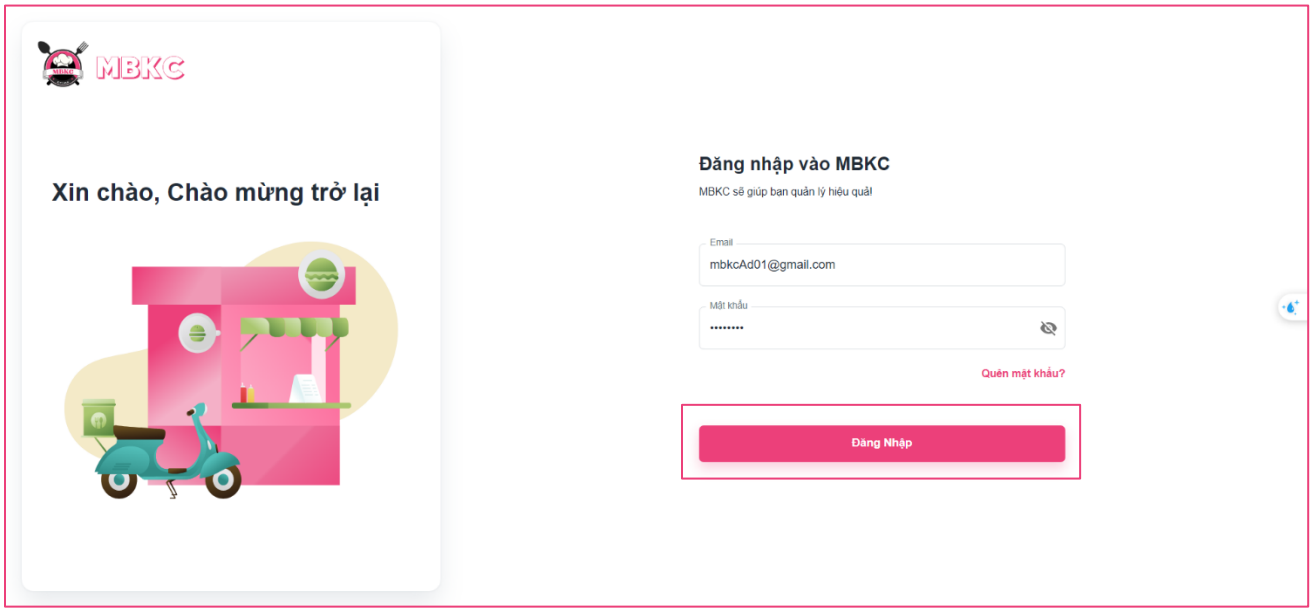


Figure 110: User Manual - Create new Kitchen Center – Step 2

Step 3: From Admin Dashboard Page, choose “Bếp Trung Tâm” from the left sidebar.

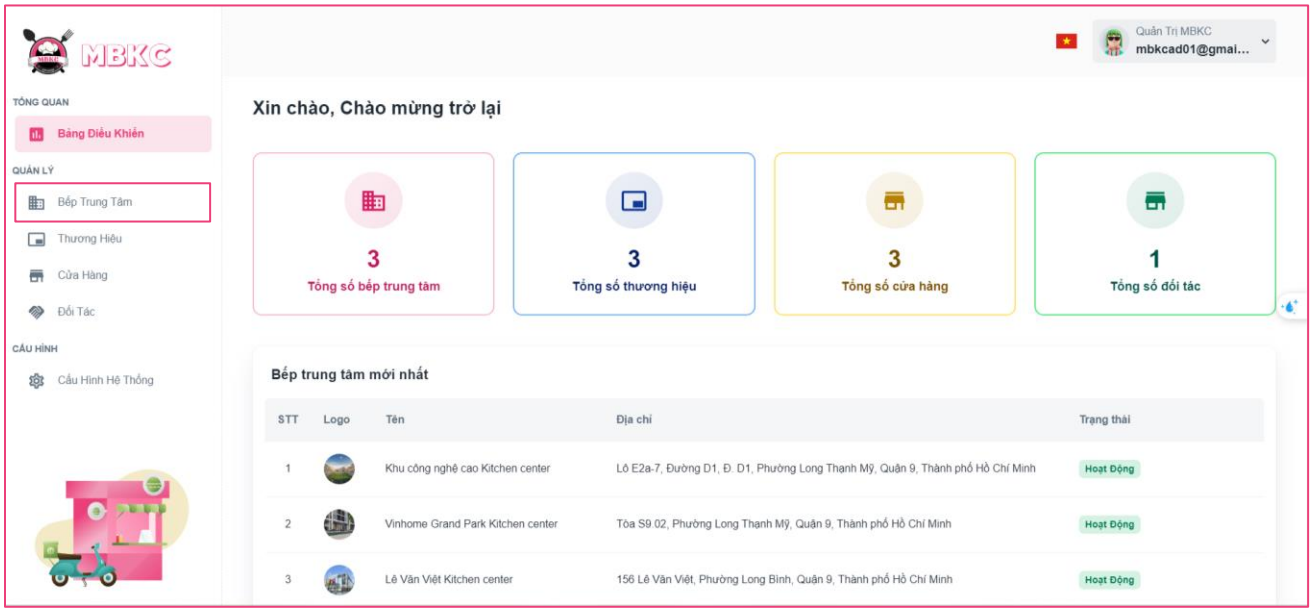


Figure 111: User Manual - Create new Kitchen Center – Step 3

Step 4: Click “Thêm Bếp Trung Tâm” button on the screen and navigate to Create New Kitchen Center Page to add a new kitchen center.

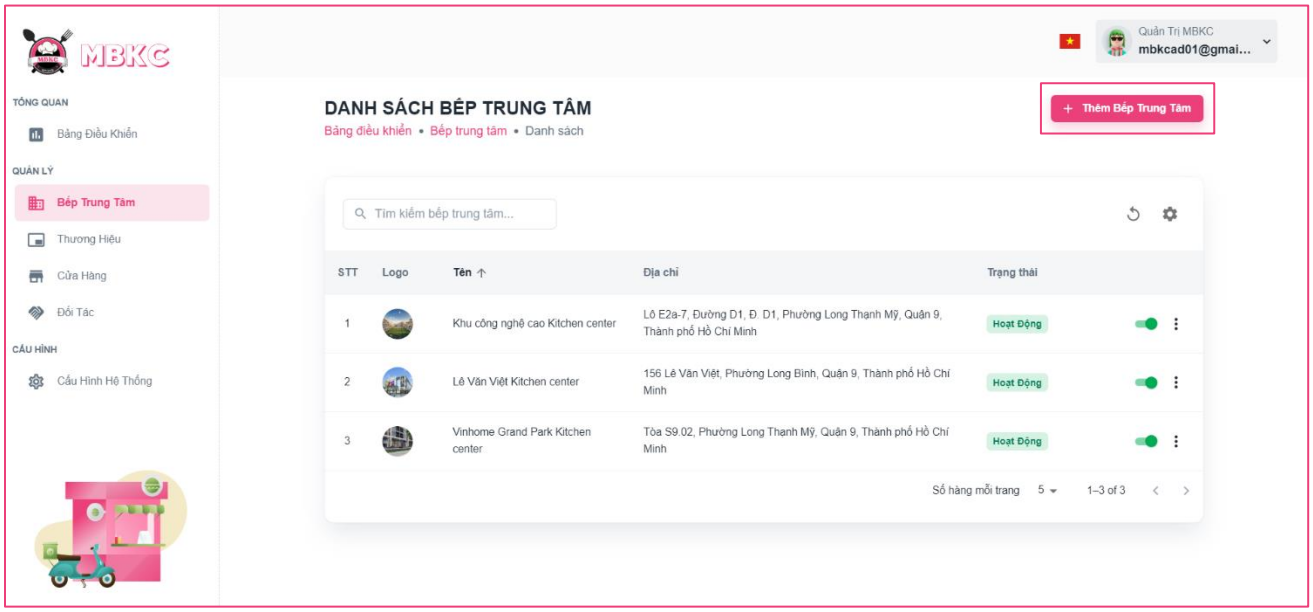


Figure 112: User Manual - Create new Kitchen Center – Step 4

Step 5: Filling all the required information to add a kitchen center including: “Logo”, “Tên bếp trung tâm", “Email người quản lý", “Địa chỉ chi tiết", “Tỉnh”, “Quận – Huyện" and “Phường”.

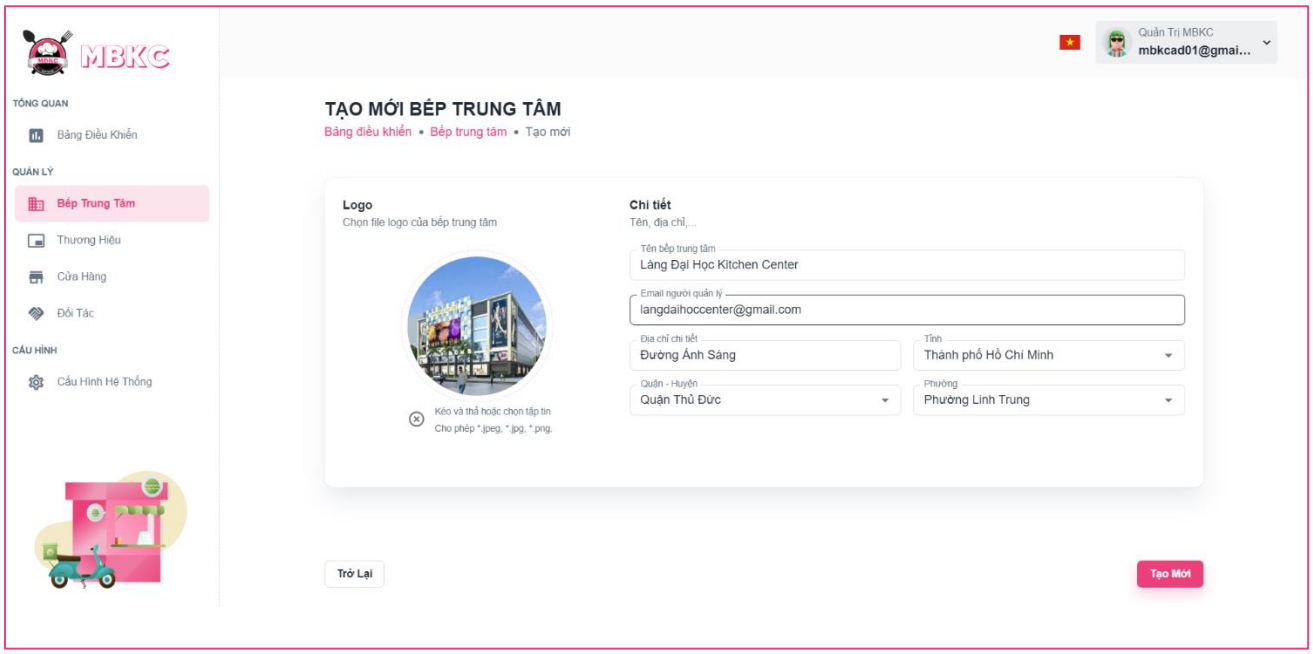


Figure 113: User Manual - Create new Kitchen Center – Step 5

Step 6: After filling in all required field, click “Tạo Mới” to finish creating a new kitchen center.

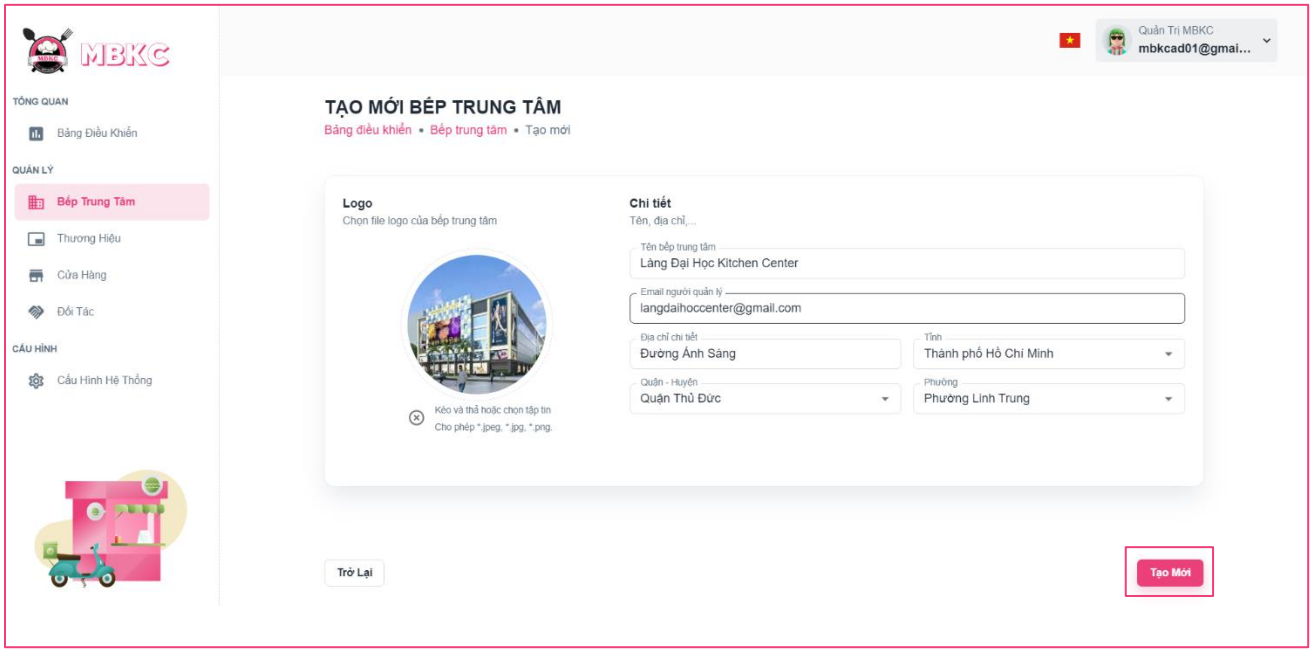


Figure 114: User Manual - Create new Kitchen Center – Step 6

Step 7: You will be navigated to List of Kitchen Center Page showing the list of kitchen center and a toast message “Tạo bếp trung tâm mới thành công”.

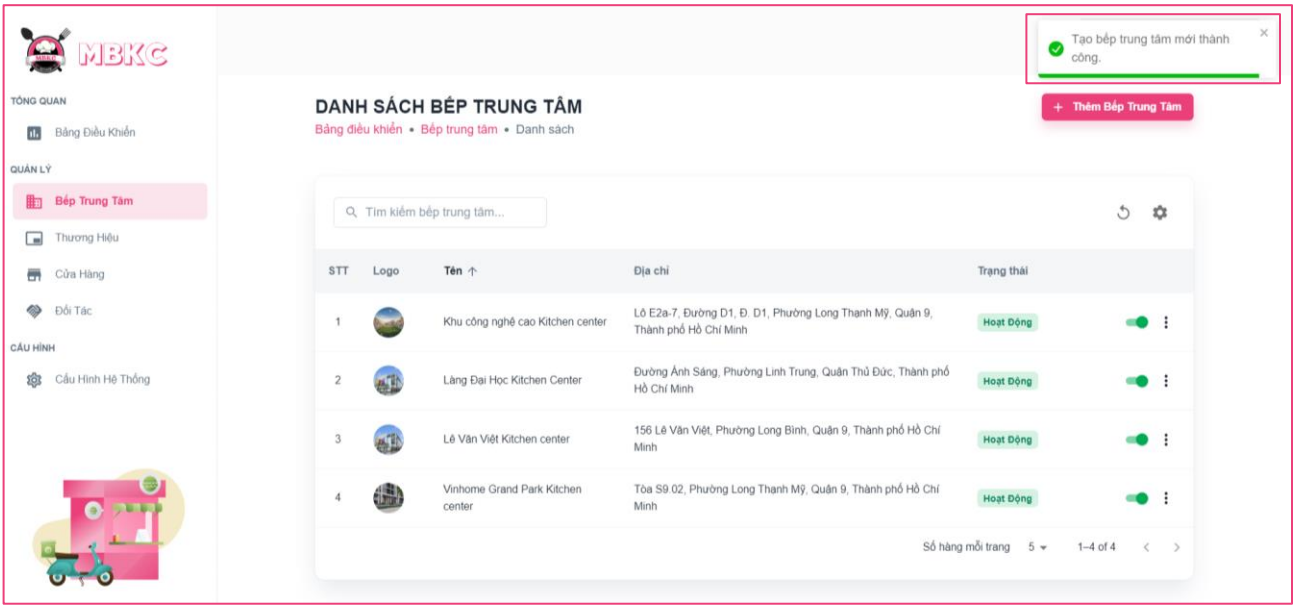


Figure 115: User Manual - Create new Kitchen Center – Step 7

3.2.2.2. View list Kitchen Center

3.2.2.2.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of the kitchen center.

3.2.2.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

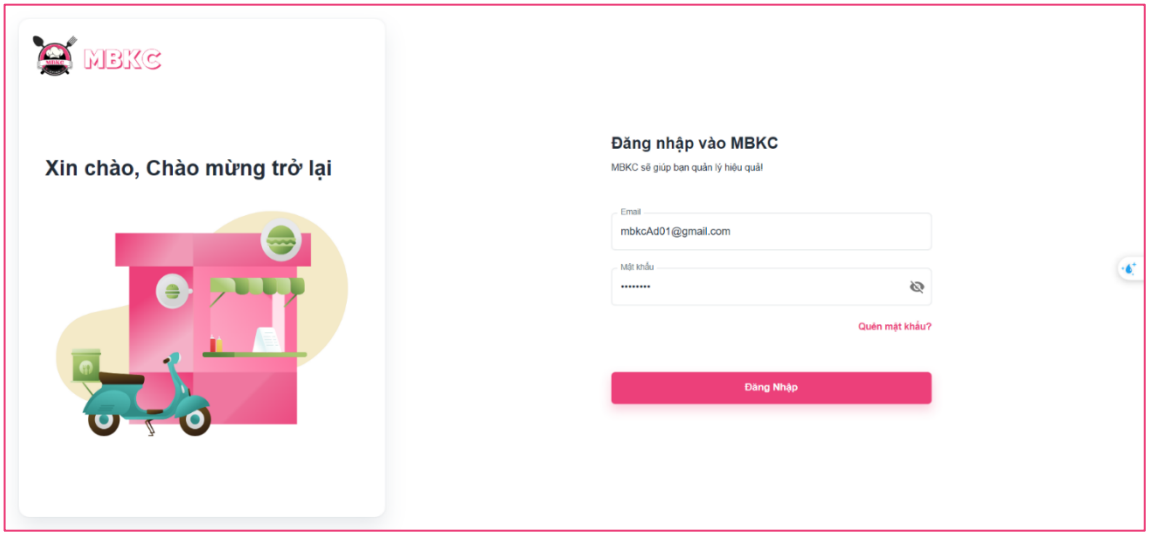


Figure 116: User Manual - View list Kitchen Center – Step 1

Step 2: Click “Đăng Nhập” button.

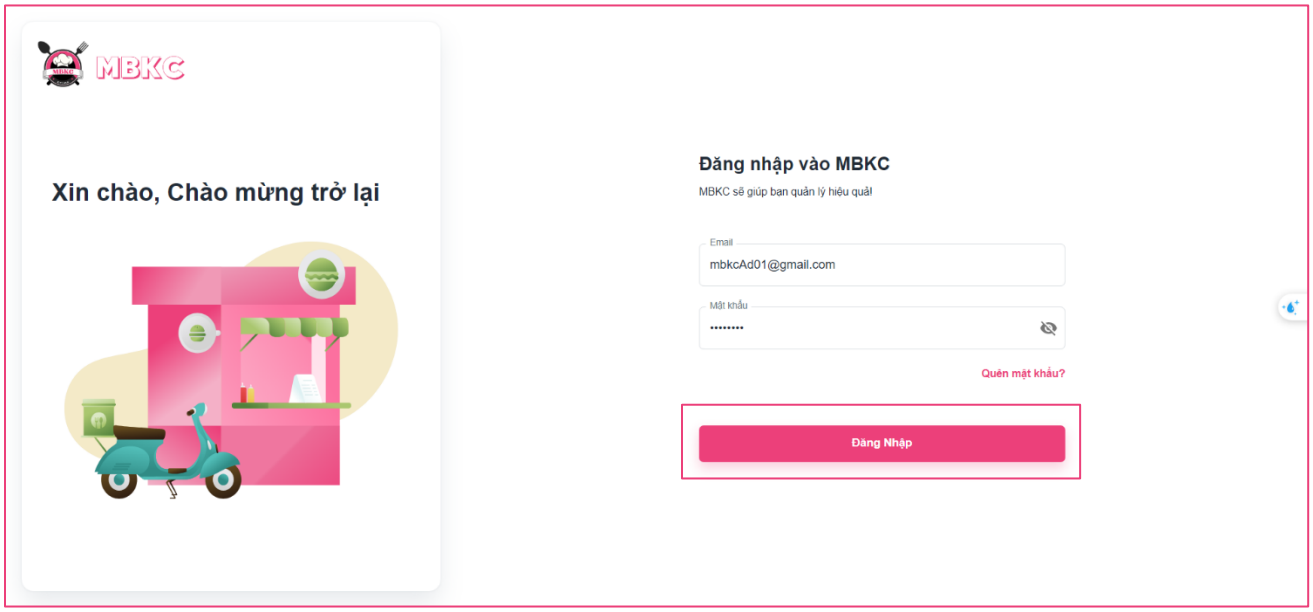


Figure 117: User Manual - View list Kitchen Center – Step 2

Step 3: From Admin Dashboard Page, choose “Bếp Trung Tâm” from the left sidebar.

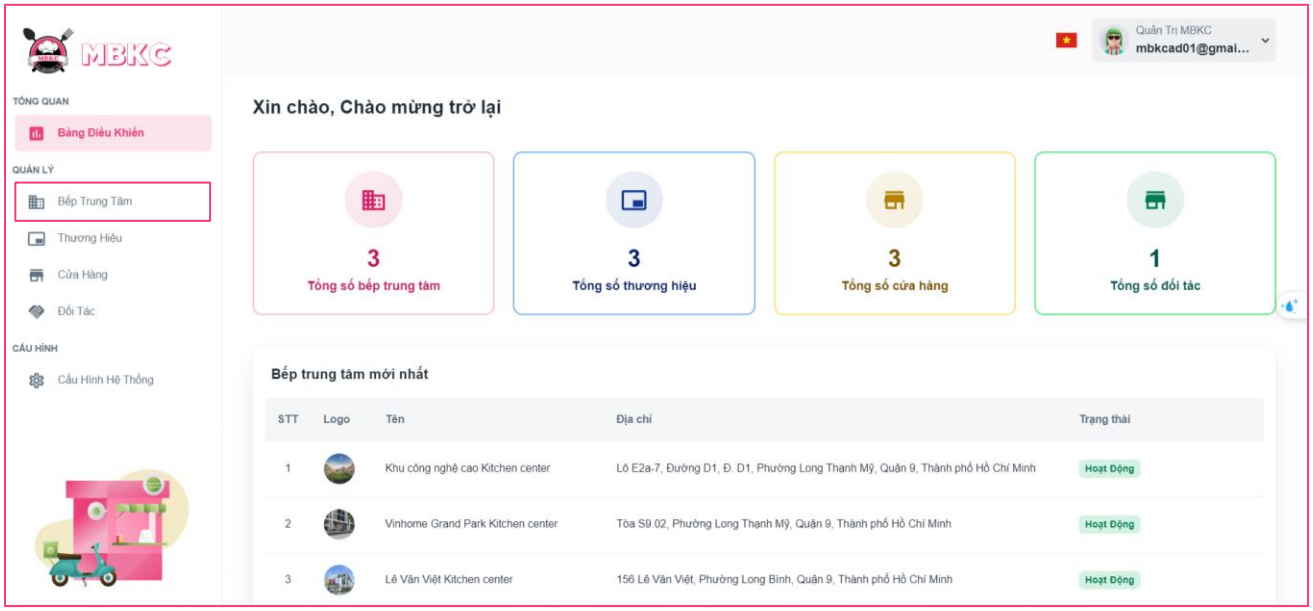


Figure 118: User Manual - View list Kitchen Center – Step 3

Step 4: Search, reload data, and show or hide columns in the table list of the kitchen center.

Case 1: Fill in the kitchen center name in the search box to search kitchen centers in the table list.

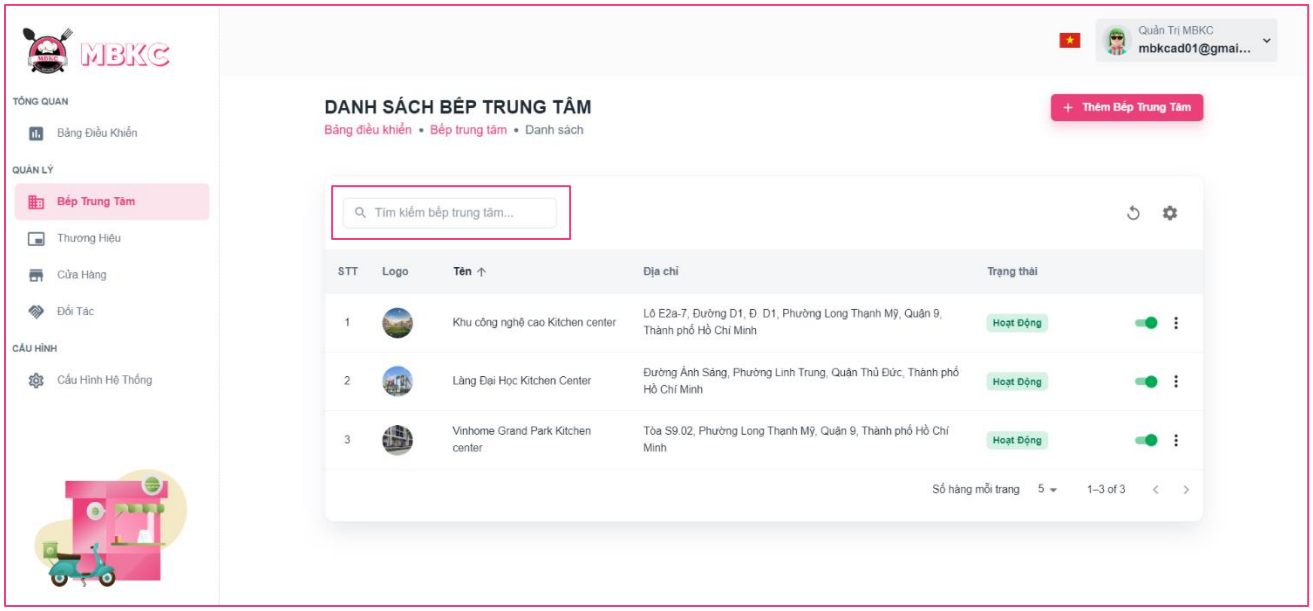


Figure 119: User Manual - View list Kitchen Center – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of kitchen centers.

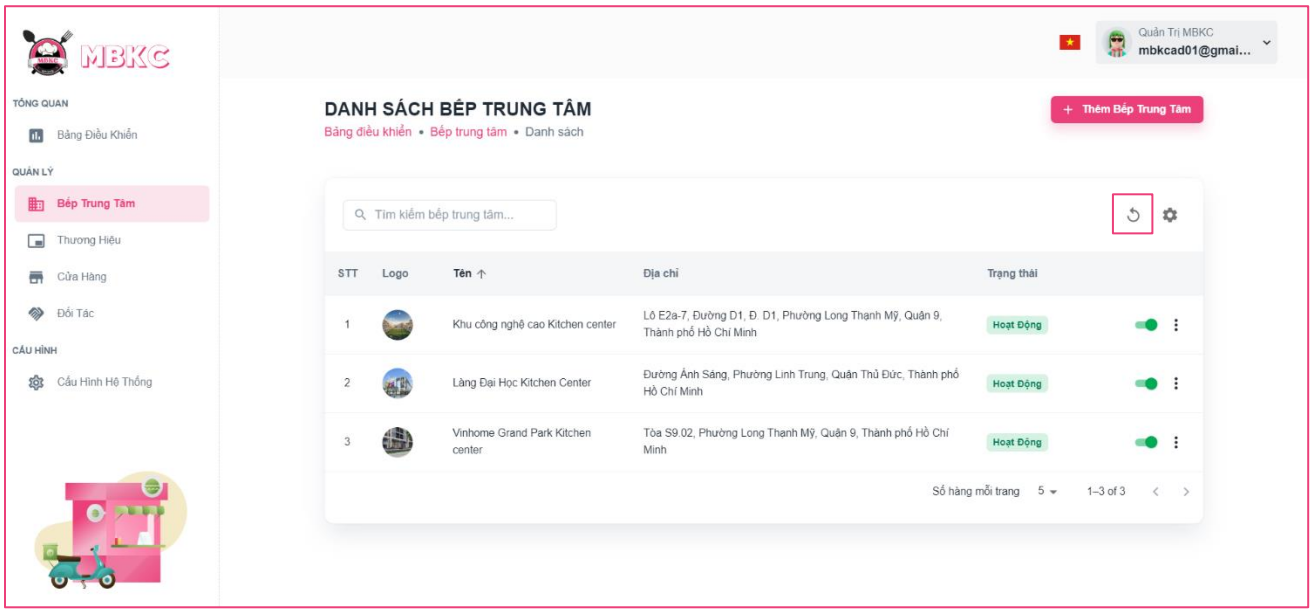


Figure 120: User Manual - View list Kitchen Center – Step 4 – Case 2

Case 3: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of the kitchen center.

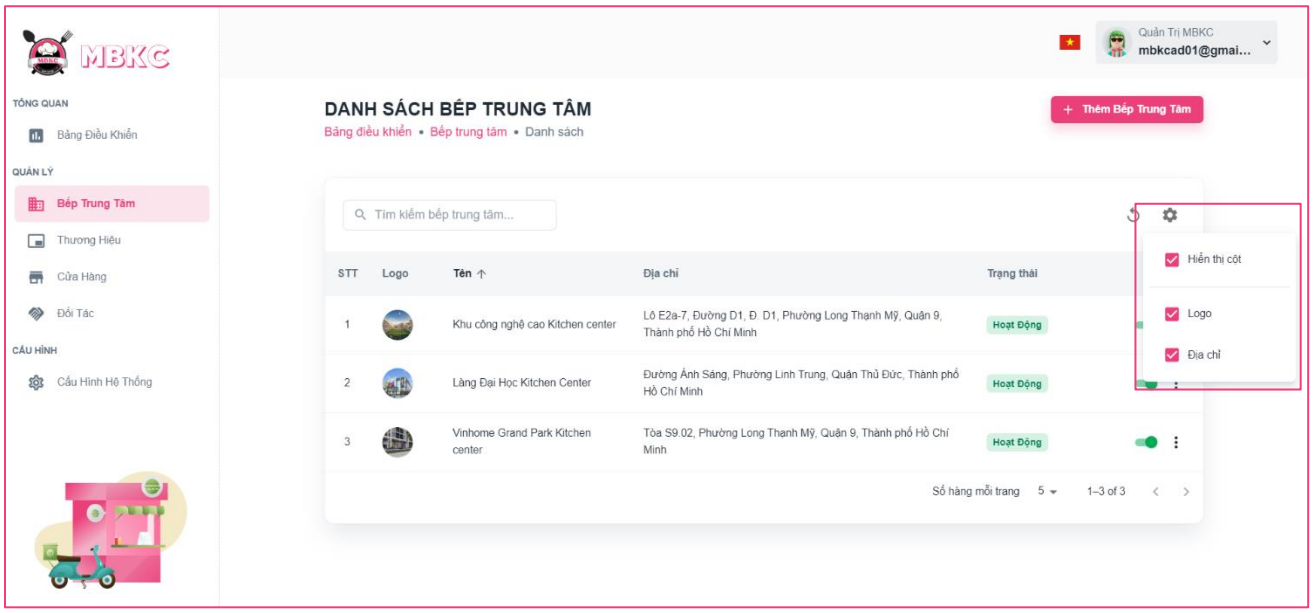


Figure 121: User Manual - View list Kitchen Center – Step 4 – Case 3

3.2.2.3. View details Kitchen Center

3.2.2.3.1. Description

Use this guideline to view details kitchen center.

3.2.2.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

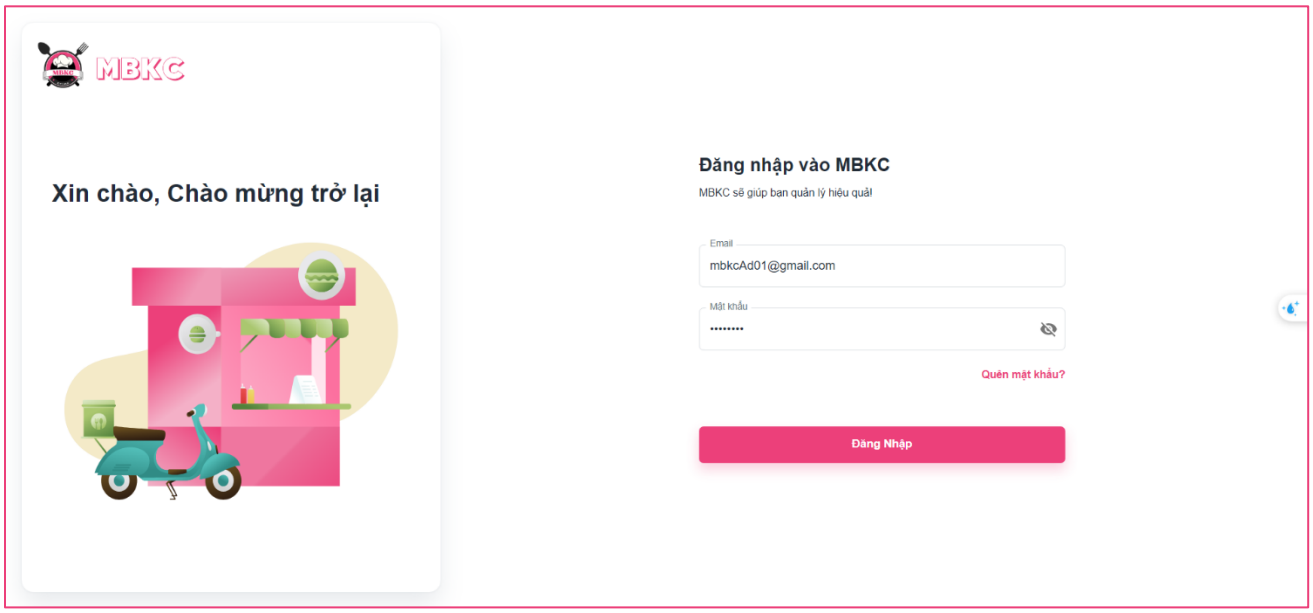


Figure 122: User Manual - View details Kitchen Center – Step 1

Step 2: Click “Đăng Nhập” button.

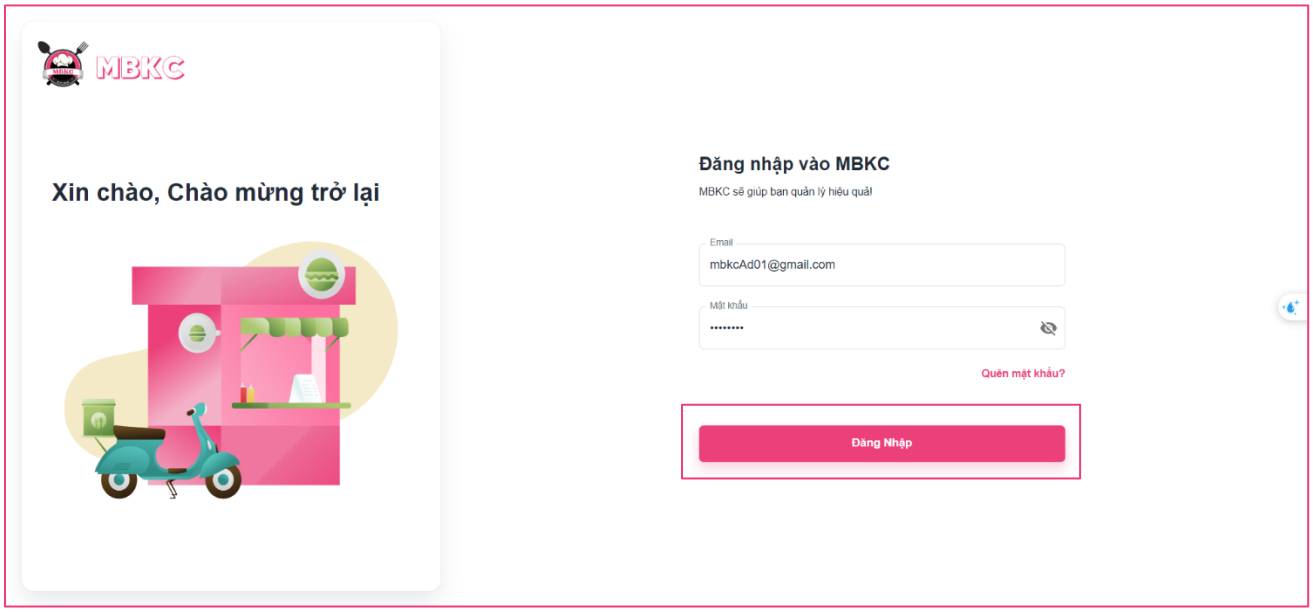


Figure 123: User Manual - View details Kitchen Center – Step 2

Step 3: From Admin Dashboard Page, choose “Bếp Trung Tâm” from the left sidebar.

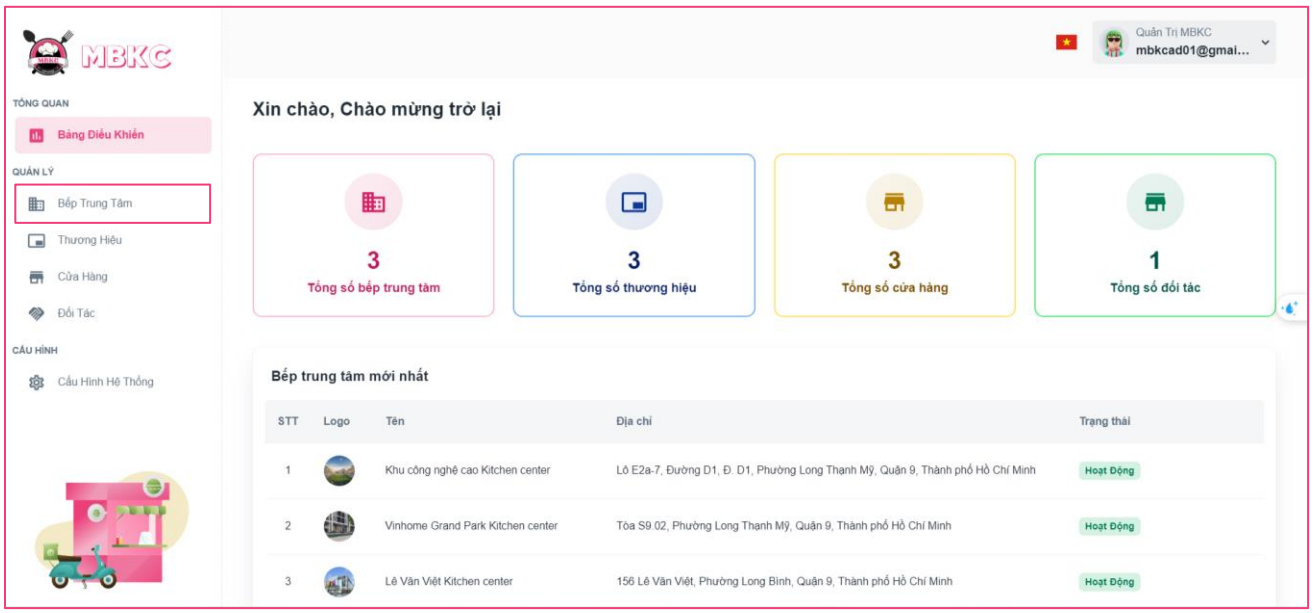


Figure 124: User Manual - View details Kitchen Center – Step 3

Step 4: Click the kitchen center row to navigate to Kitchen Center Details Page.

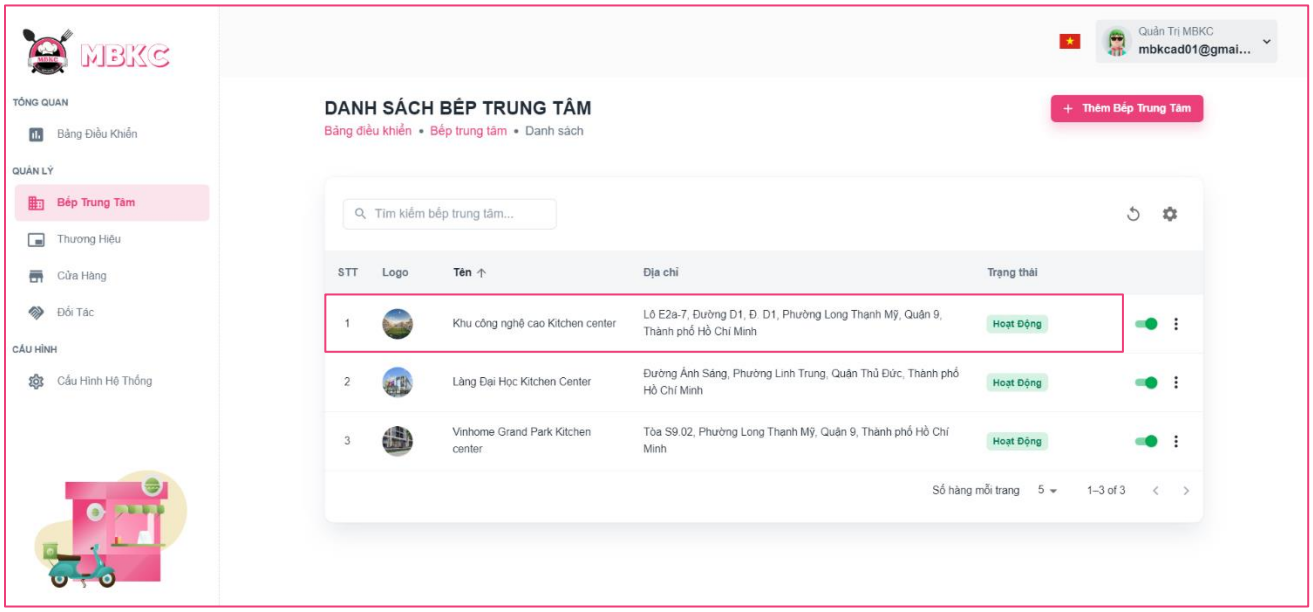


Figure 125: User Manual - View details Kitchen Center – Step 4

Step 5: Navigate to Kitchen Center Detail Page to show information details of kitchen center.

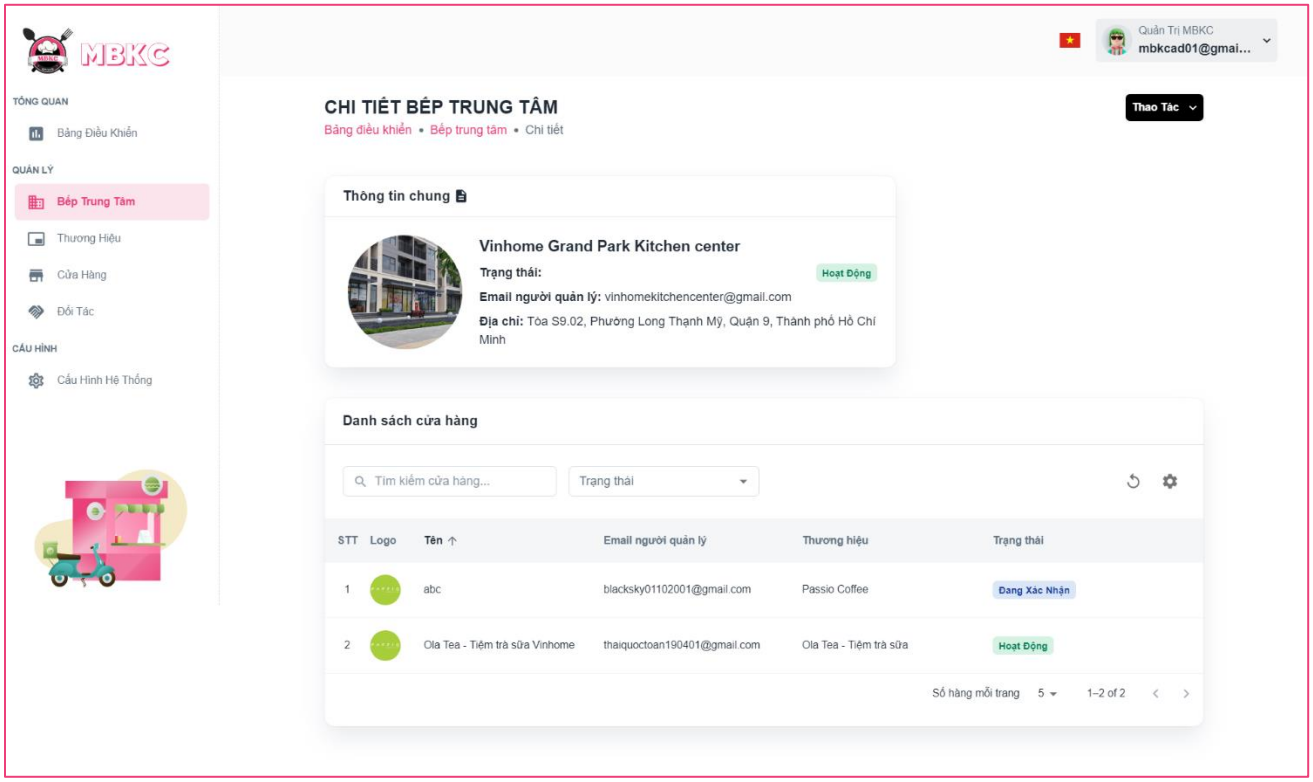


Figure 126: User Manual - View details Kitchen Center – Step 5

3.2.2.4. Update Status Kitchen Center

3.2.2.4.1. Description

Use this guideline to update the status of the kitchen center.

3.2.2.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

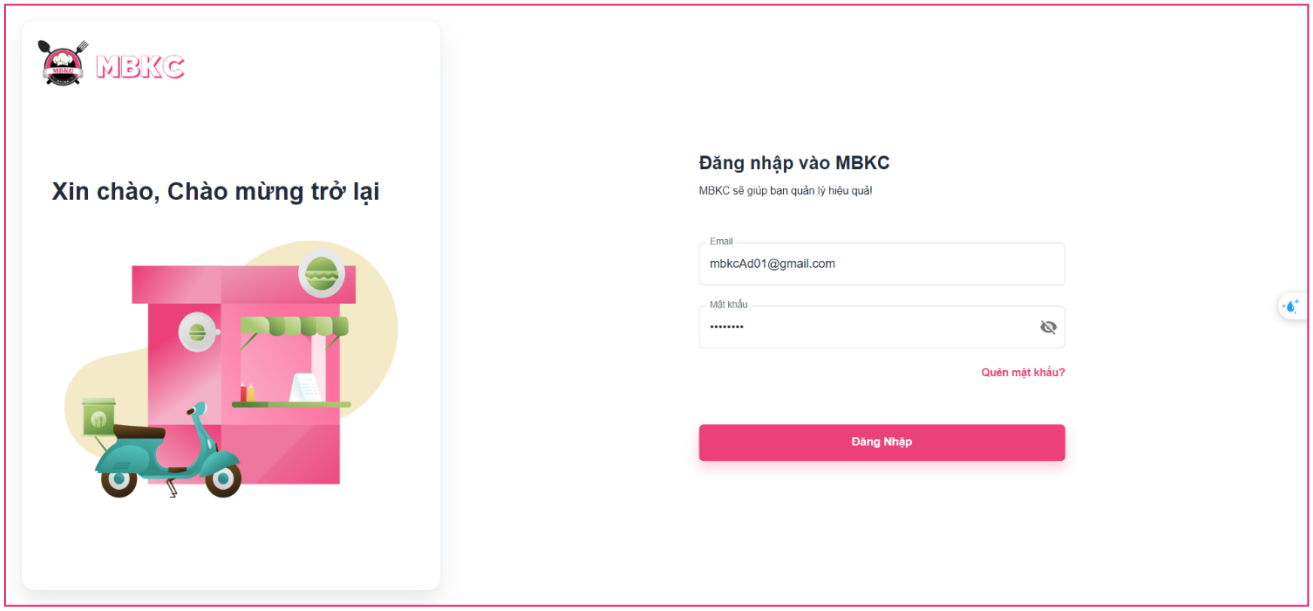


Figure 127: User Manual – Update Status Kitchen Center – Step 1

Step 2: Click “Đăng Nhập” button.

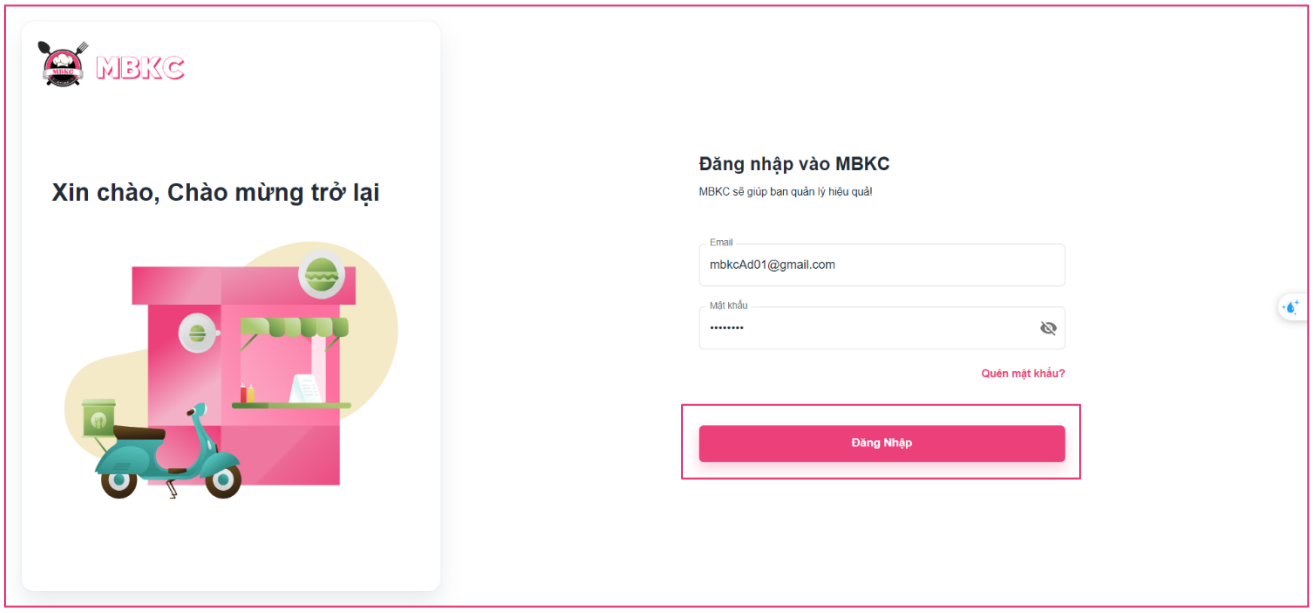


Figure 128: User Manual - Update Status Kitchen Center – Step 2

Step 3: From Admin Dashboard Page, choose “Bếp Trung Tâm” from the left sidebar.

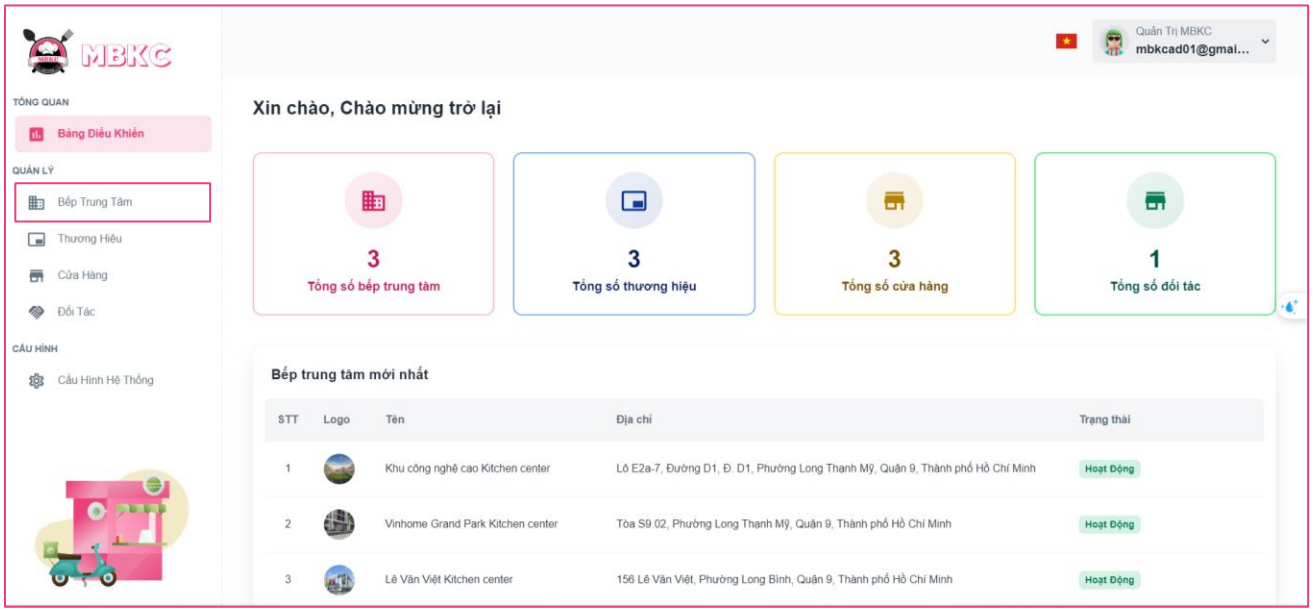


Figure 129: User Manual - Update Status Kitchen Center – Step 3

Step 4: Click on the switch button on the kitchen center row to update the status.

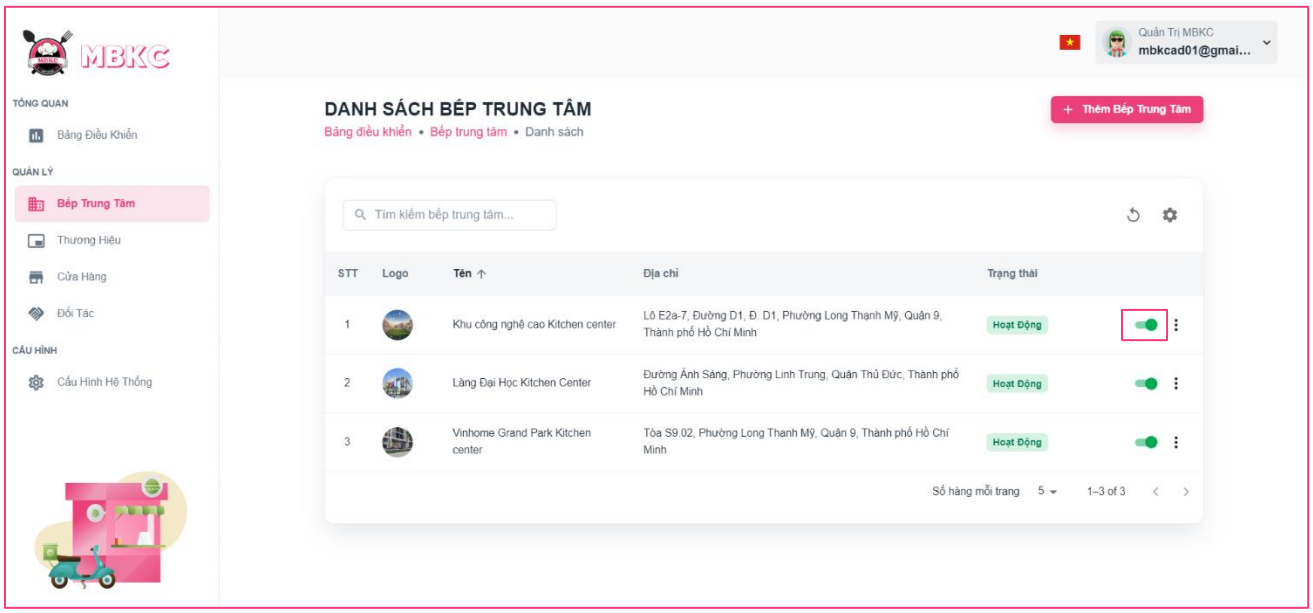


Figure 130: User Manual - Update Status Kitchen Center – Step 4

Step 5: List of kitchen centers will be reloaded and a toast message “Cập nhật trạng thái bếp trung tâm thành công”. And the status of the kitchen center has been updated.

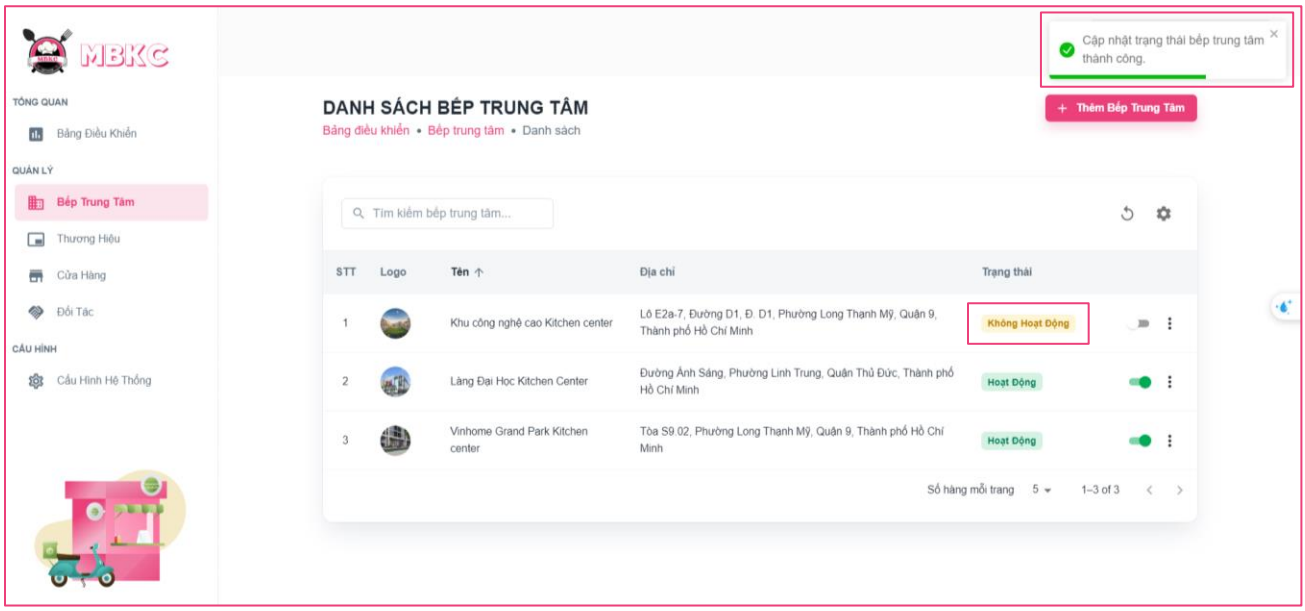


Figure 131: User Manual - Update Status Kitchen Center – Step 5

3.2.2.5. Update Information Kitchen Center

3.2.2.5.1. Description

Use this guideline to update information of the kitchen center.

3.2.2.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

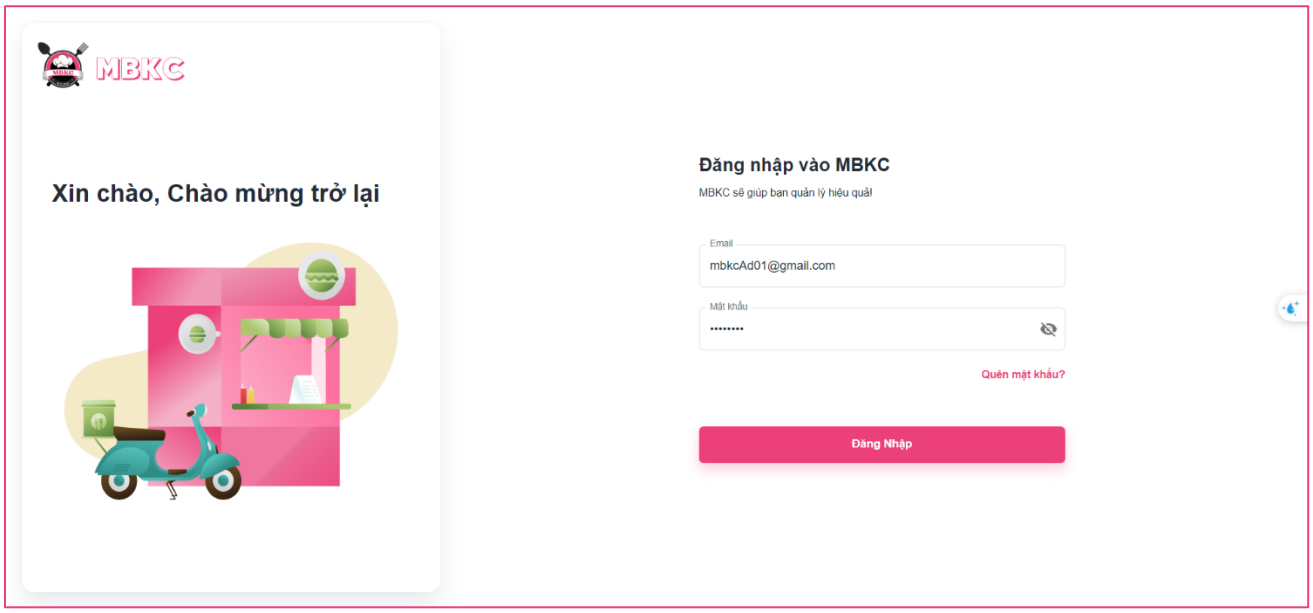


Figure 132: User Manual – Update Information Kitchen Center – Step 1

S

Step 2: Click “Đăng Nhập” button.

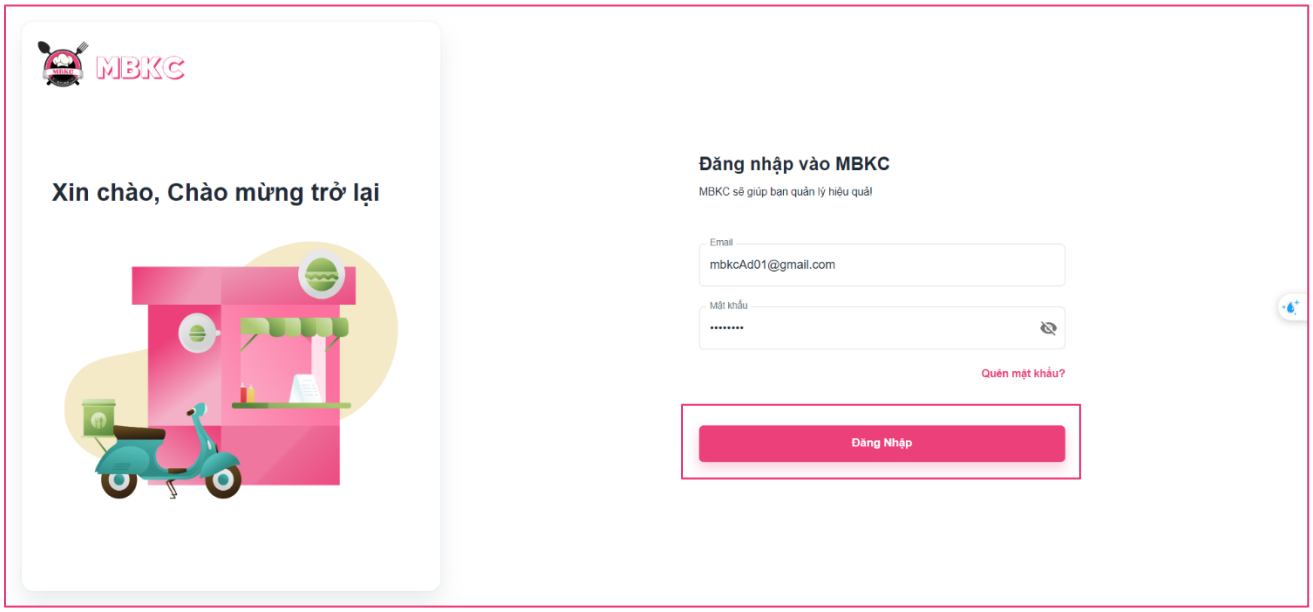


Figure 133: User Manual - Update Information Kitchen Center – Step 2

Step 3: From Admin Dashboard Page, choose “Bếp Trung Tâm” from the left sidebar.

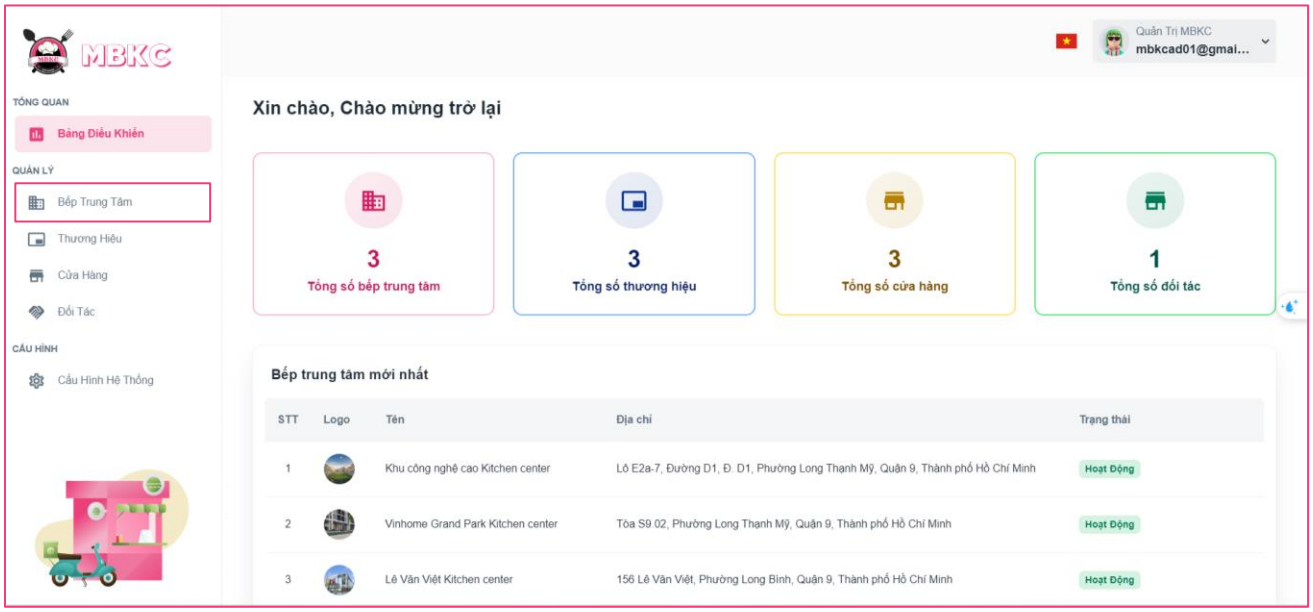


Figure 134: User Manual - Update Information Kitchen Center – Step 3

Step 4: Update information of the kitchen center.

Case 1: Update information of the kitchen center in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the kitchen center row want to navigate Update Kitchen Center Page.

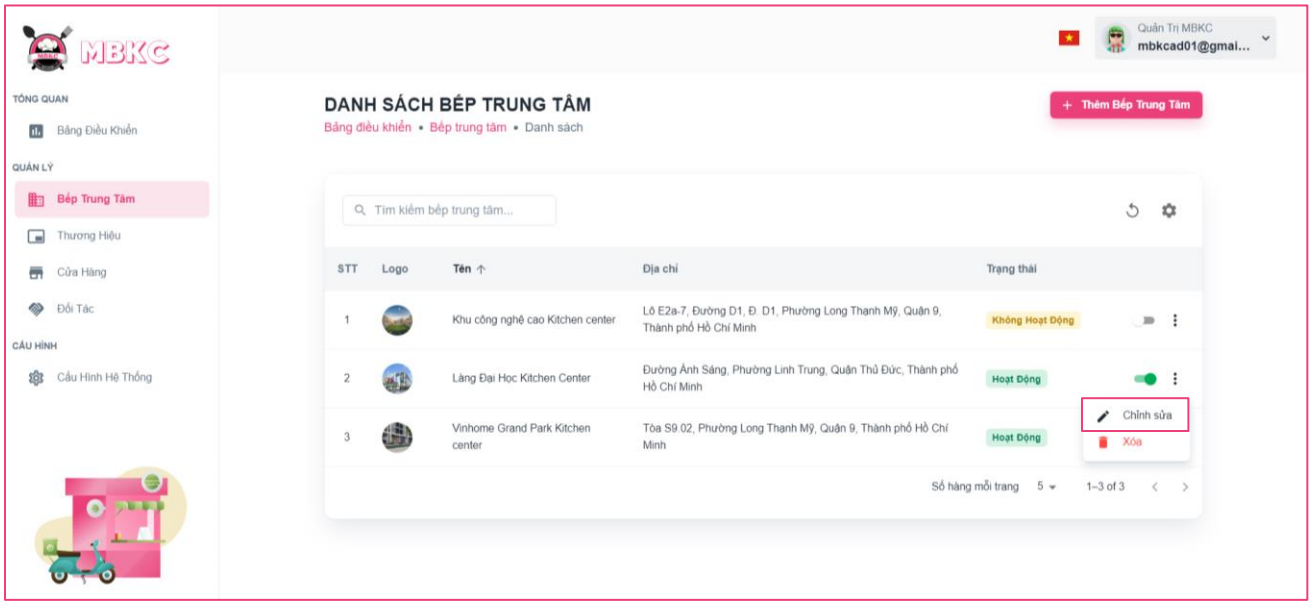


Figure 135: User Manual - Update Information Kitchen Center – Step 4 – Case 1

Case 2: Update information of the kitchen center on the details page.

1: Click the kitchen center row to navigate to Kitchen Center Details Page.

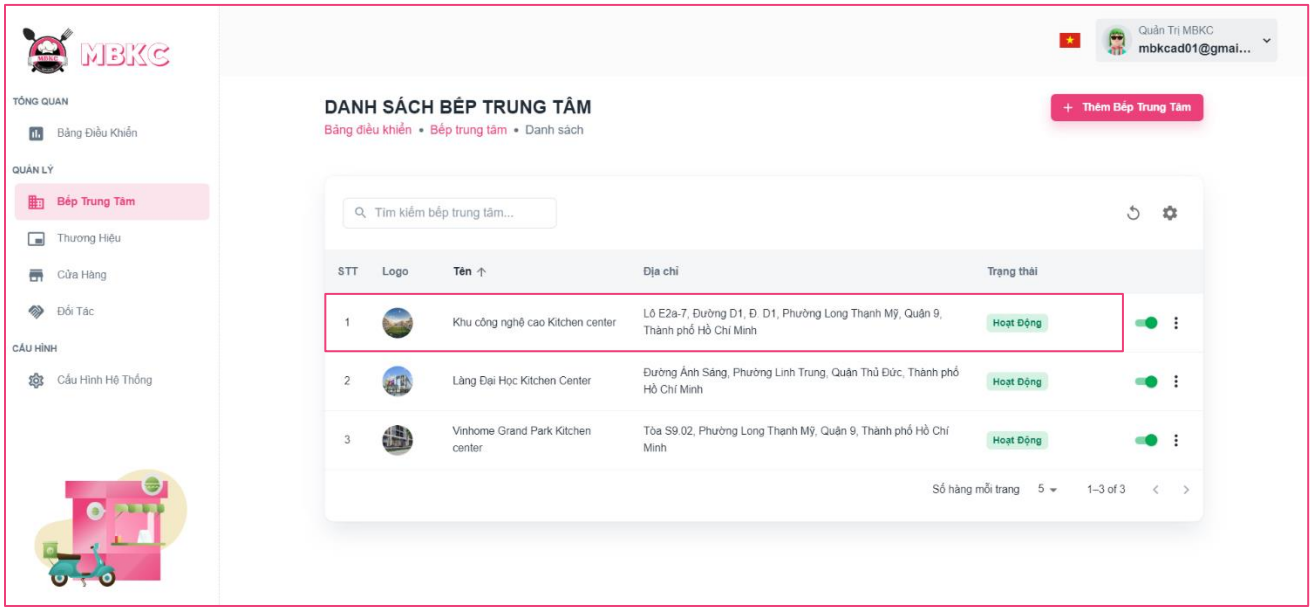


Figure 136: User Manual - Update Information Kitchen Center – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Chỉnh sửa” menu item to navigate Update Kitchen Center Page.

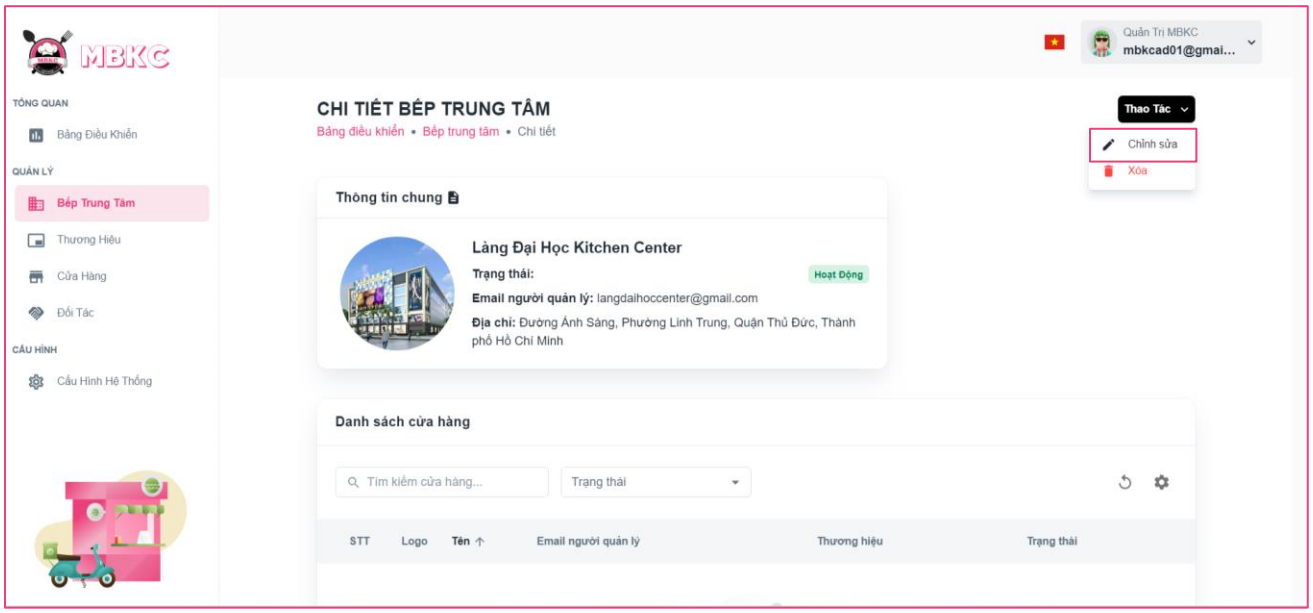


Figure 137: User Manual - Update Information Kitchen Center – Step 4 – Case 2 - 2

Step 5: Kitchen center information will be automatically filled in the form. Then edit the information as desired.

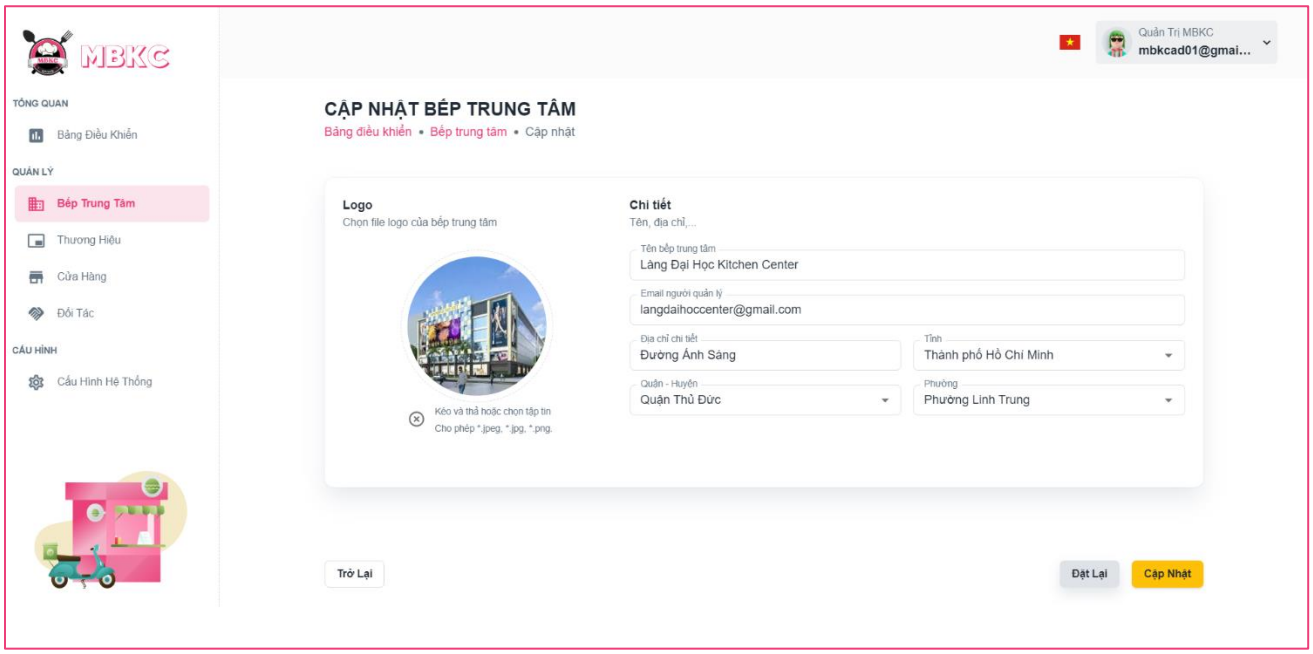


Figure 138: User Manual - Update Information Kitchen Center – Step 5

Step 6: Click “Cập nhật” button.

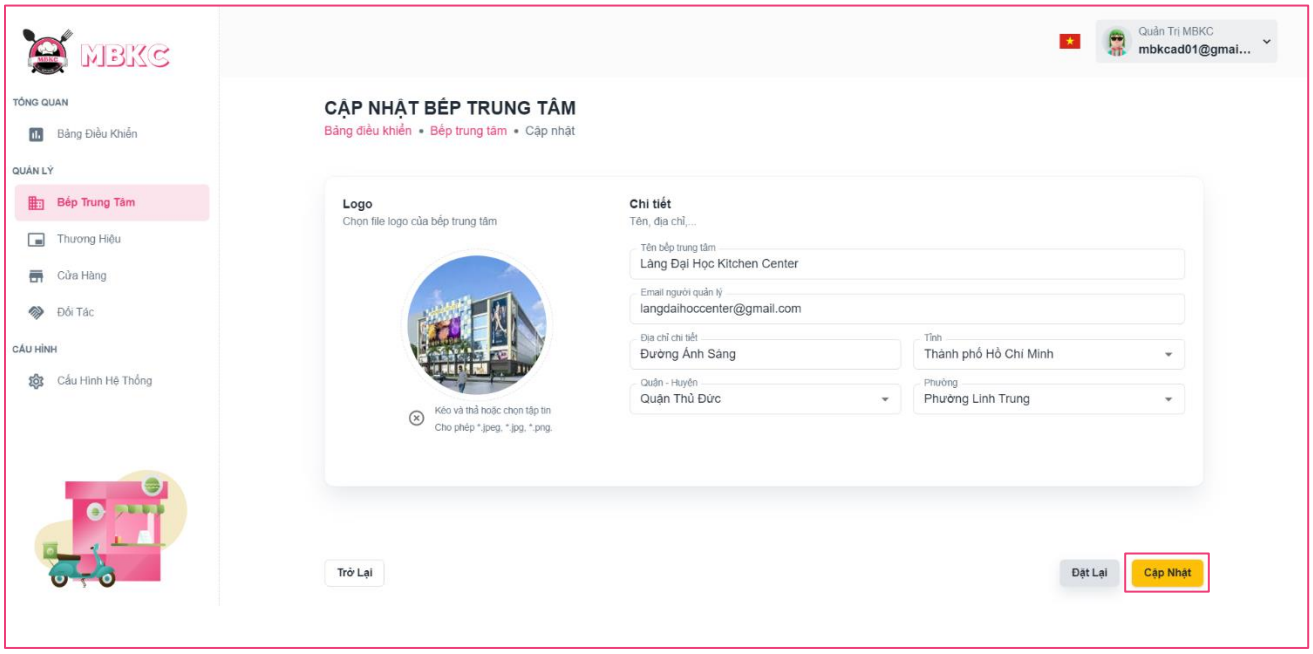


Figure 139: User Manual - Update Information Kitchen Center – Step 6

Step 7:

Case 1: You will be navigated to Kitchen Center Detail Page showing the kitchen center details when start edit information of kitchen center from Kitchen Center Detail Page and a toast message “Cập nhật thông tin bếp trung tâm thành công”.

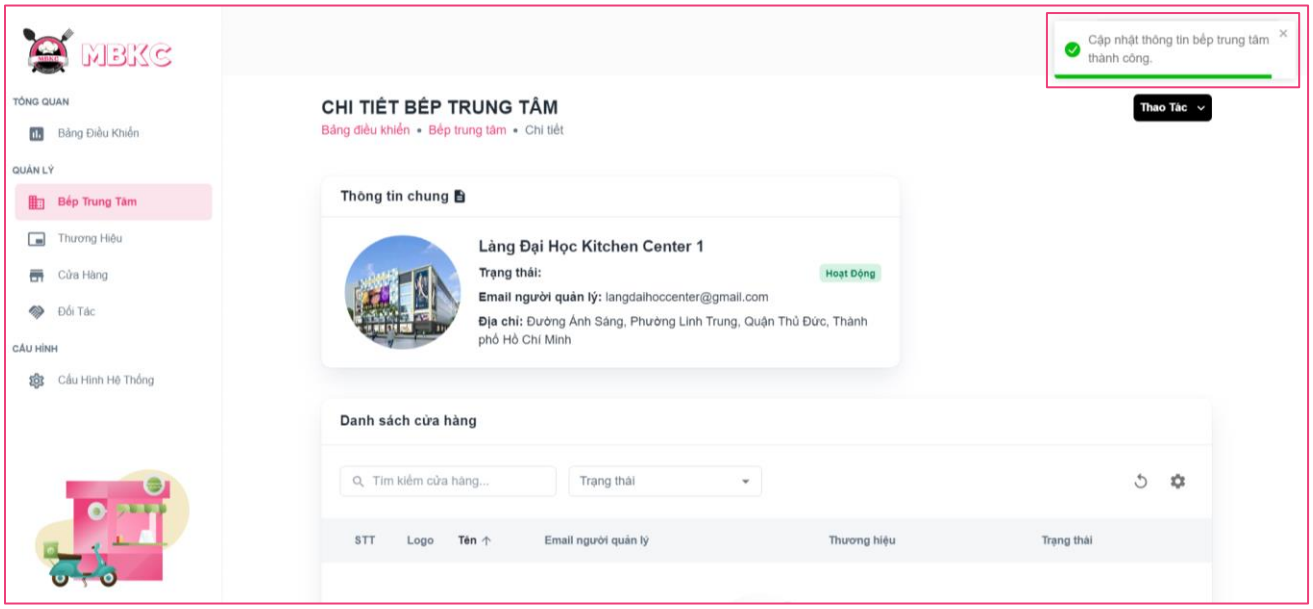


Figure 140: User Manual - Update Information Kitchen Center – Step 7 – Case 1

Case 2: You will be navigated to List of Kitchen Center Page showing the list of kitchen centers when start edit information of kitchen center from List of Kitchen Center Page and a toast message “Cập nhật thông tin bếp trung tâm thành công”.

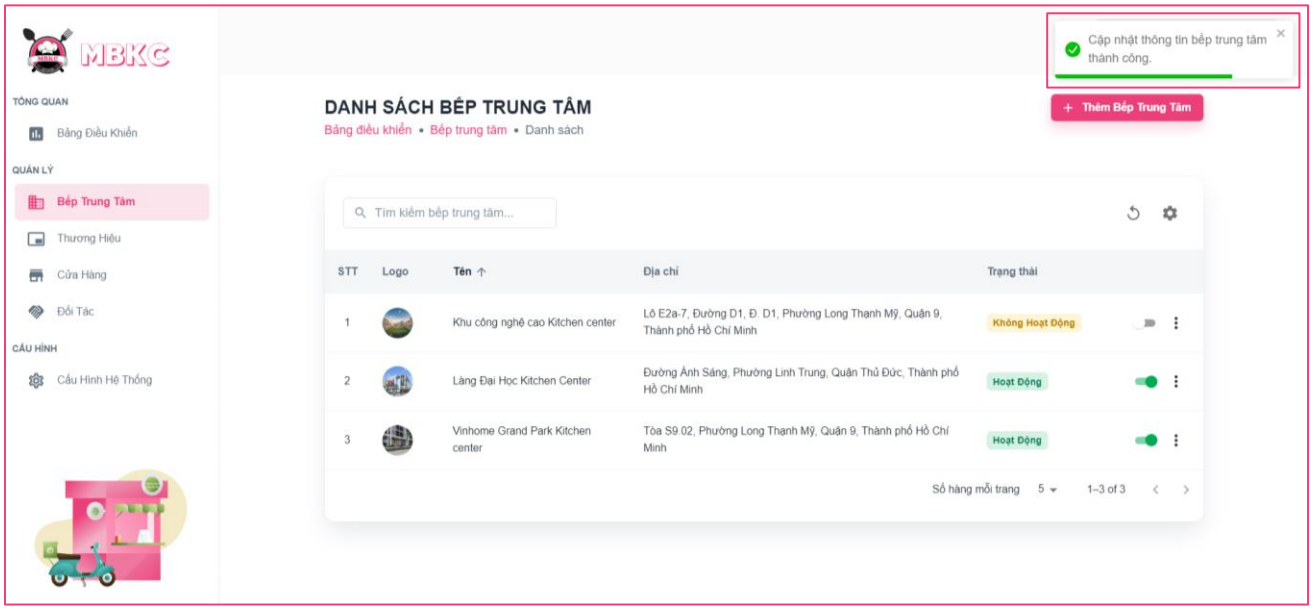


Figure 141: User Manual - Update Information Kitchen Center – Step 7 – Case 2

3.2.2.6. Delete Kitchen Center

3.2.2.6.1. Description
Use this guideline to delete kitchen center.

3.2.2.6.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

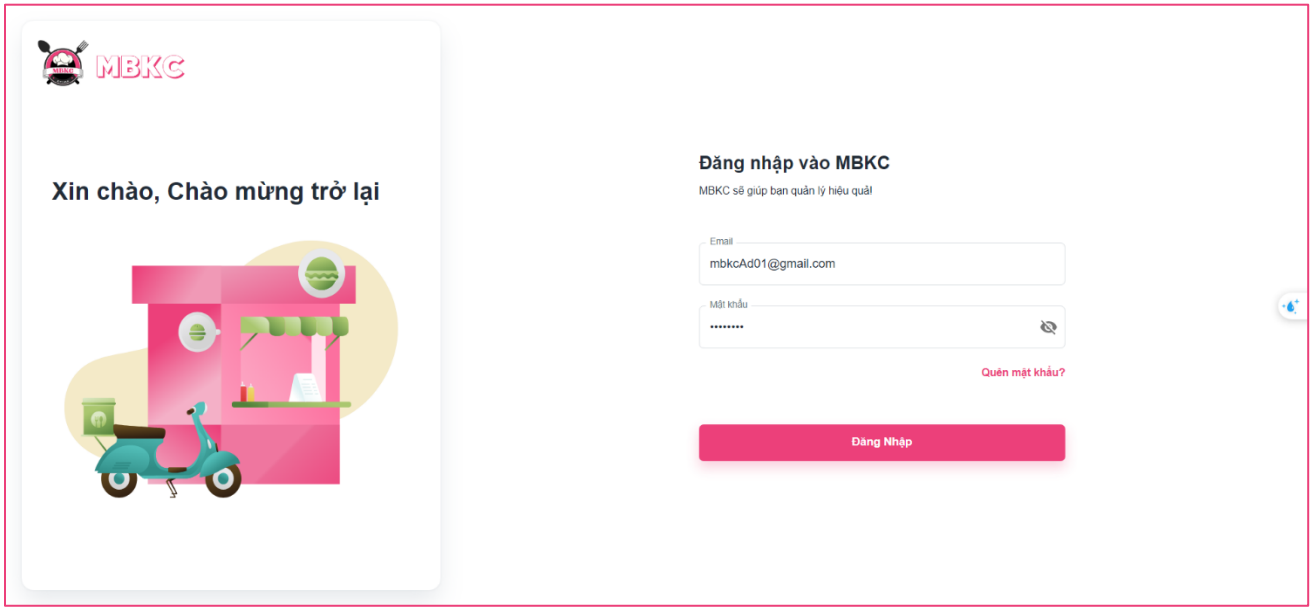


Figure 142: User Manual – Delete Kitchen Center – Step 1

Step 2: Click “Đăng Nhập” button.

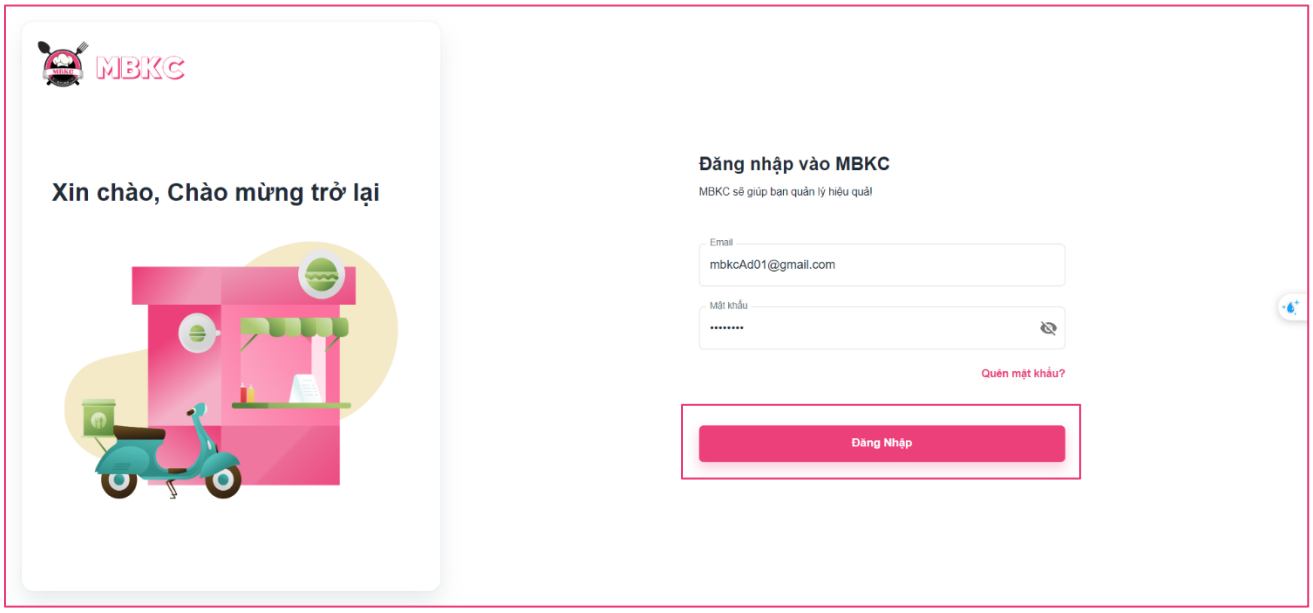


Figure 143: User Manual - Delete Kitchen Center – Step 2

Step 3: From Admin Dashboard Page, choose “Bếp Trung Tâm” from the left sidebar.

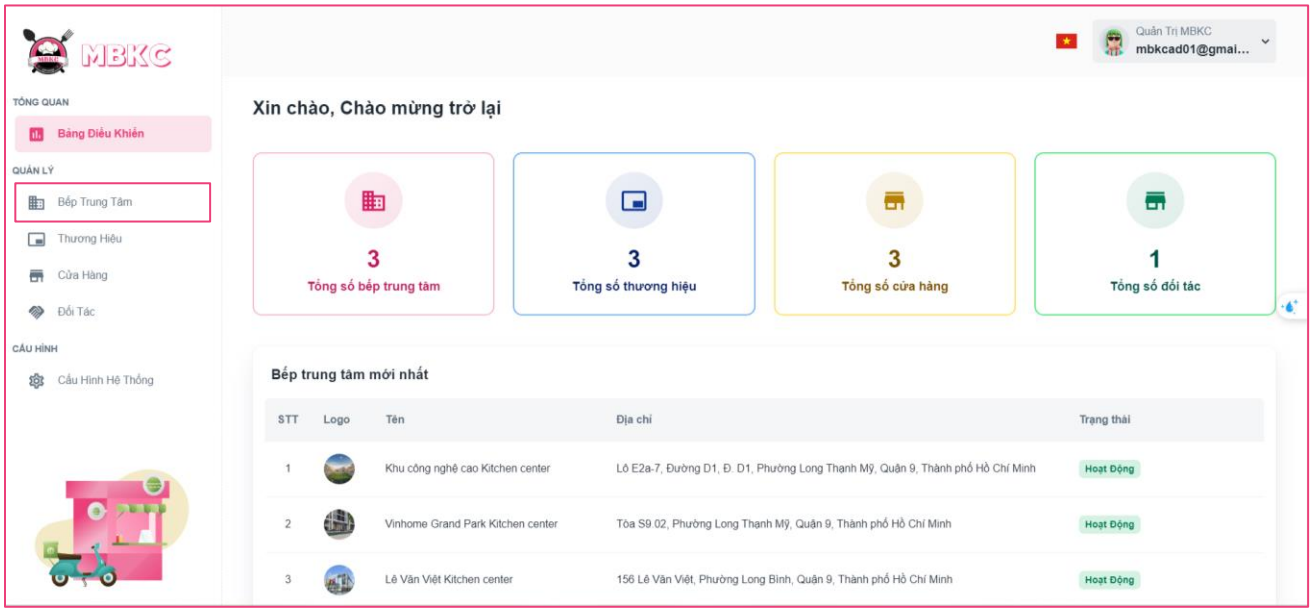


Figure 144: User Manual - Delete Kitchen Center – Step 3

Step 4: Delete kitchen center.

Case 1: Delete kitchen center in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the kitchen center row to delete.

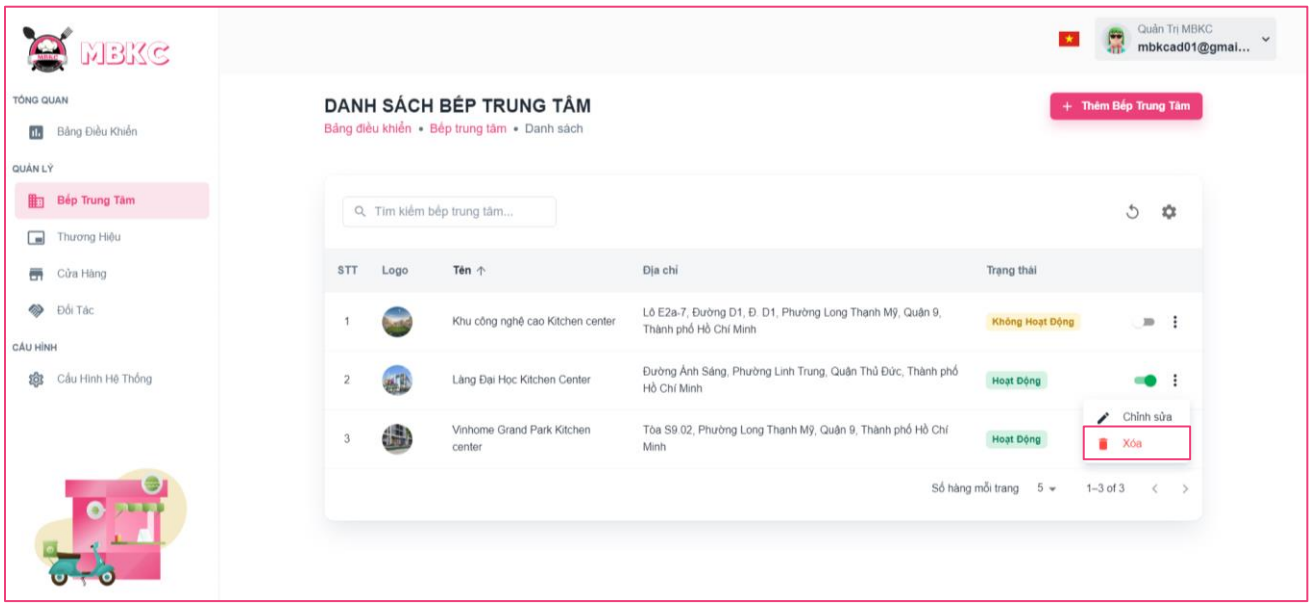


Figure 145: User Manual - Delete Kitchen Center – Step 4 – Case 1

Case 2: Delete kitchen center on the details page.

1: Click the kitchen center row to navigate to Kitchen Center Details Page.

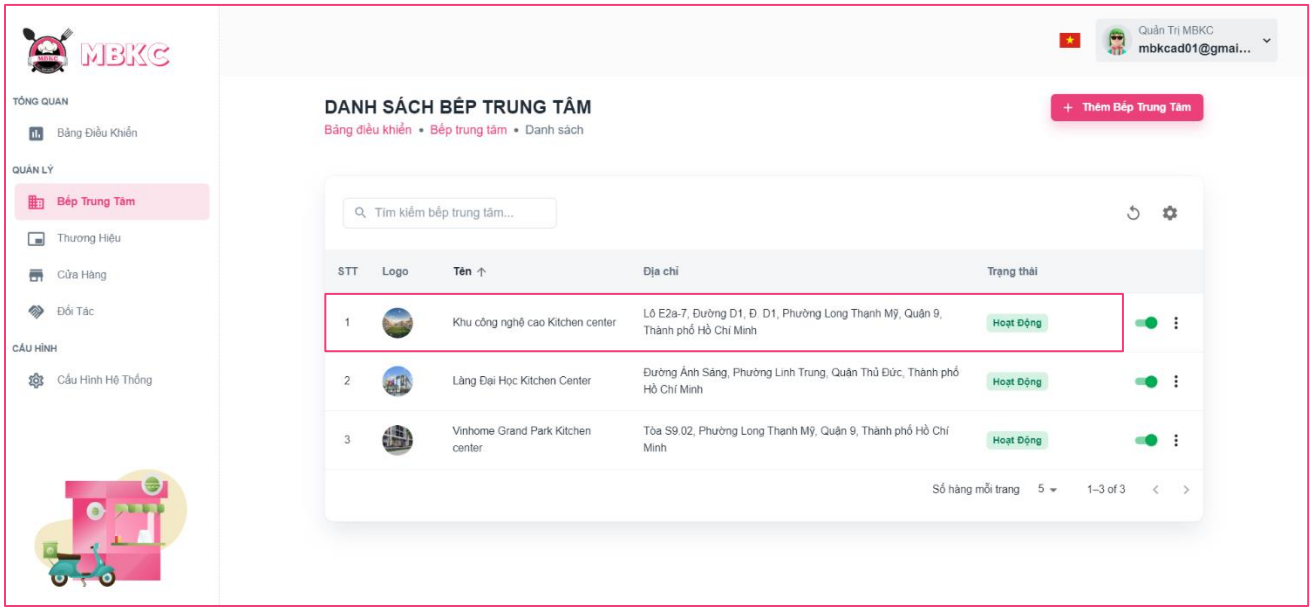


Figure 146: User Manual - Delete Kitchen Center – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item to delete.

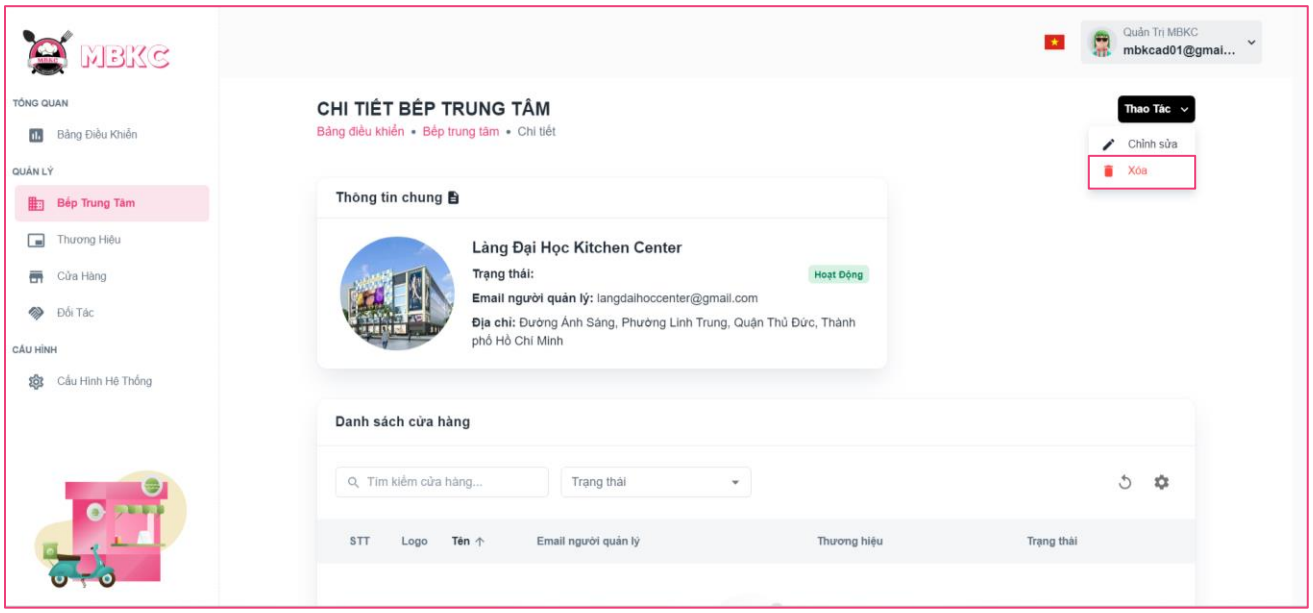


Figure 147: User Manual - Delete Kitchen Center – Step 4 – Case 2 - 2

Step 5: Click “Xác Nhận” button.

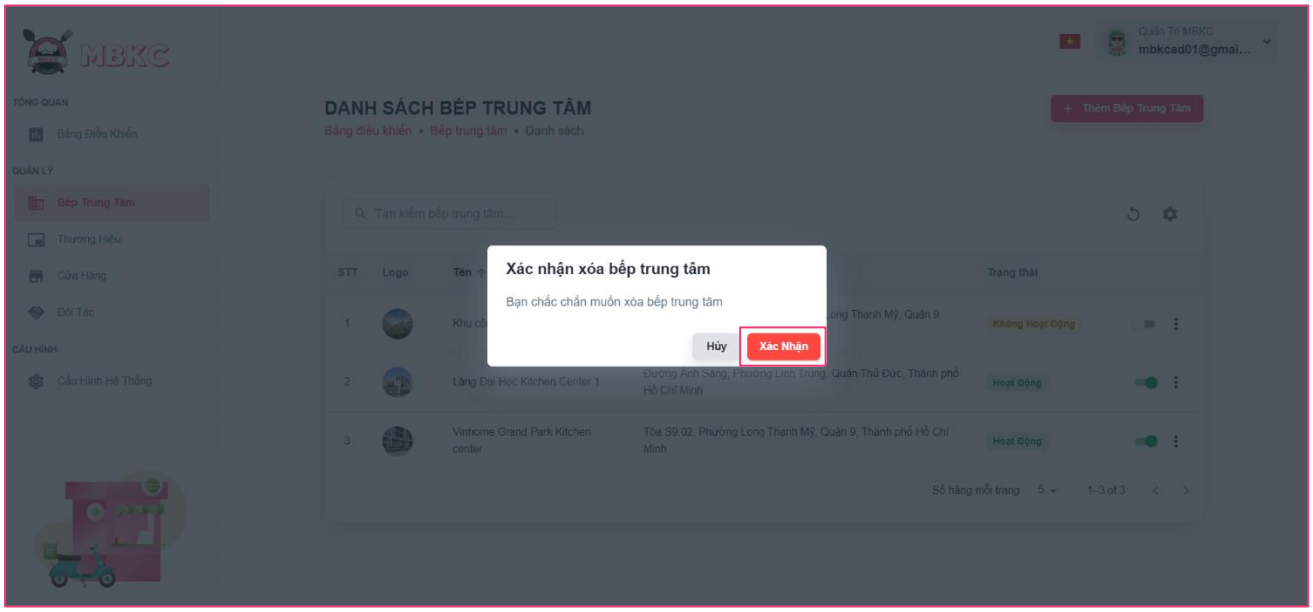


Figure 148: User Manual - Delete Kitchen Center – Step 5

Step 6: List of kitchen centers will be reloaded and a toast message “Đã xóa bếp trung tâm thành công”. And kitchen center has been deleted.

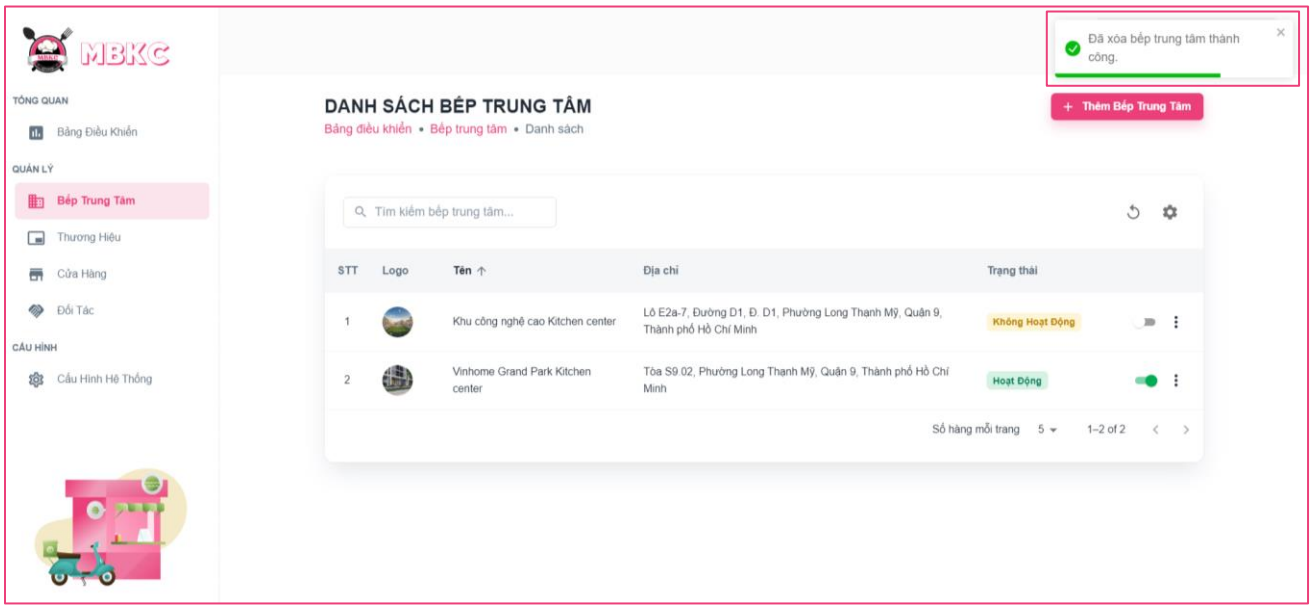


Figure 149: User Manual – Delete Kitchen Center – Step 6

3.2.3. Brand management

3.2.3.1. Create new Brand

3.2.3.1.1. Description

Use this guideline to login into MBKC account to create a new brand.

3.2.3.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

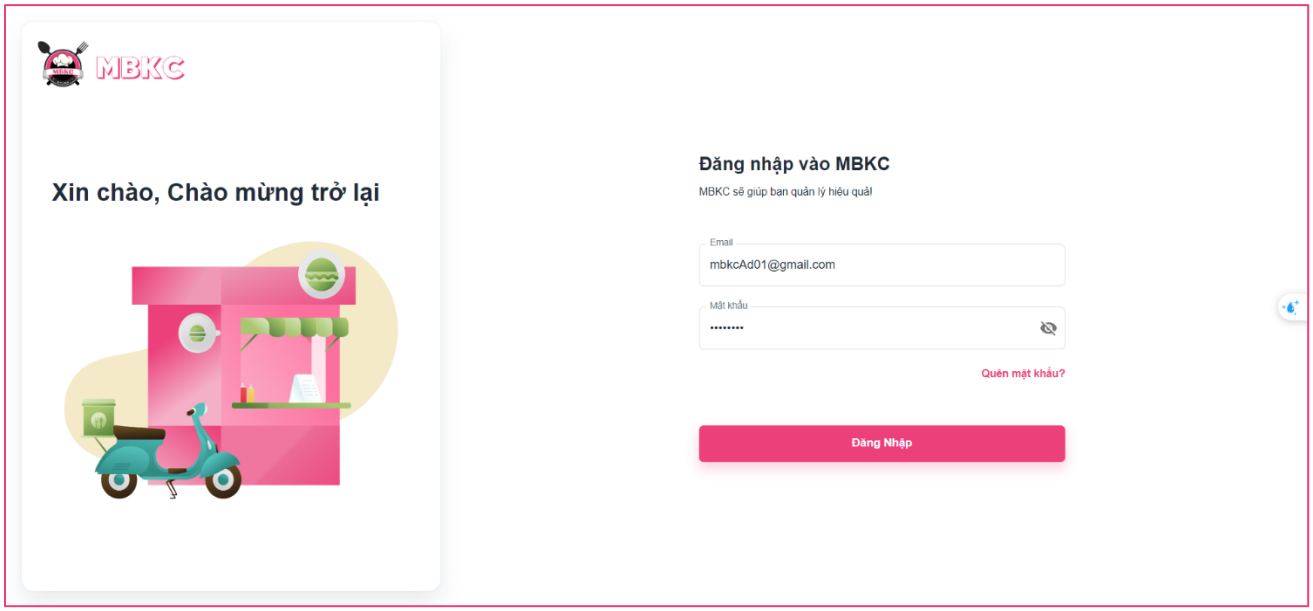


Figure 150: User Manual - Create new Brand – Step 1

Step 2: Click “Đăng Nhập” button.

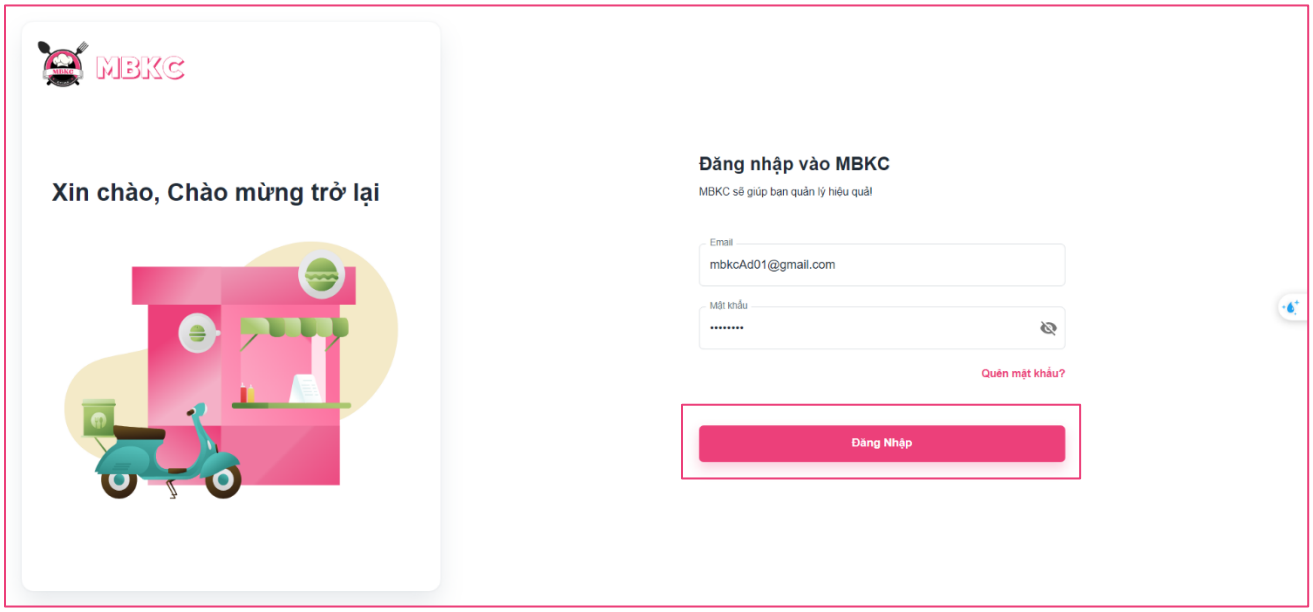


Figure 151: User Manual - Create new Brand – Step 2

Step 3: From Admin Dashboard Page, choose “Thương Hiệu” from the left sidebar.

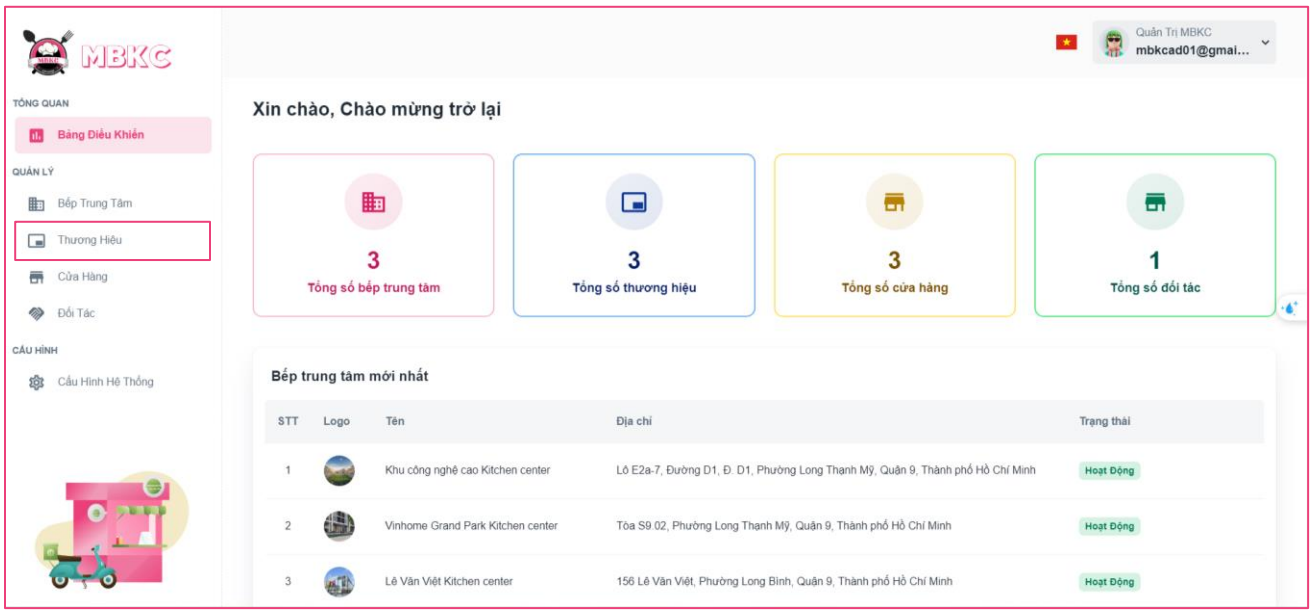


Figure 152: User Manual - Create new Brand – Step 3

Step 4: Click “Thêm Thương Hiệu” button on the screen and navigate to Create New Brand Page to add a new brand.

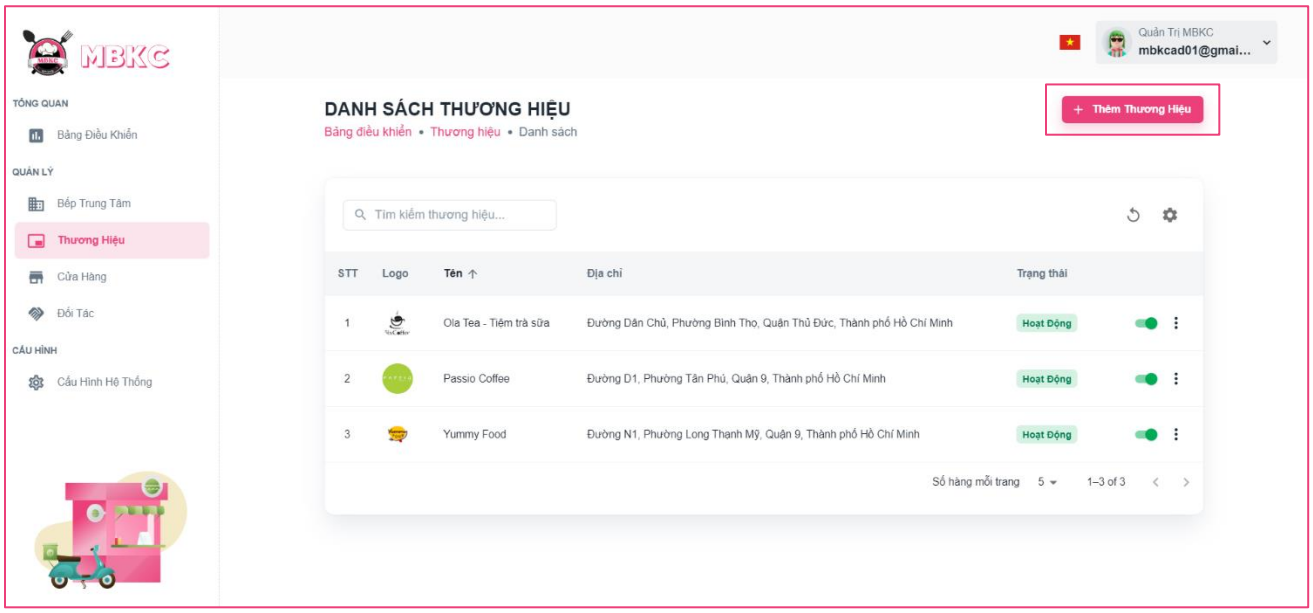


Figure 153: User Manual - Create new Brand – Step 4

Step 5: Filling all the required information to add a brand including: “Logo”, “Tên thương hiệu", “Email người quản lý", “Địa chỉ chi tiết", “Tỉnh”, “Quận – Huyện" and “Phường”.

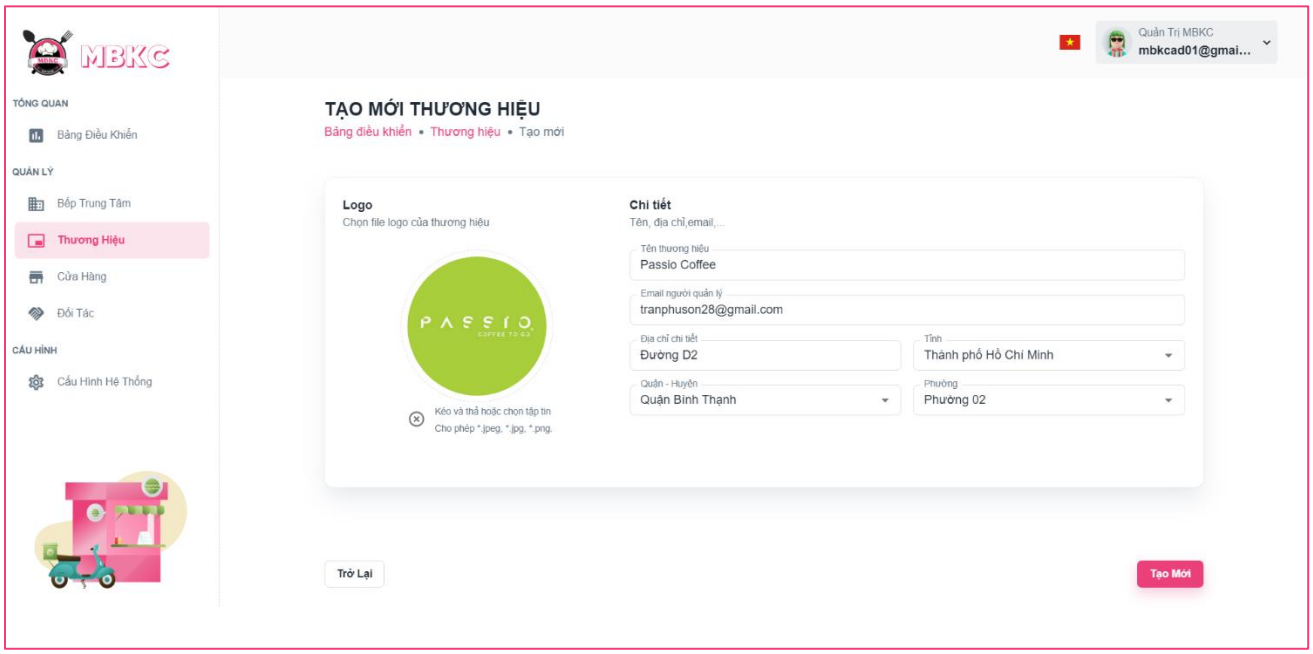


Figure 154: User Manual - Create new Brand – Step 5

Step 6: After filling in all required field, click “Tạo Mới” to finish creating a new brand.

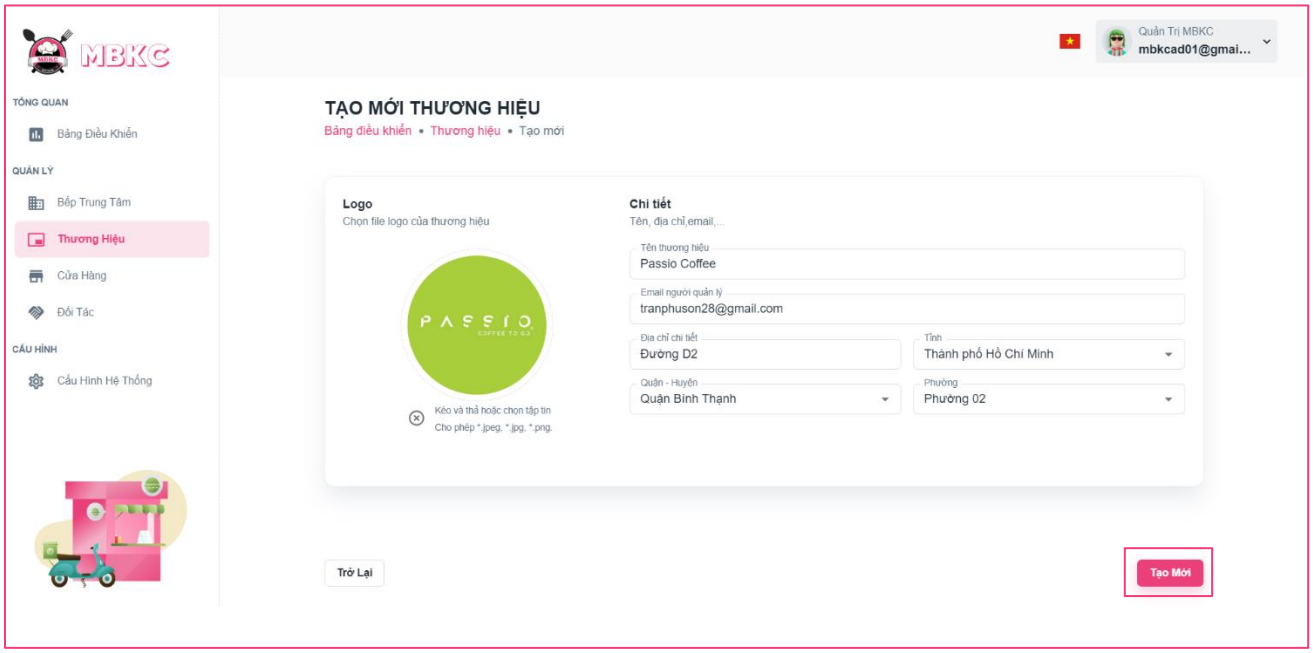


Figure 155: User Manual - Create new Brand – Step 6

Step 7: You will be navigated to List of Brand Page showing the list of brands and a toast message “Tạo mới thương hiệu thành công”.

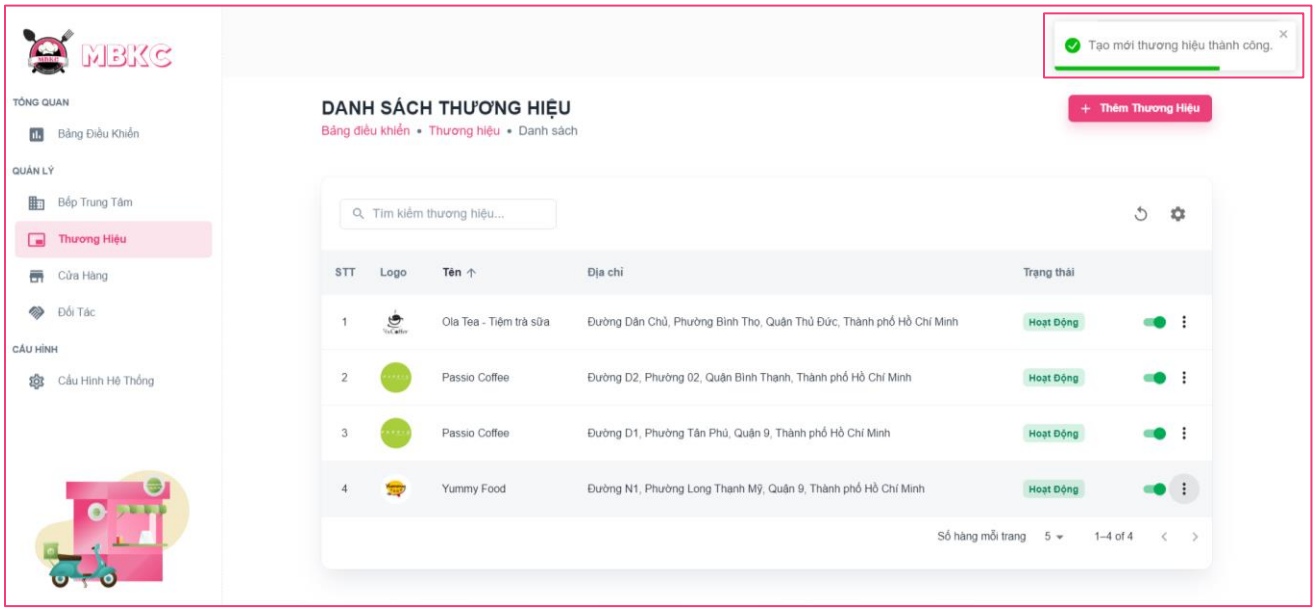


Figure 156: User Manual – Create new Brand – Step 7

3.2.3.2. View list Brand

3.2.3.2.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of the brand.

3.2.3.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

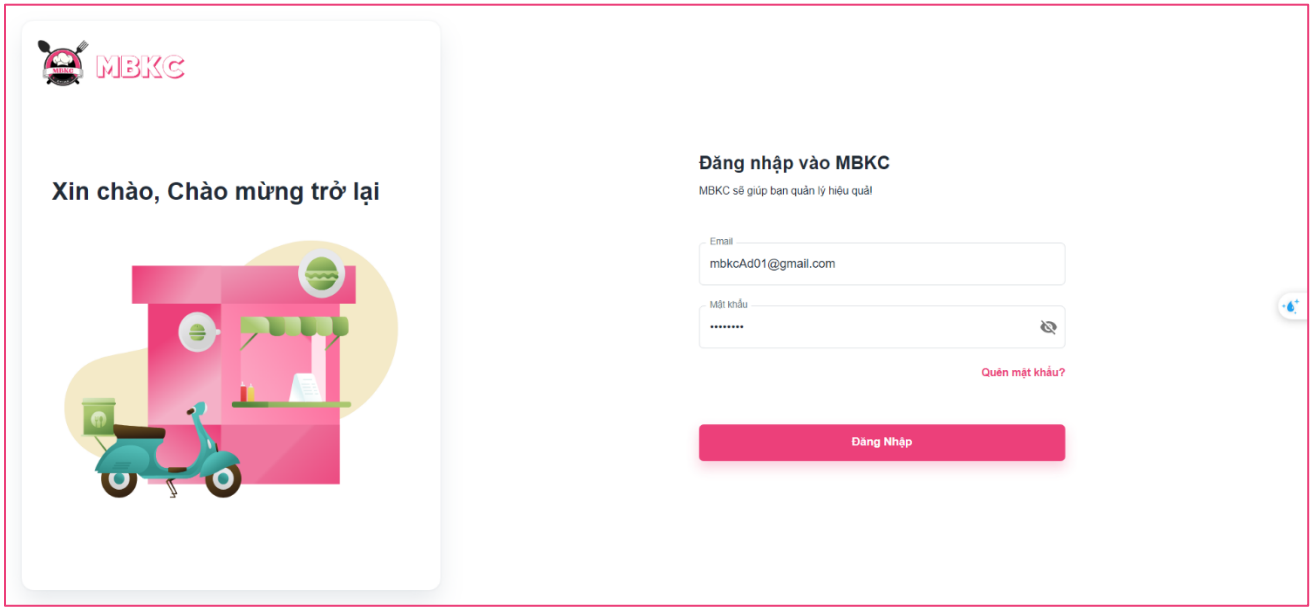


Figure 157: User Manual - View list Brand – Step 1

Step 2: Click “Đăng Nhập” button.

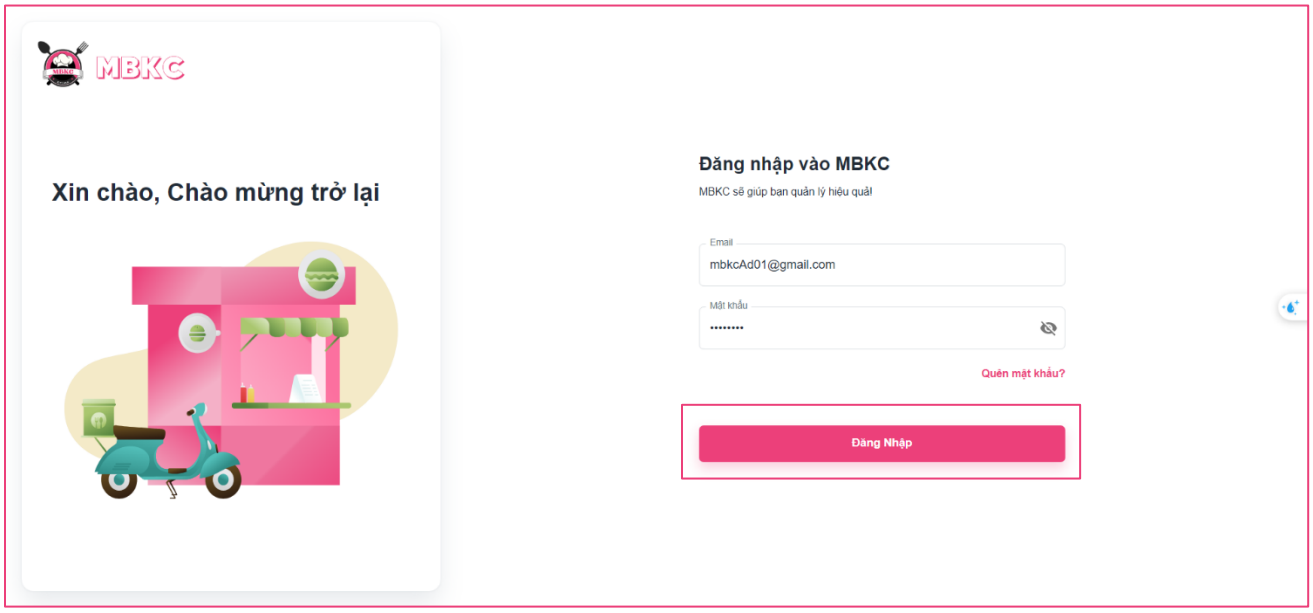


Figure 158: User Manual - View list Brand – Step 2

Step 3: From Admin Dashboard Page, choose “Thương Hiệu” from the left sidebar.

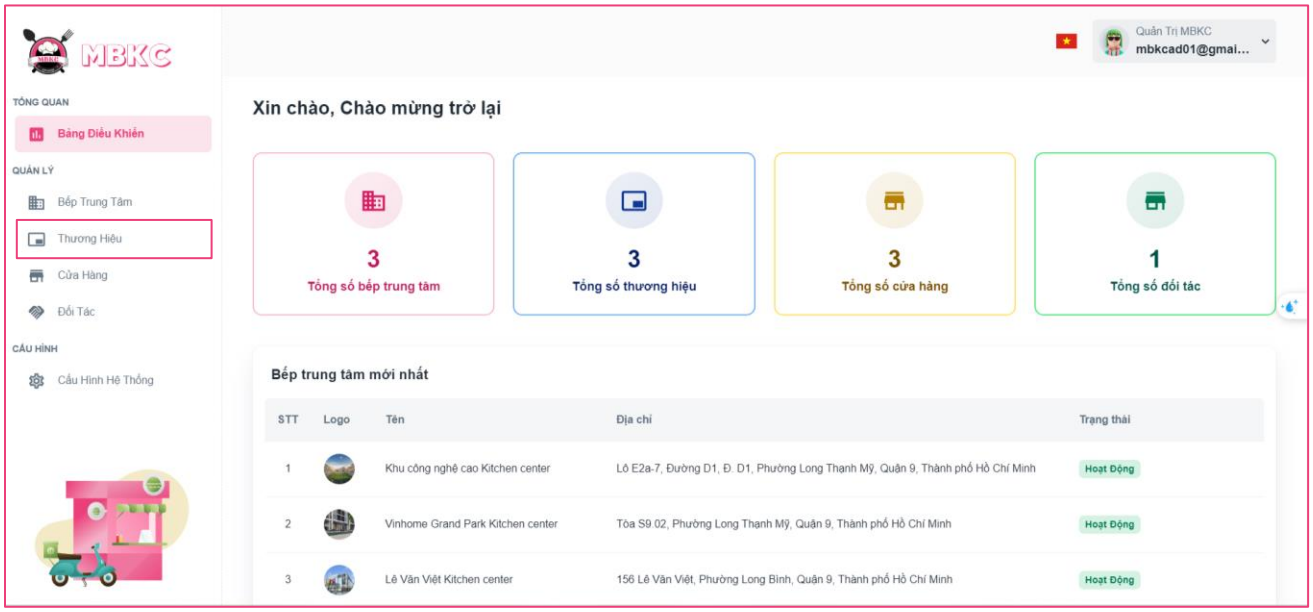


Figure 159: User Manual - View list Brand – Step 3

Step 4: Search, reload data, and show or hide columns in the table list of the brand.

Case 1: Fill in the brand name in the search box to search brands in the table list.

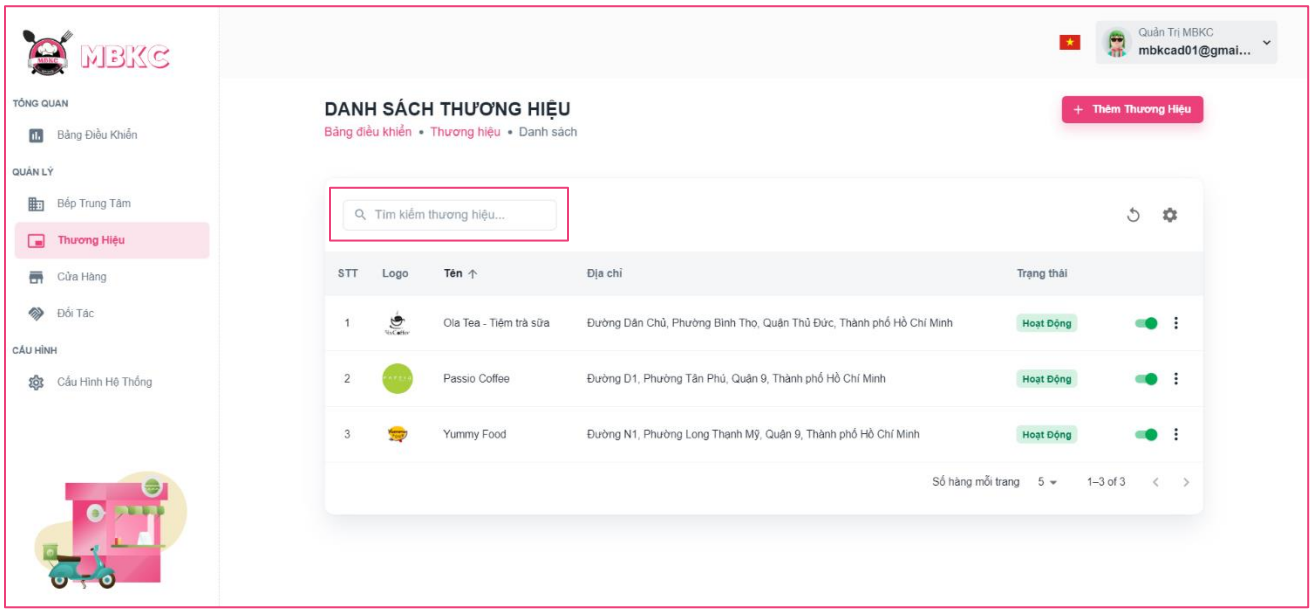


Figure 160: User Manual - View list Brand – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of brands.

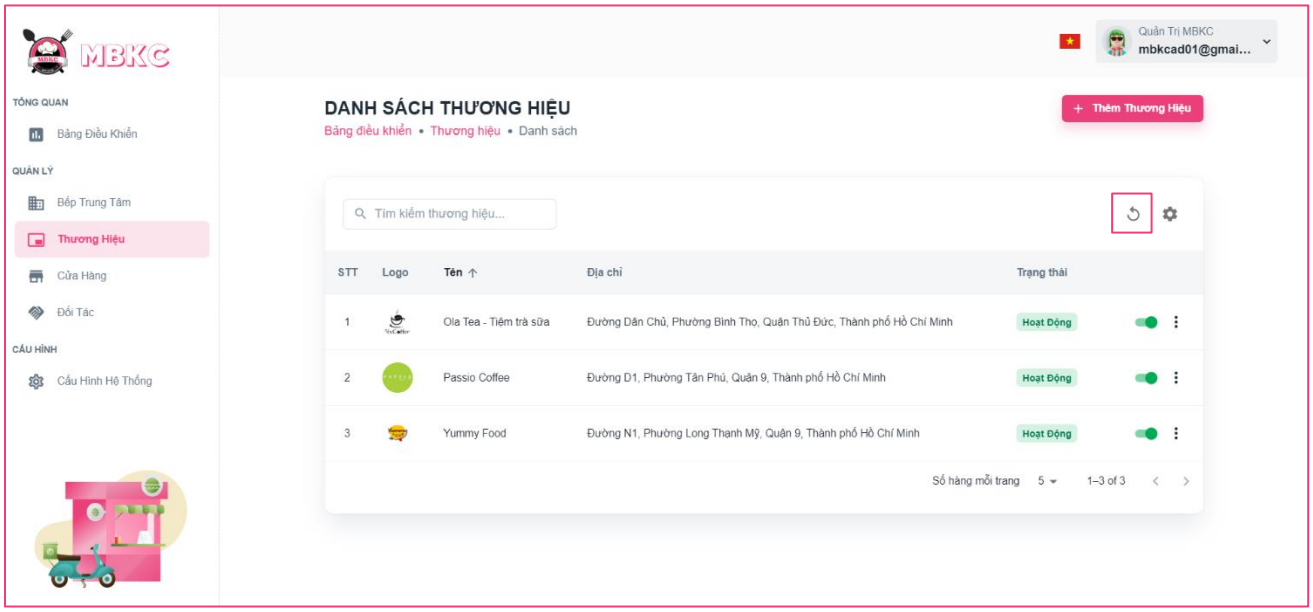


Figure 161: User Manual - View list Brand – Step 4 – Case 2

Case 3: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of the brand.

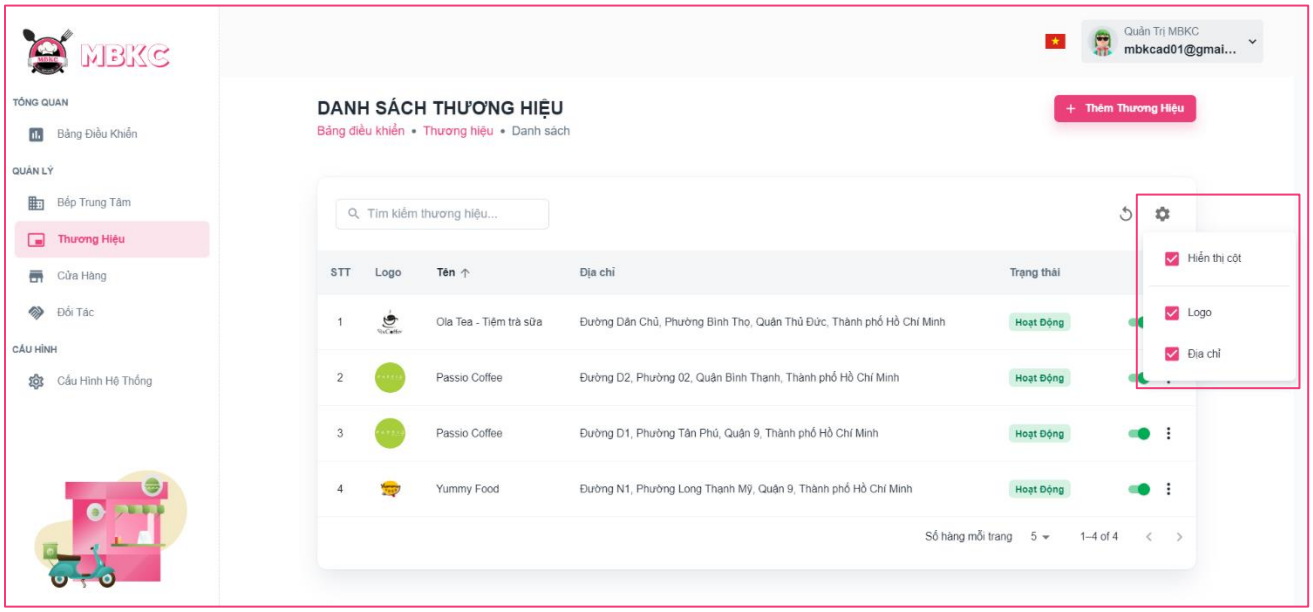


Figure 162: User Manual - View list Brand – Step 4 – Case 3

3.2.3.3. View details Brand

3.2.3.3.1. Description
Use this guideline to view details brand.

3.2.3.3.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

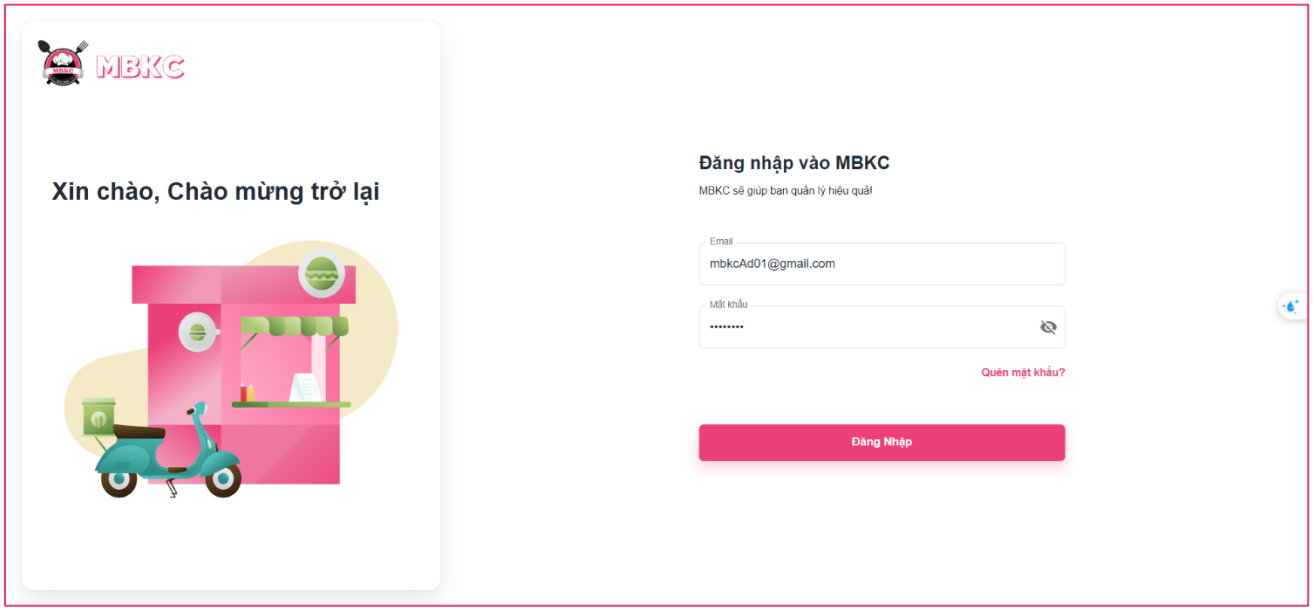


Figure 163: User Manual - View details Brand – Step 1

Step 2: Click “Đăng Nhập” button.

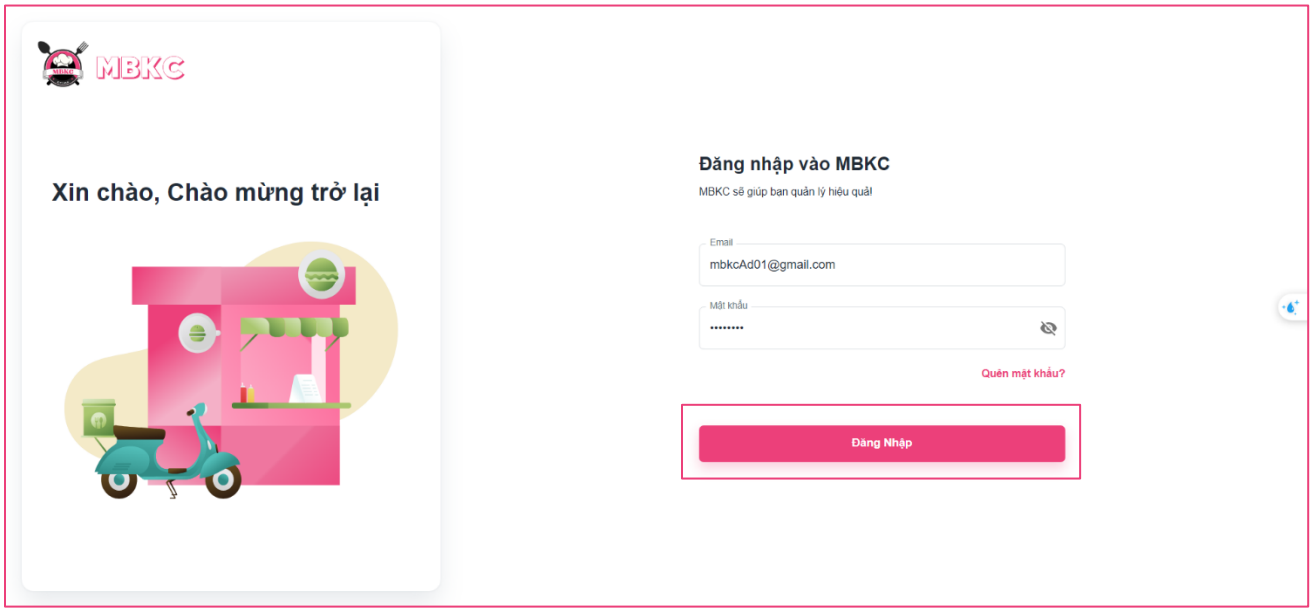


Figure 164: User Manual - View details Brand – Step 2

Step 3: From Admin Dashboard Page, choose “Thương Hiệu” from the left sidebar.

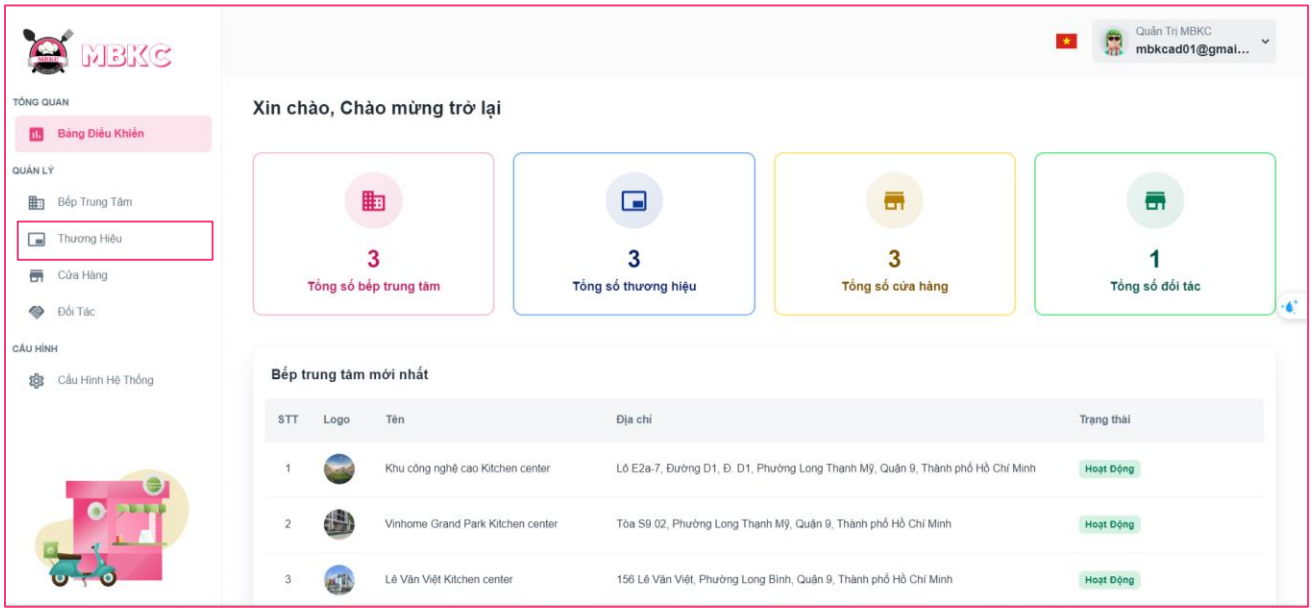


Figure 165: User Manual - View details Brand – Step 3

Step 4: Click the brand row to navigate to Brand Details Page.

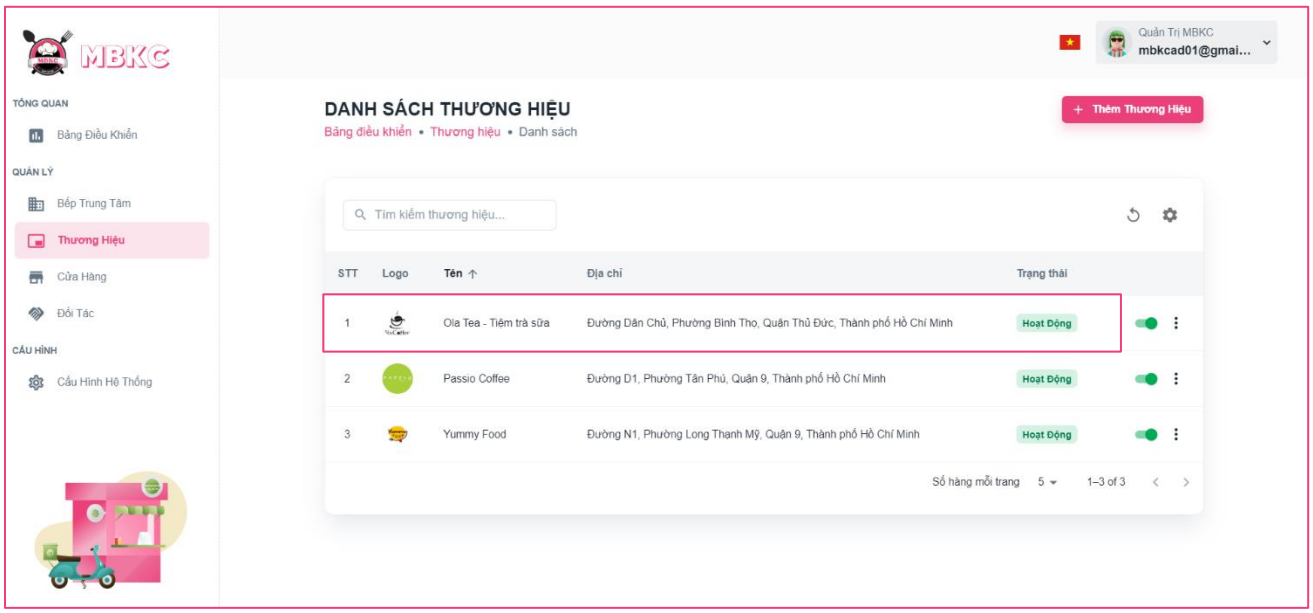


Figure 166: User Manual - View details Brand – Step 4

Step 5: Navigate to Brand Detail Page to show information details of brand.

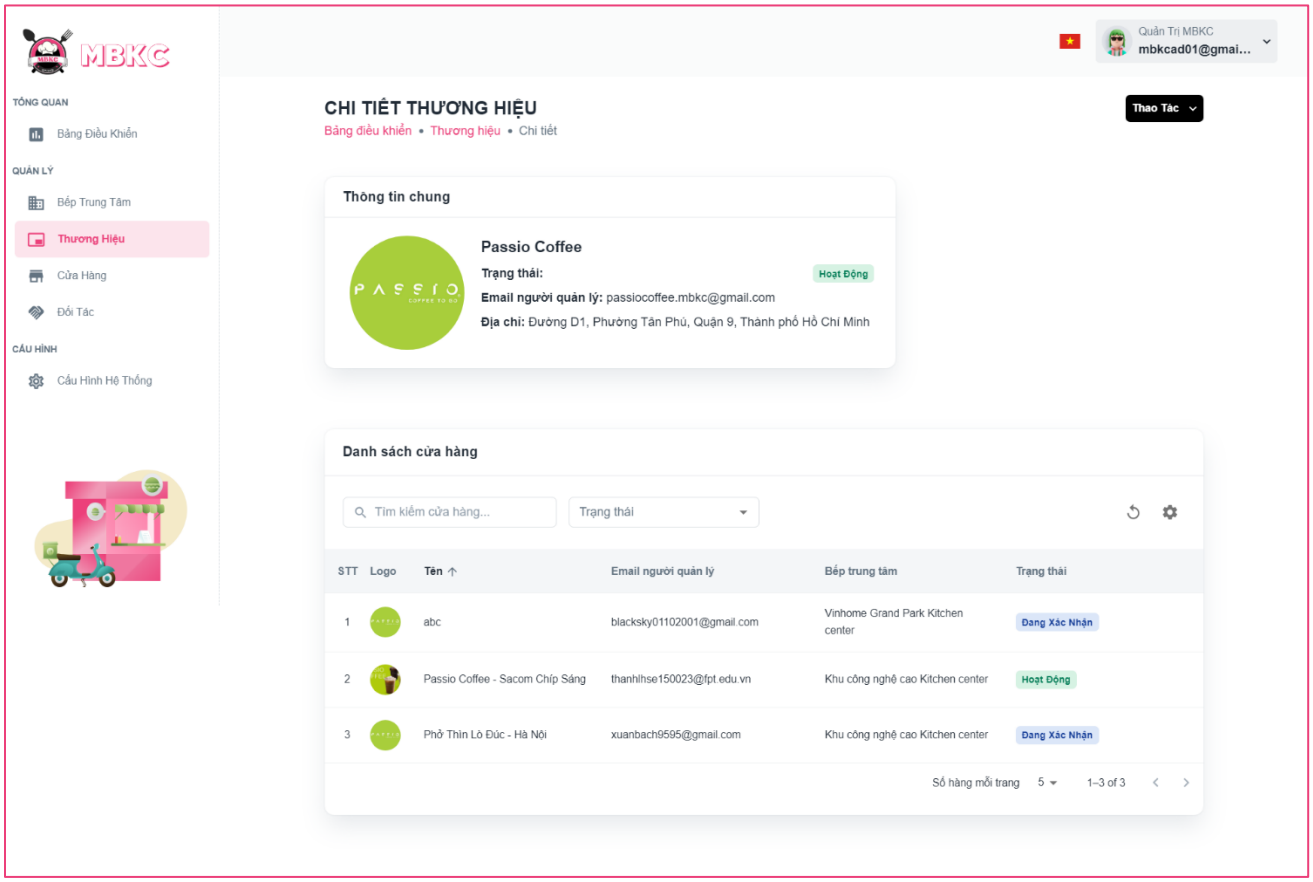


Figure 167: User Manual - View details Brand – Step 5

3.2.3.4. Update Status Brand

3.2.3.4.1. Description

Use this guideline to update the status of the brand.

3.2.3.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

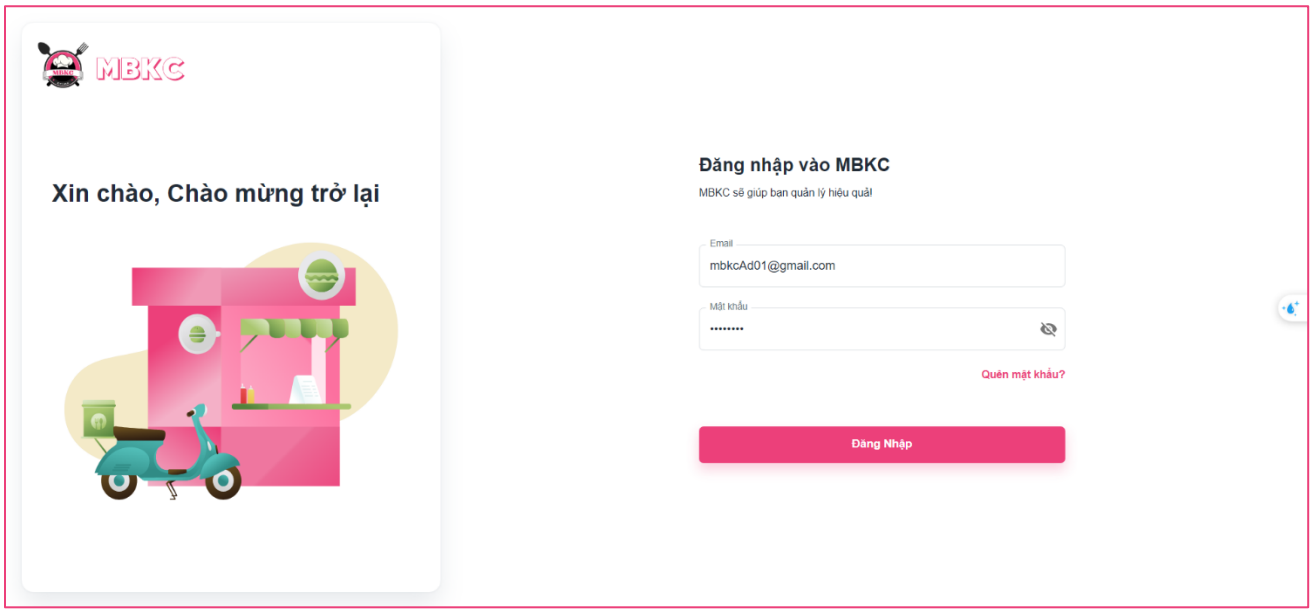


Figure 168: User Manual – Update Status Brand – Step 1

Step 2: Click “Đăng Nhập” button.

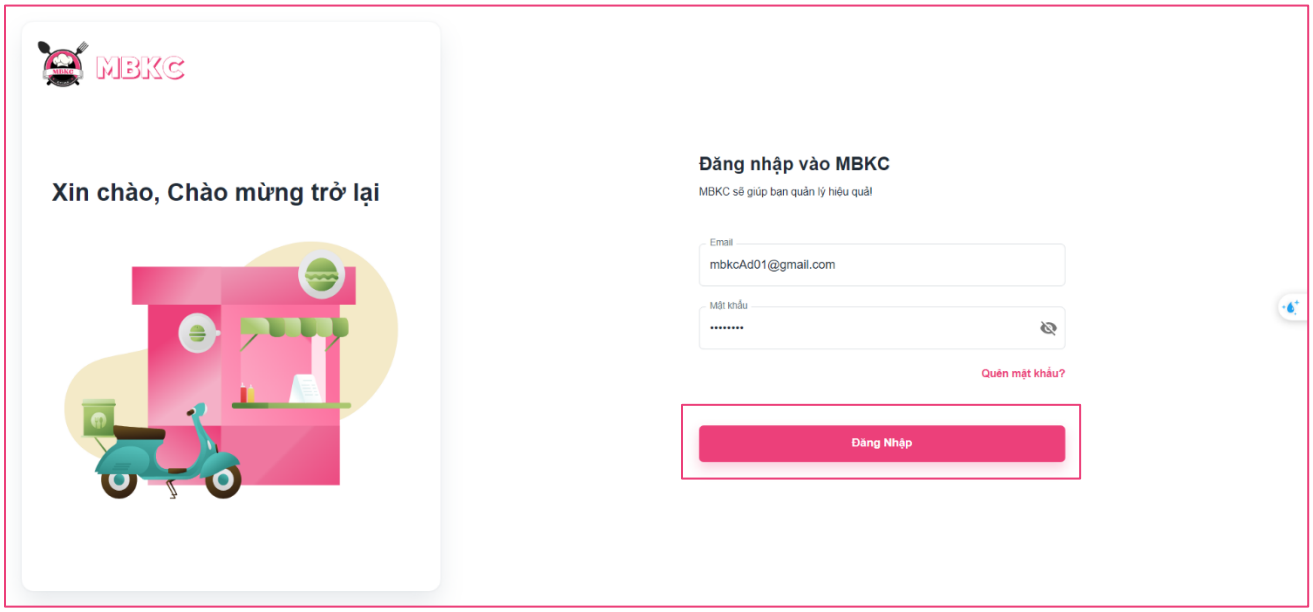


Figure 169: User Manual - Update Status Brand – Step 2

Step 3: From Admin Dashboard Page, choose “Thương Hiệu” from the left sidebar.

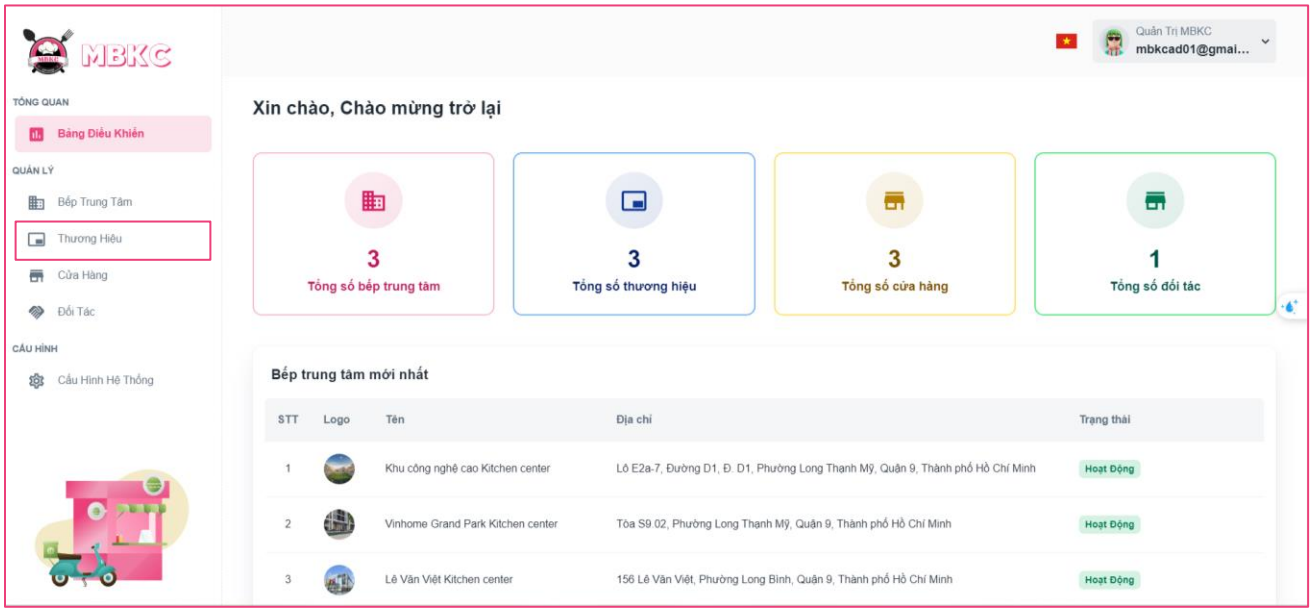


Figure 170: User Manual - Update Status Brand – Step 3

Step 4: Click on the switch button on the brand row to update the status.

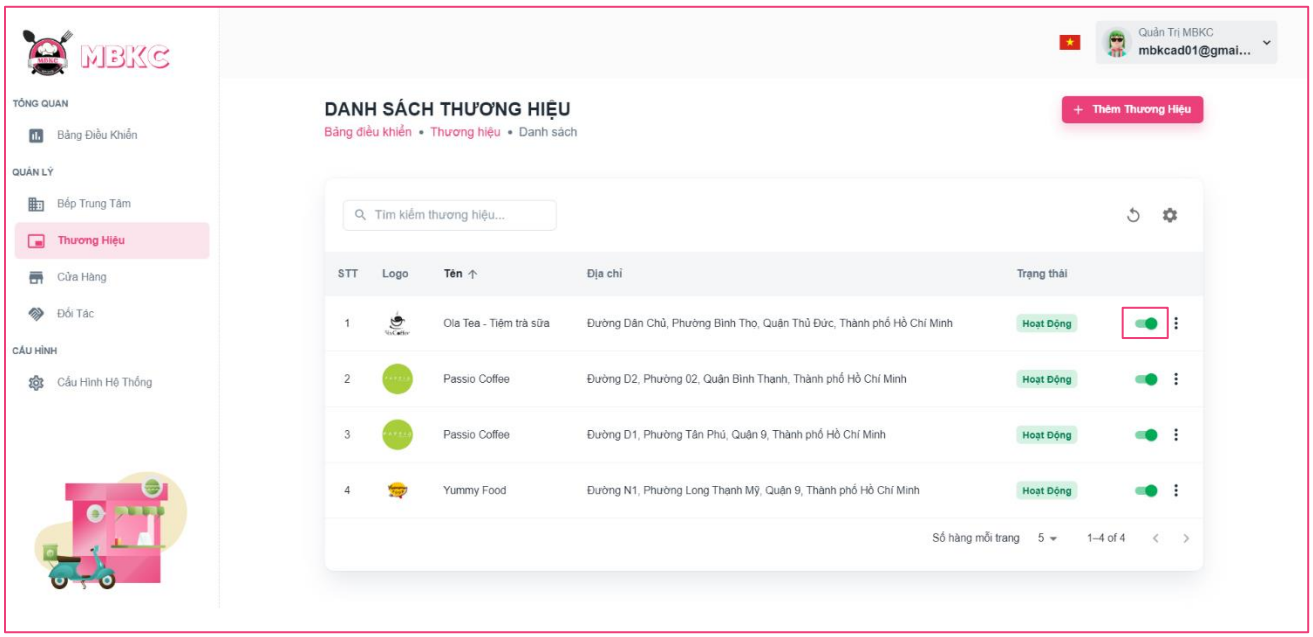


Figure 171: User Manual - Update Status Brand – Step 4

Step 5: List of brands will be reloaded and a toast message “Cập nhật trạng thái thương hiệu thành công”. And the status of the brand has been updated.

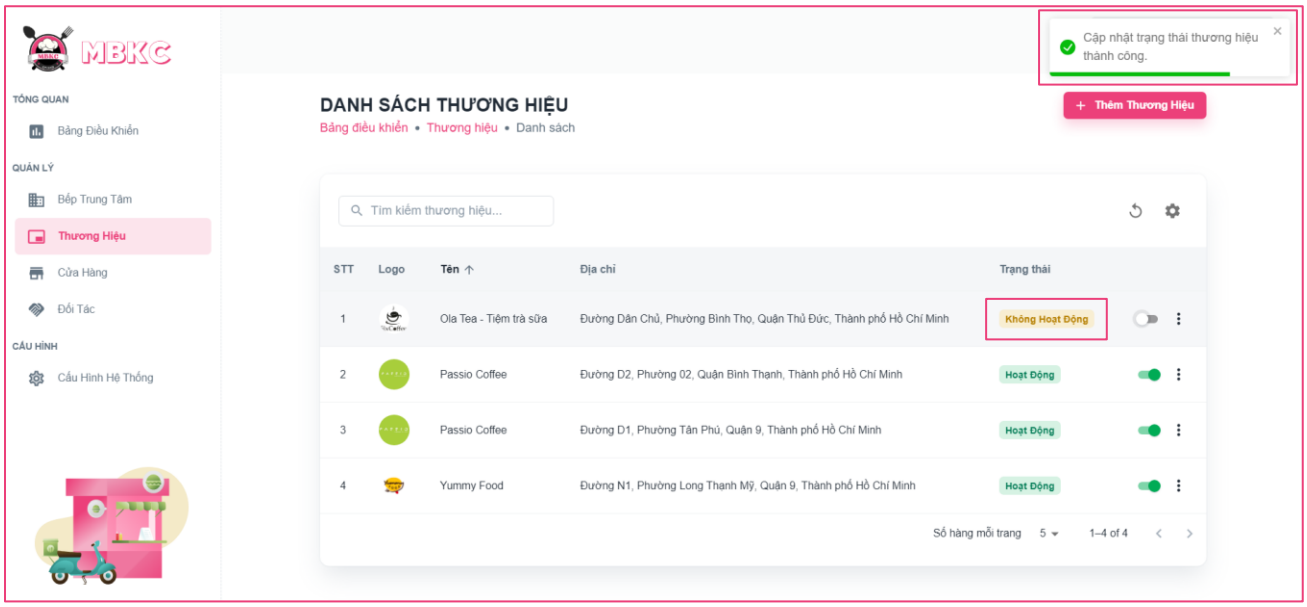


Figure 172: User Manual - Update Status Brand – Step 5

3.2.3.5. Update Information Brand

3.2.3.5.1. Description

Use this guideline to update information of the brand.

3.2.3.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

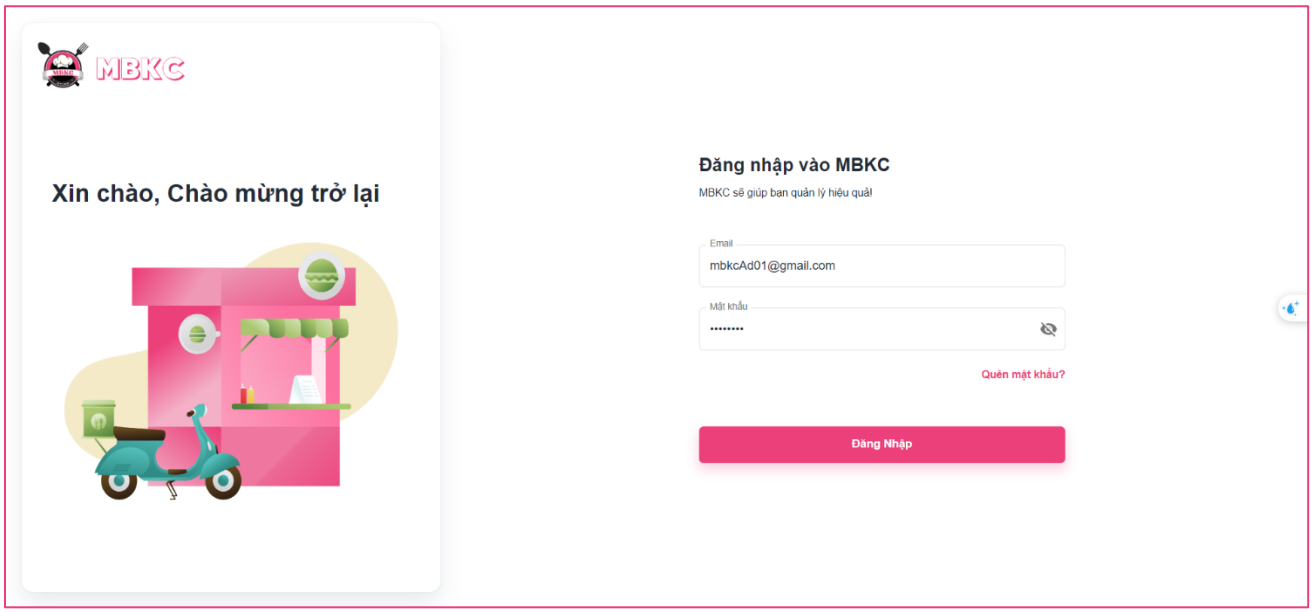


Figure 173: User Manual – Update Information Brand – Step 1

Step 2: Click “Đăng Nhập” button.

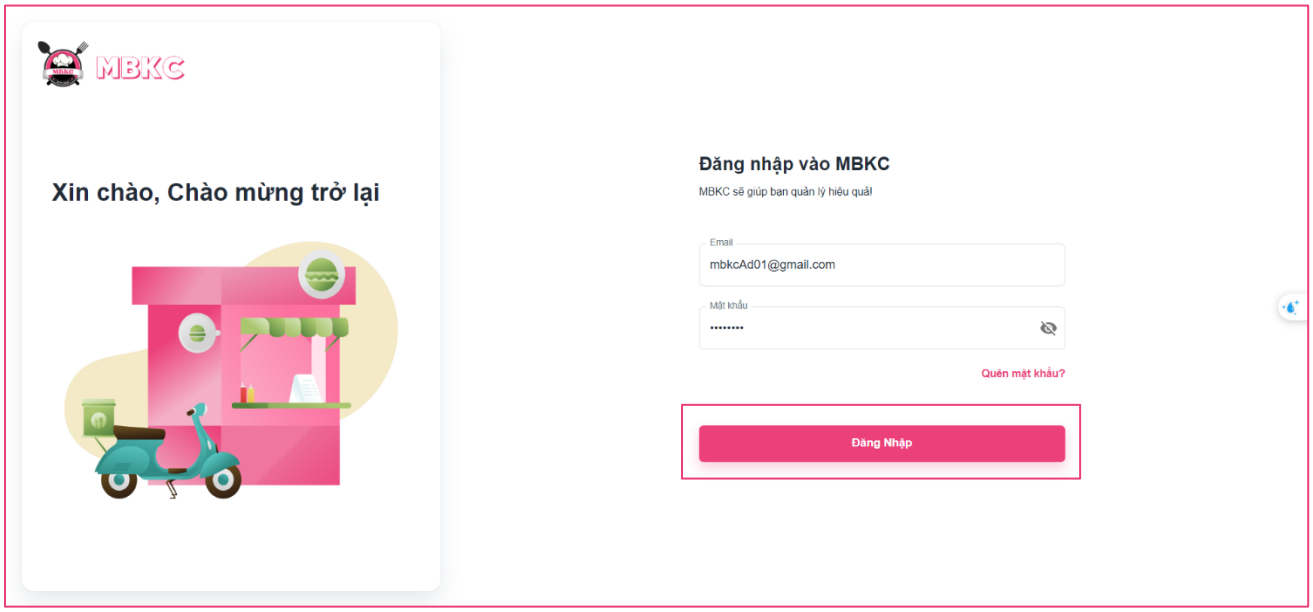


Figure 174: User Manual - Update Information Brand – Step 2

Step 3: From Admin Dashboard Page, choose “Thương Hiệu” from the left sidebar.

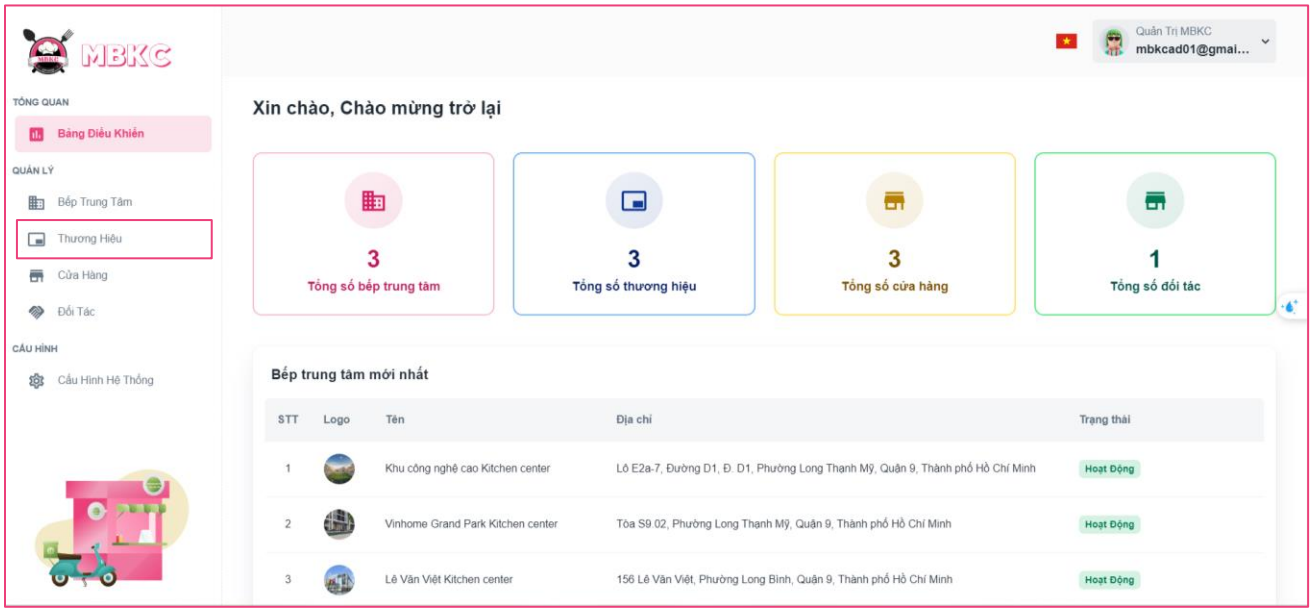


Figure 175: User Manual - Update Information Brand – Step 3

Step 4: Update information of the brand.

Case 1: Update information of the brand in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the brand row want to navigate Update Brand Page.

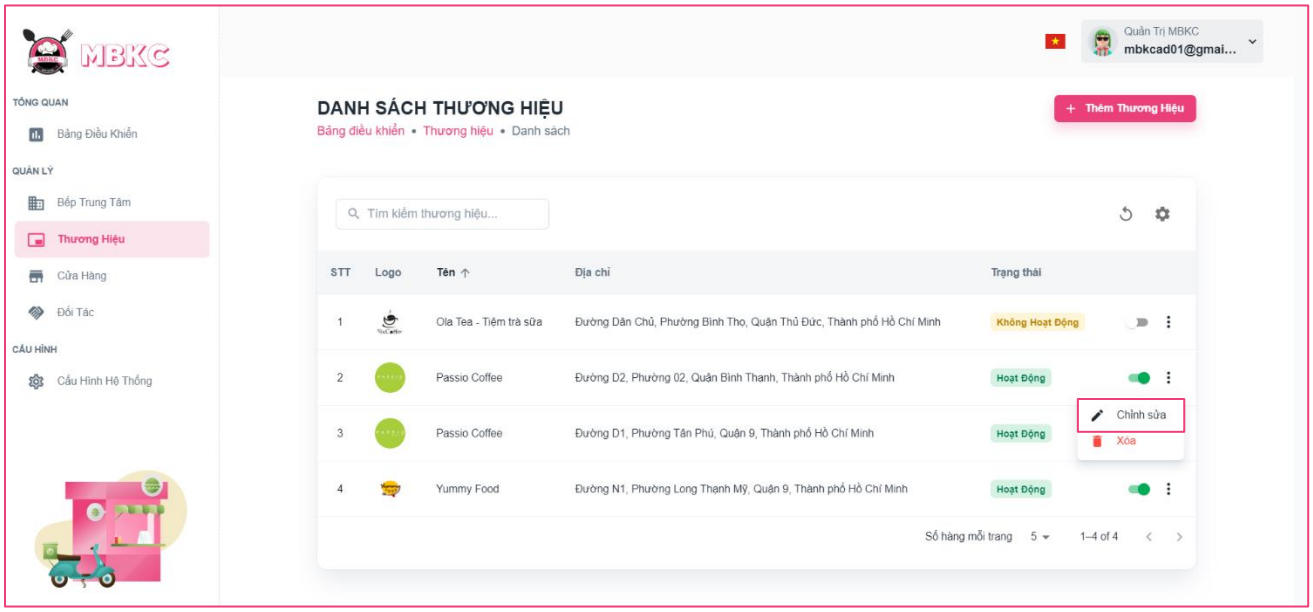


Figure ...: User Manual - Update Information Brand – Step 4 – Case 1

Case 2: Update information of the brand on the details page.

1: Click the brand row to navigate to Brand Details Page.

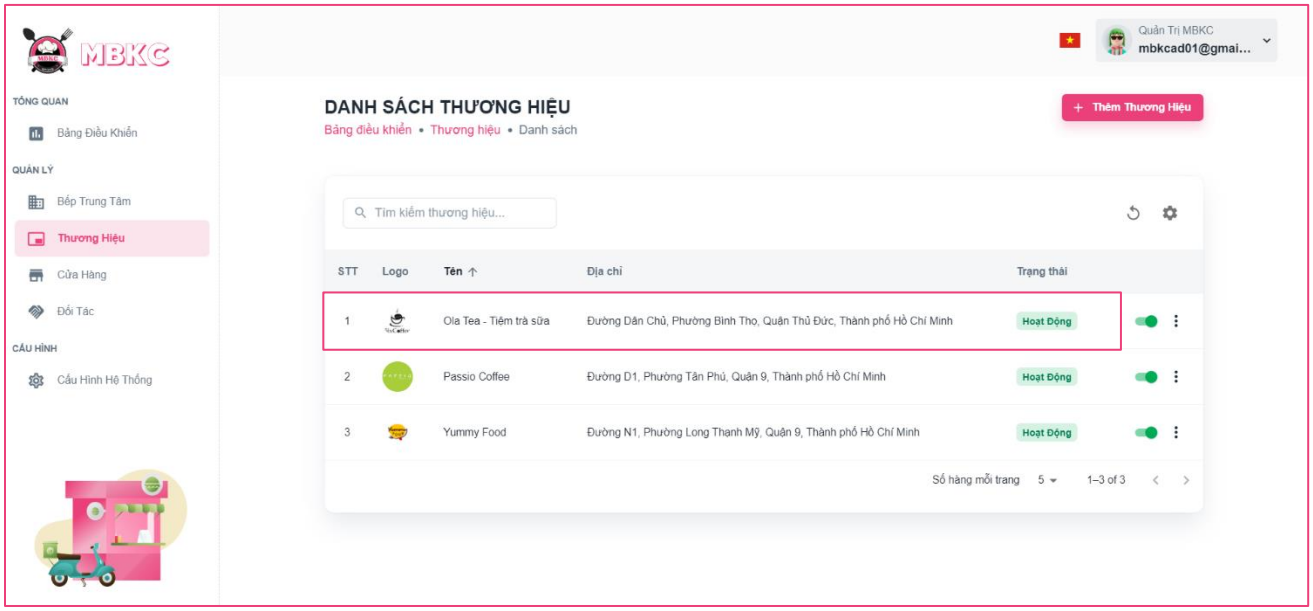


Figure 176: User Manual - Update Information Brand – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Chỉnh sửa” menu item to navigate Update Brand Page.

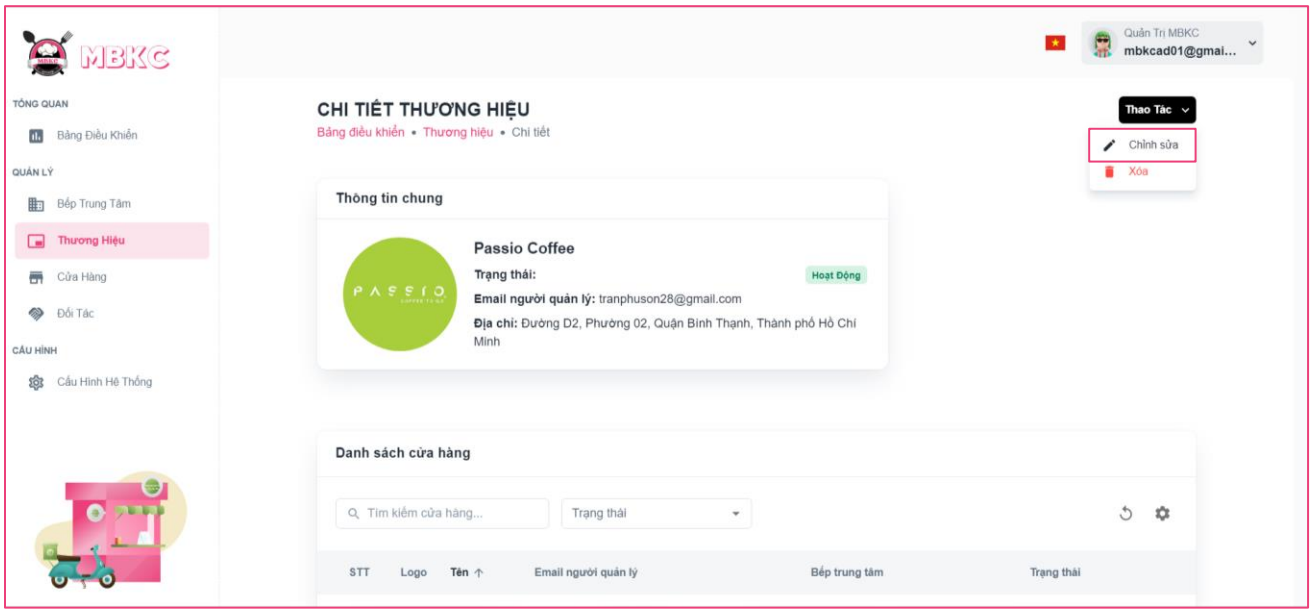


Figure 177: User Manual - Update Information Brand – Step 4 – Case 2 - 2

Step 5: Brand information will be automatically filled in the form. Then edit the information as desired.

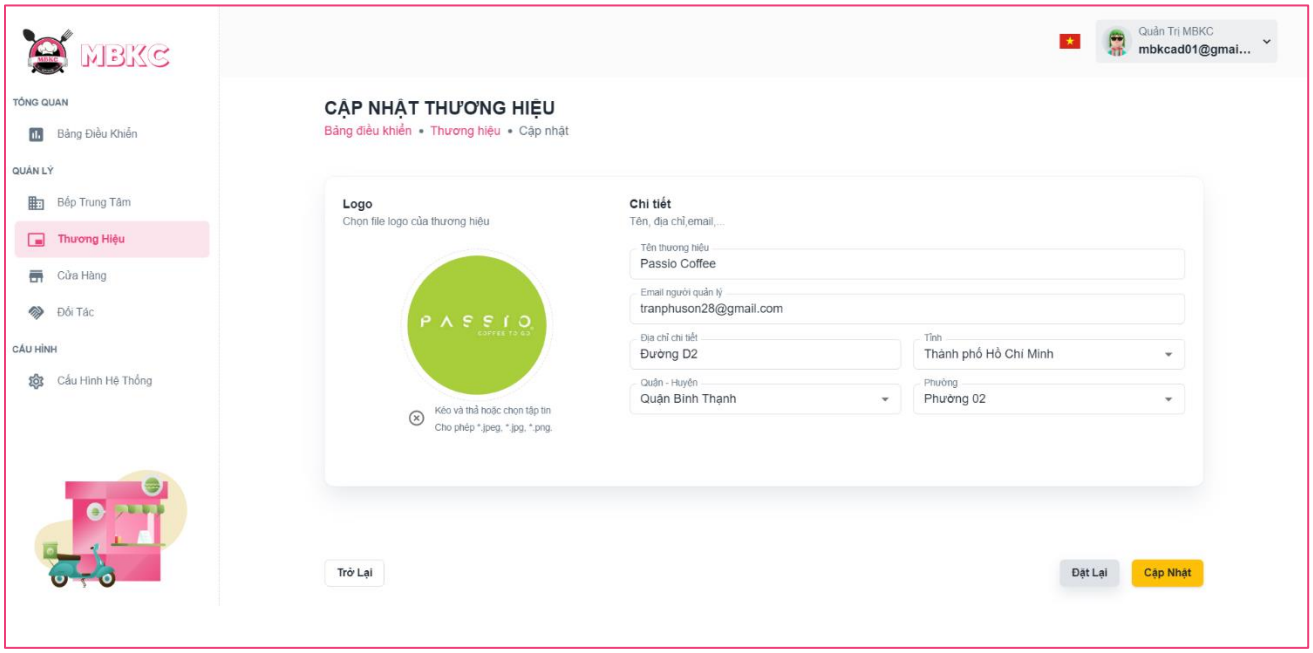


Figure 178: User Manual - Update Information Brand – Step 5

Step 6: Click "Cập nhật" button.

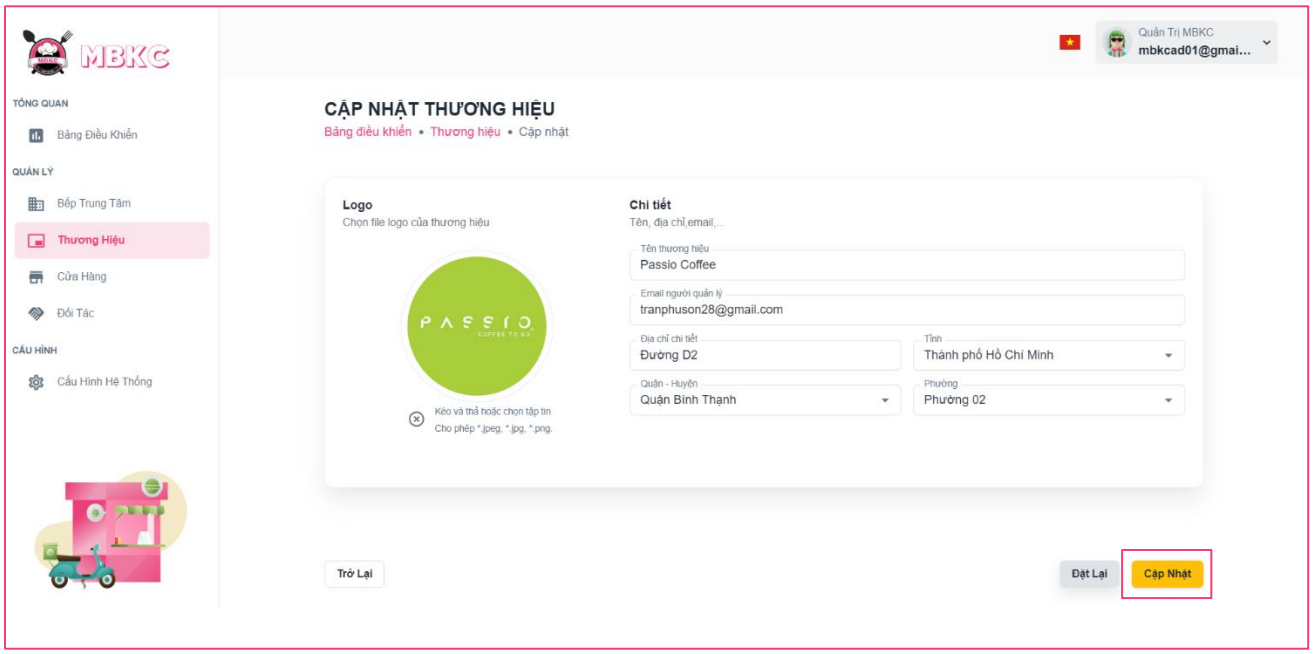


Figure 179: User Manual - Update Information Brand – Step 6

Step 7:

Case 1: You will be navigated to Brand Detail Page showing the brand details when start edit information of brand from Brand Detail Page and a toast message “Cập nhật thương hiệu thành công”.

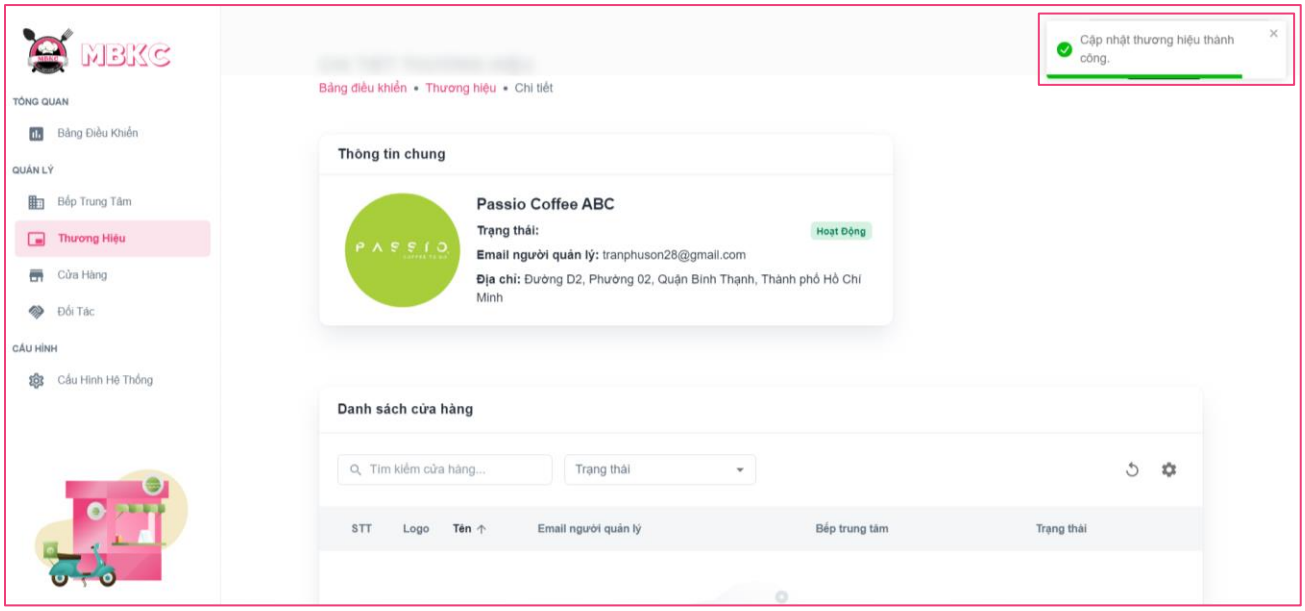


Figure 180: User Manual - Update Information Brand – Step 7 – Case 1

Case 2: You will be navigated to List of Brand Page showing the list of brands when start edit information of brand from List of Brand Page and a toast message “Cập nhật thương hiệu thành công”.

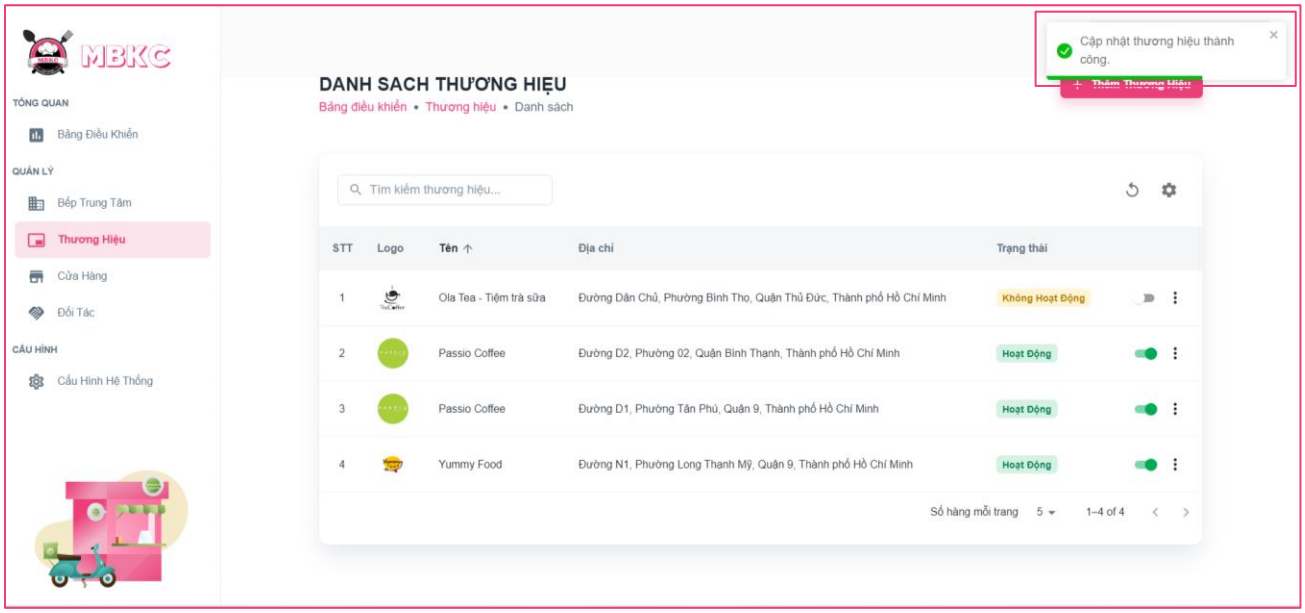


Figure 181: User Manual - Update Information Brand – Step 7 – Case 2

3.2.3.6. Delete Brand

3.2.3.6.1. Description
Use this guideline to delete brand.

3.2.3.6.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

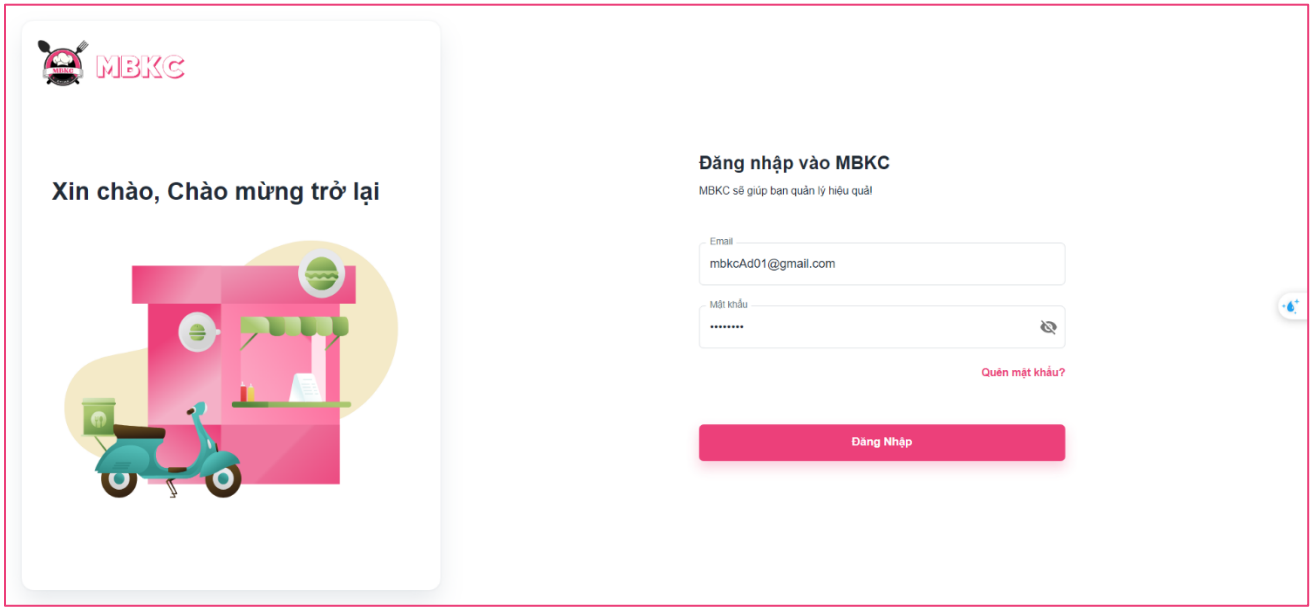


Figure 182: User Manual – Delete Brand – Step 1

Step 2: Click “Đăng Nhập” button.

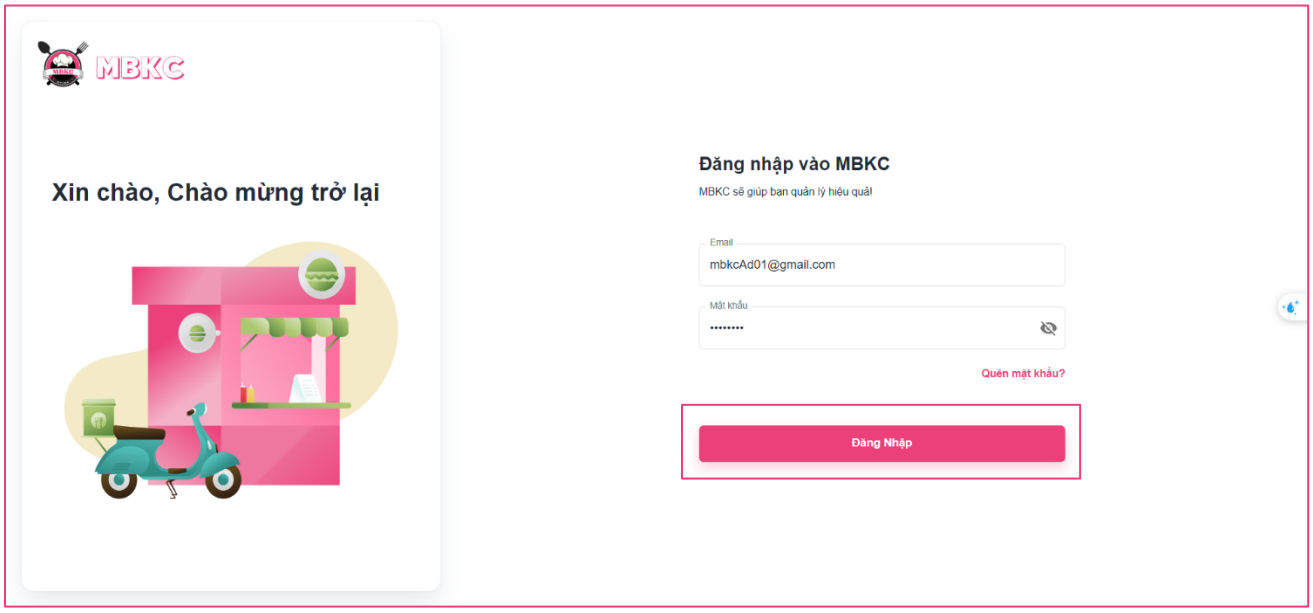


Figure 183: User Manual - Delete Brand – Step 2

Step 3: From Admin Dashboard Page, choose “Thương Hiệu” from the left sidebar.

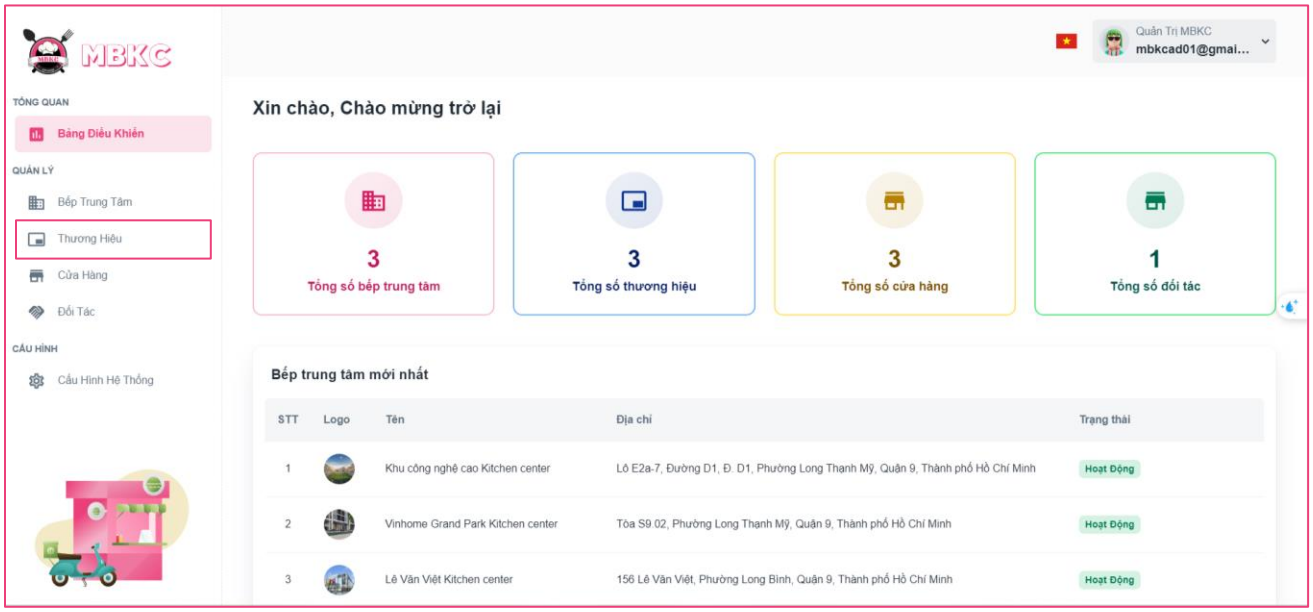


Figure 184: User Manual - Delete Brand – Step 3

Step 4: Delete brand.

Case 1: Delete brand in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the brand row to delete.

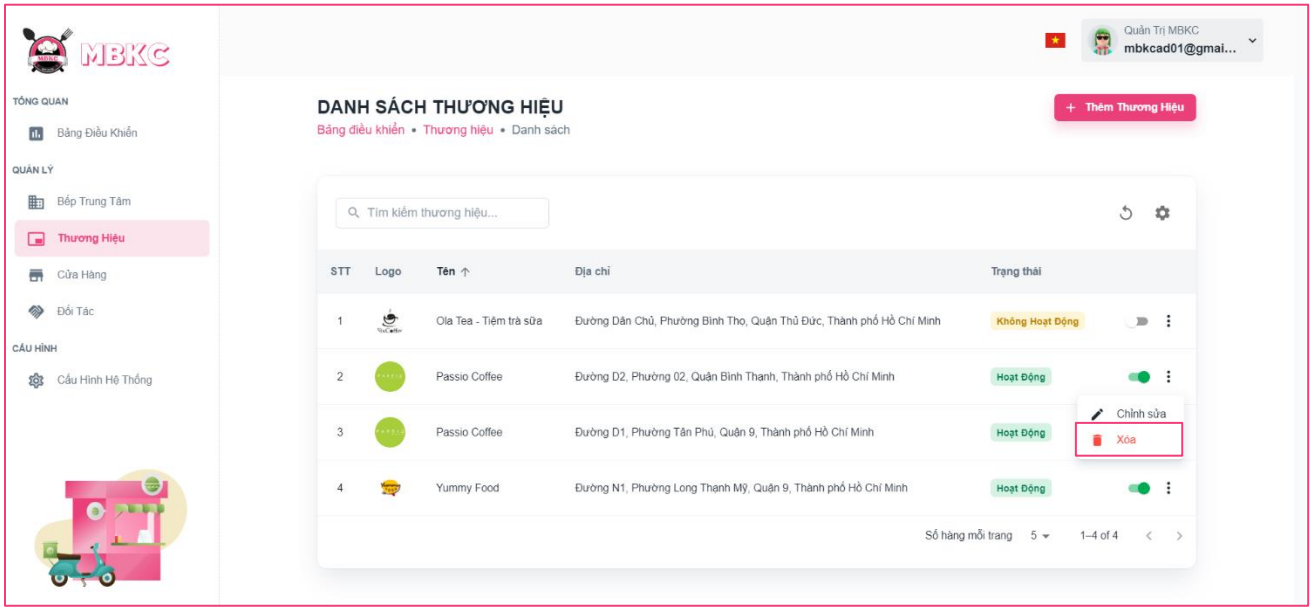


Figure 185: User Manual - Delete Brand – Step 4 – Case 1

Case 2: Delete brand on the details page.

1: Click the brand row to navigate to Brand Details Page.

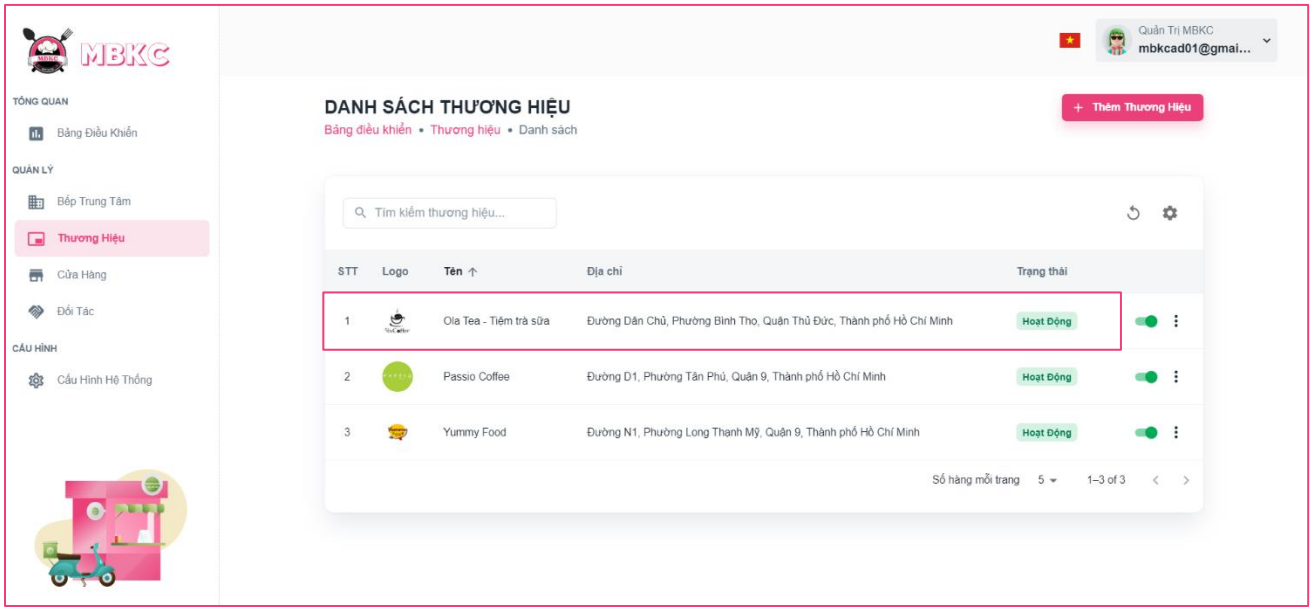


Figure 186: User Manual - Delete Brand – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item to delete.

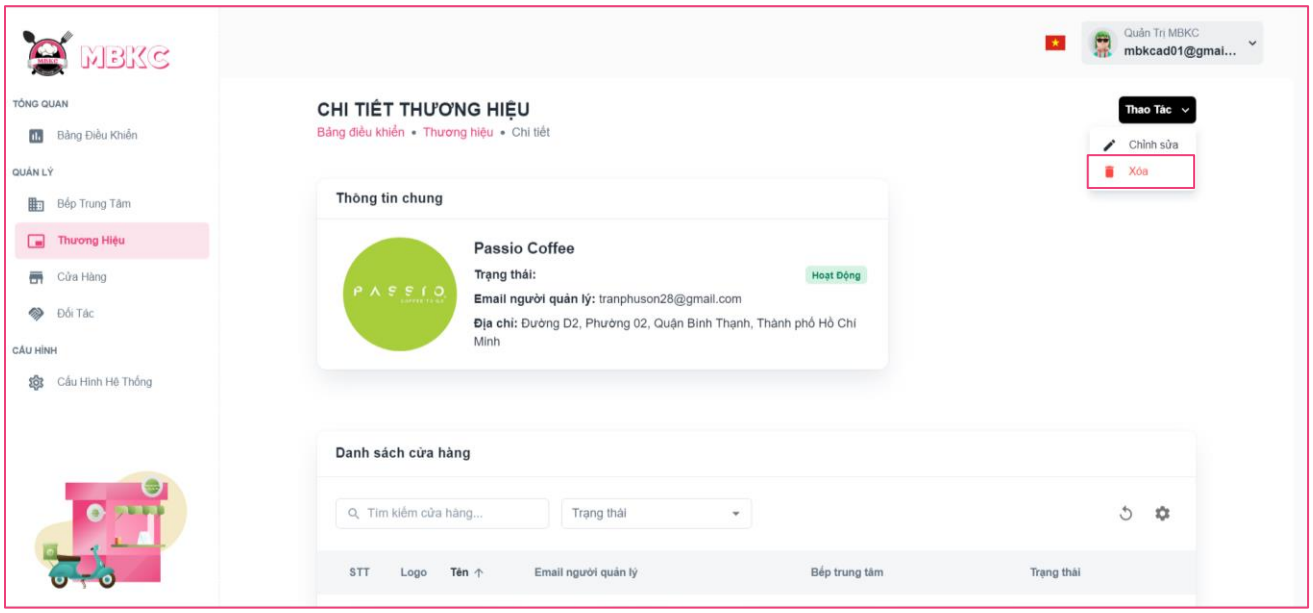


Figure 187: User Manual - Delete Brand – Step 4 – Case 2 - 2

Step 5: Click “Xác Nhận” button.

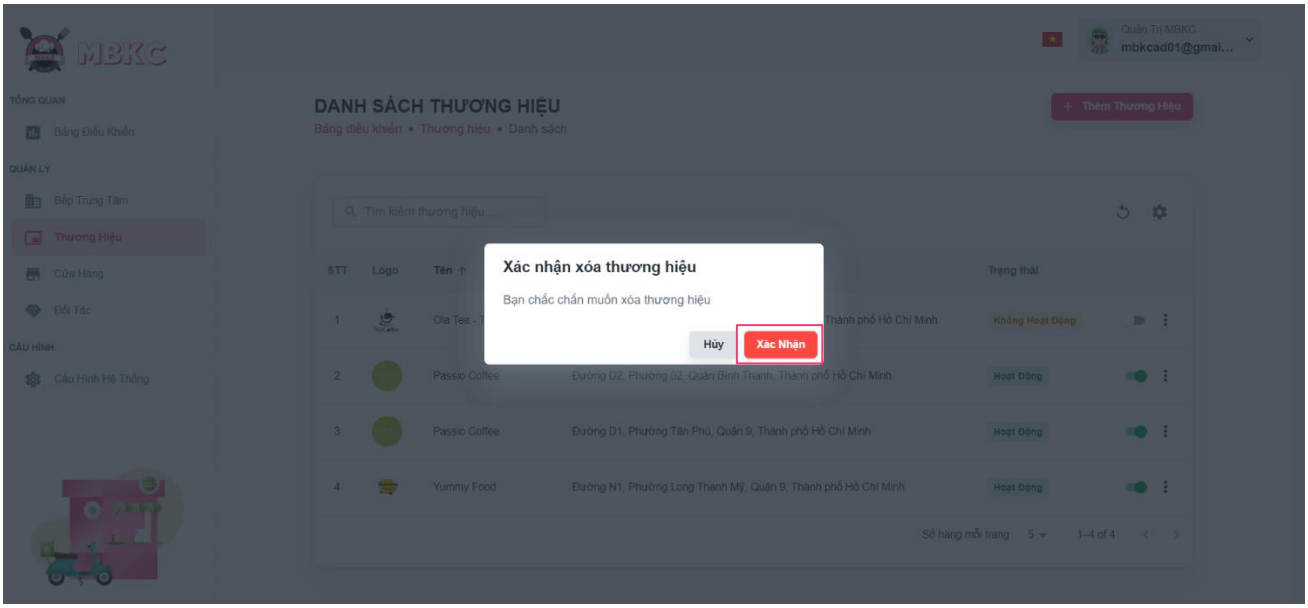


Figure 188: User Manual - Delete Brand – Step 5

Step 6: List of brands will be reloaded and a toast message “Đã xóa thương hiệu thành công”. And brand has been deleted.

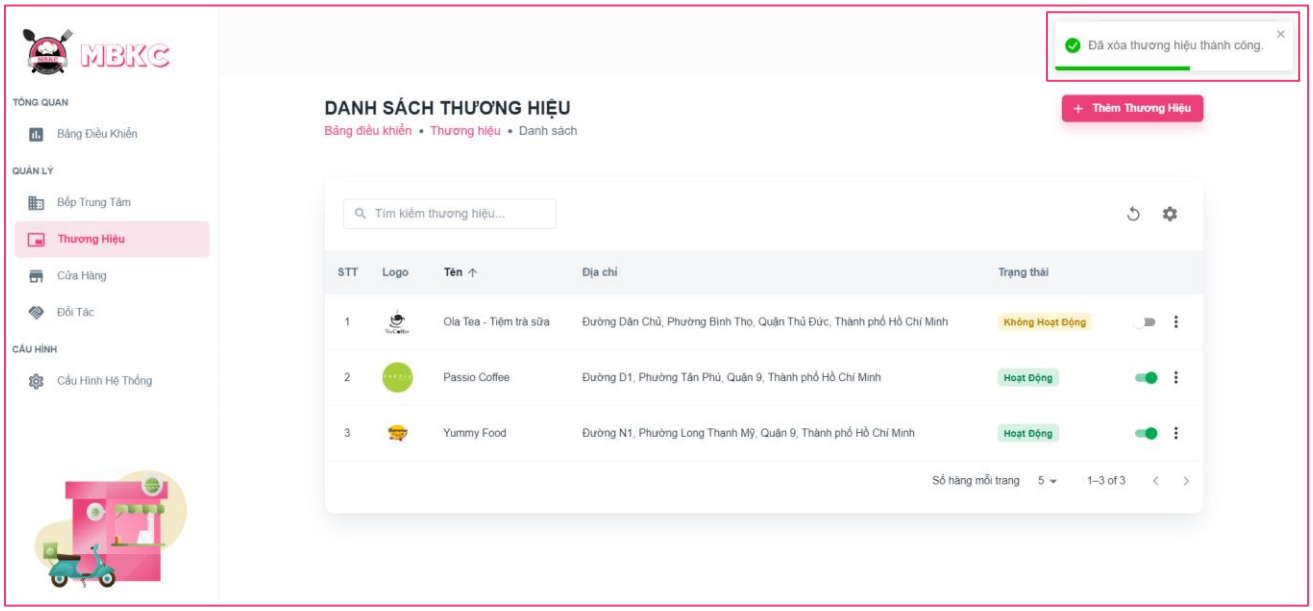


Figure 189: User Manual – Delete Brand – Step 6

3.2.4. Store management

3.2.4.1. Register new Store

3.2.4.1.1. Description
Use this guideline to register a new store.

This web application supports register a new store with account as a Brand Manager.

3.2.4.1.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

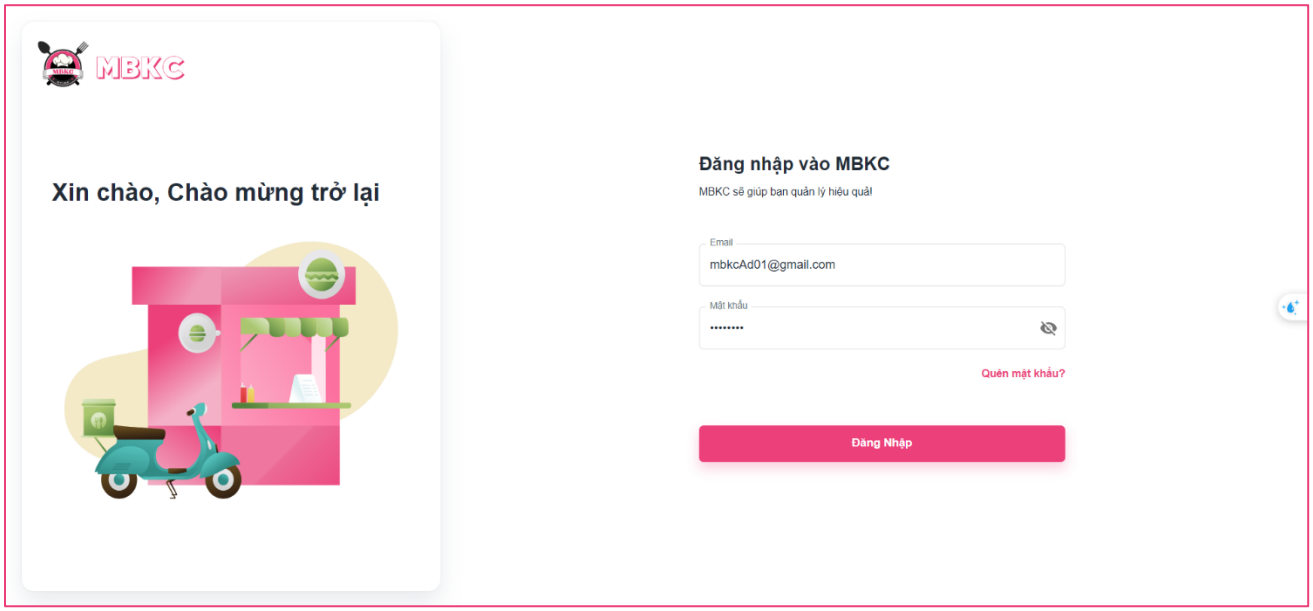


Figure 190: User Manual - Register new Store – Step 1

Step 2: Click “Đăng Nhập” button.

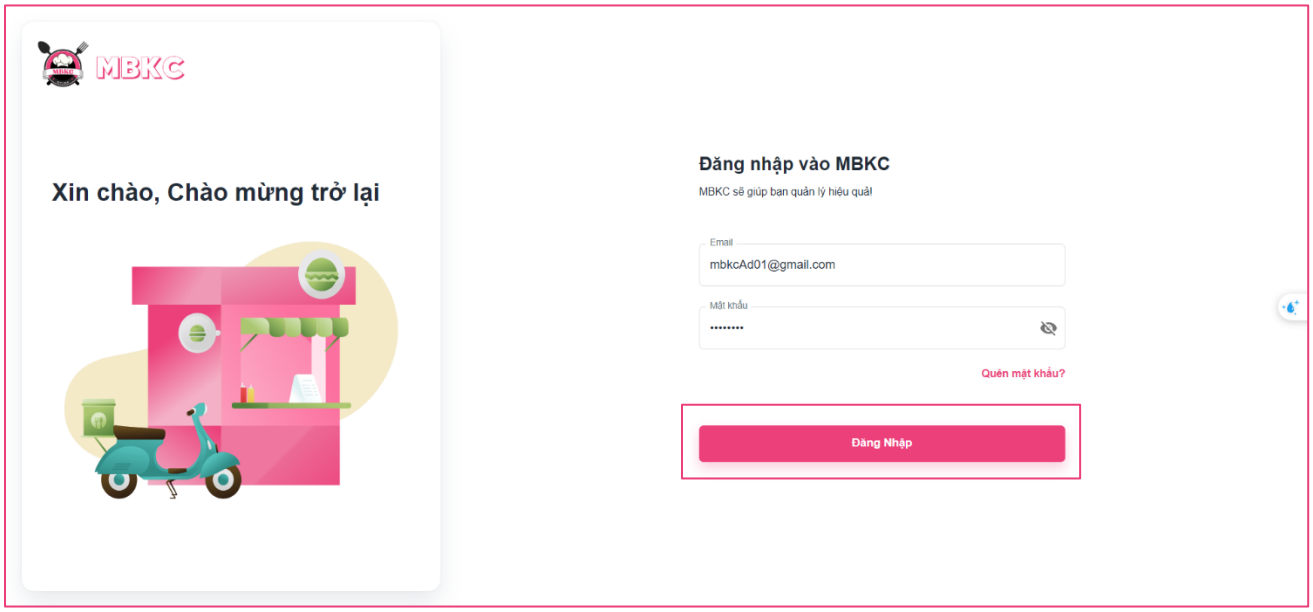


Figure 191: User Manual - Register new Store – Step 2

Step 3: From Brand Dashboard Page, choose “Cửa Hàng” from the left sidebar.

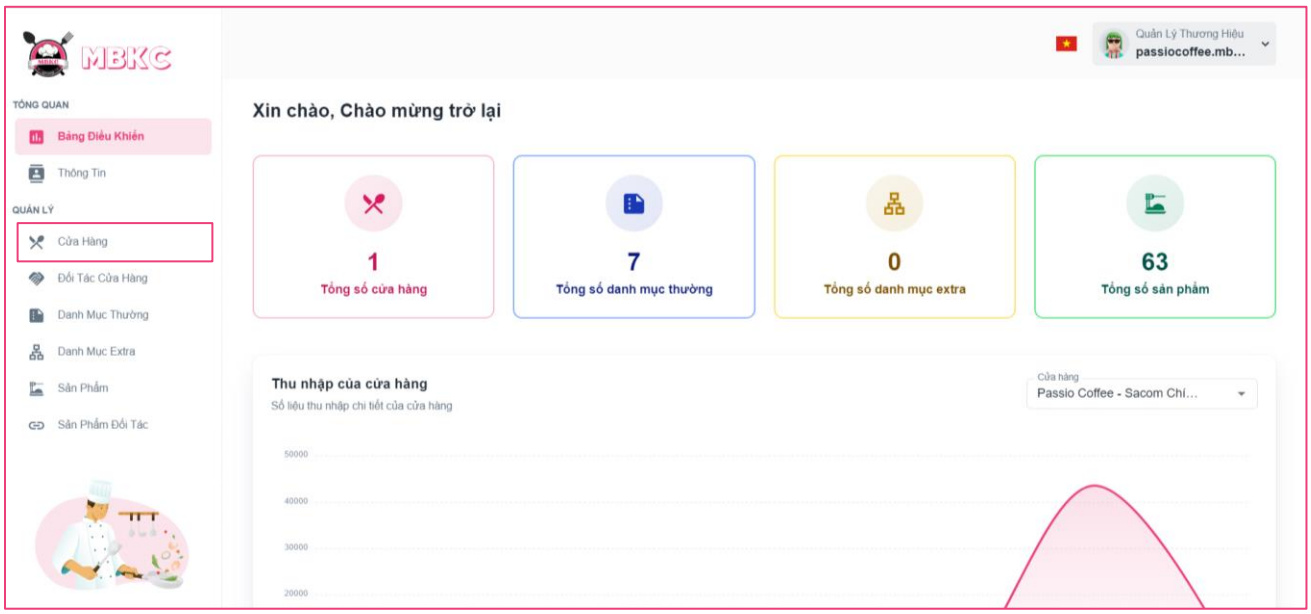


Figure 192: User Manual - Register new Store – Step 3

Step 4: Click “Đăng Kí Cửa Hàng” button on the screen and navigate to Register New Store Page to register a new store.

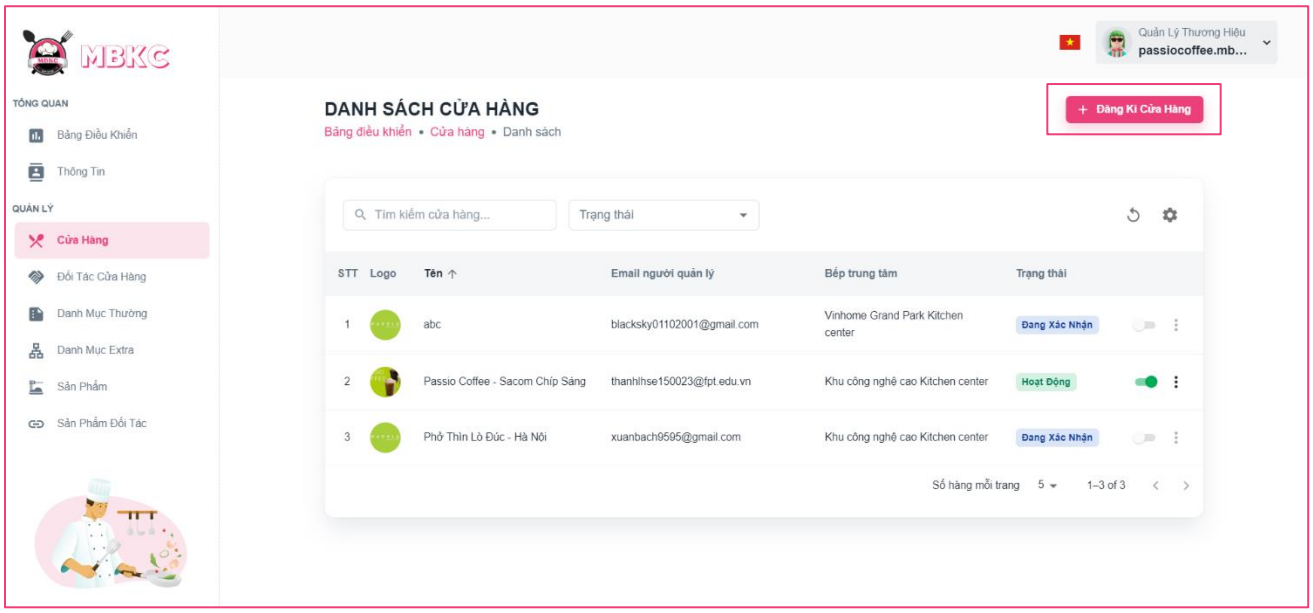


Figure 193: User Manual - Register new Store – Step 4

Step 5: Filling all the required information to register a new store including: “Logo”, “Tên cửa hàng", “Email người quản lý", “Bếp trung tâm".

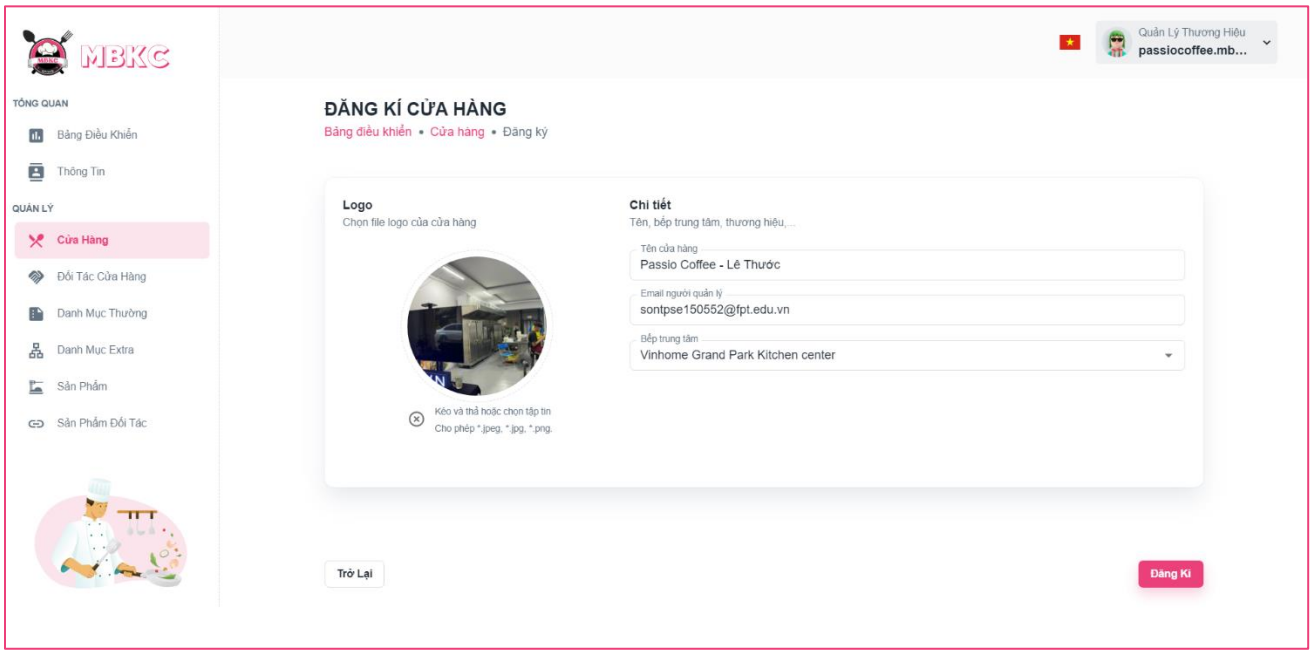


Figure 194: User Manual - Register new Store – Step 5

Step 6: After filling in all required field, click “Đăng Kí” to finish register a new store.

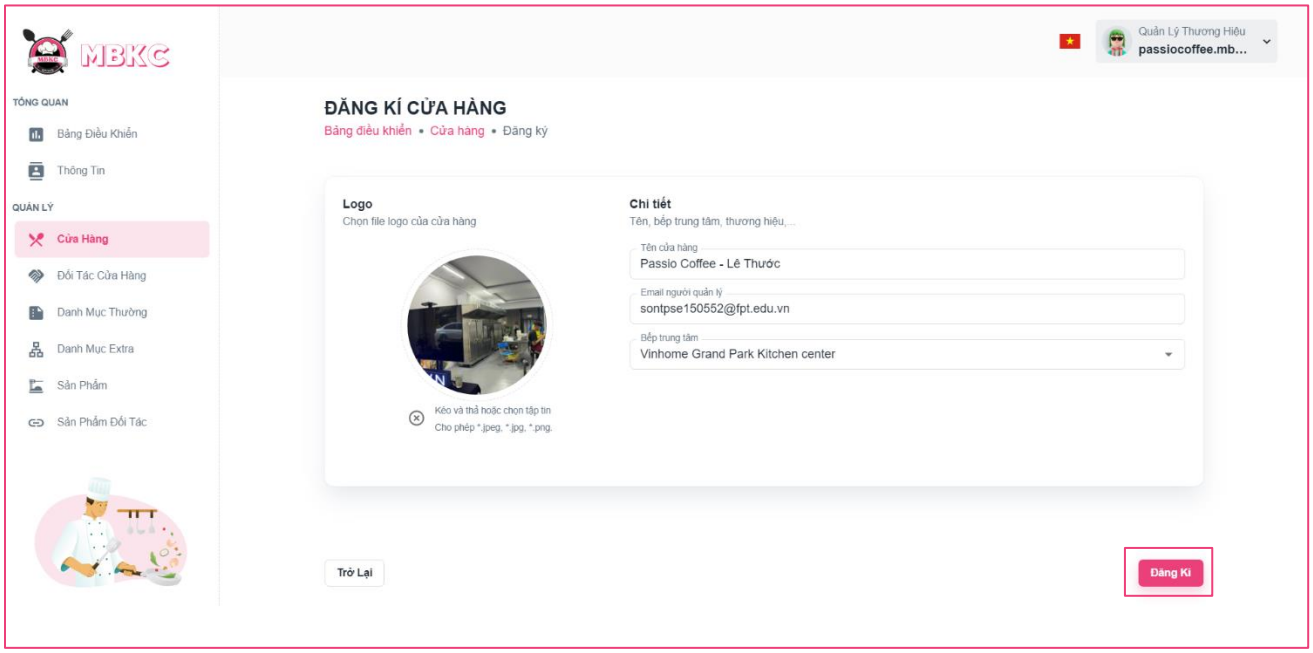


Figure 195: User Manual - Register new Store – Step 6

Step 7: You will be navigated to List of Store Page showing the list of stores and a toast message “Đăng ký cửa hàng mới thành công”.

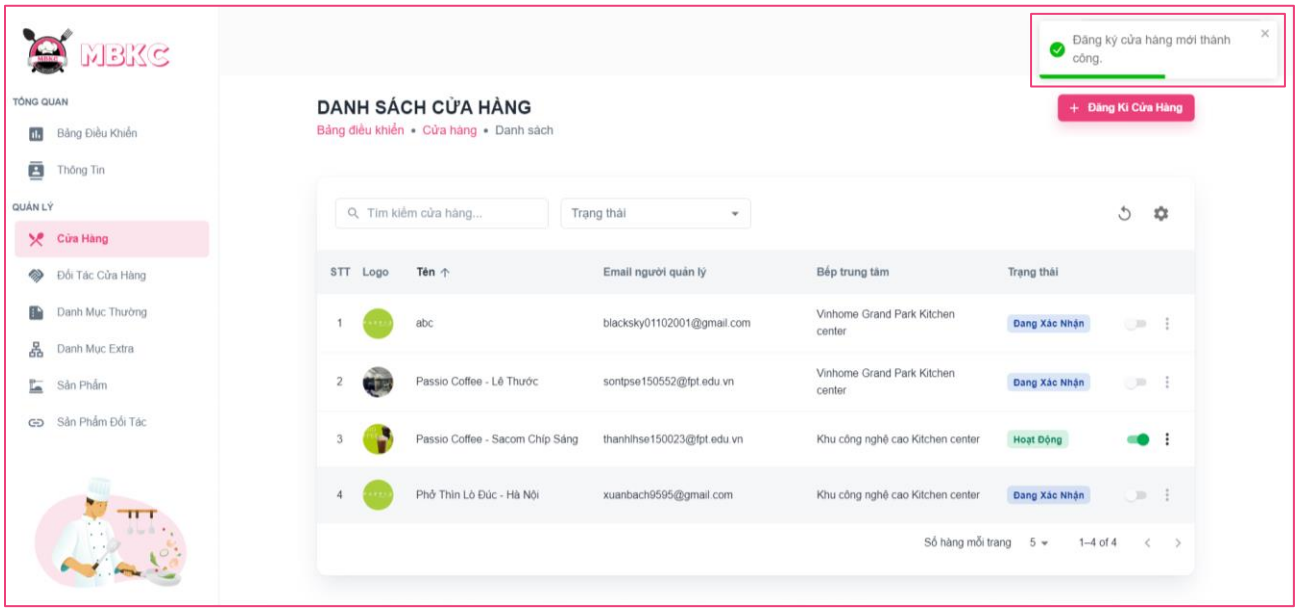


Figure 196: User Manual - Register new Store – Step 7

3.2.4.2. View list Store

3.2.4.2.1. Description

Use this guideline to search, filter store status, reload data, and show or hide columns in the table list of stores.

This web application supports view list of stores with account as a MBKC Admin, Kitchen Center Manager, and Brand Manager.

3.2.4.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password”.

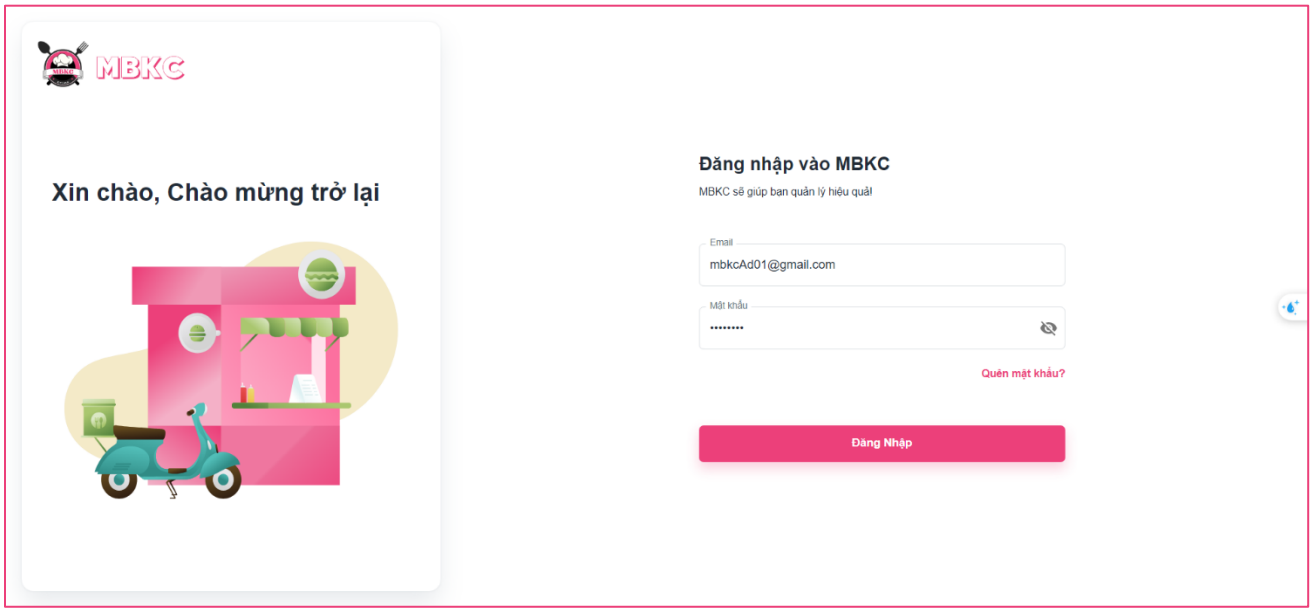


Figure 197: User Manual - View list Store – Step 1

Step 2: Click “Đăng Nhập” button.

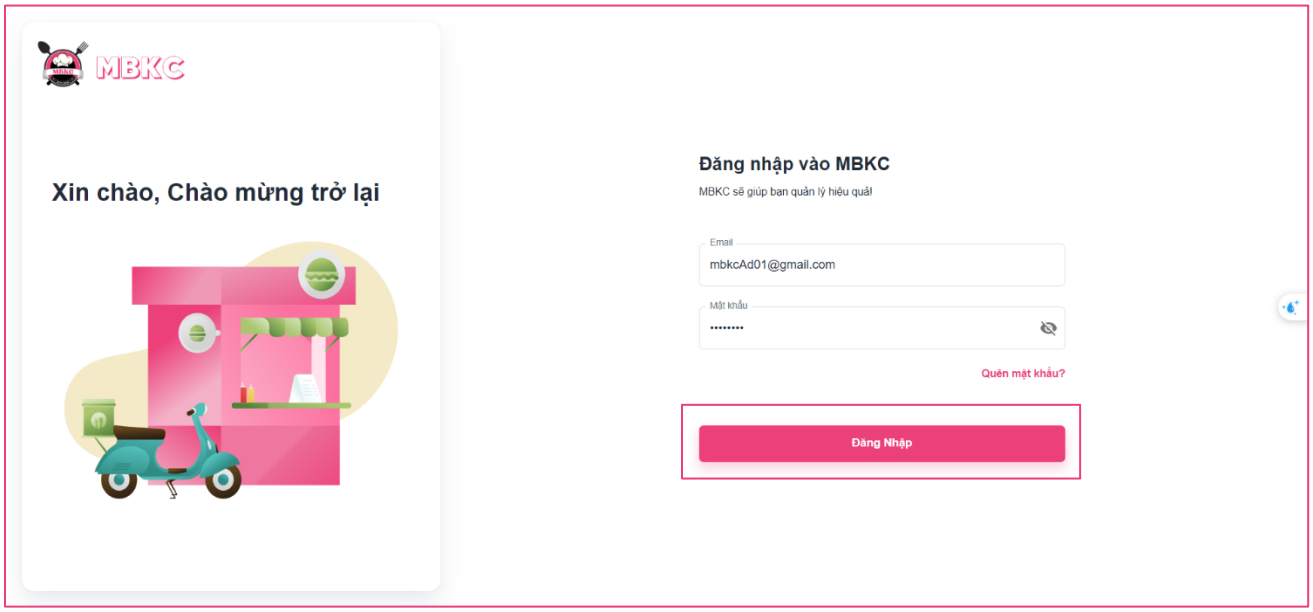


Figure 198: User Manual - View list Store – Step 2

Step 3: From Dashboard Page, choose “Cửa Hàng” from the left sidebar.

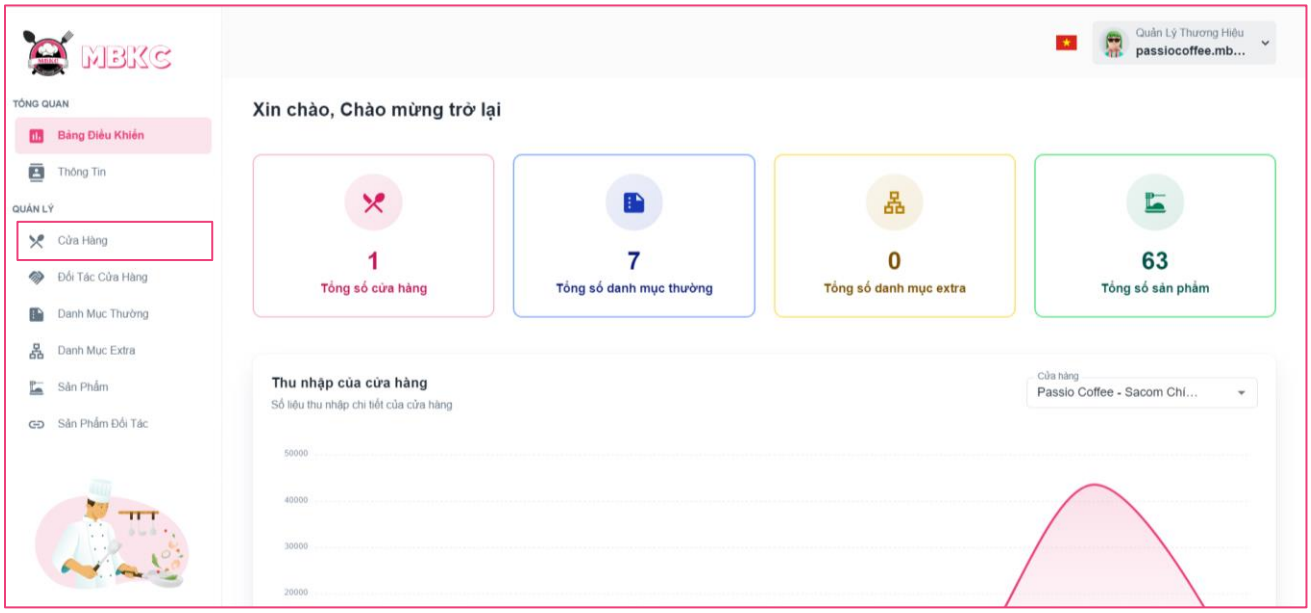


Figure 199: User Manual - View list Store – Step 3

Step 4: Search, filter store status, reload data, and show or hide columns in the table list of stores.

Case 1: Fill in the store name in the search box to search stores in the table list.

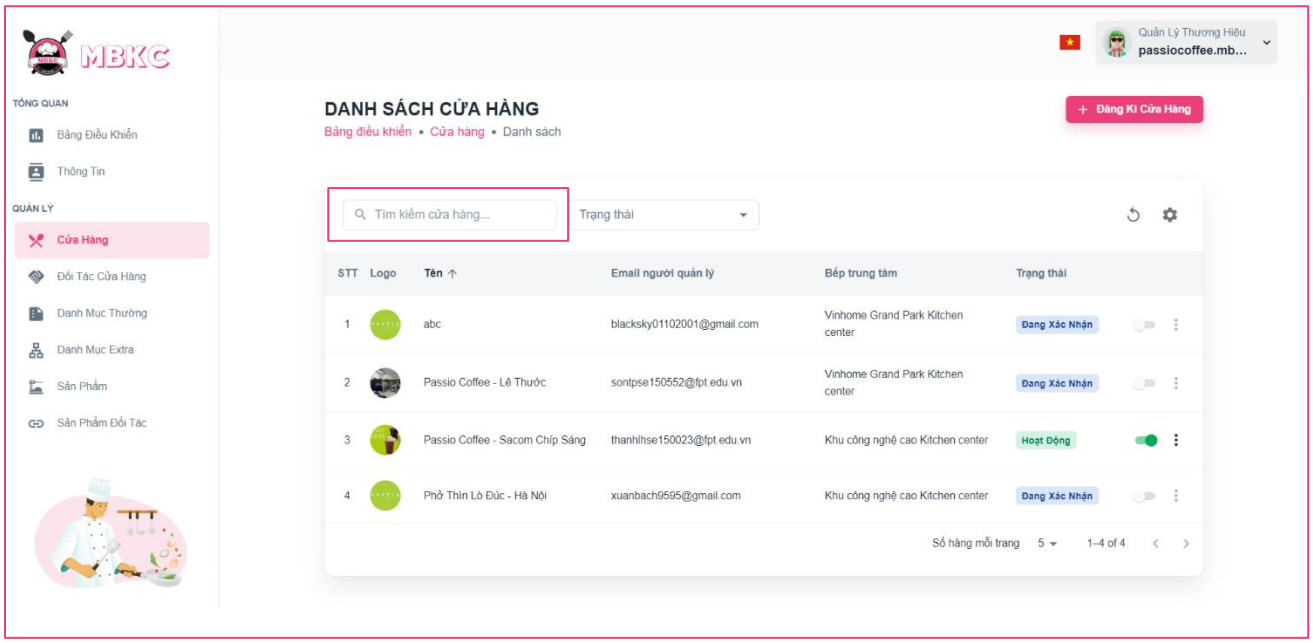


Figure 200: User Manual - View list Store – Step 4 – Case 1

Case 2: Click the box to select filter status “Trạng thái” and select one status item in the dropdown.

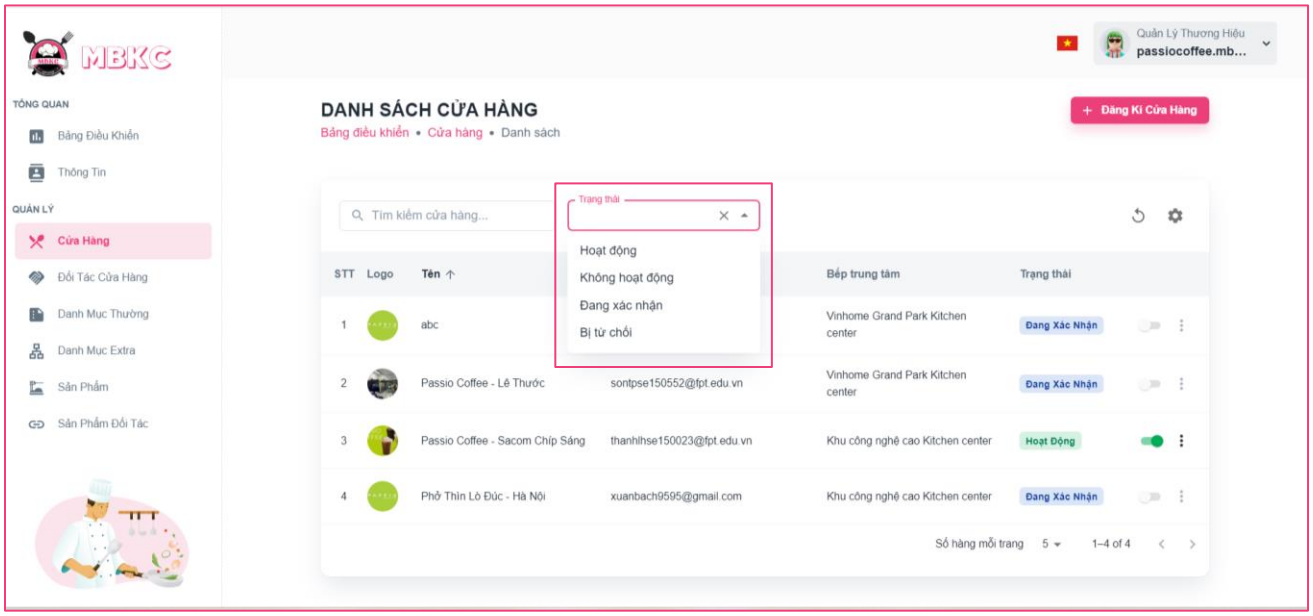


Figure 201: User Manual - View list Store – Step 4 – Case 2

Case 3: Click the Reload icon button to reload the list of stores.

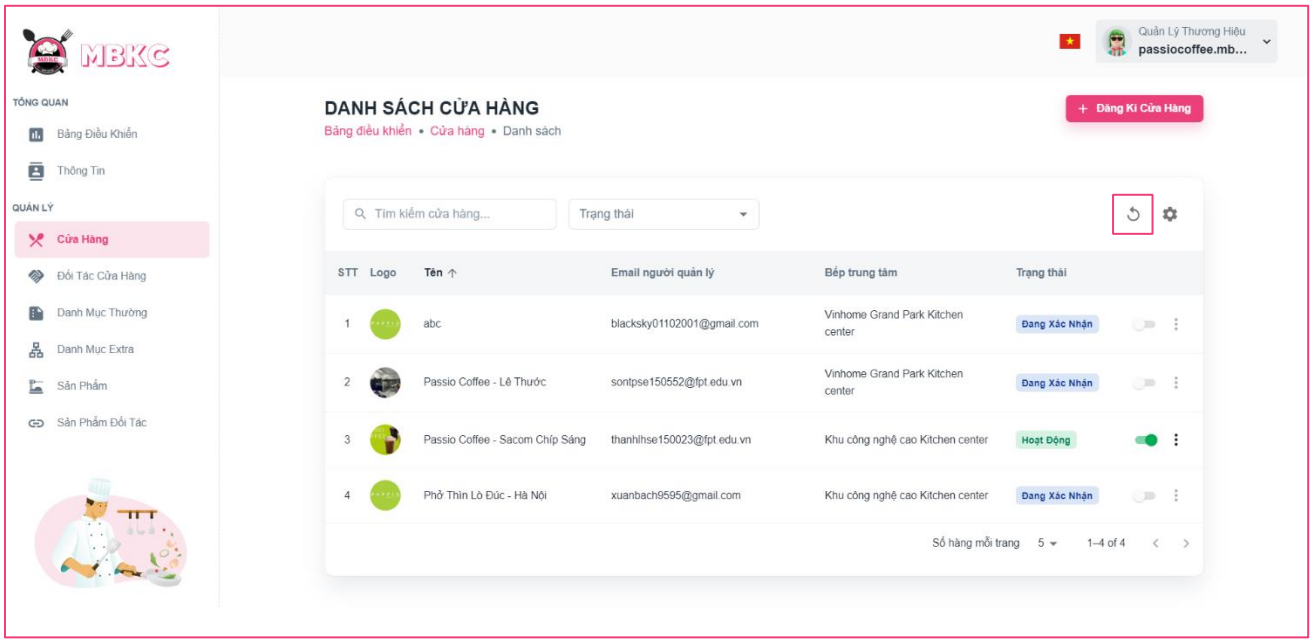


Figure 202: User Manual - View list Store – Step 4 – Case 2

Case 4: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of the store.

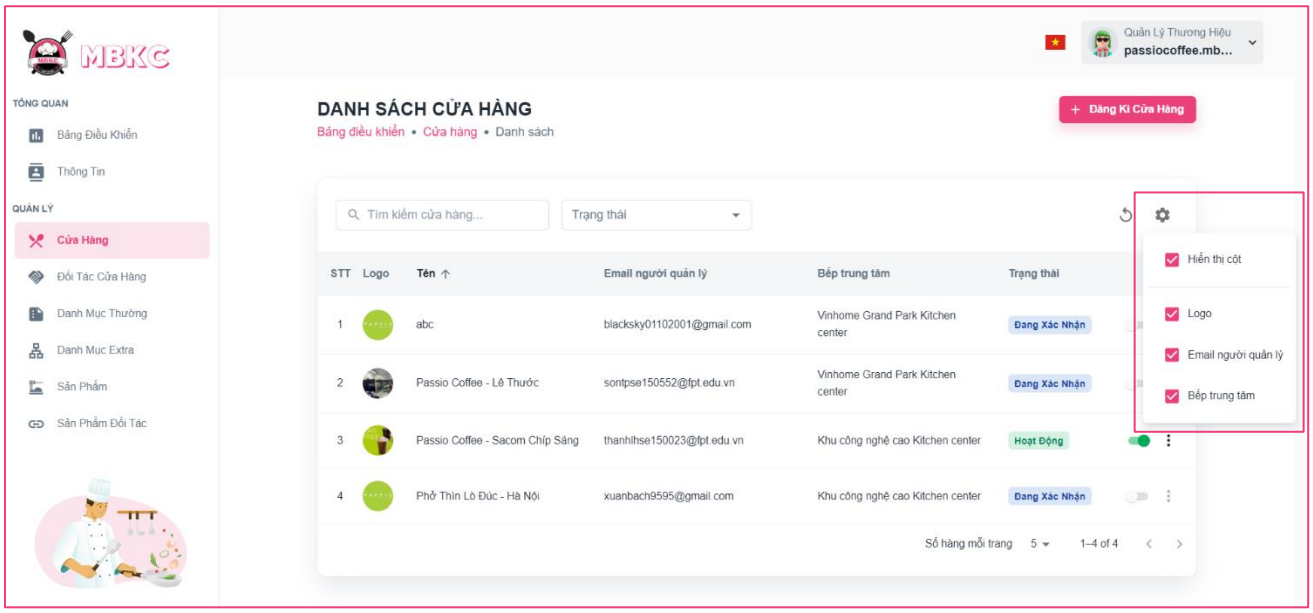


Figure 203: User Manual - View list Store – Step 4 – Case 4

3.2.4.3. View details Store

3.2.4.3.1. Description

Use this guideline to view details store.

This web application supports view details store with account as a MBKC Admin, Kitchen Center Manager, and Brand Manager.

3.2.4.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password”.

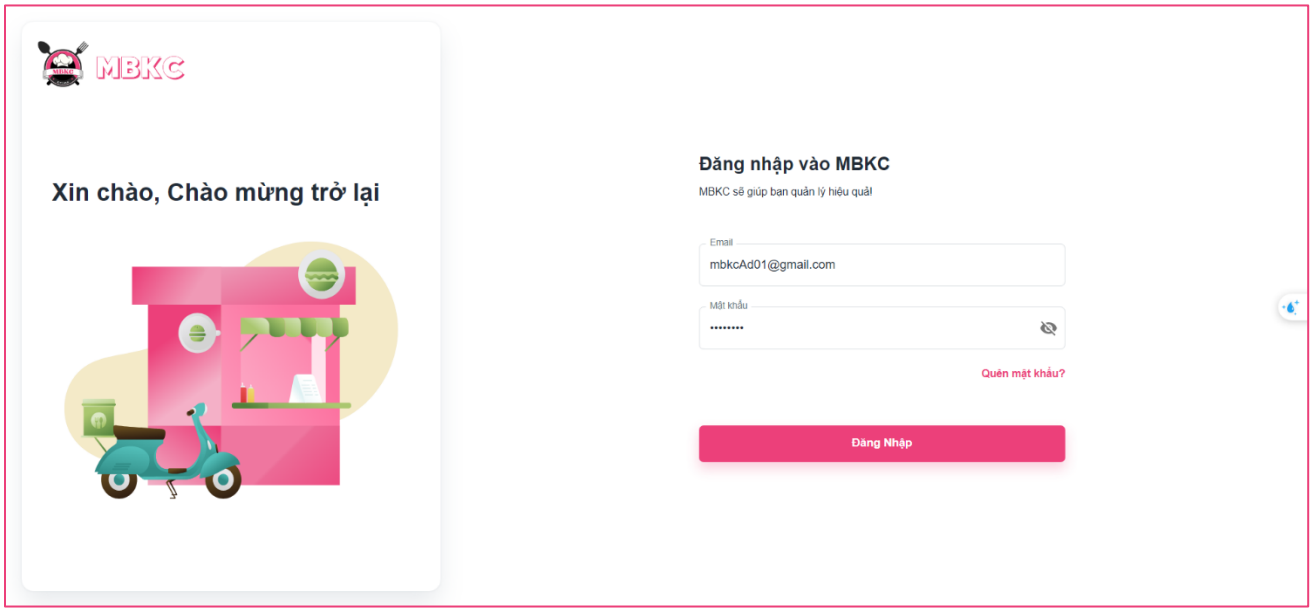


Figure 204: User Manual - View details Store – Step 1

Step 2: Click “Đăng Nhập” button.

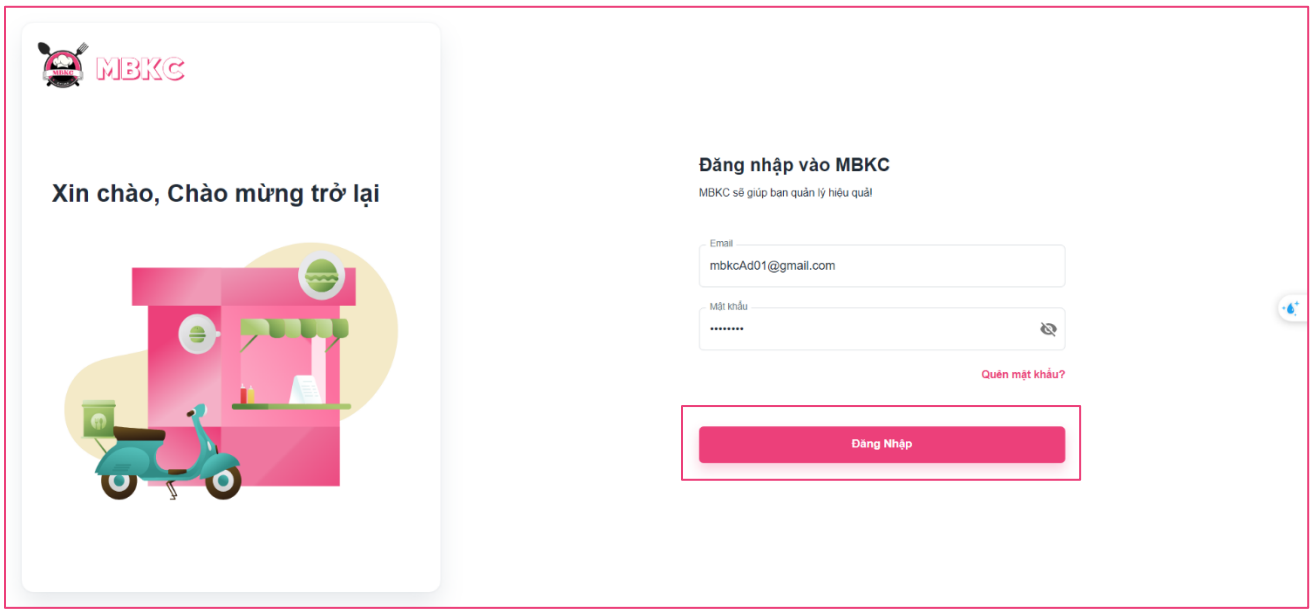


Figure 205: User Manual - View details Store – Step 2

Step 3: From Dashboard Page, choose “Cửa Hàng” from the left sidebar.

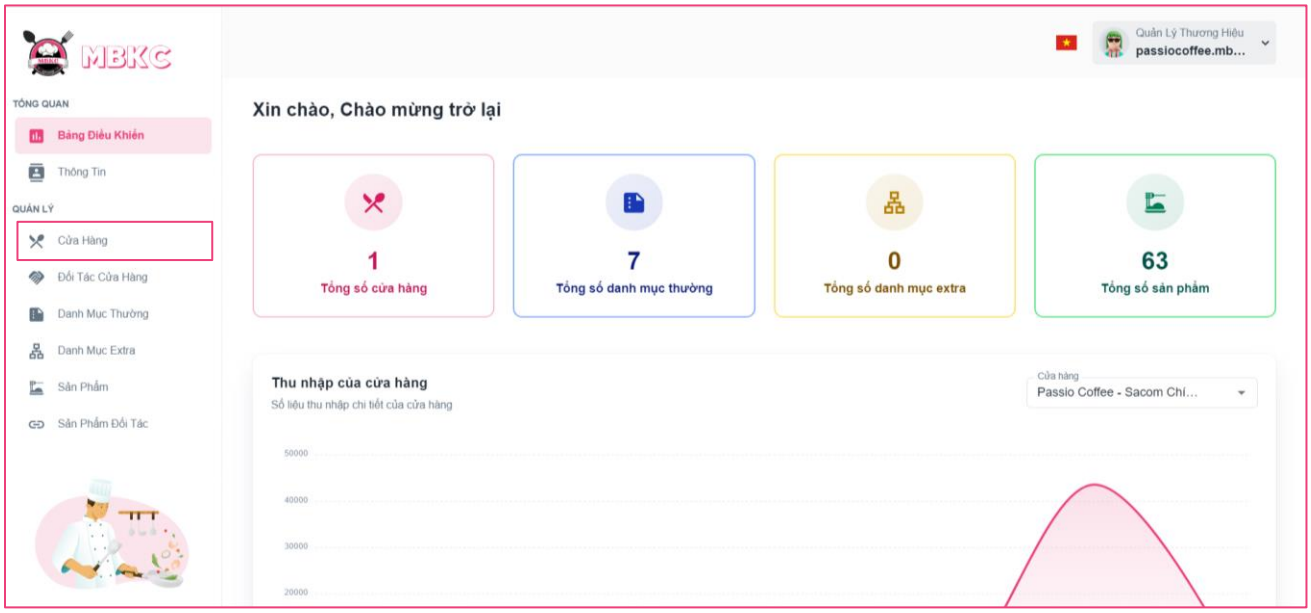


Figure 206: User Manual - View details Store – Step 3

Step 4: Click the store row to navigate to Store Details Page.

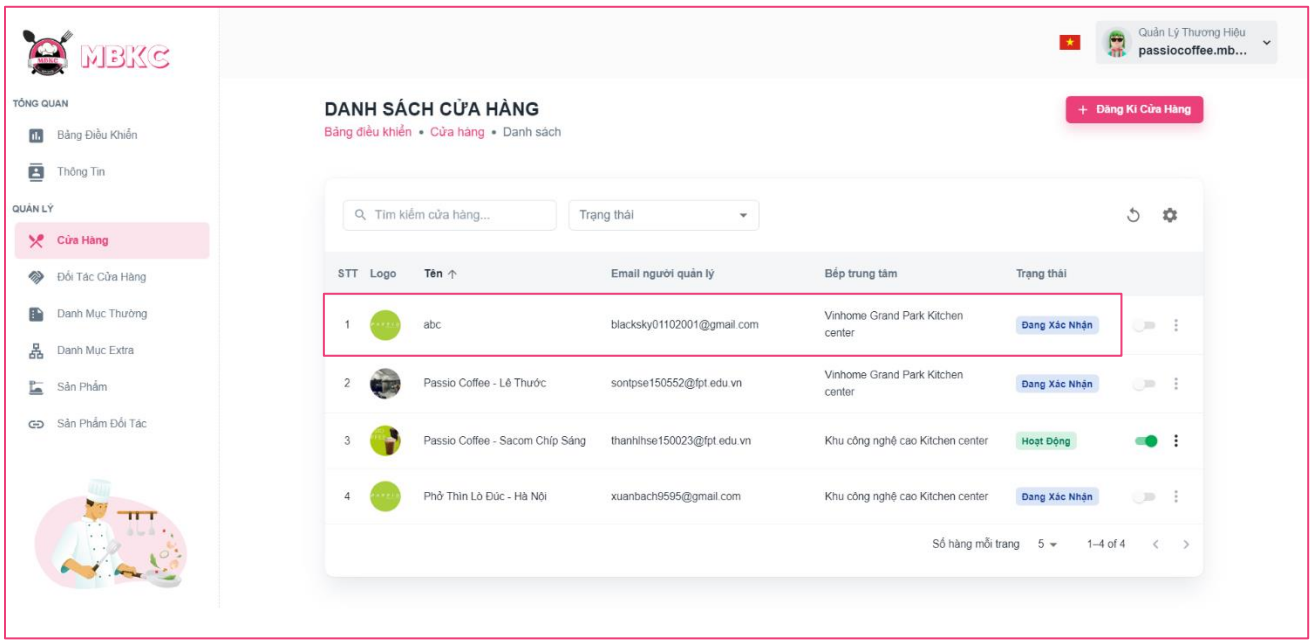


Figure 207: User Manual - View details Store – Step 4

Step 5: Navigate to Store Details Page to show information details of store.

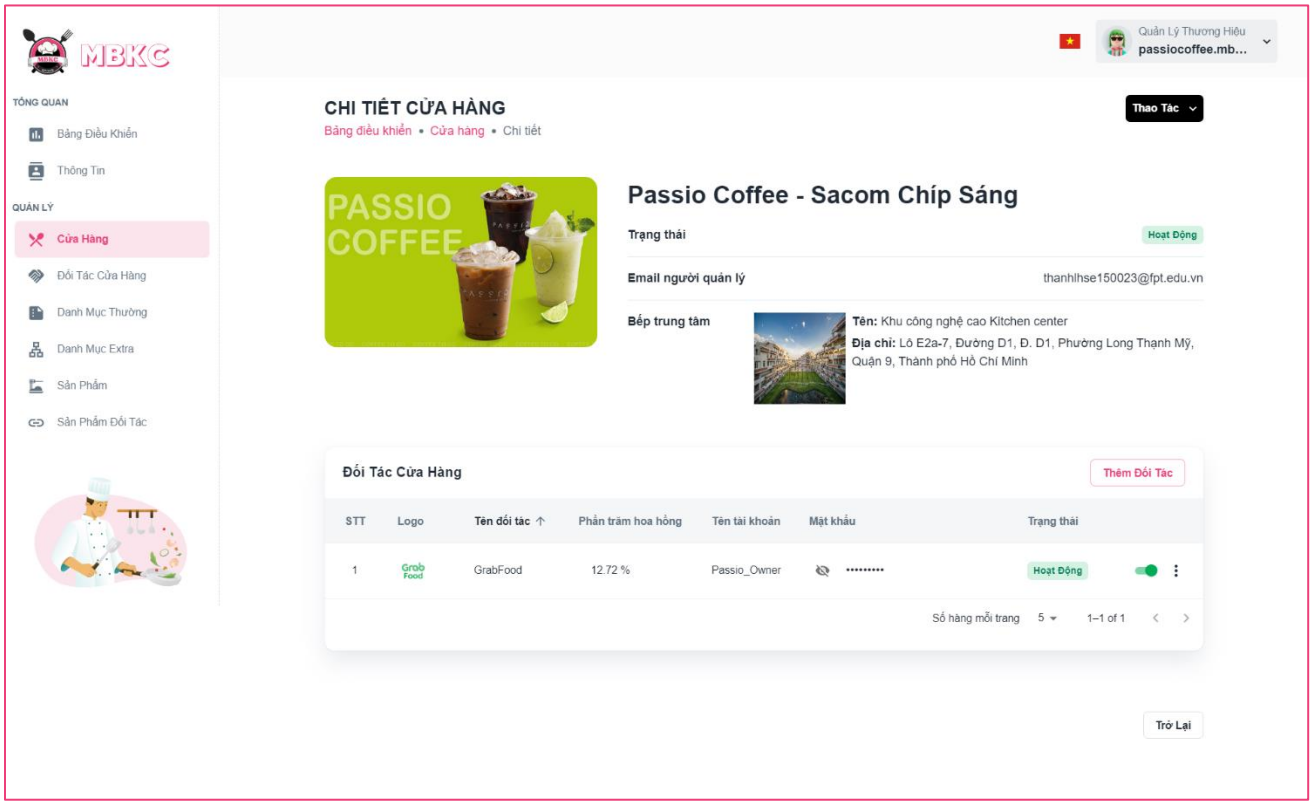


Figure 208: User Manual - View details Store – Step 5

3.2.4.4. Confirm Store Registration

3.2.4.4.1. Description

Use this guideline to confirm store registration.

This web application supports confirm store registration with account as a MBKC Admin.

3.2.4.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

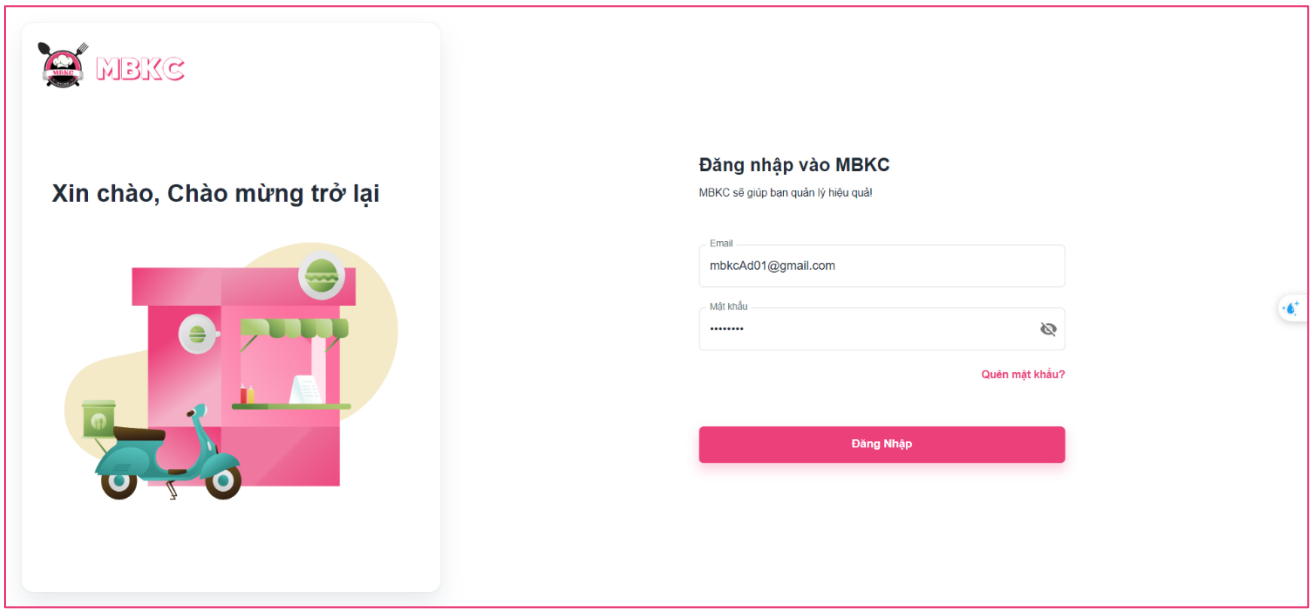


Figure 209: User Manual – Confirm Store Registration – Step 1

Step 2: Click “Đăng Nhập” button.

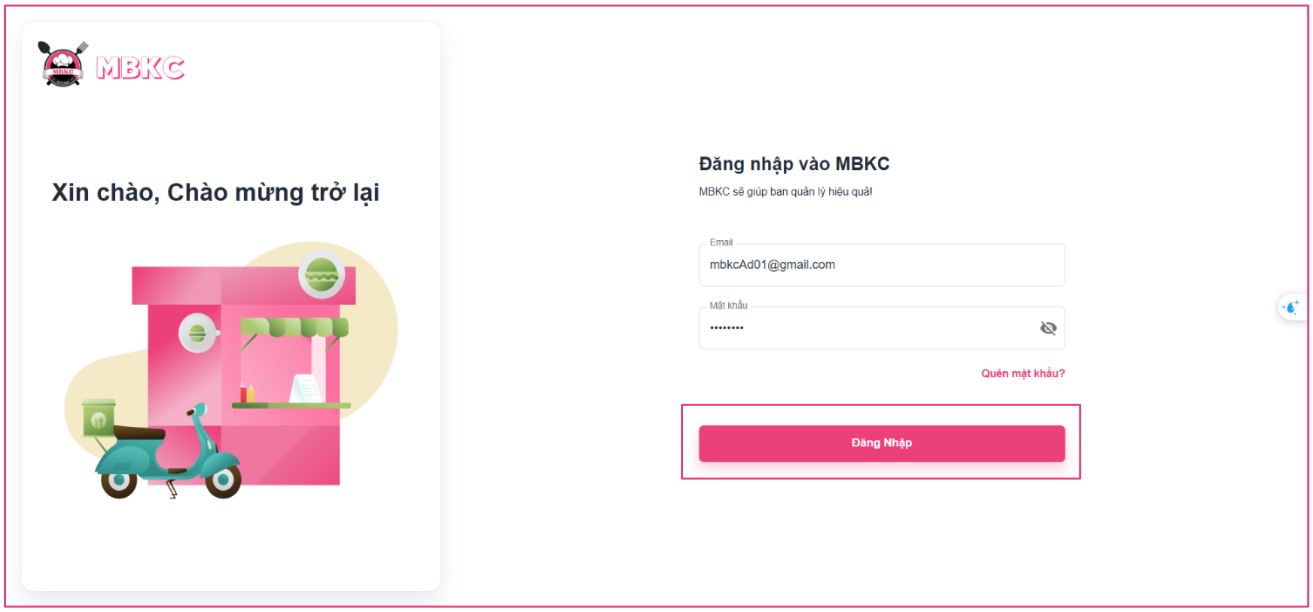


Figure 210: User Manual - Confirm Store Registration – Step 2

Step 3: From Admin Dashboard Page, choose “Cửa Hàng” from the left sidebar.

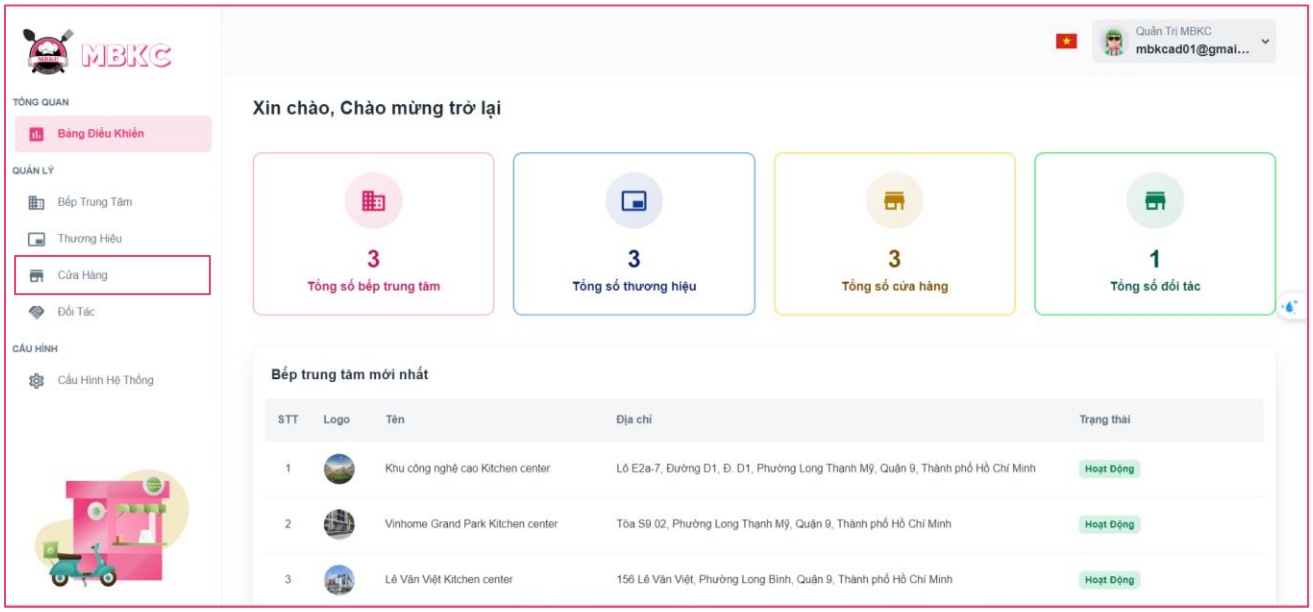


Figure 211: User Manual - Confirm Store Registration – Step 3

Case 1: Accept store registration.

Step 4: Click on Menu icon button and click “Chấp nhận” menu item on the store row having the status “Đang Xác Nhận” – “Confirming”.

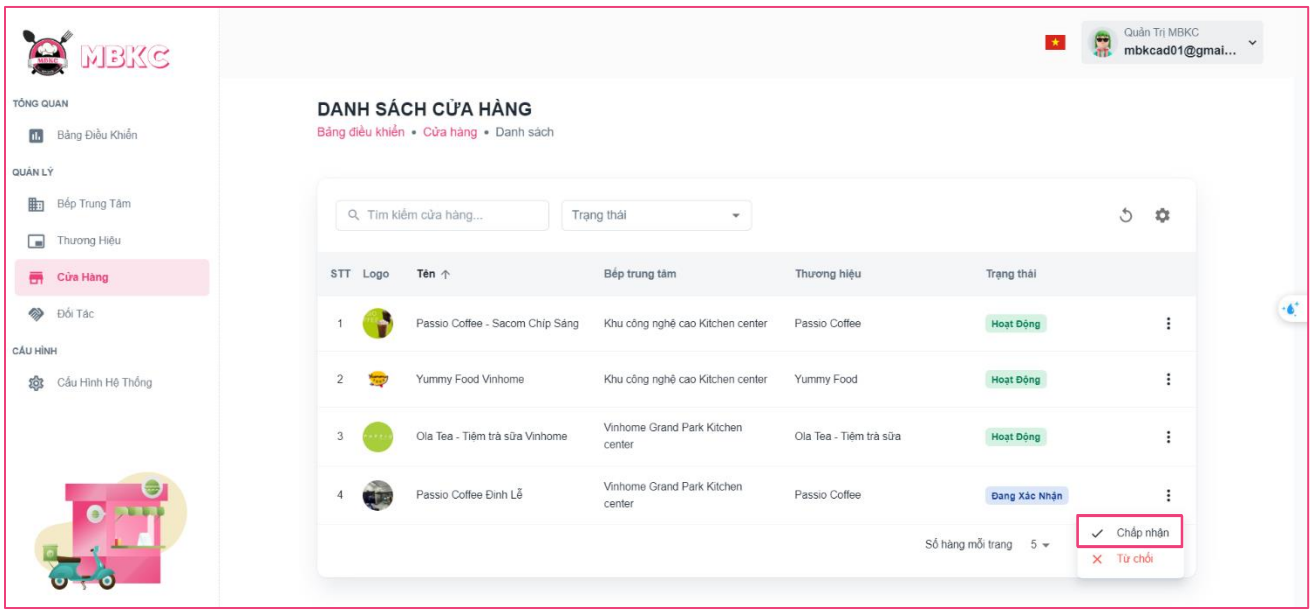


Figure 212: User Manual - Confirm Store Registration - Case 1 - Step 4

Step 5: Click “Chấp Nhận” button to accept store registration.

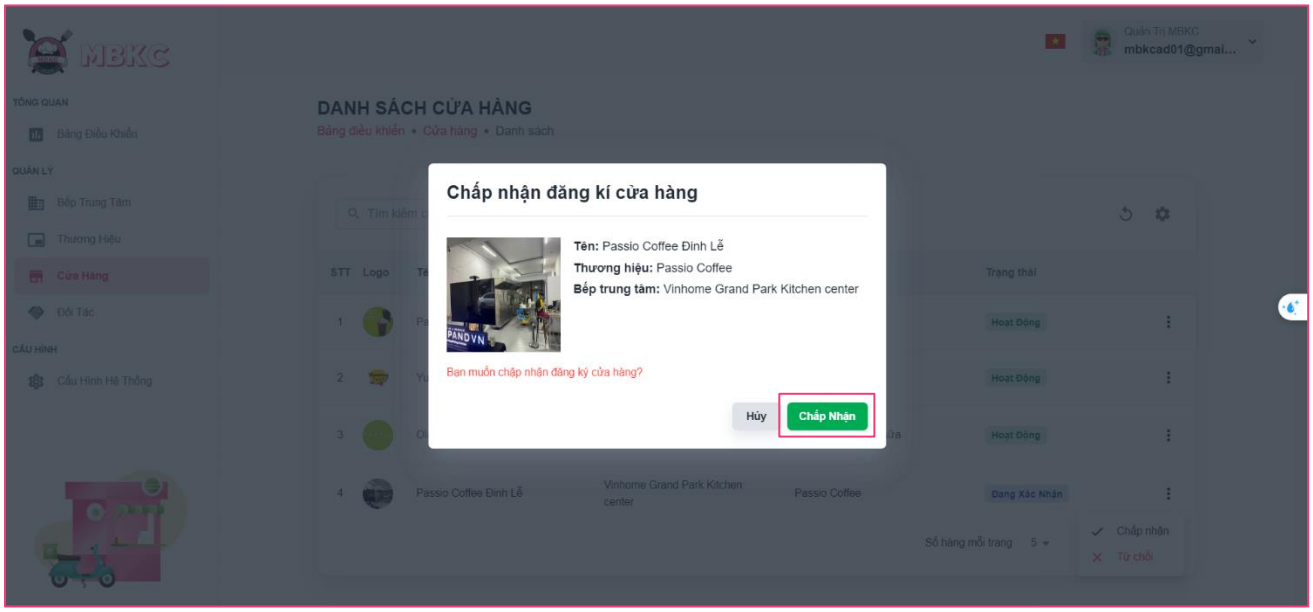


Figure 213: User Manual - Confirm Store Registration - Case 1 - Step 5

Step 6: List of stores will be reloaded and a toast message “Xác nhận đăng ký cửa hàng thành công”. And the status of the store has been updated.

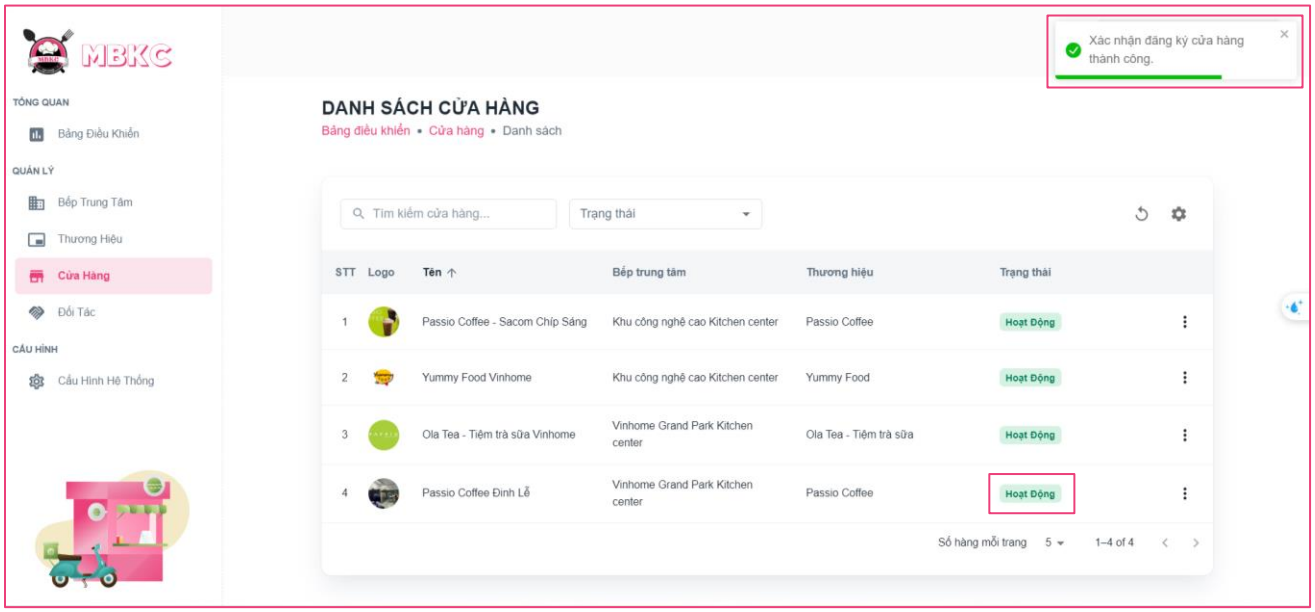


Figure 214: User Manual - Confirm Store Registration - Case 1 - Step 6

Case 2: Reject store registration.

Step 4: Click on Menu icon button and click “Từ chối” menu item on the store row having the status “Đang Xác Nhận” – “Confirming”.

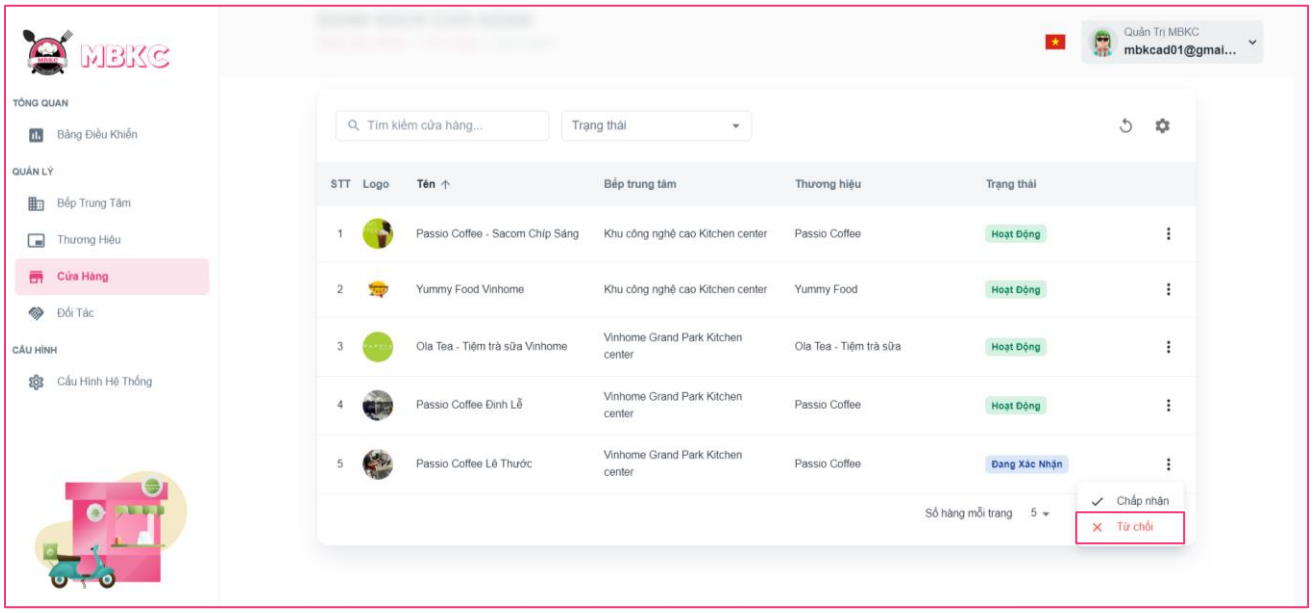


Figure 215: User Manual - Confirm Store Registration - Case 2 - Step 4

Step 5: Fill in “Nội dung” as the reason for rejecting store registration. And then click “Từ Chối” button to reject store registration.

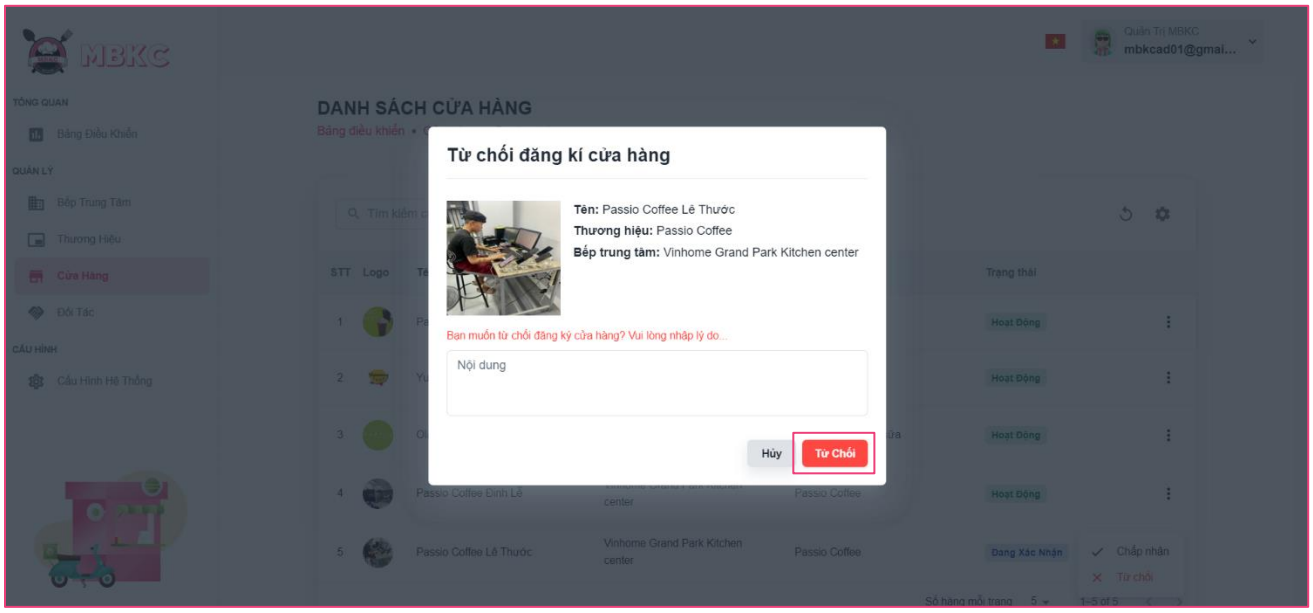


Figure 216: User Manual - Confirm Store Registration - Case 1 - Step 5

Step 6: List of stores will be reloaded and a toast message “Xác nhận đăng ký cửa hàng thành công”. And the status of the store has been updated.

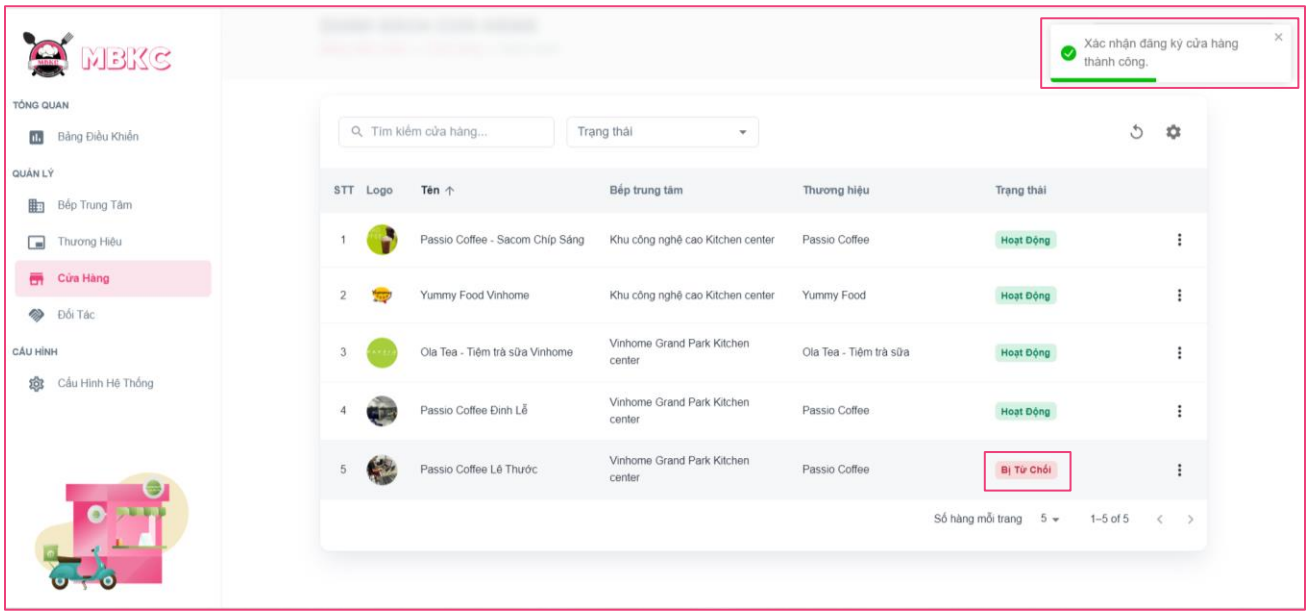


Figure 217: User Manual - Confirm Store Registration - Case 1 - Step 6

3.2.4.5. Update Status Store

3.2.4.5.1. Description

Use this guideline to update the status of the store.

This web application supports update the status of the store with account as a Brand Manager.

3.2.4.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Band Manager.

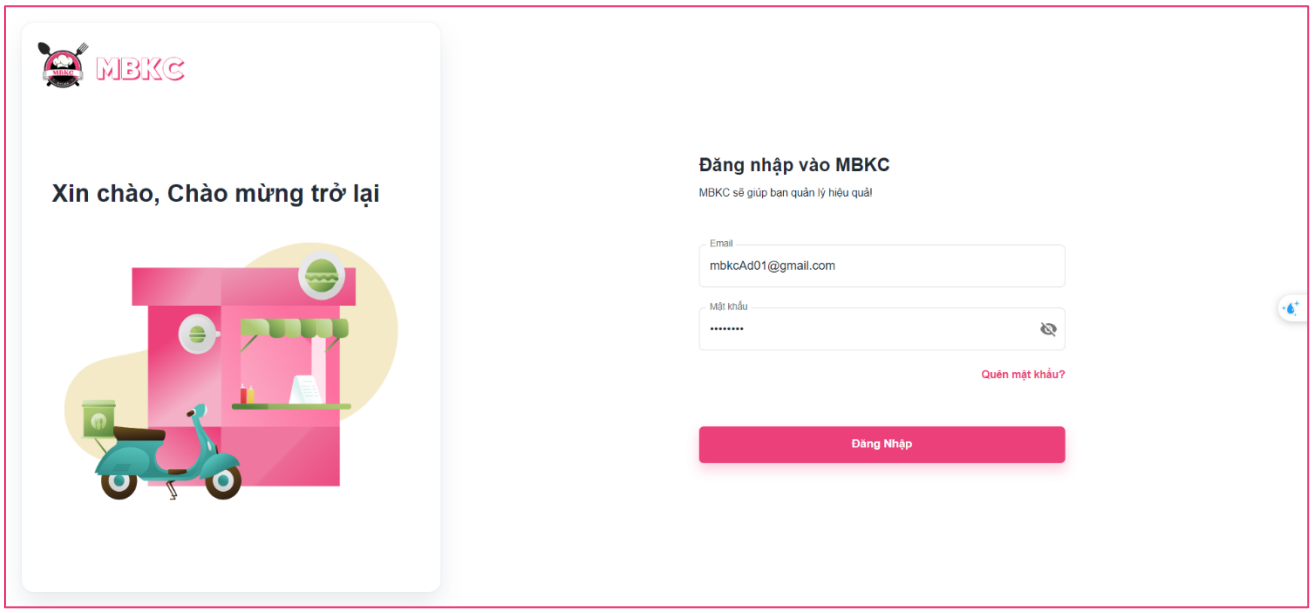


Figure 218: User Manual – Update Status Store – Step 1

Step 2: Click “Đăng Nhập” button.

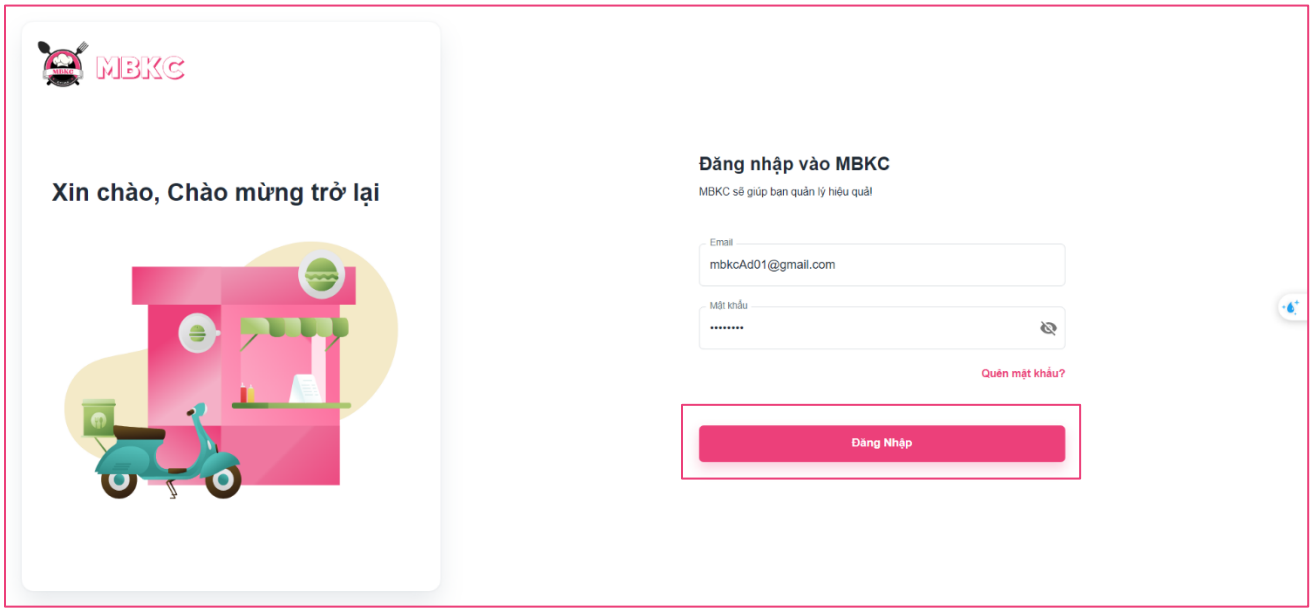


Figure 219: User Manual - Update Status Store – Step 2

Step 3: From Brand Dashboard Page, choose “Cửa Hàng” from the left sidebar.

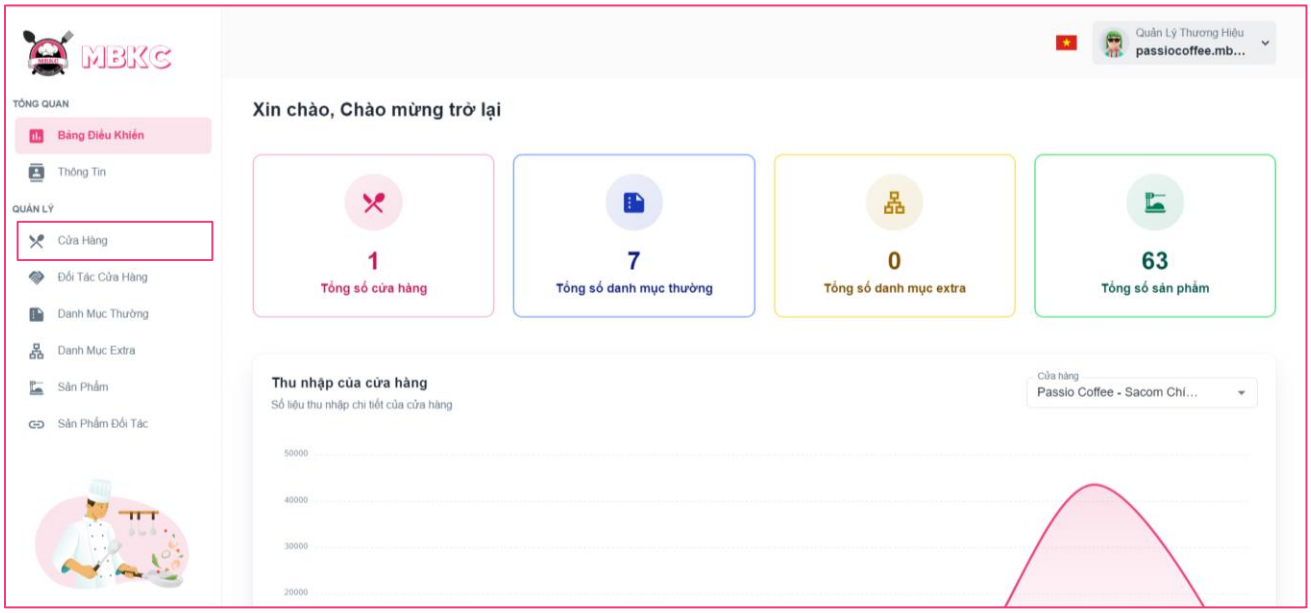


Figure 220: User Manual - Update Status Store – Step 3

Step 4: Click on the switch button on the store row have status “Hoạt Động” – “Active” or “Không Hoạt Động” – “Inactive” to update the status.

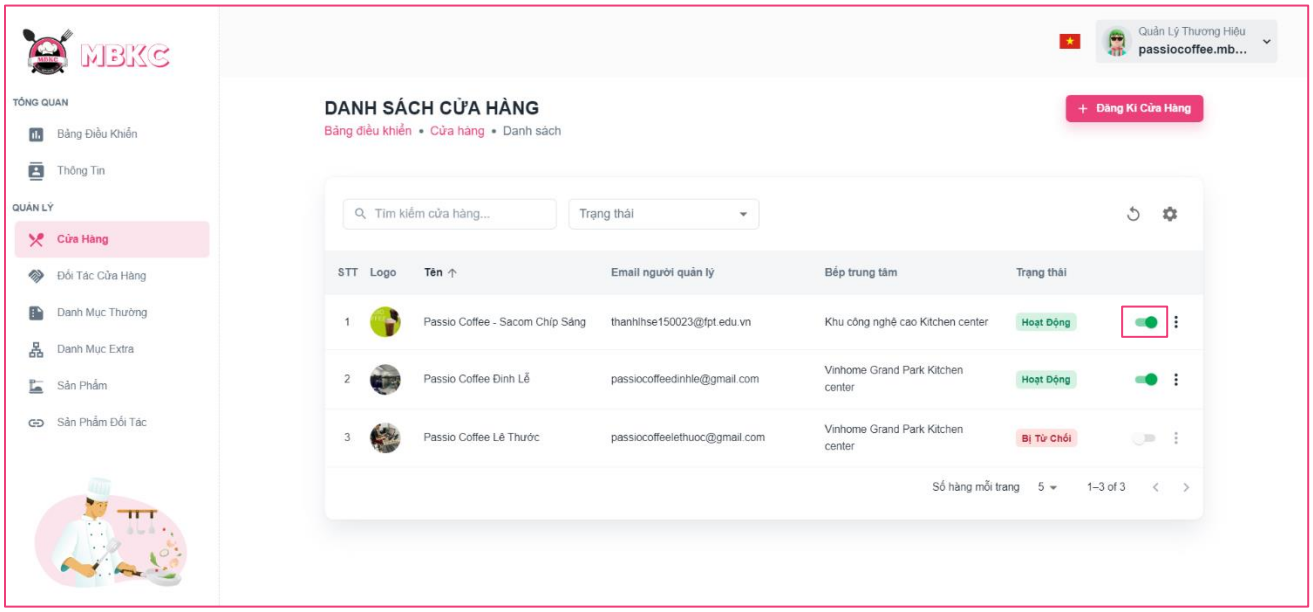


Figure 221: User Manual - Update Status Store – Step 4

Step 5: List of stores will be reloaded and a toast message “Đã cập nhật trạng thái cửa hàng thành công”. And the status of the store has been updated.

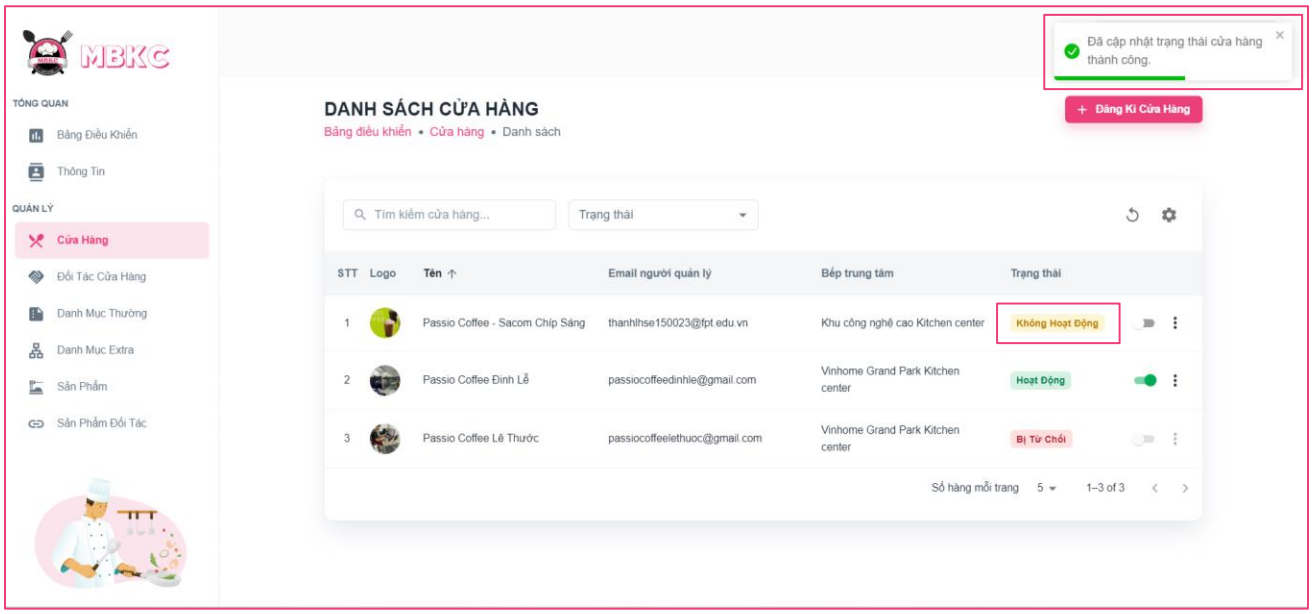


Figure 222: User Manual - Update Status Store – Step 5

3.2.4.6. Update Information Store

3.2.4.6.1. Description

Use this guideline to update information of the store.

This web application supports update information of the store with account as a Brand Manager.

3.2.4.6.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Band Manager.

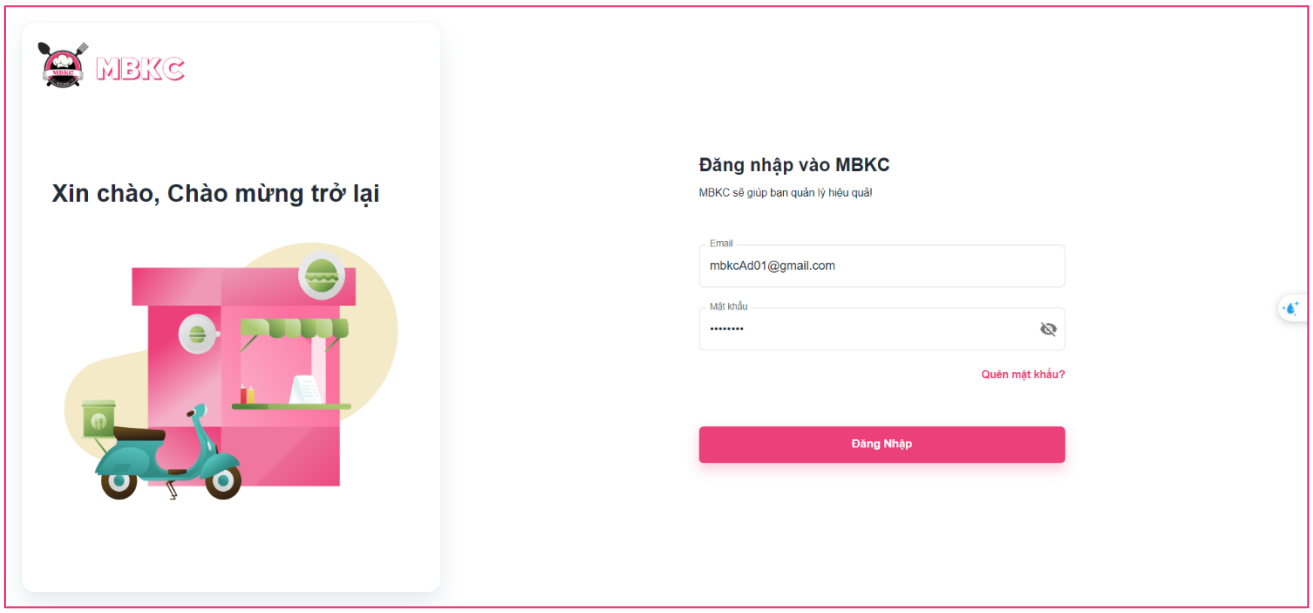


Figure 223: User Manual – Update Information Store – Step 1

Step 2: Click “Đăng Nhập” button.

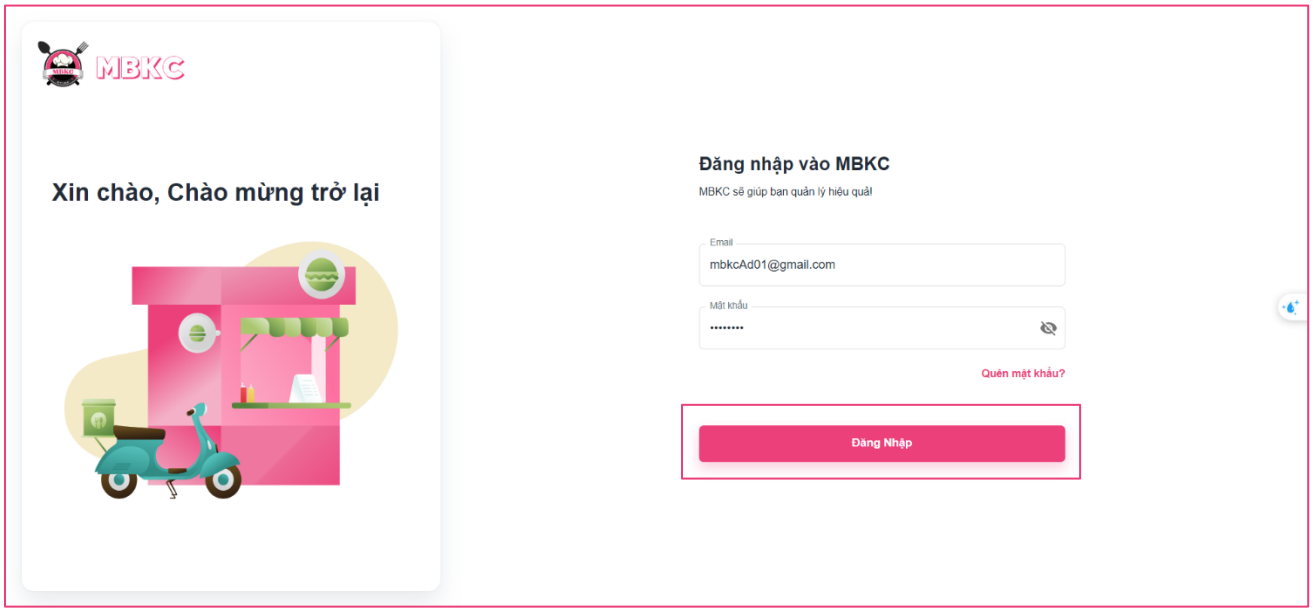


Figure 224: User Manual - Update Information Store – Step 2

Step 3: From Brand Dashboard Page, choose “Cửa Hàng” from the left sidebar.

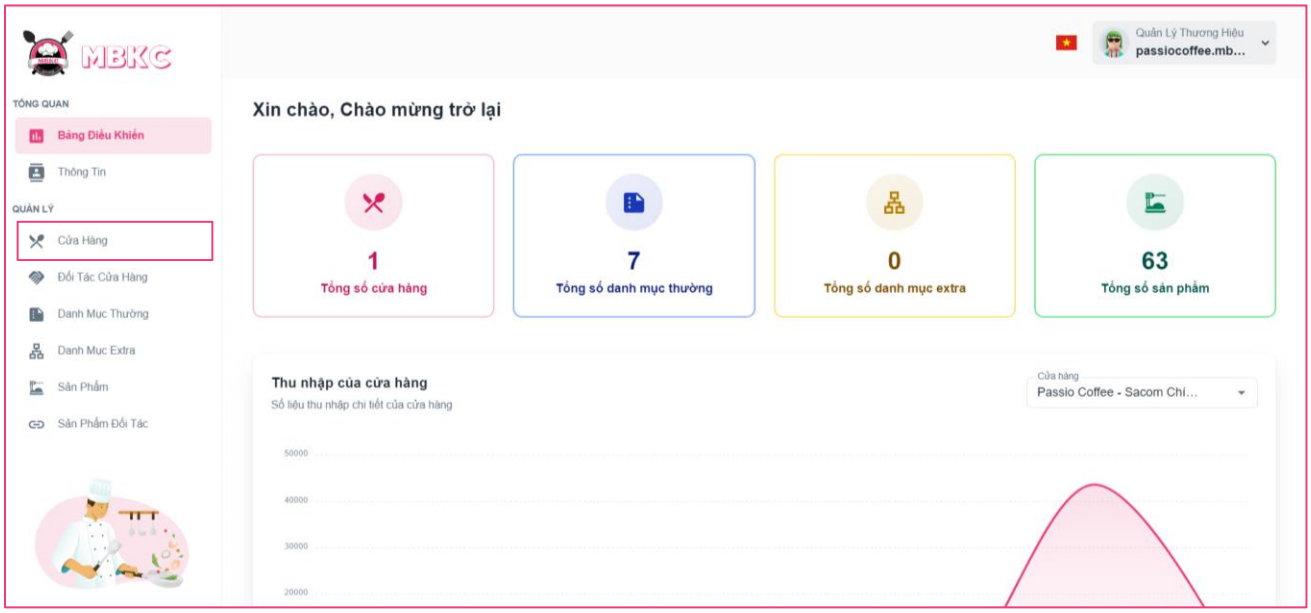


Figure 225: User Manual - Update Information Store – Step 3

Step 4: Update information of the store have status “Hoạt Động” – “Active” or “Không Hoạt Động” – “Inactive” to update the information.

Case 1: Update information of the store in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the store row want to navigate Update Store Page.

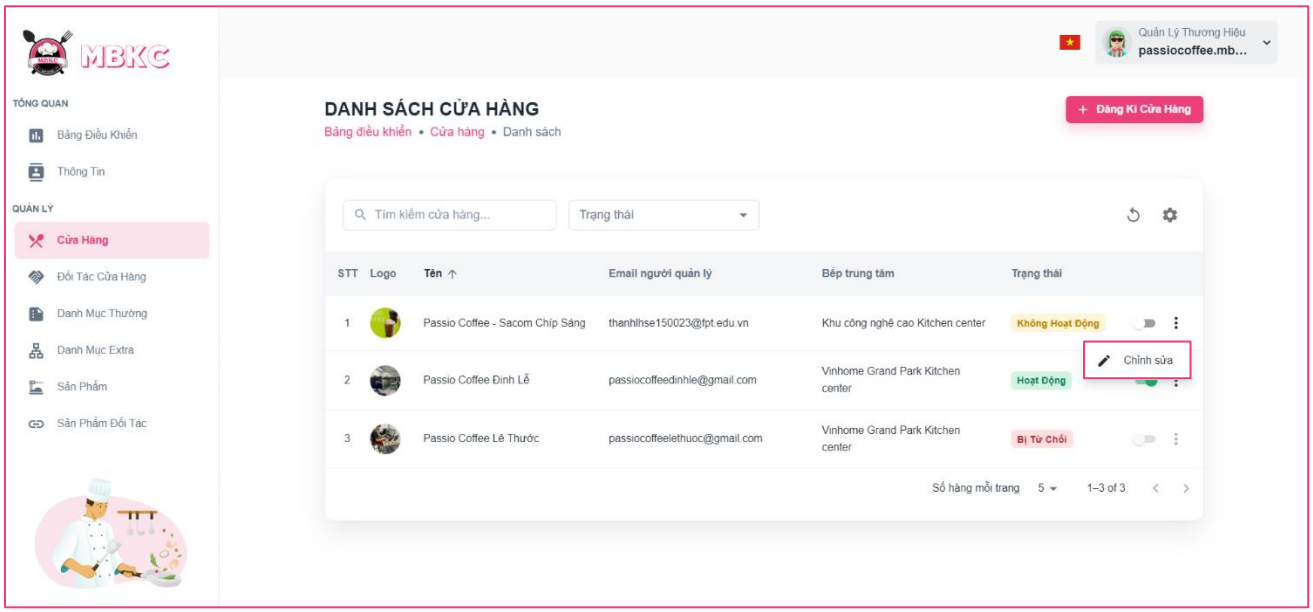


Figure 226: User Manual - Update Information Store – Step 4 – Case 1

Case 2: Update information of the store on the details page.

1: Click the store row to navigate to Store Details Page.

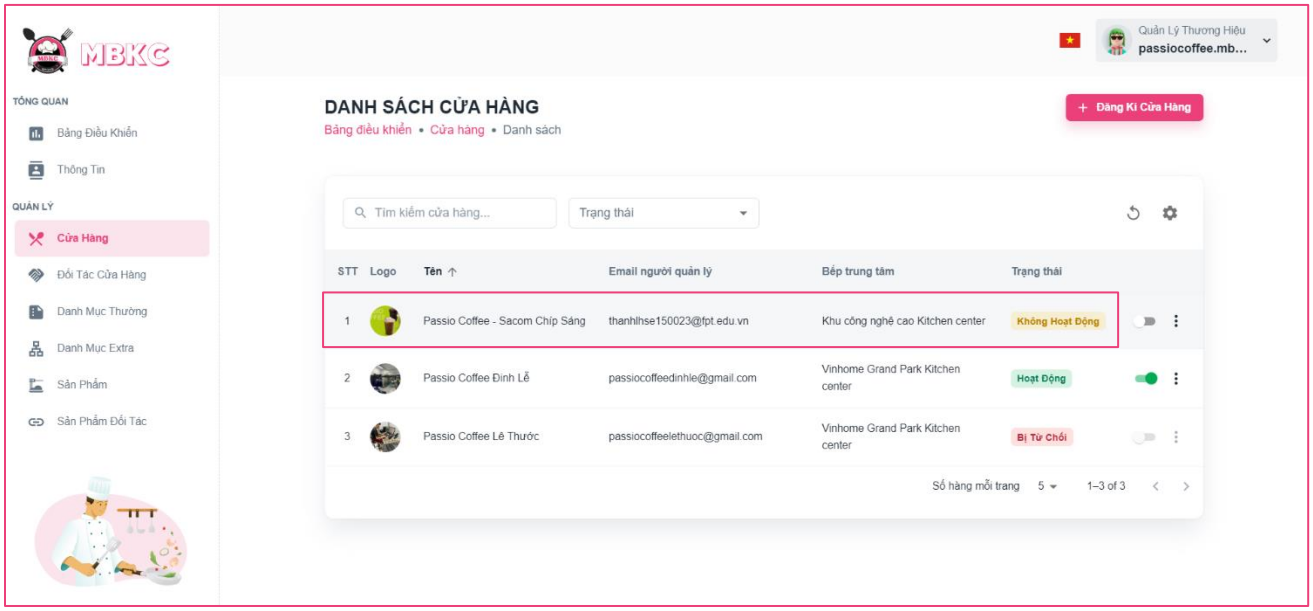


Figure 227: User Manual - Update Information Store – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Chỉnh sửa” menu item to navigate Update Store Page.

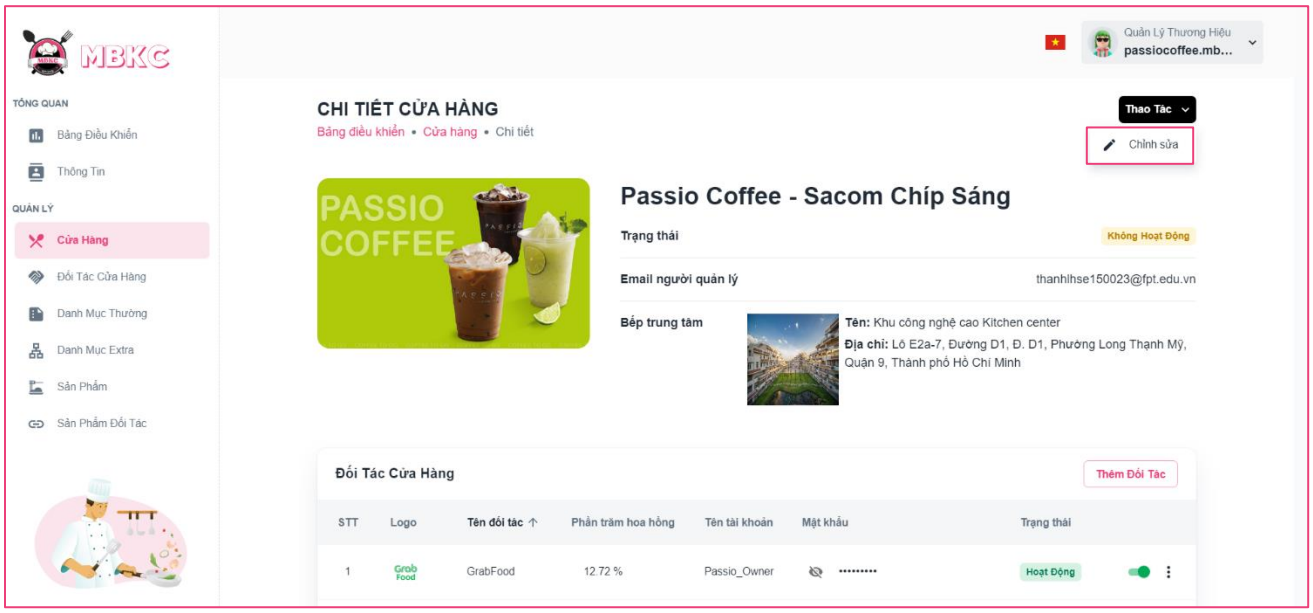


Figure 228: User Manual - Update Information Store – Step 4 – Case 2 - 2

Step 5: Store information will be automatically filled in the form. Then edit the information as desired.

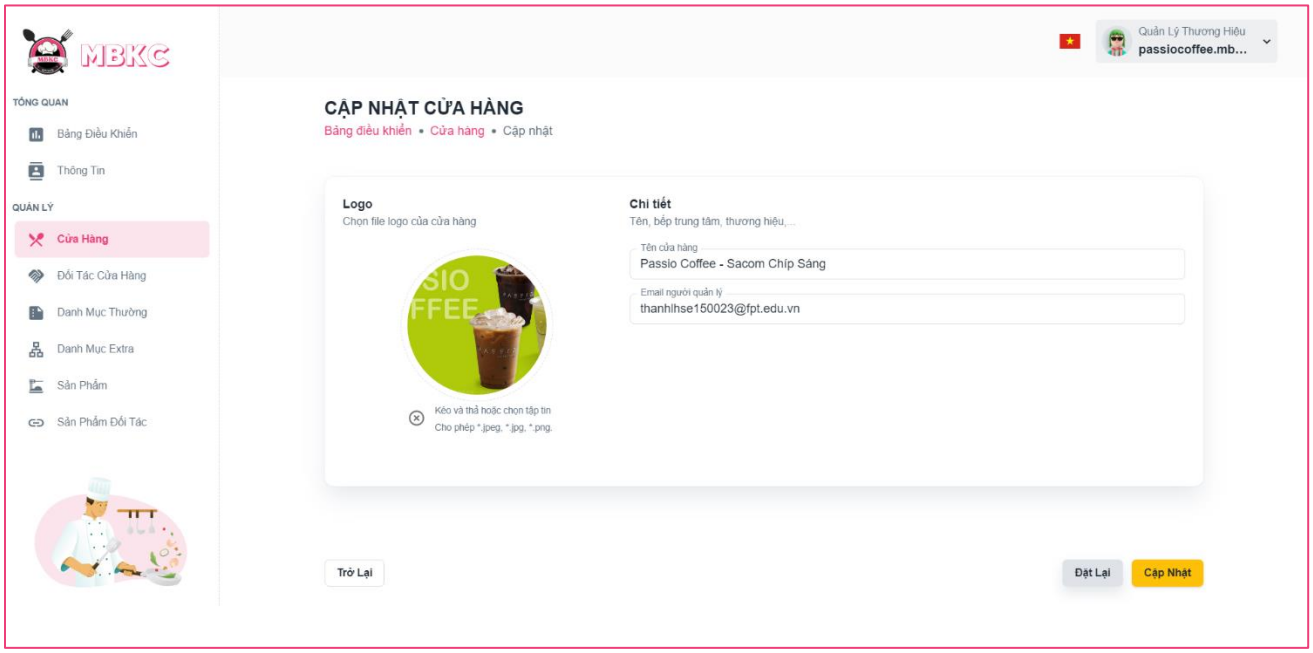


Figure 229: User Manual - Update Information Store – Step 5

Step 6: Click “Cập nhật” button.

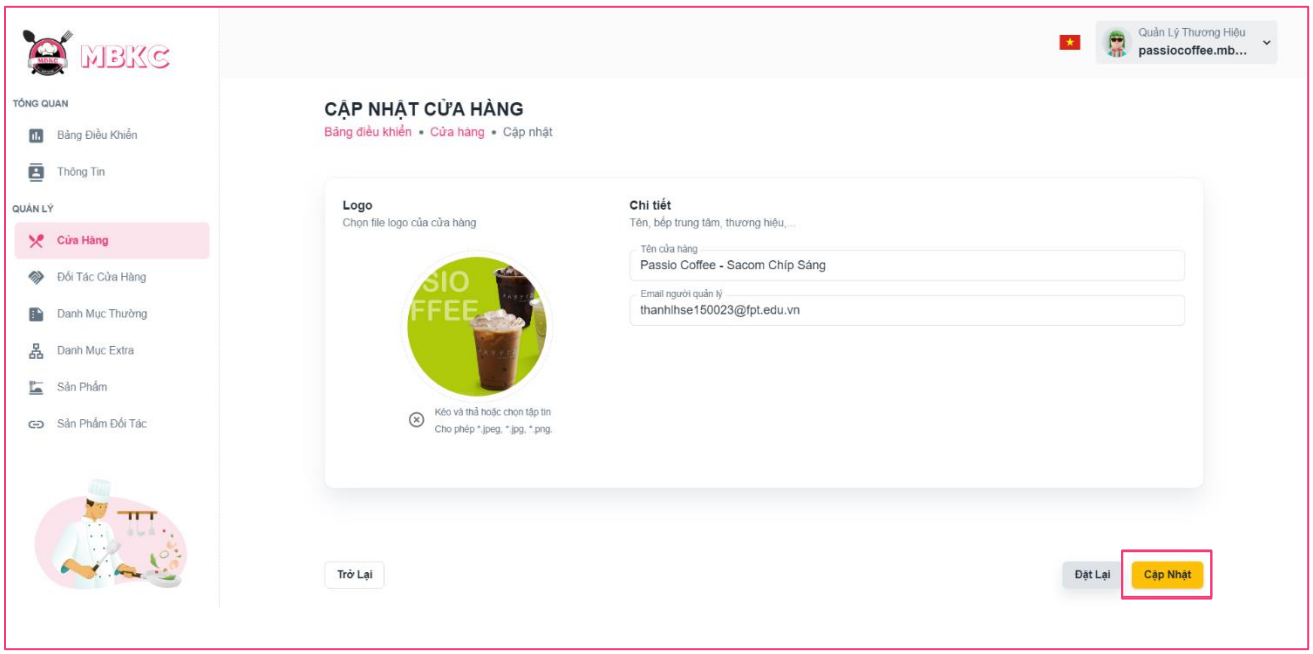


Figure 230: User Manual - Update Information Store – Step 6

Step 7:

Case 1: You will be navigated to Store Detail Page showing the store details when start edit information of store from Store Detail Page and a toast message “Cập nhật thông tin cửa hàng thành công”.

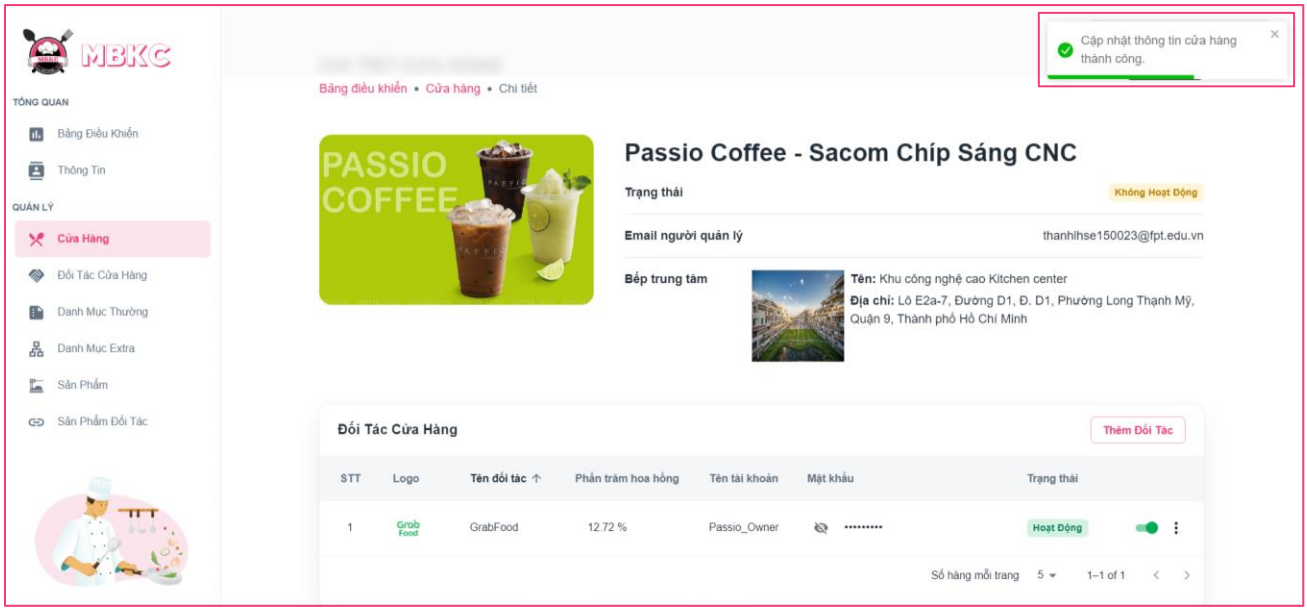


Figure 231: User Manual - Update Information Store – Step 7 – Case 1

Case 2: You will be navigated to List of Store Page showing the list of stores when start edit information of store from List of Store Page and a toast message “Cập nhật thông tin cửa hàng thành công”.

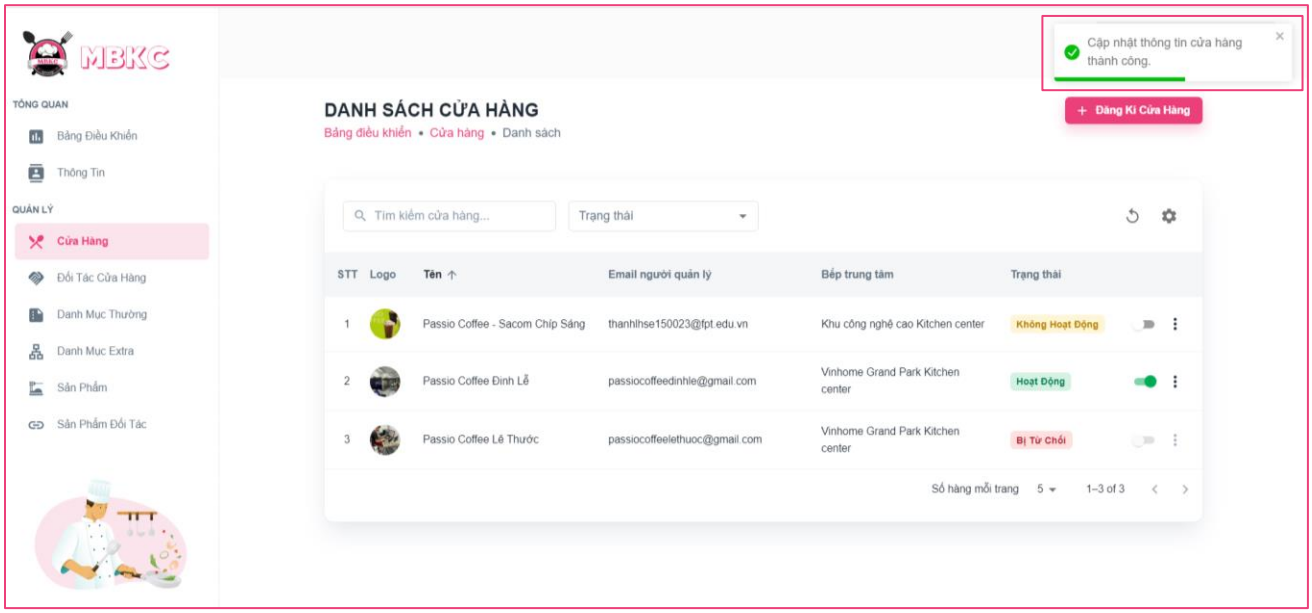


Figure 232: User Manual - Update Information Store – Step 7 – Case 2

3.2.4.7. Delete store

3.2.4.7.1. Description
Use this guideline to delete store.

This web application supports delete store with account as a MBKC Admin.

3.2.4.7.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

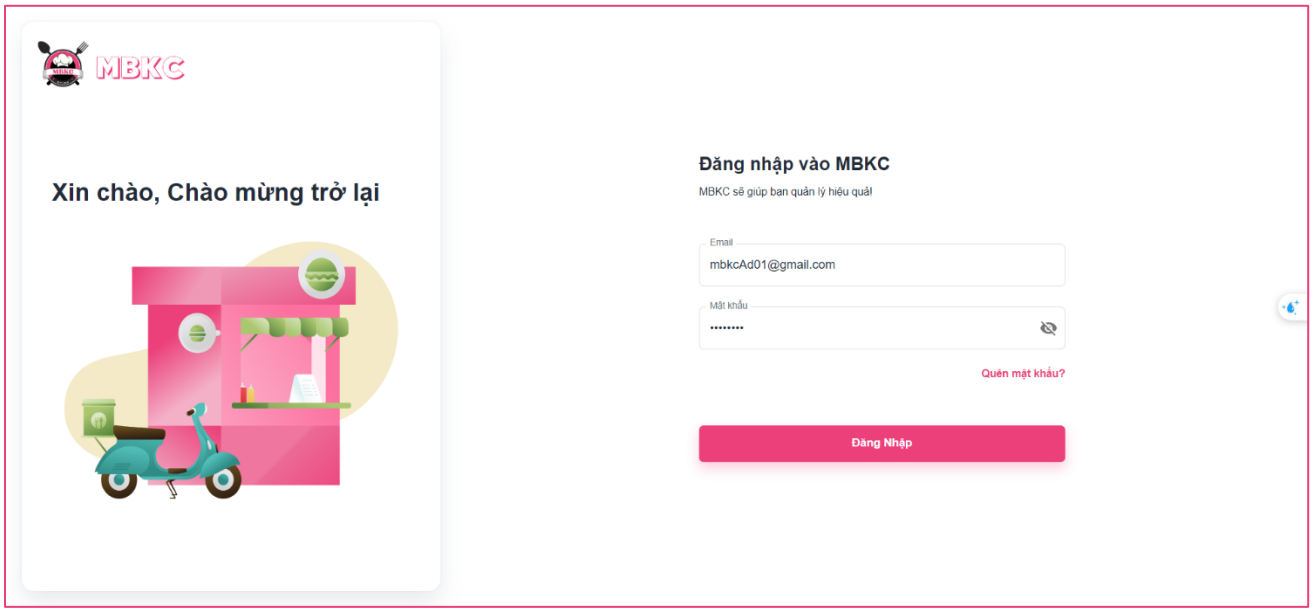


Figure 233: User Manual – Delete Store – Step 1

Step 2: Click “Đăng Nhập” button.

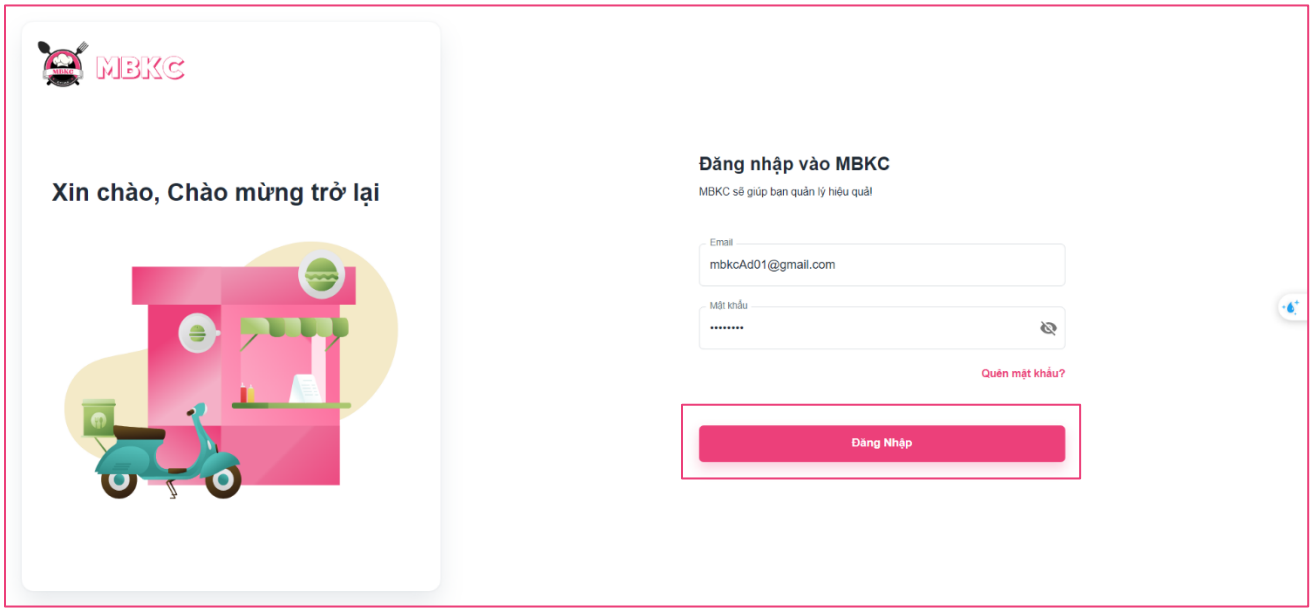


Figure 234: User Manual - Delete Store – Step 2

Step 3: From Admin Dashboard Page, choose “Cửa Hàng” from the left sidebar.

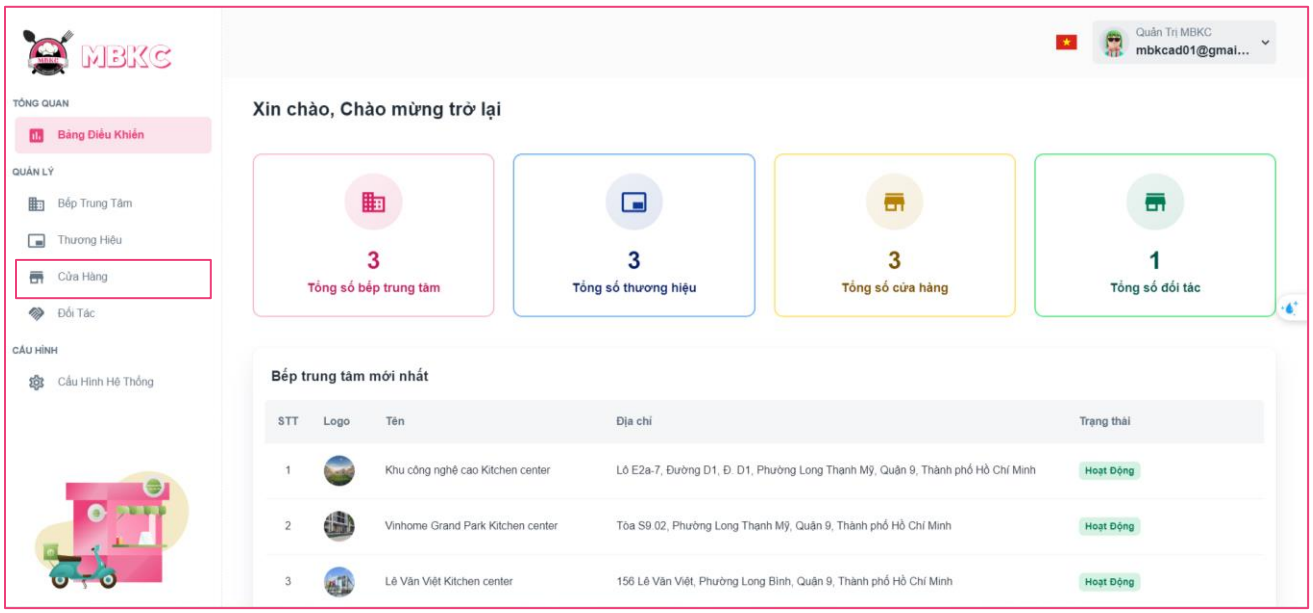


Figure 235: User Manual - Delete Store – Step 3

Step 4: Delete store.

Case 1: Delete store in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the store row have the status “Hoạt Động” – “Active” or “Không Hoạt Động” – “Inactive” or “Bị Từ Chối” – “Rejected” to delete.

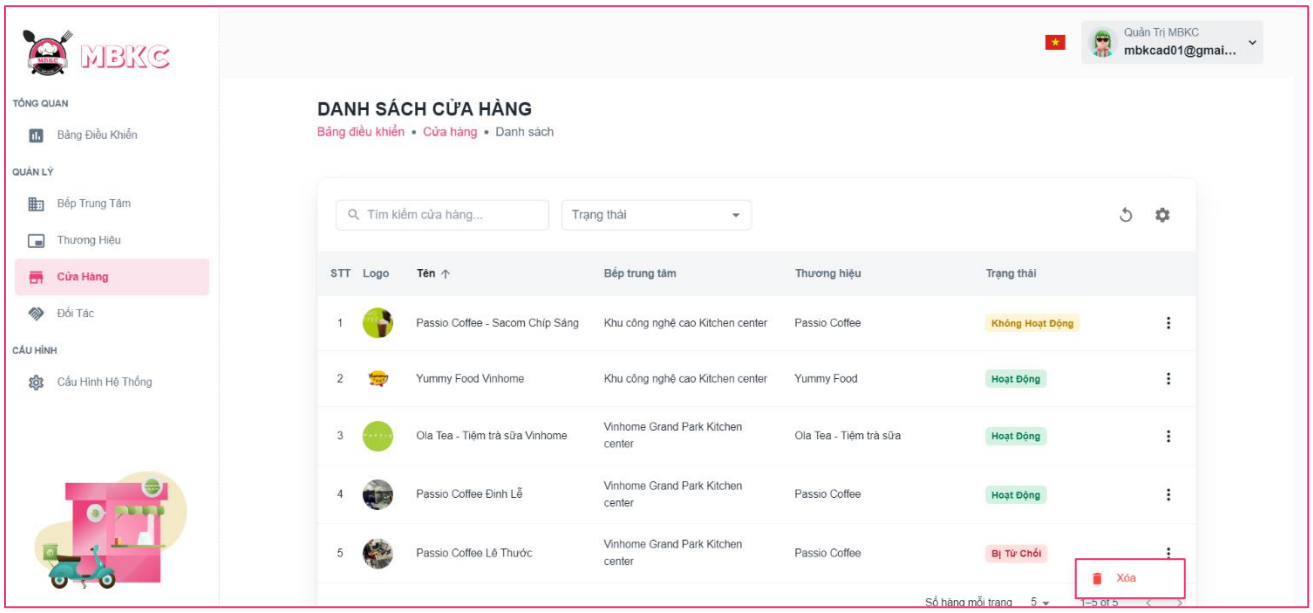


Figure 236: User Manual - Delete Store – Step 4 – Case 1

Case 2: Delete store on the details page.

1: Click the store row to navigate to Store Details Page.

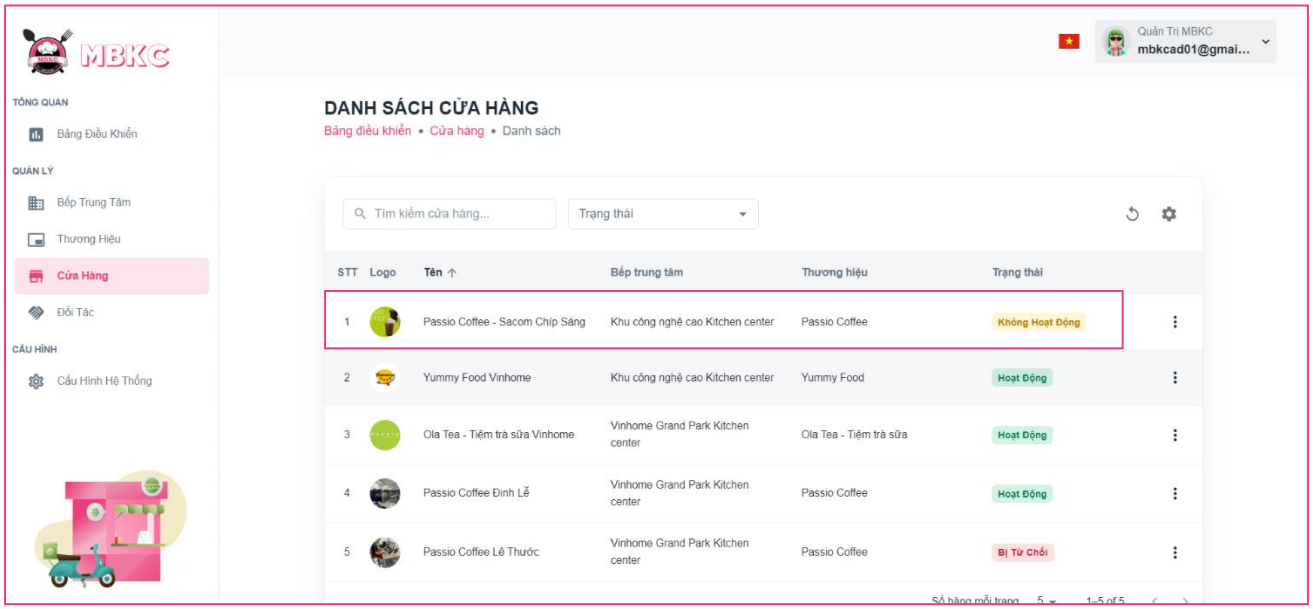


Figure 237: User Manual - Delete Store – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item with store information details have the status “Hoạt Động” – “Active” or “Không Hoạt Động” – “Inactive” or “Bị Từ Chối” – “Rejected” to delete.

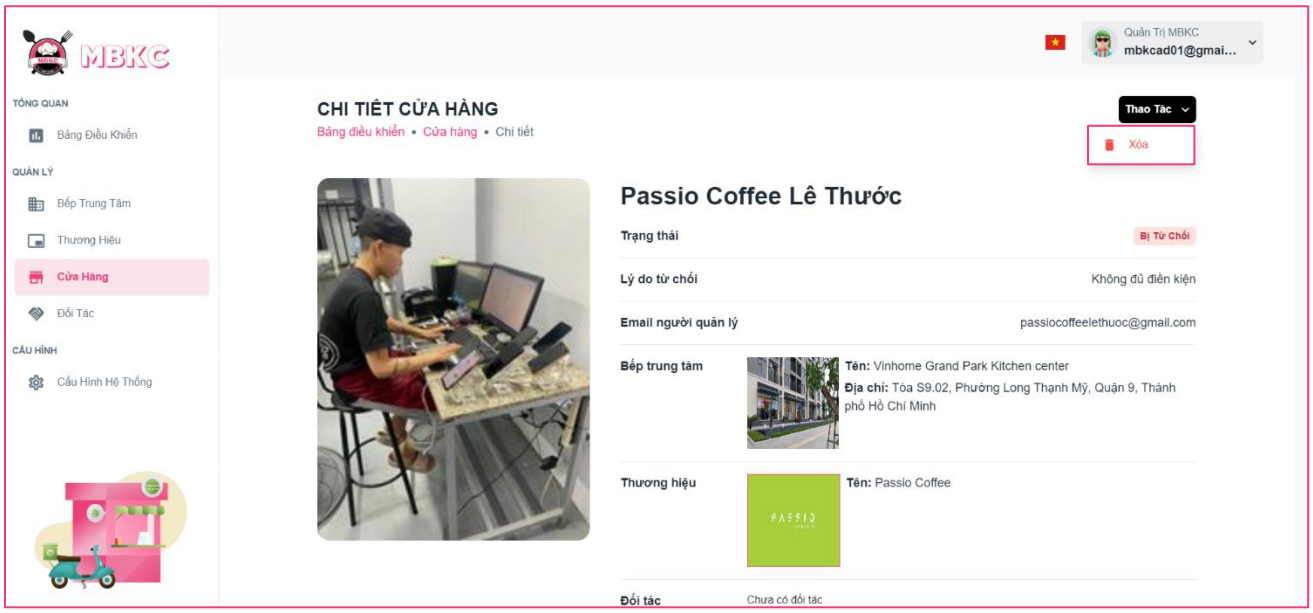


Figure 238: User Manual - Delete Store – Step 4 – Case 2 – 2

Step 5: Click “Xác Nhận” button.

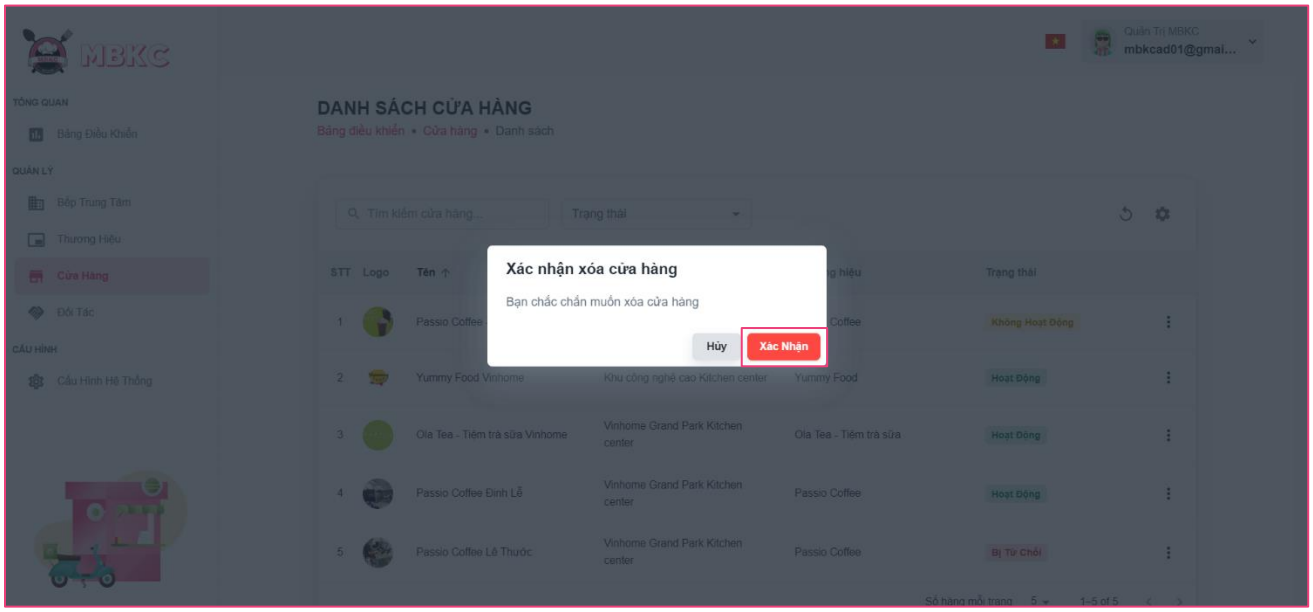


Figure 239: User Manual - Delete Store – Step 5

Step 6: Navigate to List of Stores, and then list of stores will be reloaded and a toast message “Đã xóa cửa hàng thành công”. And store has been deleted.

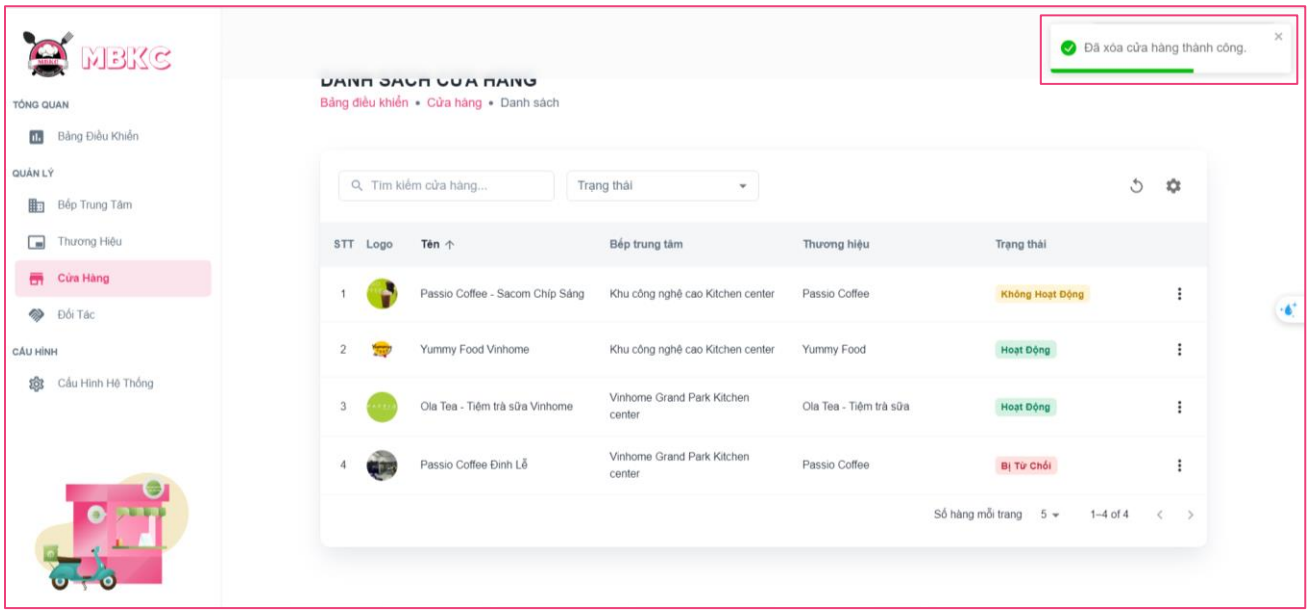


Figure 240: User Manual – Delete Store – Step 6

3.2.5. Partner management

3.2.5.1. View list Partner

3.2.5.1.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of partners.

This web application supports view list of partners with account as a MBKC Admin.

3.2.5.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

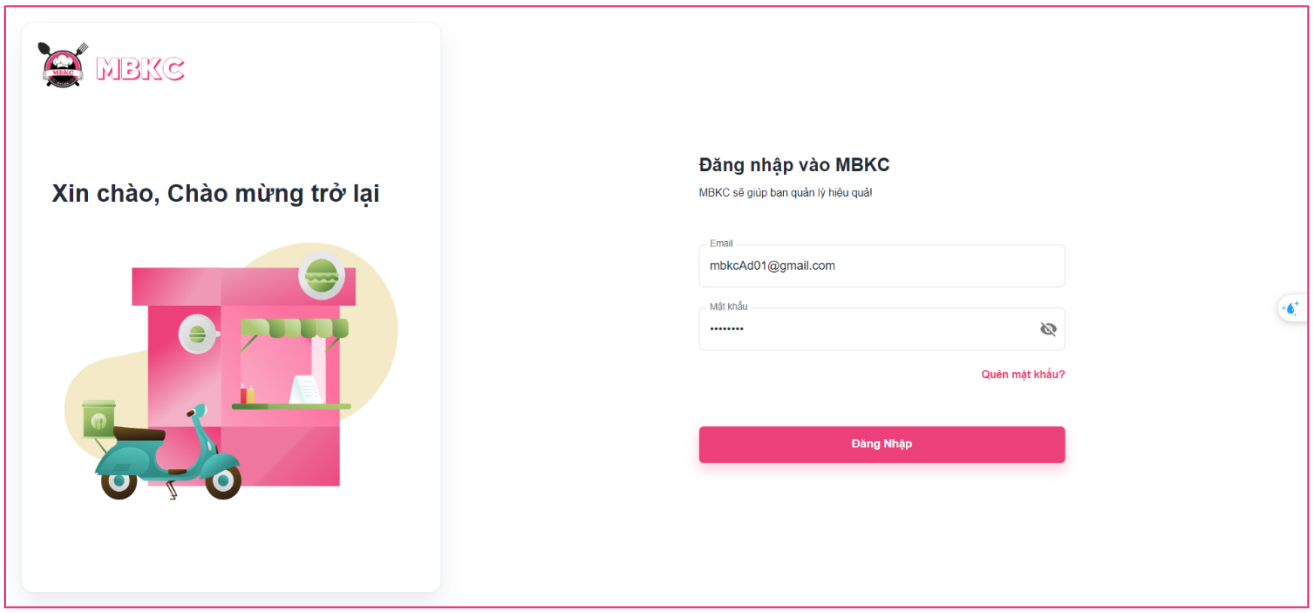


Figure 241: User Manual - View list Partner – Step 1

Step 2: Click “Đăng Nhập” button.

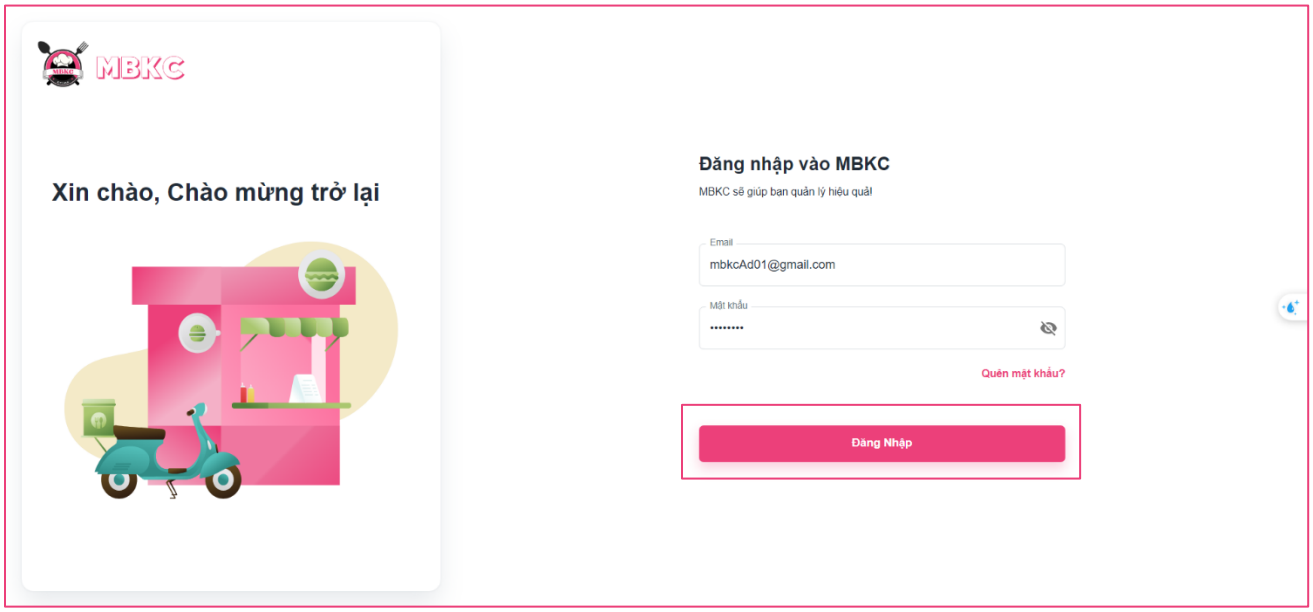


Figure 242: User Manual - View list Partner – Step 2

Step 3: From Admin Dashboard Page, choose “Đối Tác” from the left sidebar.

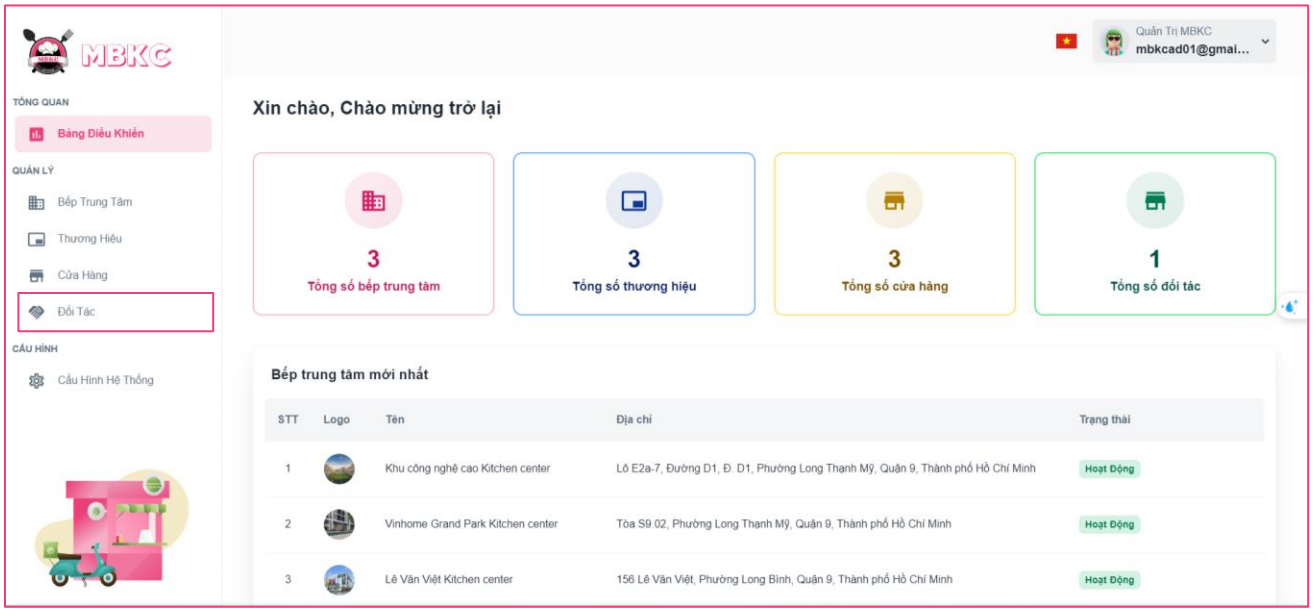


Figure 243: User Manual - View list Partner – Step 3

Step 4: Search, reload data, and show or hide columns in the table list of partners.

Case 1: Fill in the partner’s name in the search box to search partners in the table list.

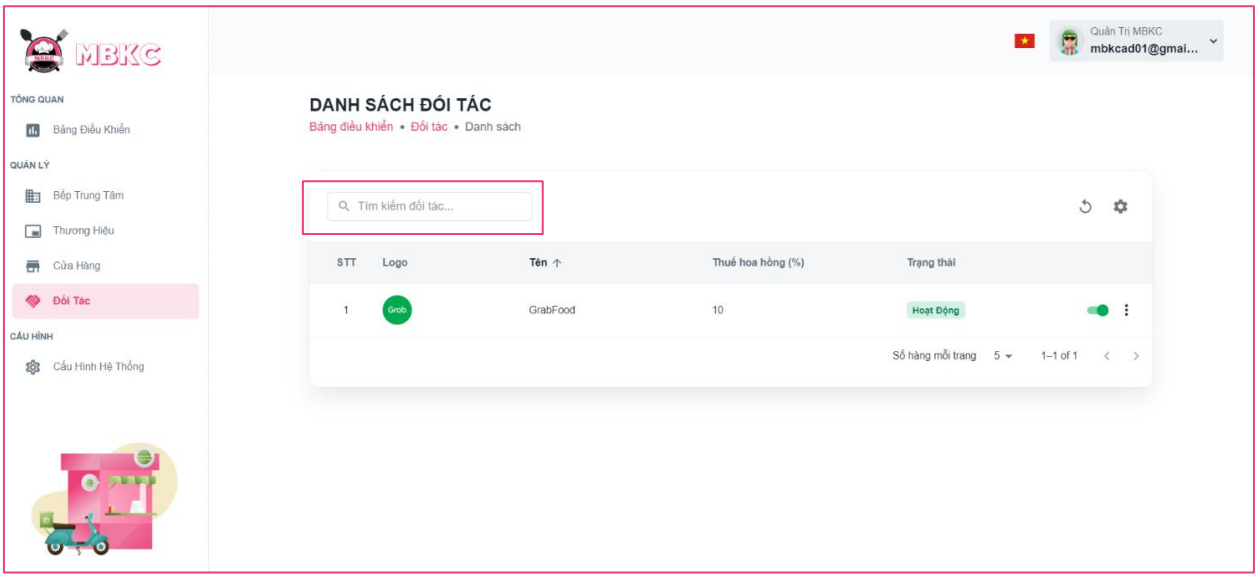


Figure 244: User Manual - View list Partner – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of partners.

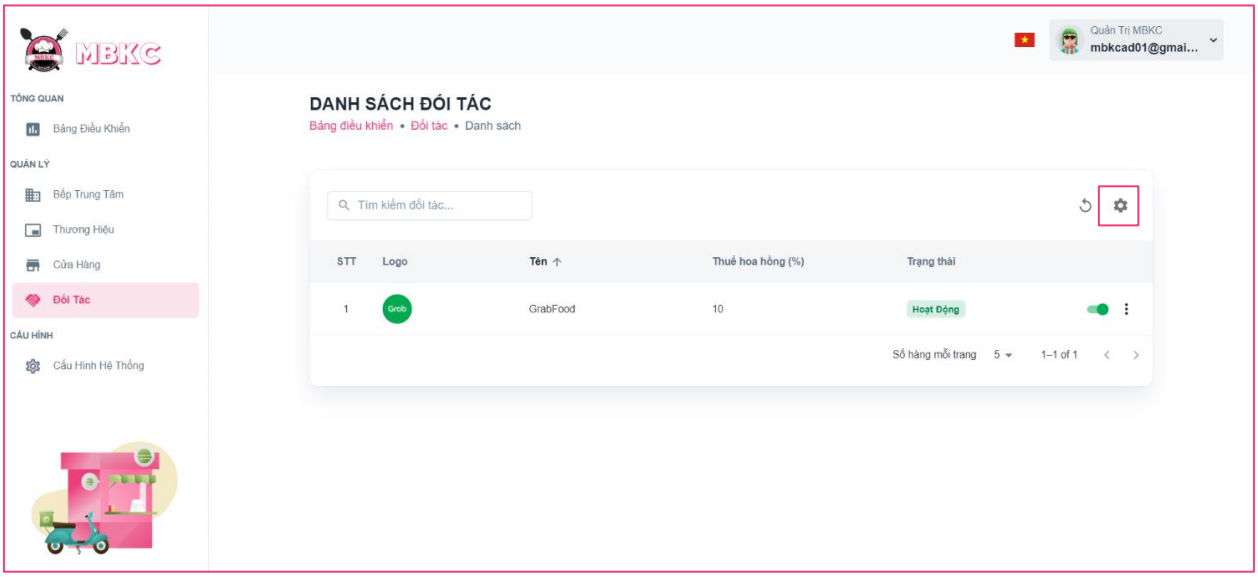


Figure 245: User Manual - View list Partner – Step 4 – Case 2

Case 3: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of the partner.

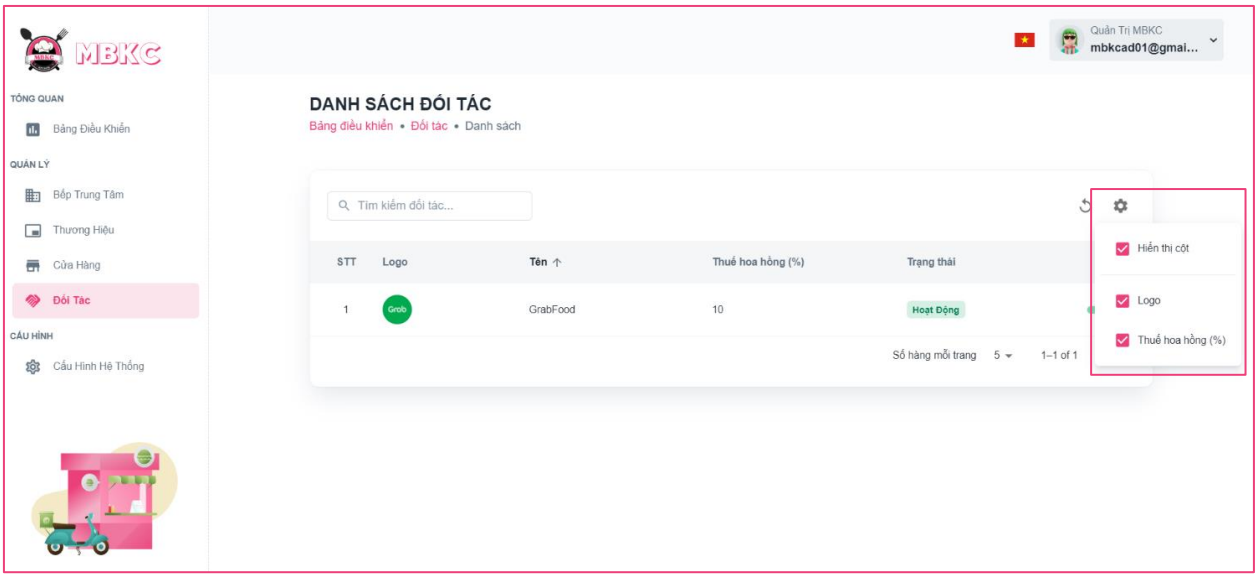


Figure 246: User Manual - View list Partner – Step 4 – Case 3

3.2.5.2. View details Partner

3.2.5.2.1. Description

Use this guideline to view details partner.

This web application supports view details partner with account as a MBKC Admin.

3.2.5.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

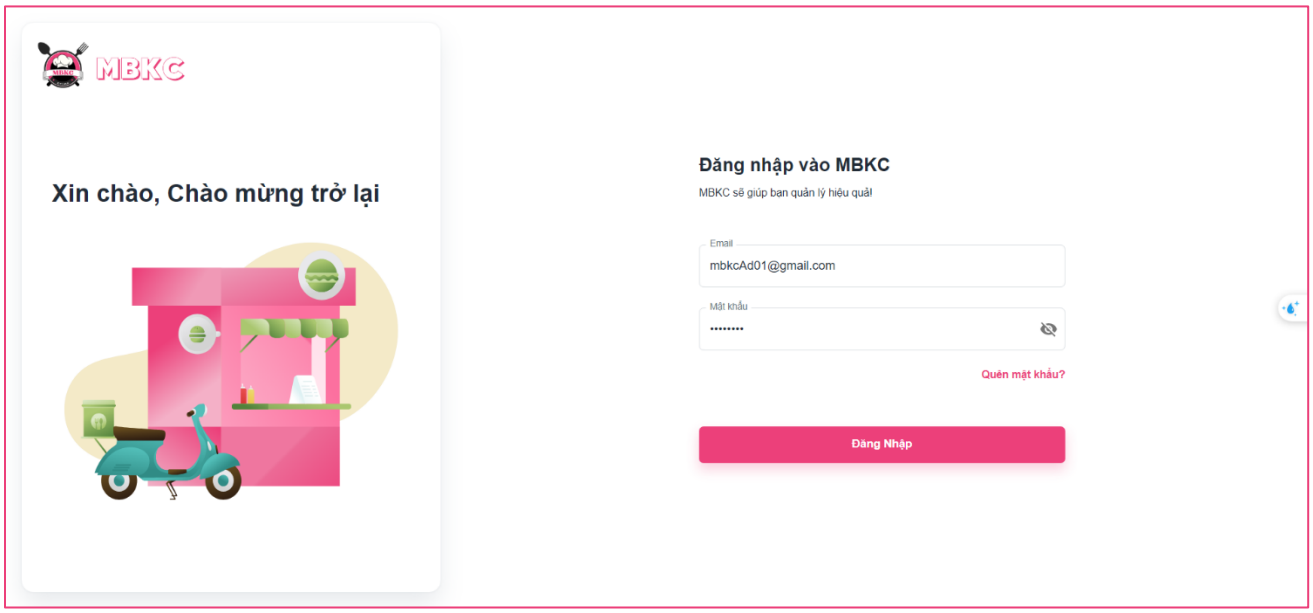


Figure 247: User Manual - View details Partner – Step 1

Step 2: Click “Đăng Nhập” button.

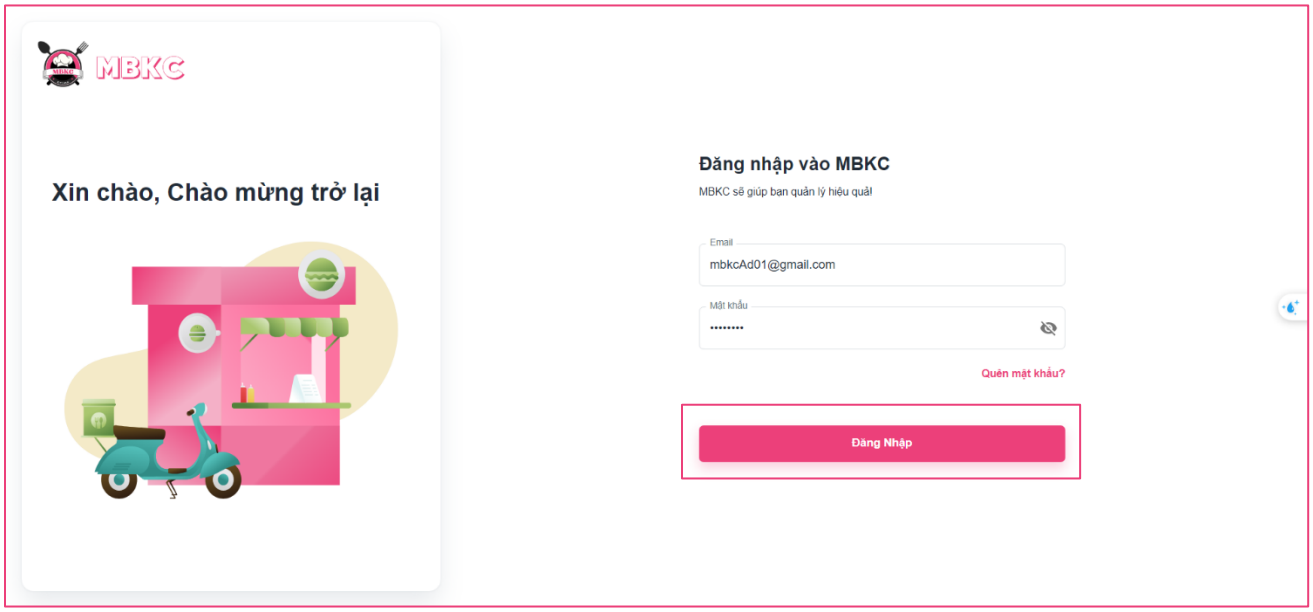


Figure 248: User Manual - View details Partner – Step 2

Step 3: From Admin Dashboard Page, choose “Đối Tác” from the left sidebar.

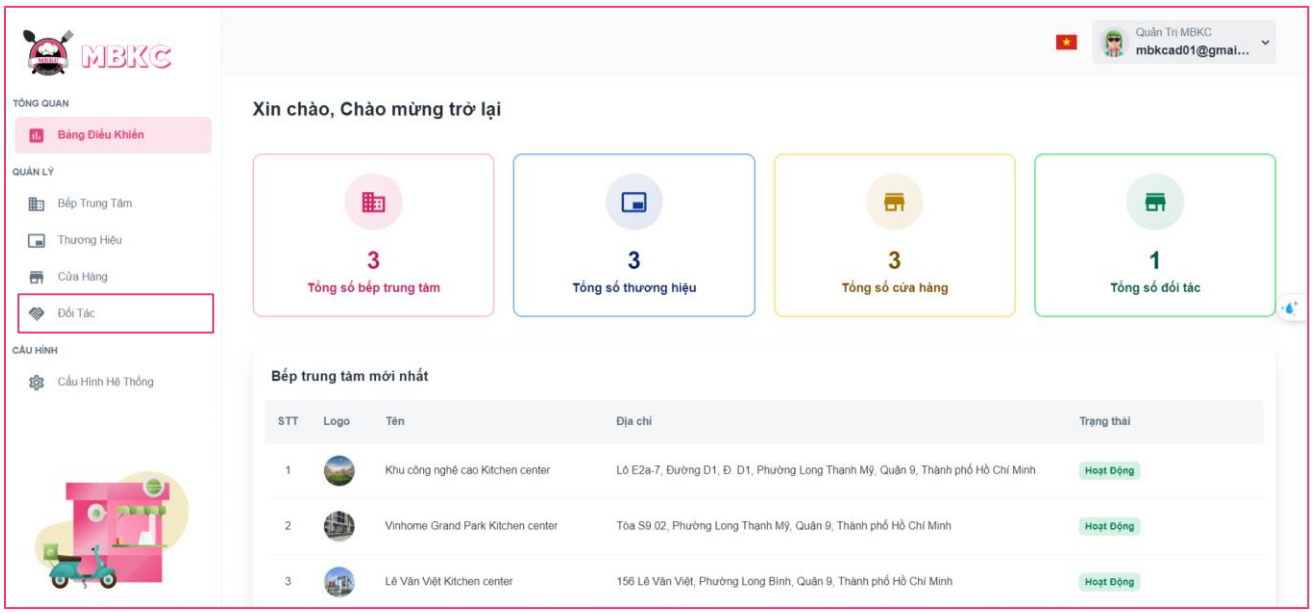


Figure 249: User Manual - View details Partner – Step 3

Step 4: Click the partner row to pop up the Partner Details Modal.

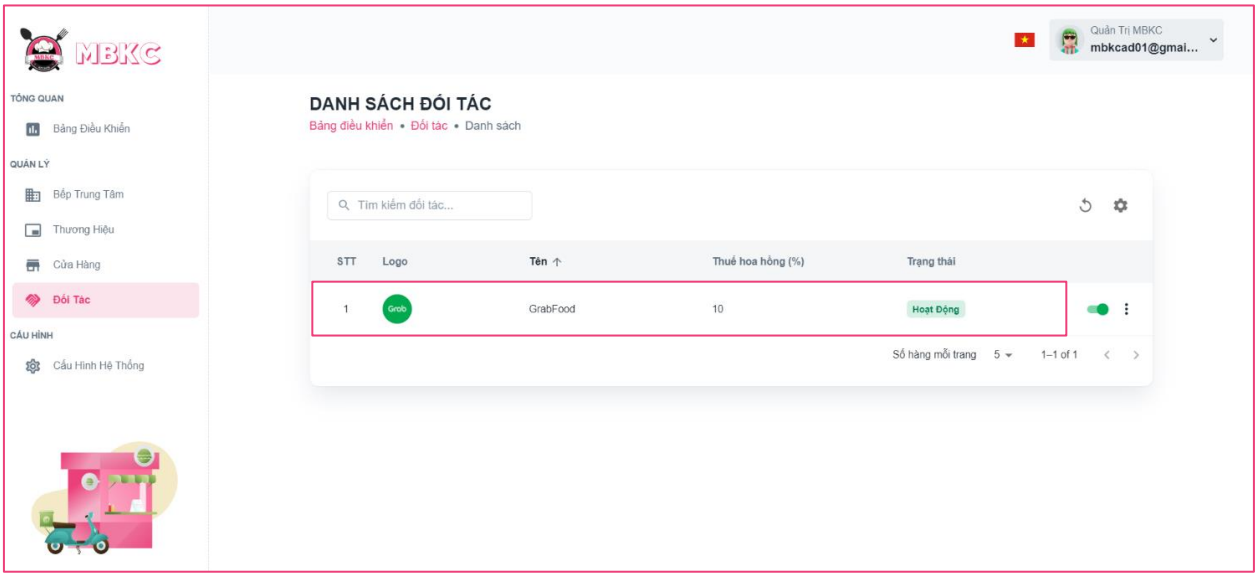


Figure 250: User Manual - View details Partner – Step 4

Step 5: Pop up the Partner Details Modal to show information details of partner.

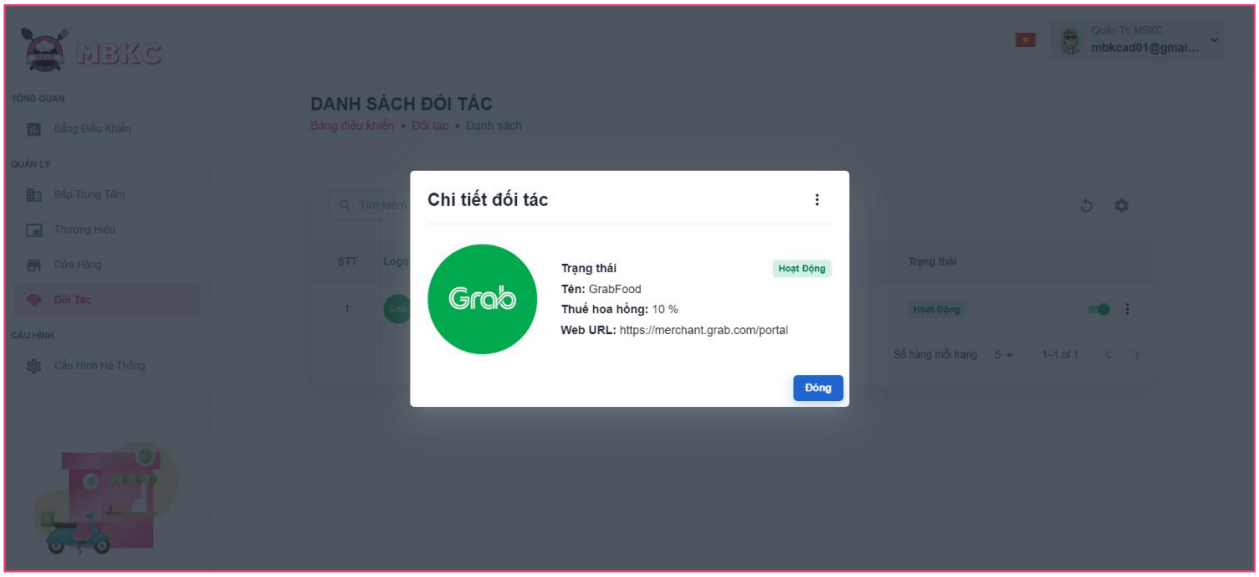


Figure 251: User Manual - View details Partner – Step 5

3.2.5.3. Update Status Partner

3.2.5.3.1. Description

Use this guideline to update the status of the partner.

This web application supports update the status of the partner with account as a MBKC Admin.

3.2.5.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

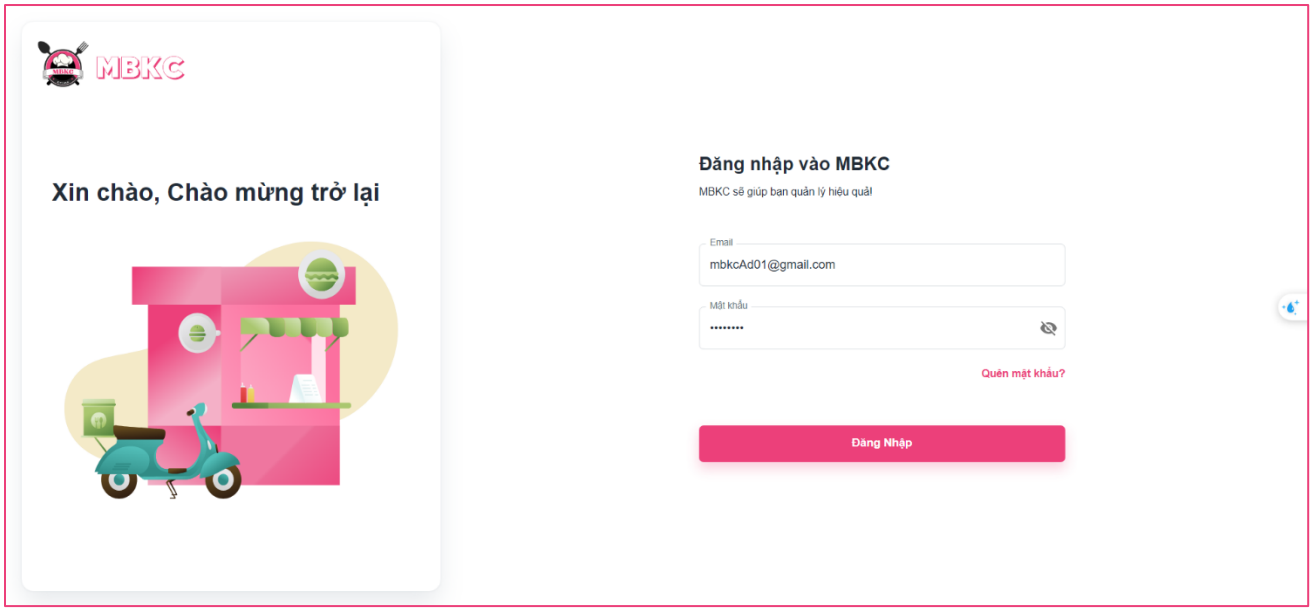


Figure 252: User Manual – Update Status Partner – Step 1

Step 2: Click “Đăng Nhập” button.

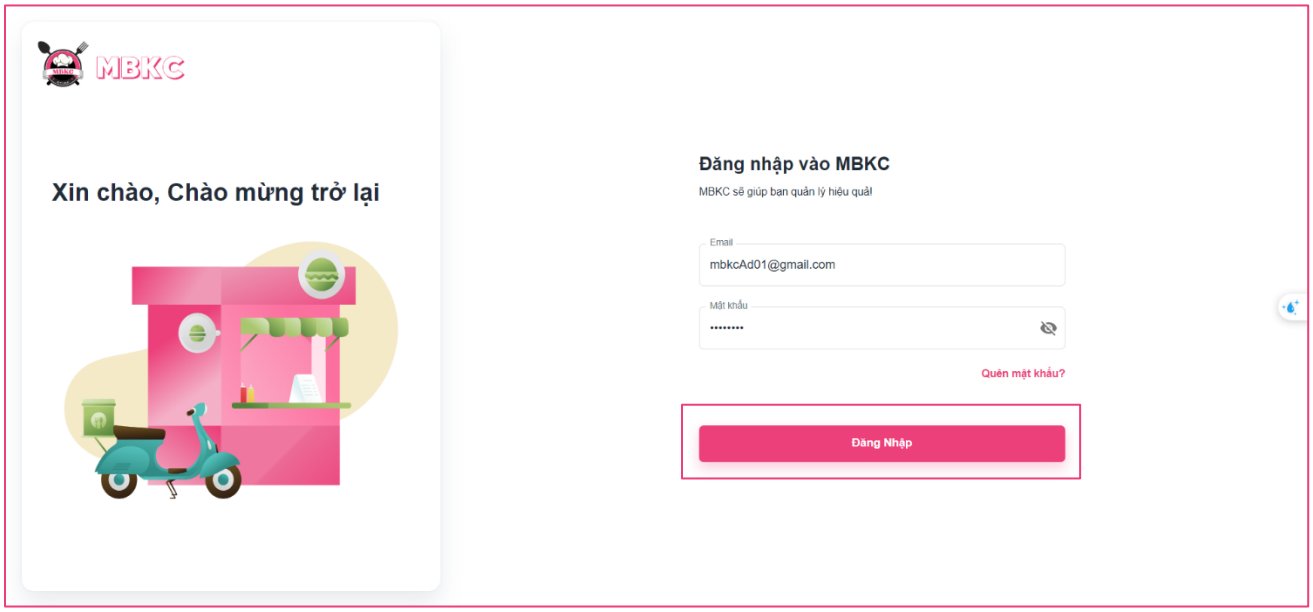


Figure 253: User Manual - Update Status Partner – Step 2

Step 3: From Admin Dashboard Page, choose “Đối Tác” from the left sidebar.

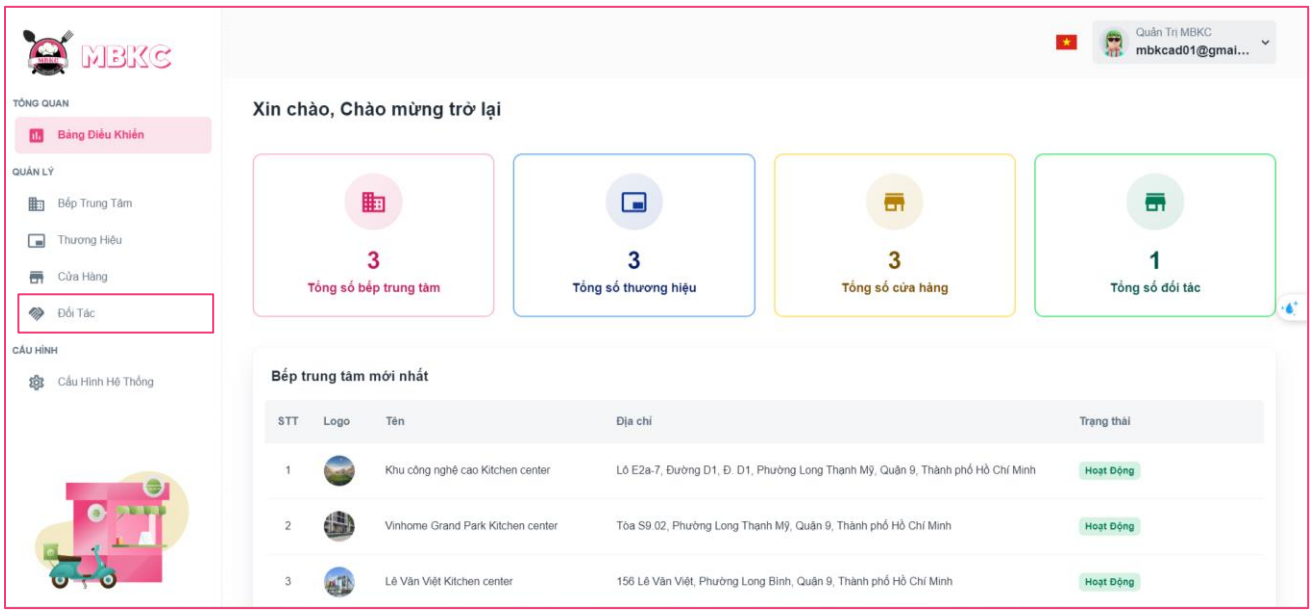


Figure 254: User Manual - Update Status Partner – Step 3

Step 4: Click on the switch button on the partner row to update the status.

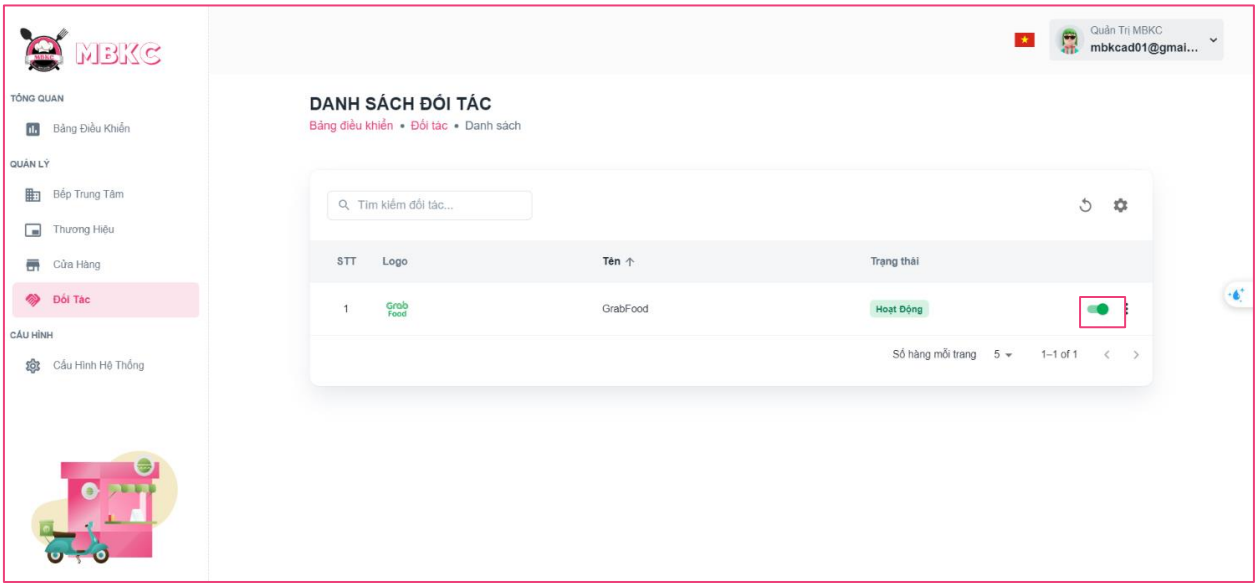


Figure 255: User Manual - Update Status Partner – Step 4

List of partners will be reloaded and a toast message "Đã cập nhật trạng thái đối tác thành công". And the status of the partner has been updated.

3.2.5.4. Update Information Partner

3.2.5.4.1. Description

Use this guideline to update information of the partner.

This web application supports update information of the partner with account as a MBKC Admin.

3.2.5.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

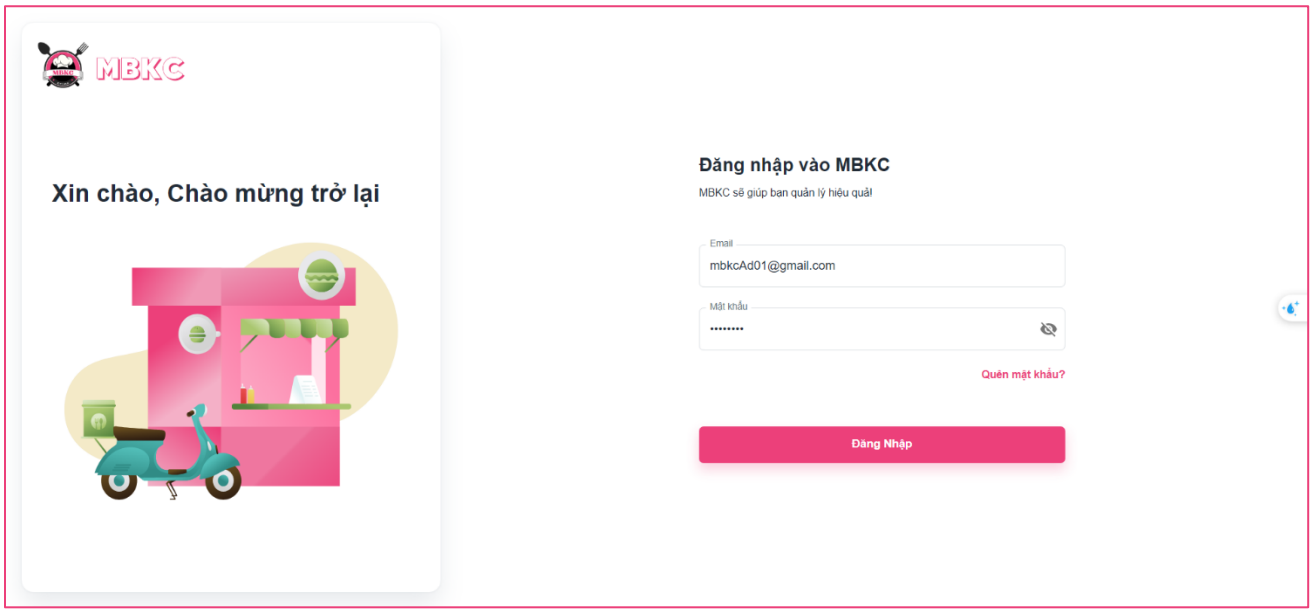


Figure 256: User Manual – Update Information Partner – Step 1

Step 2: Click “Đăng Nhập” button.

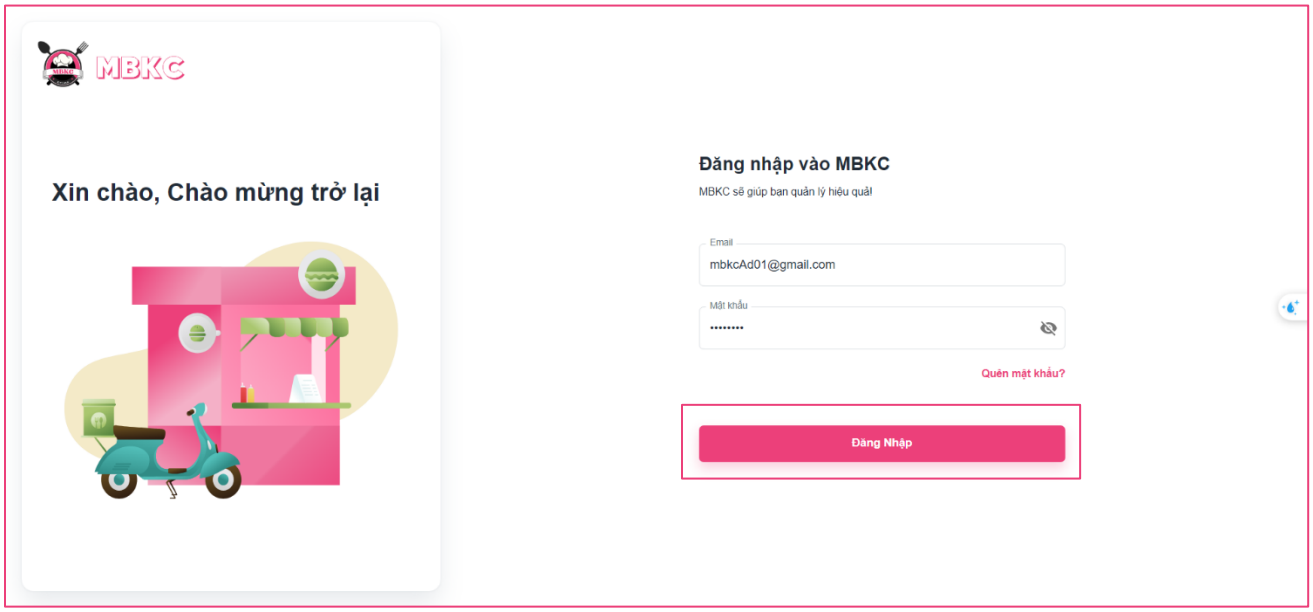


Figure 257: User Manual - Update Information Partner – Step 2

Step 3: From Admin Dashboard Page, choose “Đối Tác” from the left sidebar.

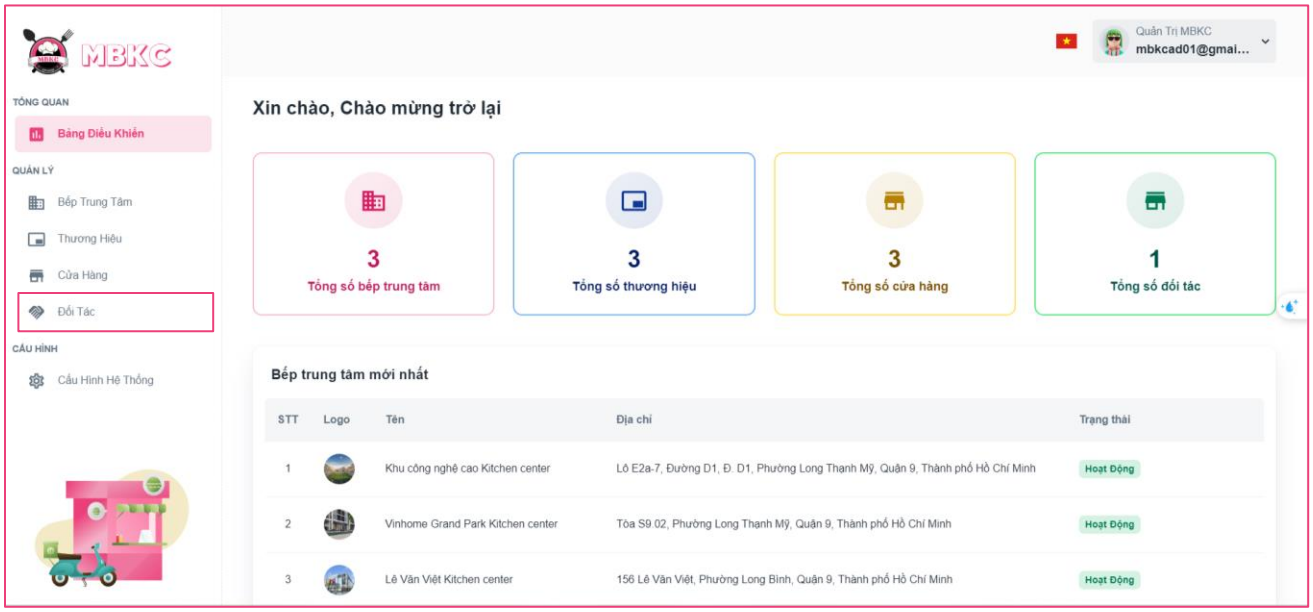


Figure 258: User Manual - Update Information Partner – Step 3

Step 4: Update information of the partner.

Case 1: Update information of the partner in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the partner row want to pop up Update Information Partner Modal to update information.

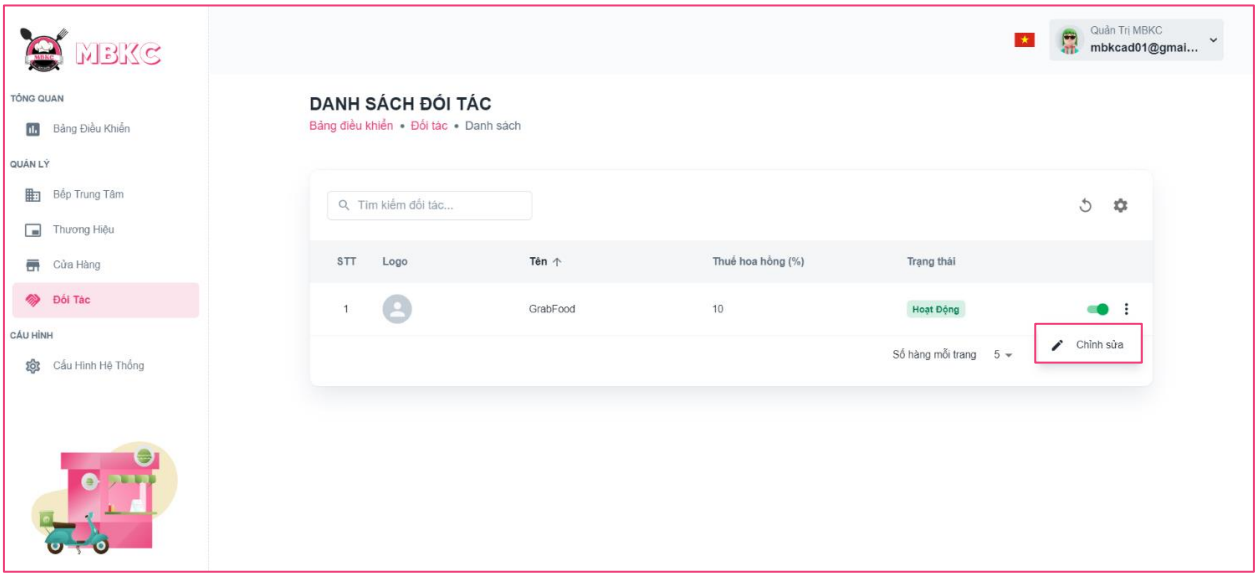


Figure 259: User Manual - Update Information Store – Step 4 – Case 1

Case 2: Update information of the partner on the detail modal.

1: Click the partner row to pop up Partner Details Modal.

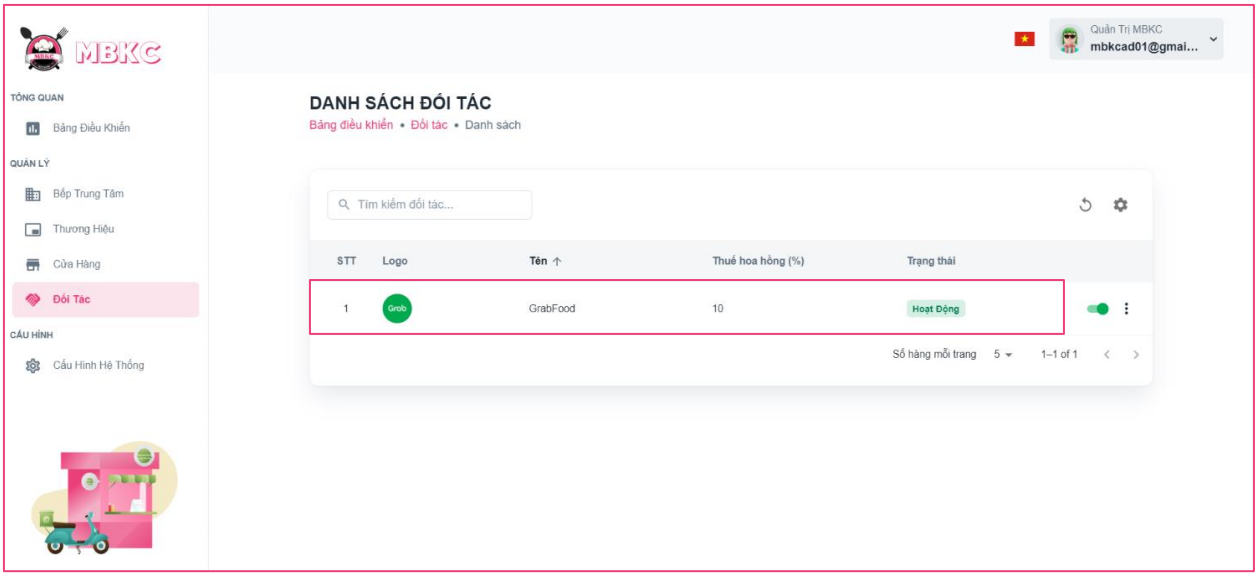


Figure 260: User Manual - Update Information Partner – Step 4 – Case 2 – 1

2: Click on the Menu icon button. Then click "Chỉnh sửa" menu item on the Partner Details Modal to update information.

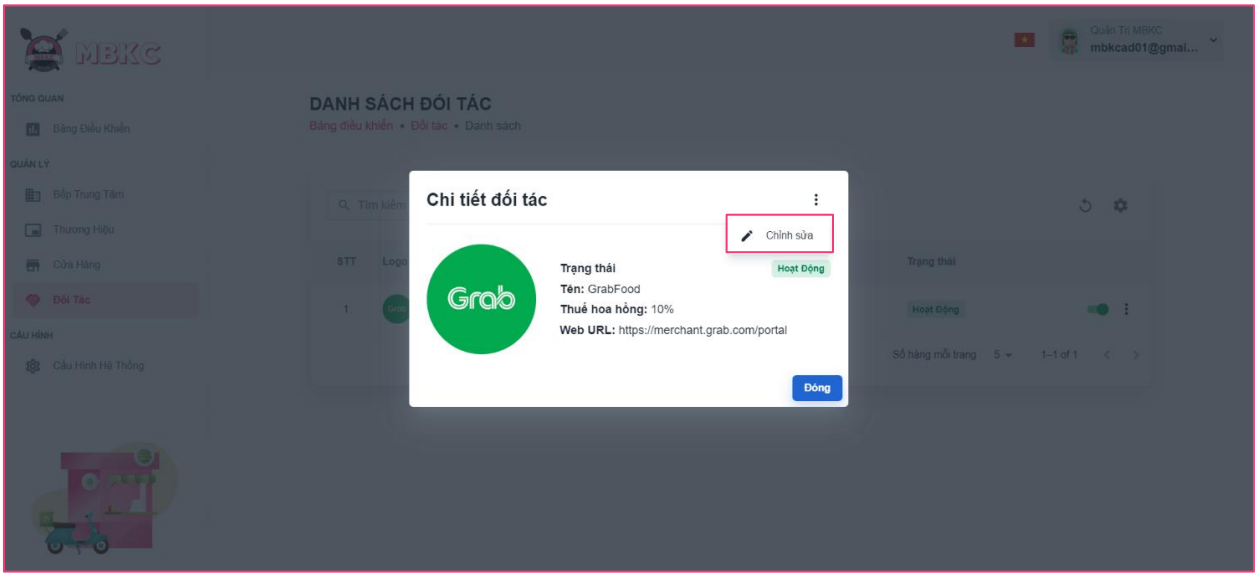


Figure 261: User Manual - Update Information Partner – Step 4 – Case 2 - 2

Step 5: Update Partner Information Modal will be showed, and partner information will be automatically filled in the form. Then edit the information as desired.

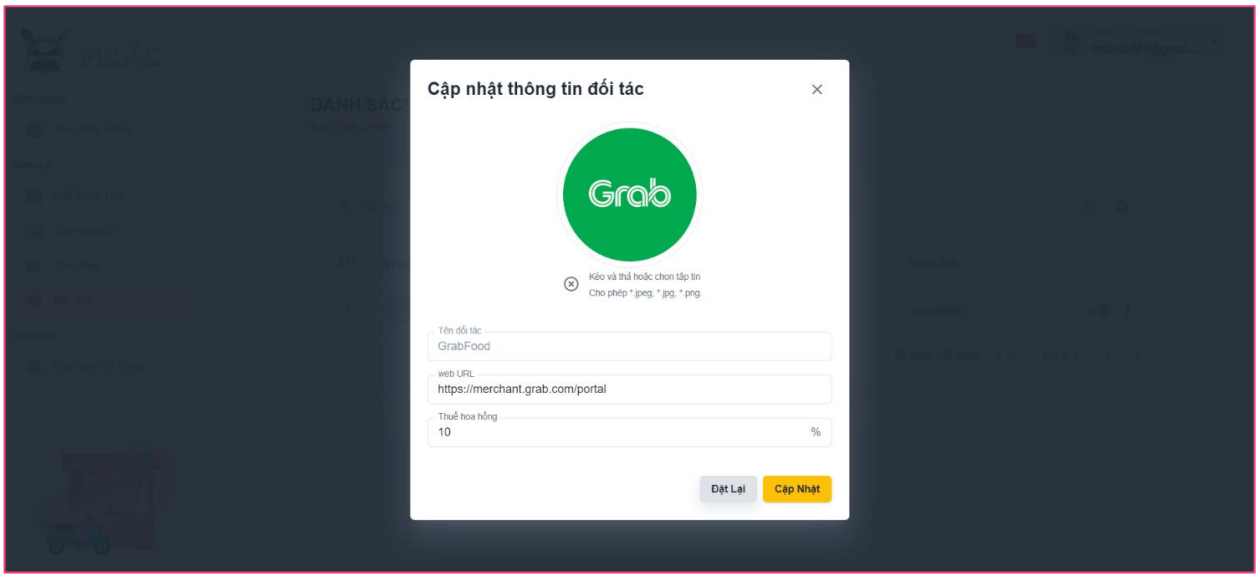


Figure 262: User Manual - Update Information Partner – Step 5

Step 6: Click "Cập nhật" button.

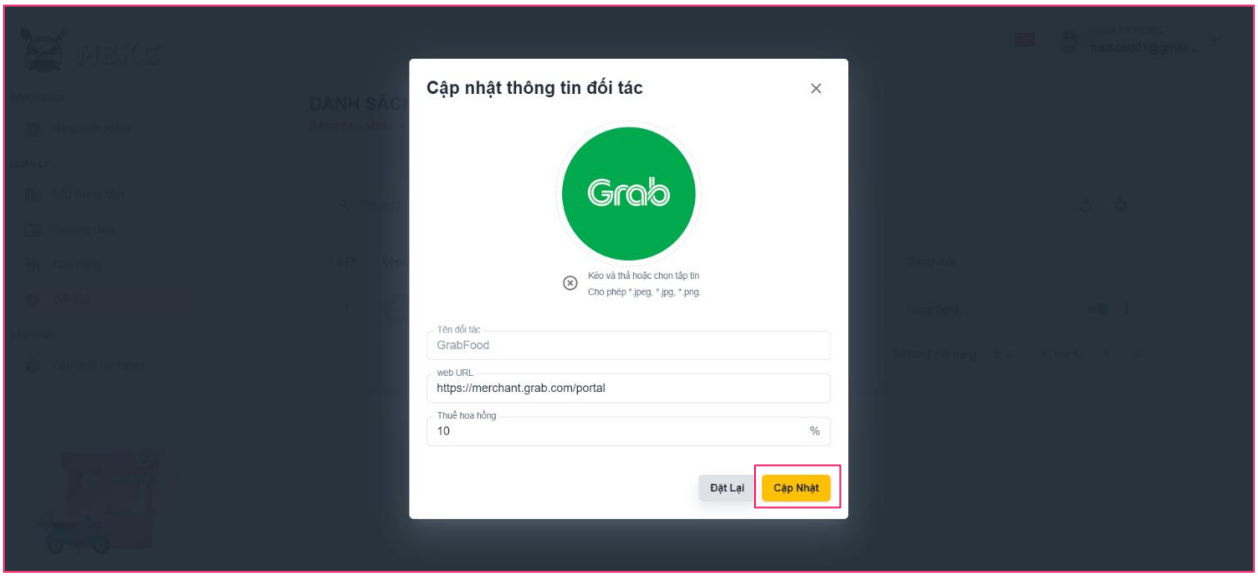


Figure 263: User Manual - Update Information Store – Step 6

Step 7: List of partners will be reloaded and a toast message “Cập nhật thông tin đối tác thành công”.

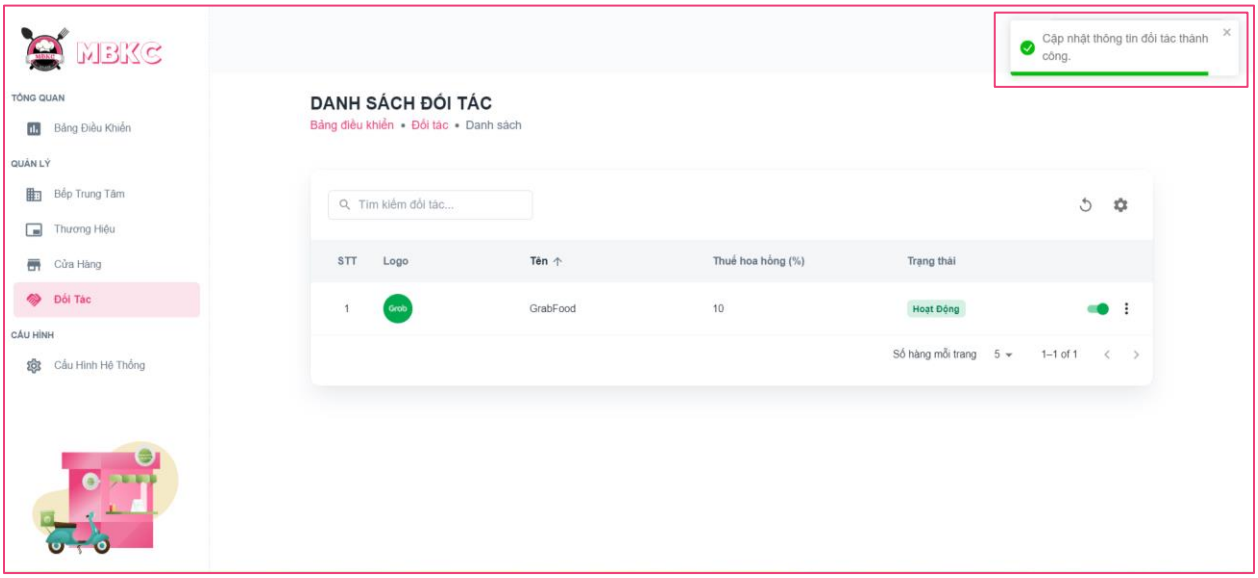


Figure 264: User Manual - Update Information Partner – Step 7

3.2.6. Normal Category management

3.2.6.1. Create new Normal Category

3.2.6.1.1. Description

Use this guideline to login into MBKC account to create a new normal category.

This web application supports create new a normal category with account as a Brand Manager.

3.2.6.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

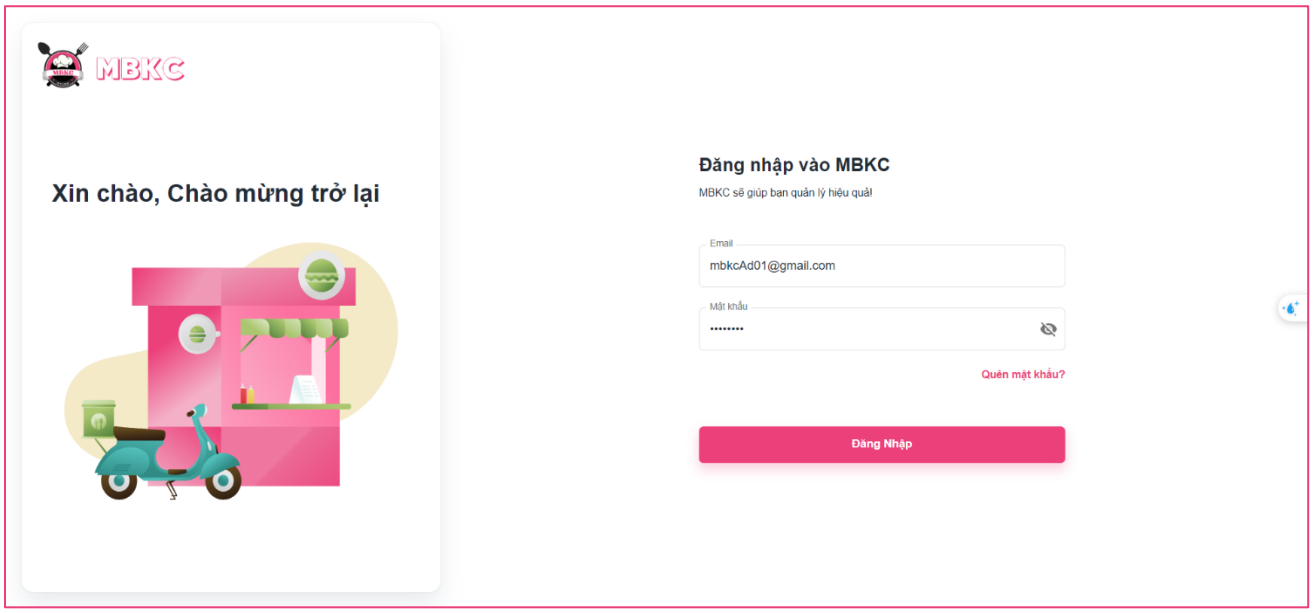


Figure 265: User Manual - Create new Normal Category – Step 1

Step 2: Click “Đăng Nhập” button.

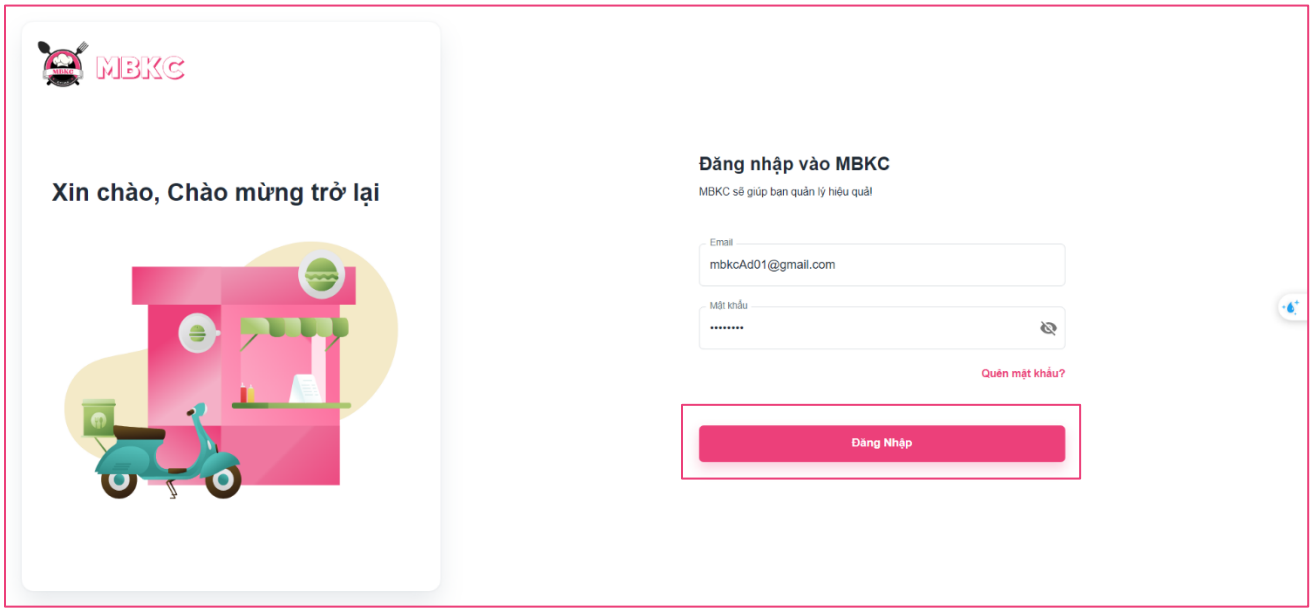


Figure 266: User Manual - Create new Normal Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Thường” from the left sidebar.

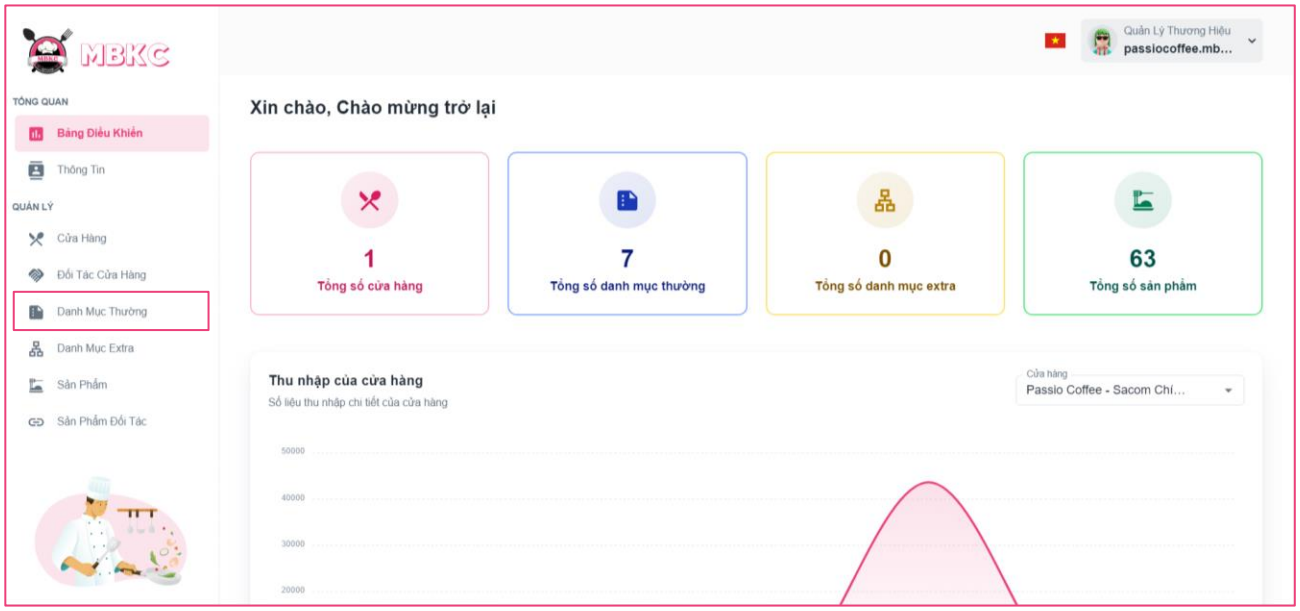


Figure 267: User Manual - Create new Normal Category - Step 3

Step 4: Click “Thêm Danh Mục Thường” button on the screen and navigate to Create New Normal Category Page to add a new normal category.

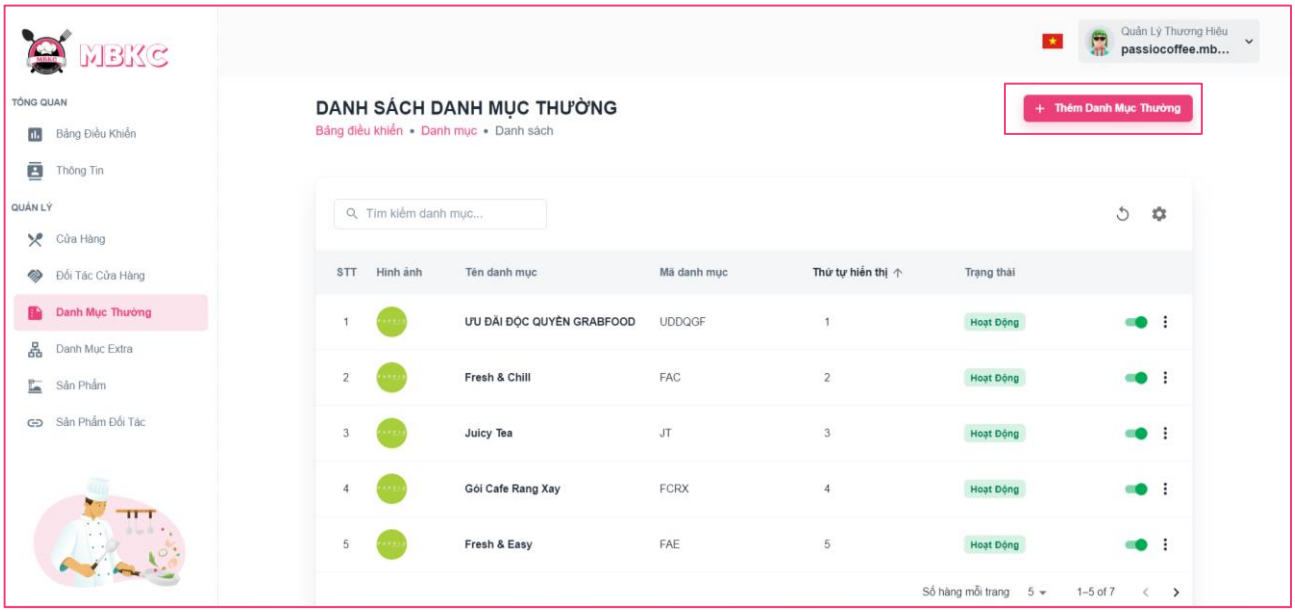


Figure 268: User Manual - Create new Normal Category - Step 4

Step 5: Filling all the required information to add a normal category including: “Logo”, “Tên danh mục", “Mã danh mục", “Thứ tự hiển thị", “Mô tả”.

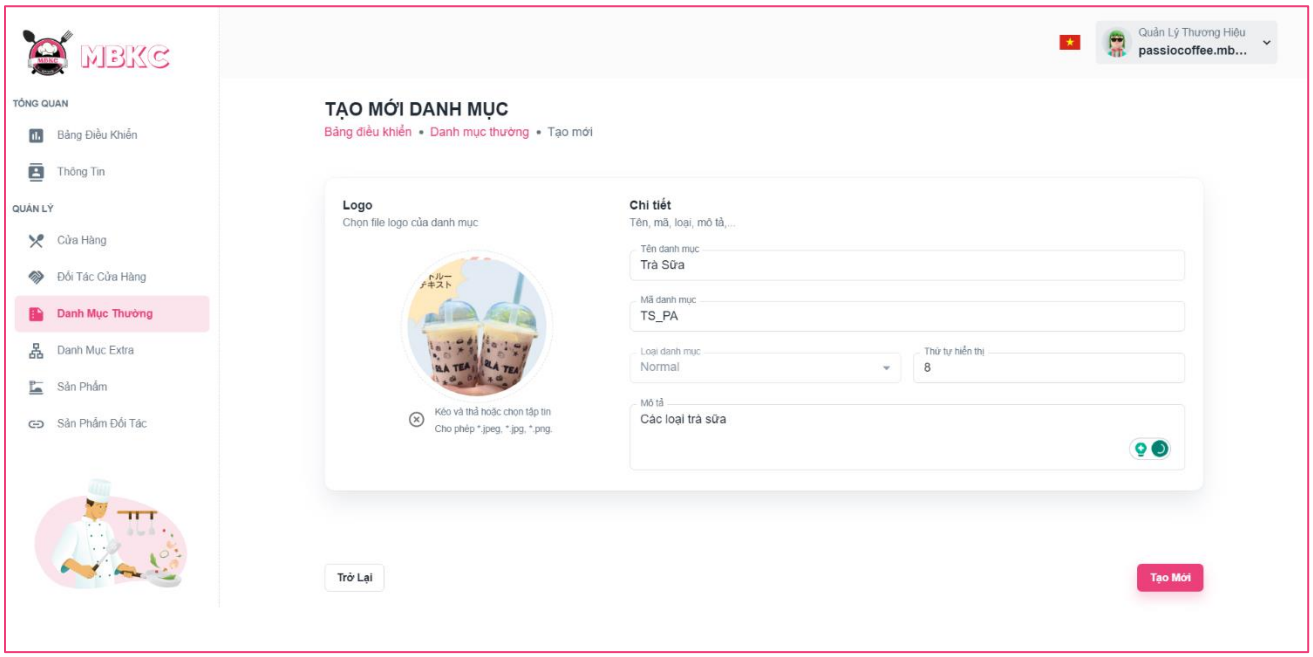


Figure 269: User Manual - Create new Normal Category - Step 5

Step 6: After filling in all required field, click “Tạo Mới” to finish creating a new normal category.

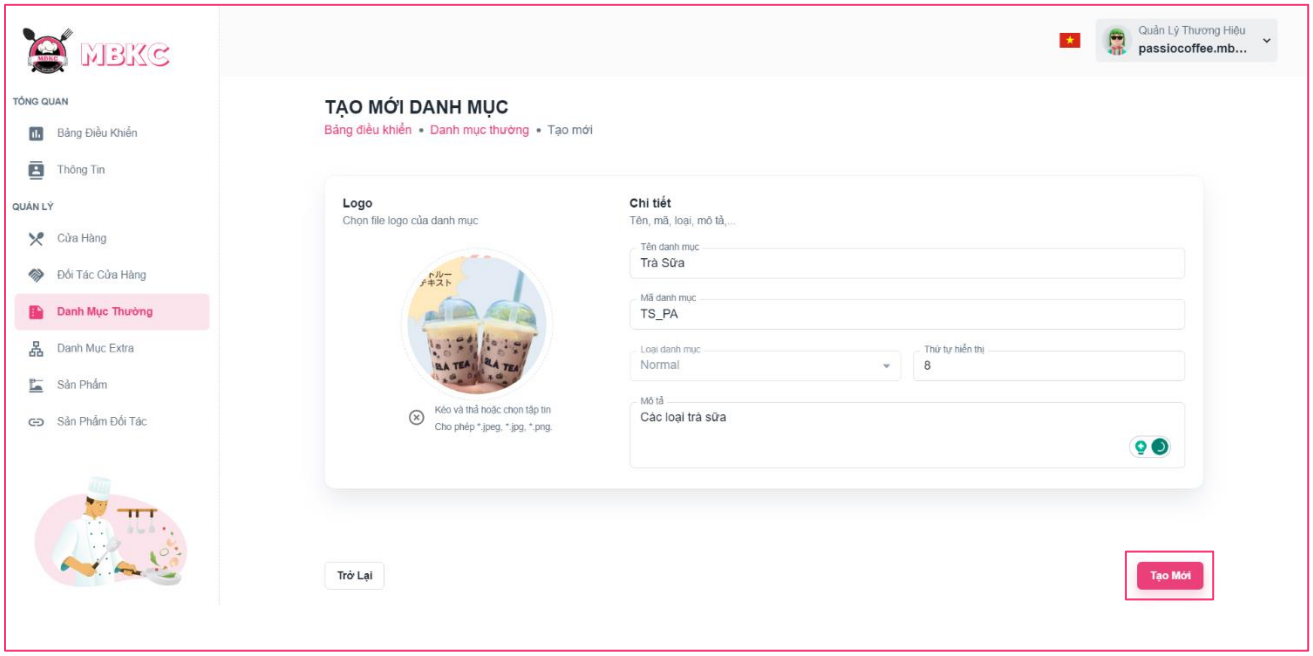


Figure 270: User Manual - Create new Normal Category – Step 6

Step 7: You will be navigated to List of Normal Categories Page showing the list of normal categories and a toast message “Đã tạo danh mục mới thành công”.

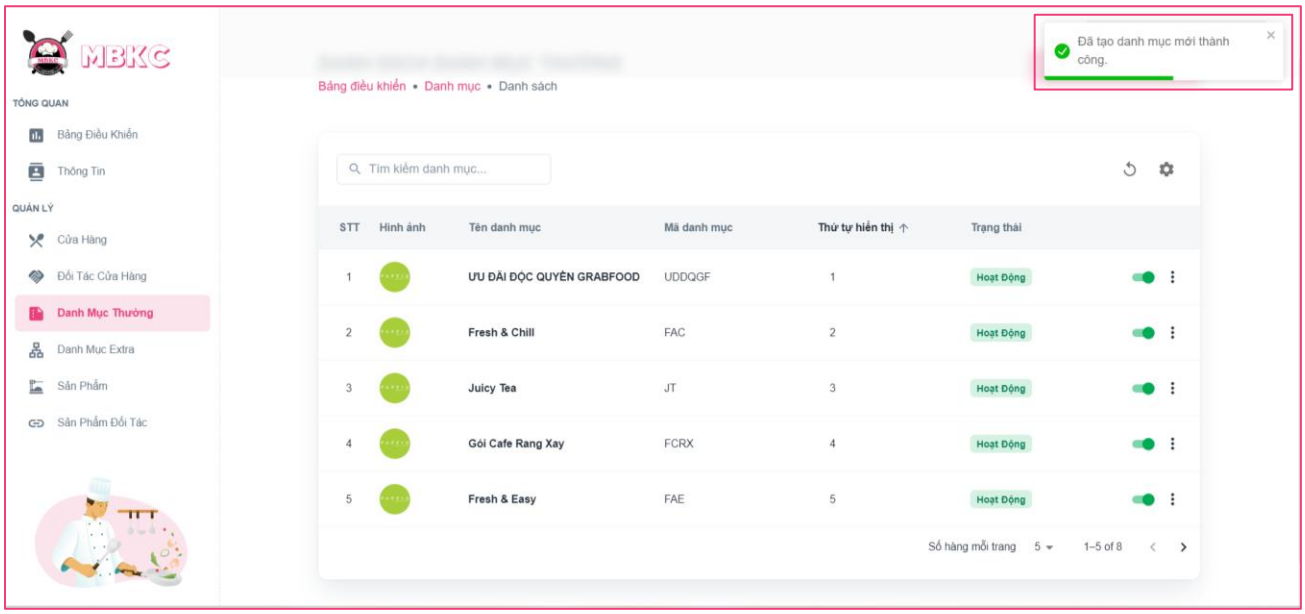


Figure 271: User Manual - Create new Normal Category - Step 7

3.2.6.2. View list of Normal Categories

3.2.6.2.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of normal categories.

This web application supports view list of normal categories with account as a Brand Manager.

3.2.6.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

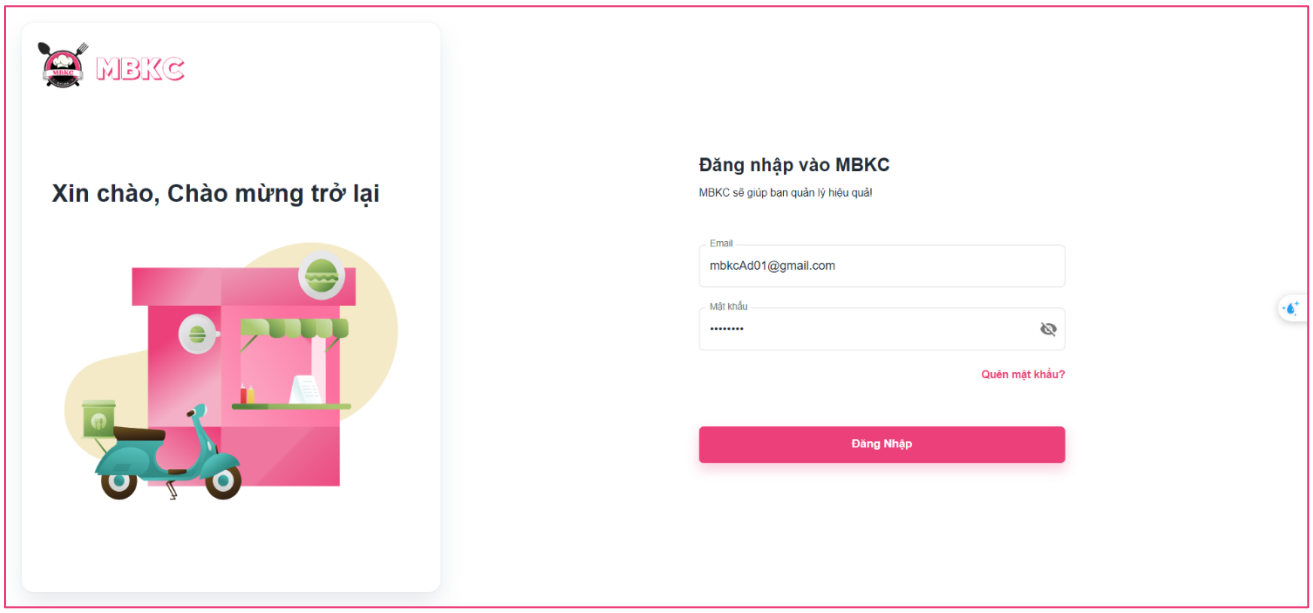


Figure 272: User Manual - View list of Normal Categories – Step 1

Step 2: Click “Đăng Nhập” button.

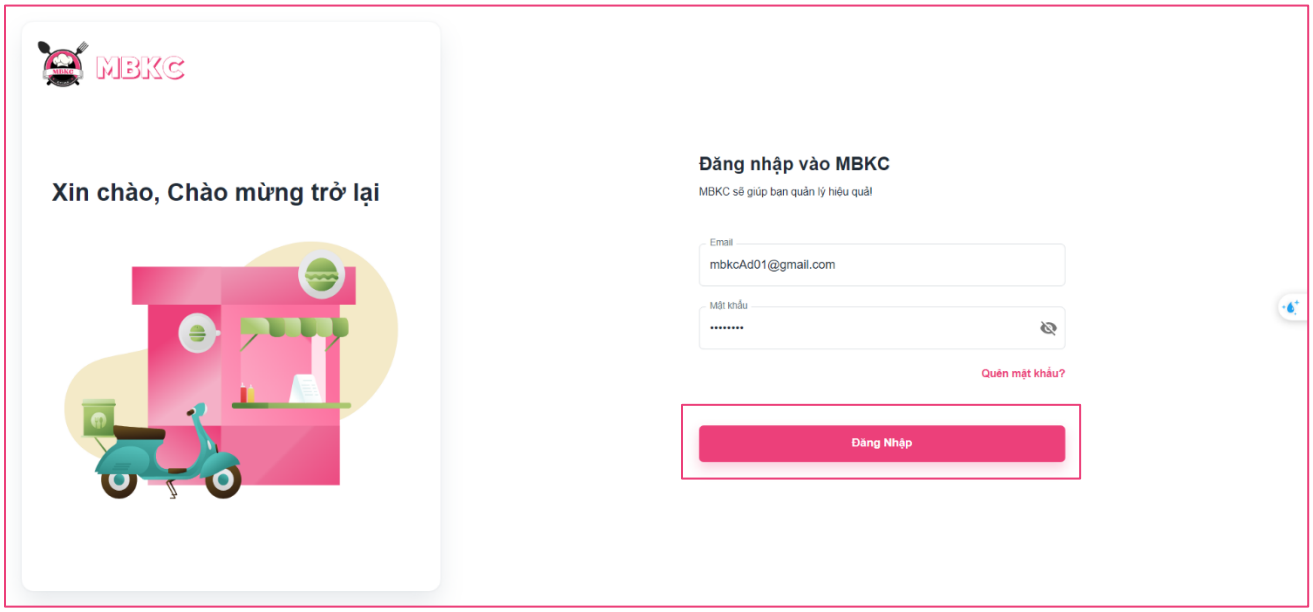


Figure 273: User Manual - View list of Normal Categories – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Thường” from the left sidebar.

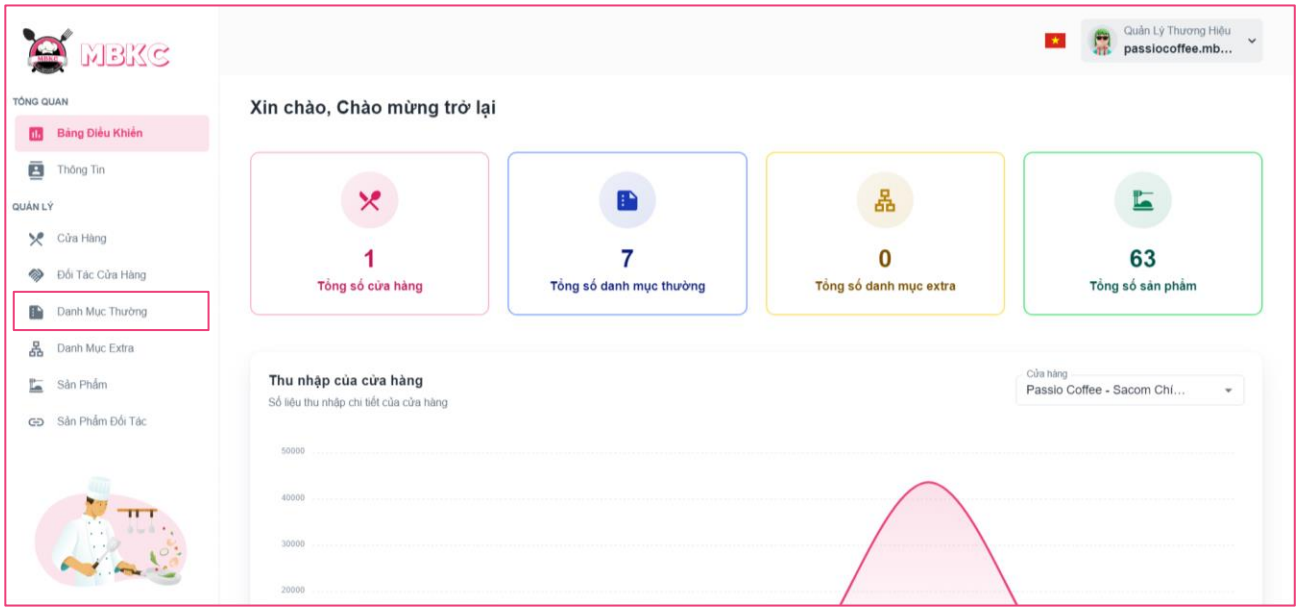


Figure 274: User Manual - View list of Normal Categories - Step 3

Step 4: Search, reload data, and show or hide columns in the table list of normal categories.

Case 1: Fill in the normal category’s name in the search box to search normal categories in the table list.

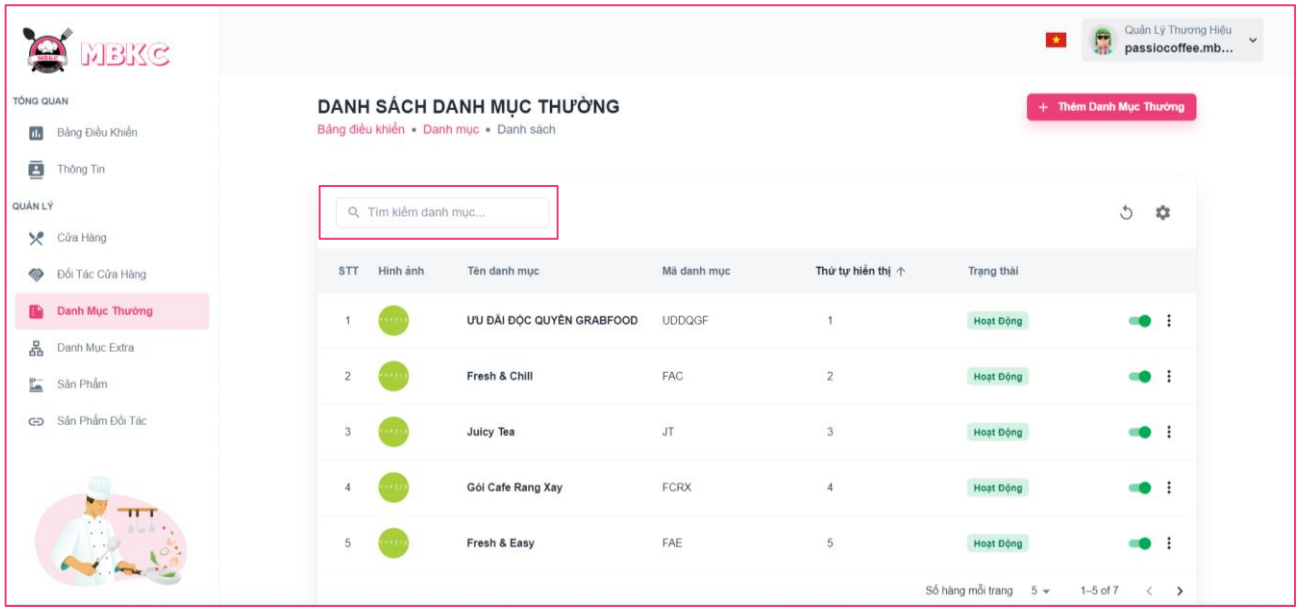


Figure 275: User Manual - View list of Normal Category – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of normal categories.

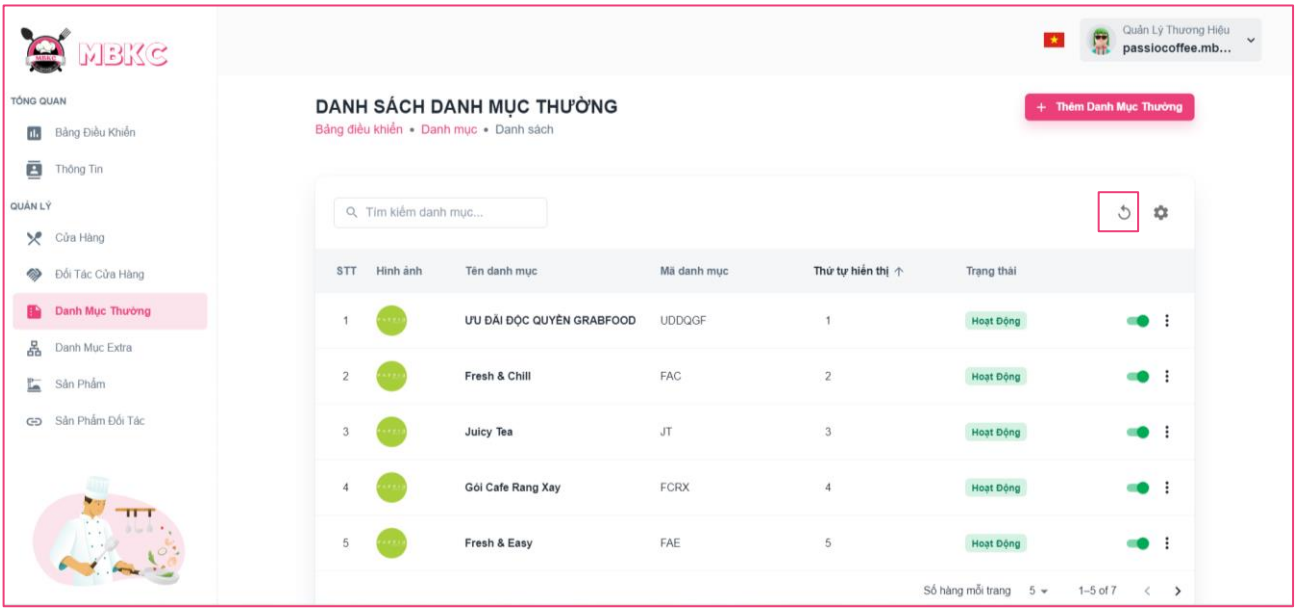


Figure 276: User Manual - View list of Normal Categories - Step 4 - Case 2

Case 3: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of normal categories.

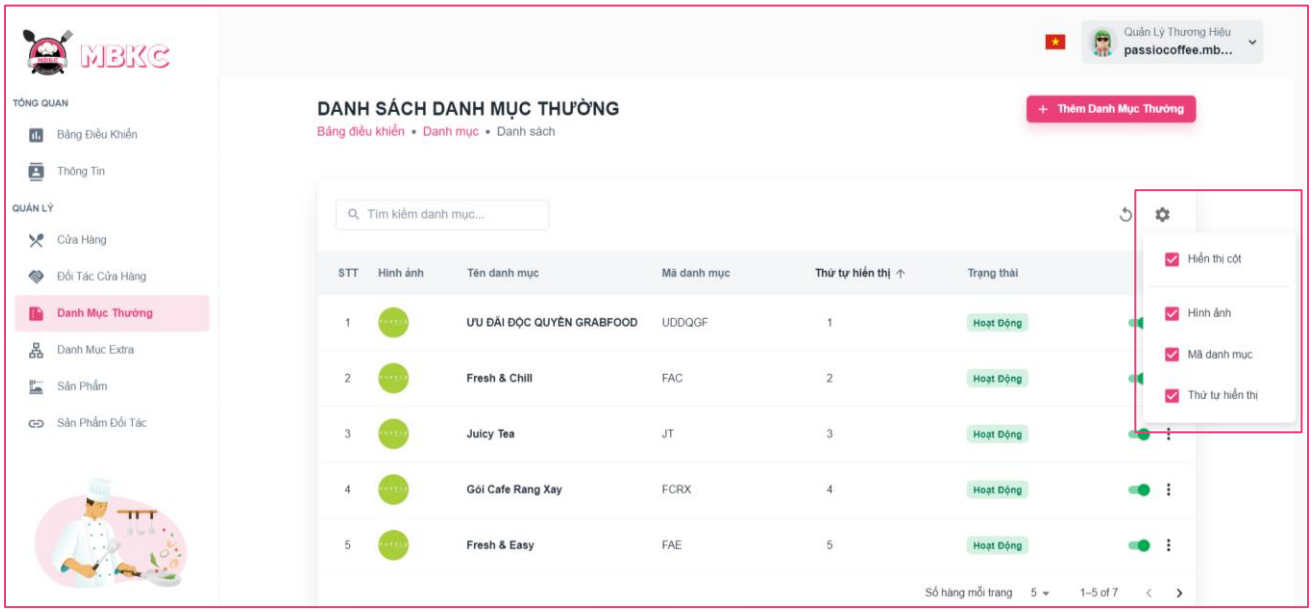


Figure 277: User Manual - View list of Normal Categories - Step 4 - Case 3

3.2.6.3. View details Normal Category

3.7.3.1. Description
Use this guideline to view details normal category.

This web application supports view details normal category with account as a Brand Manager.

3.2.6.3.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

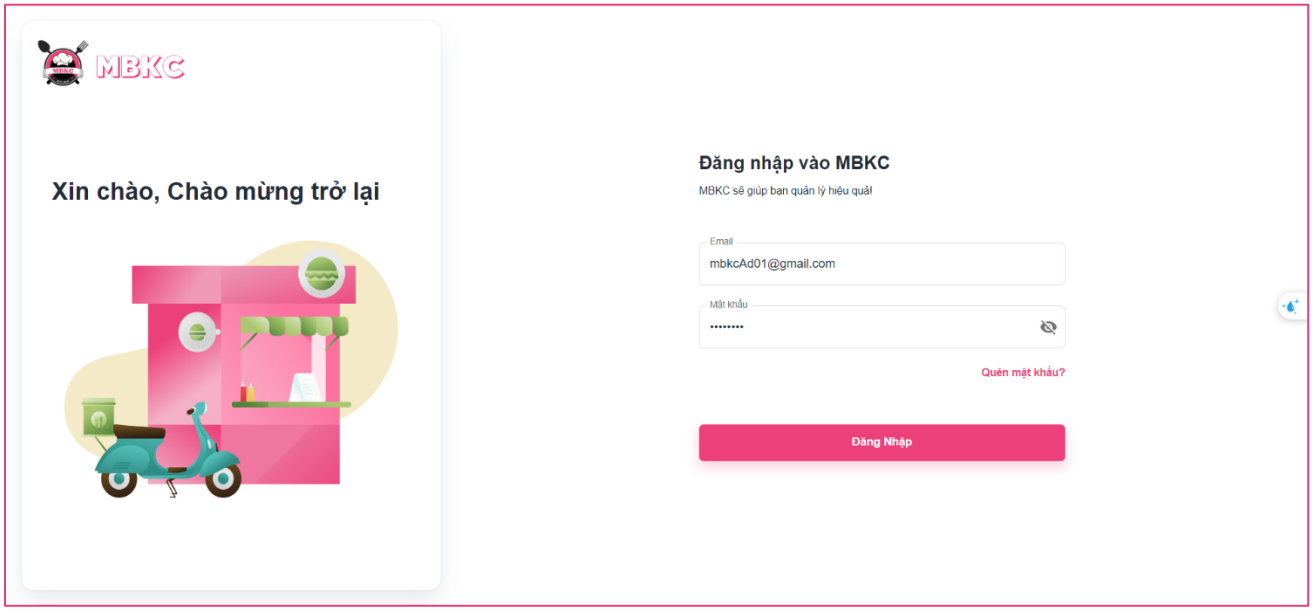


Figure 278: User Manual - View details Normal Categories - Step 1

Step 2: Click “Đăng Nhập” button.

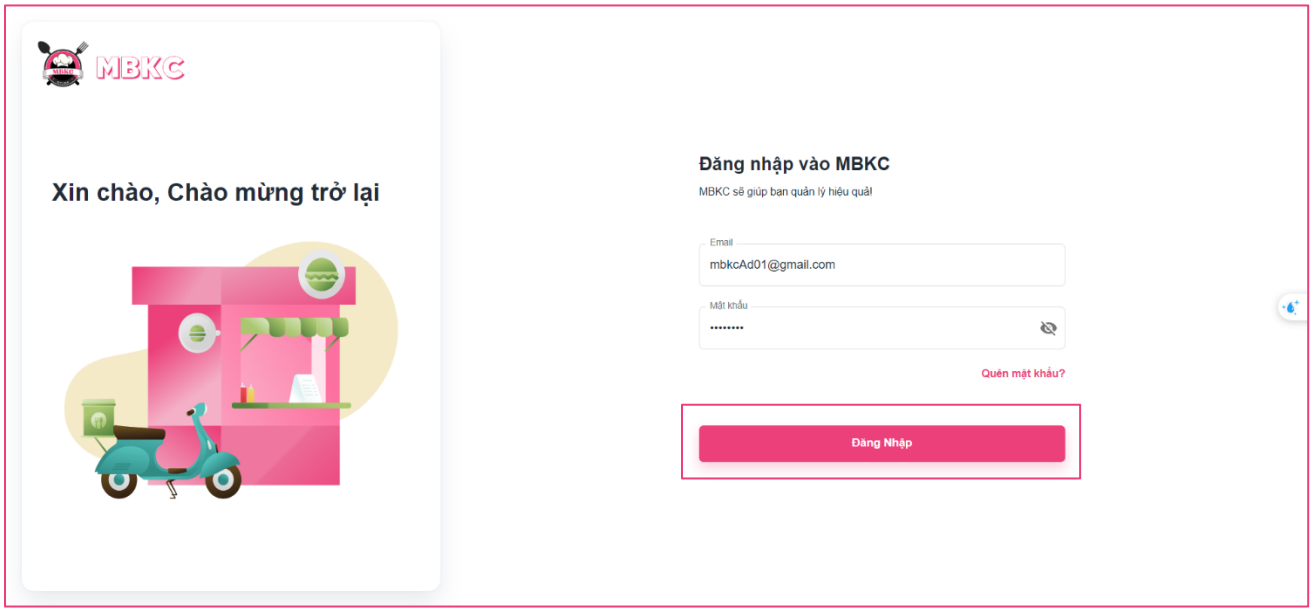


Figure 279: User Manual - View details Normal Categories - Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Thường” from the left sidebar.

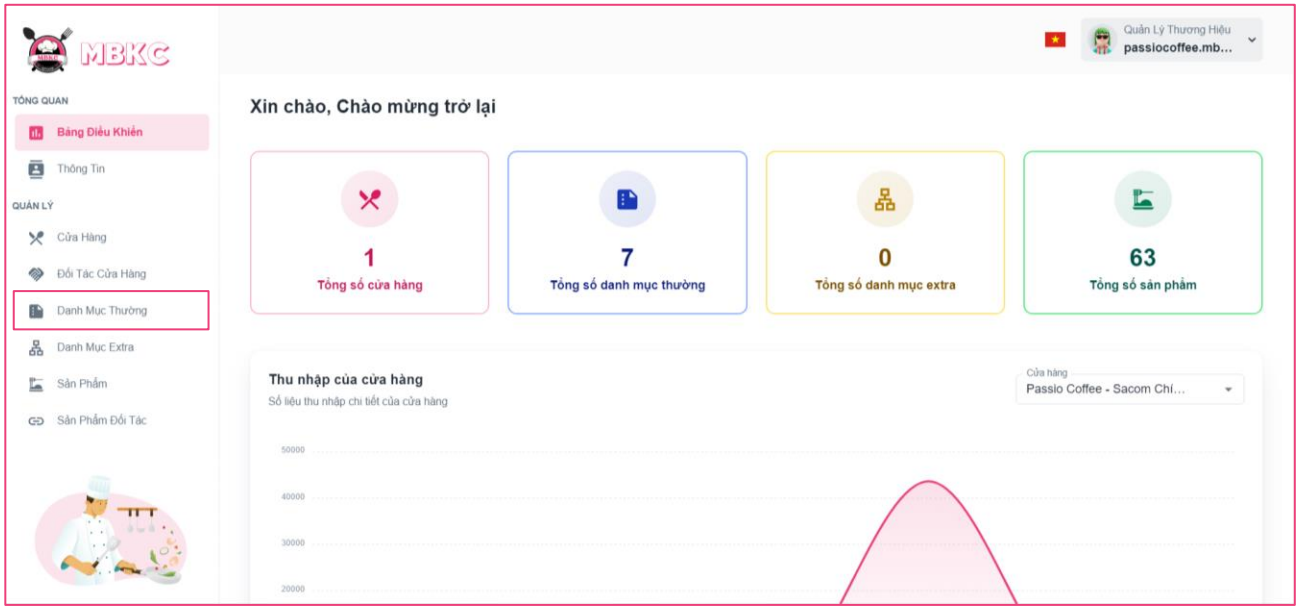


Figure 280: User Manual - View details Normal Categories - Step 3

Step 4: Click the normal category row to navigate to Normal Category Details Page.

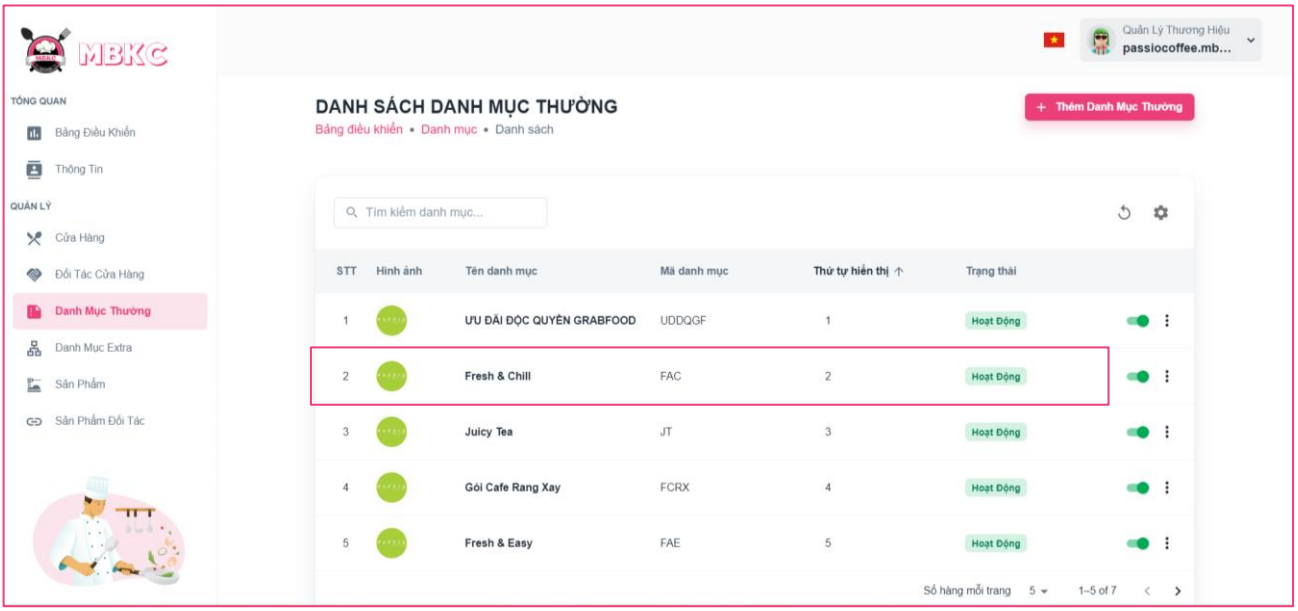


Figure 281: User Manual - View details Normal Categories - Step 4

Step 5: Navigate to Normal Category Details Page to show information details of normal category.

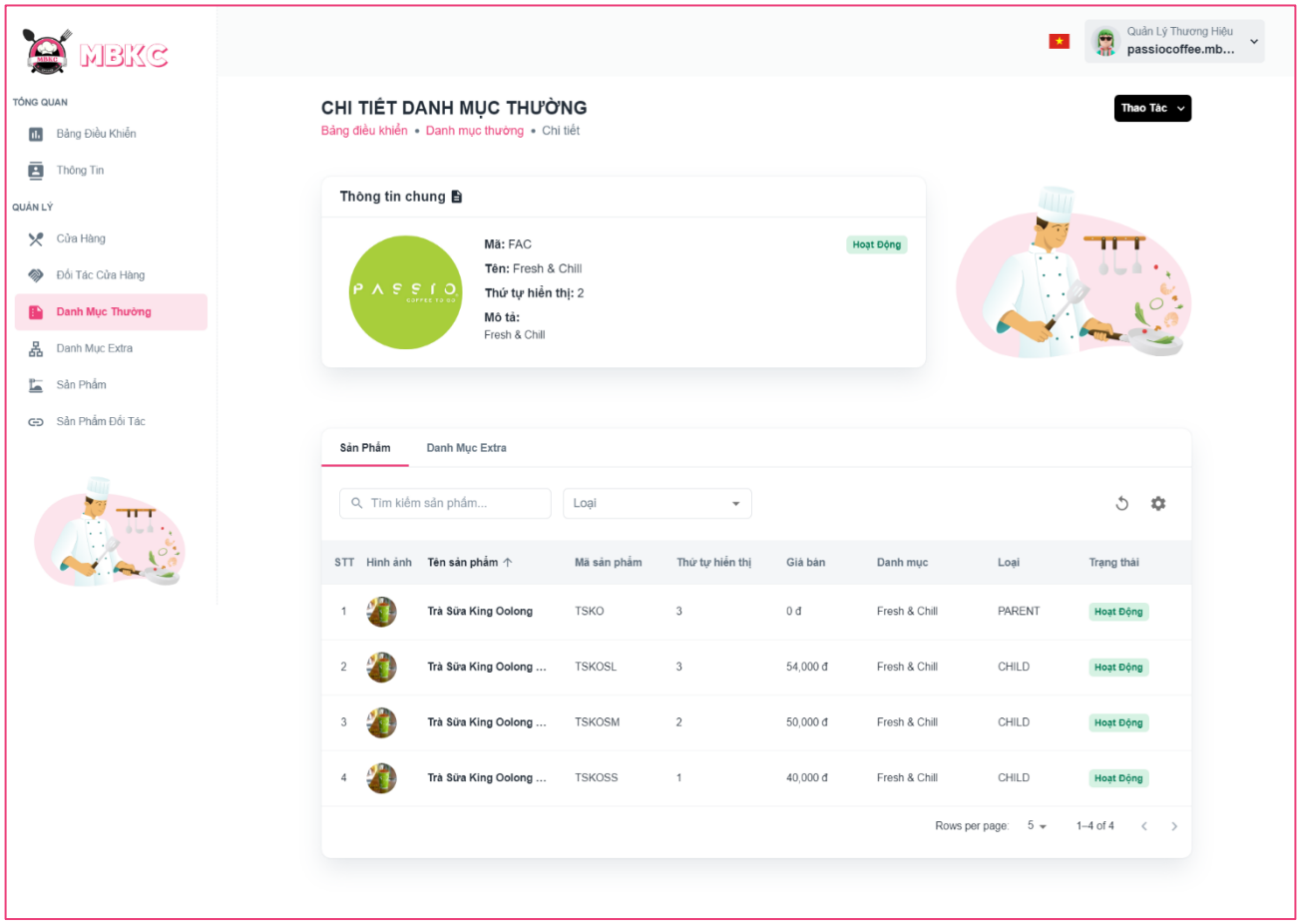


Figure 282: User Manual - View details Normal Categories - Step 5

Case: Click “Danh Mục Extra” tab to view the list of extra categories belonging to this normal category.

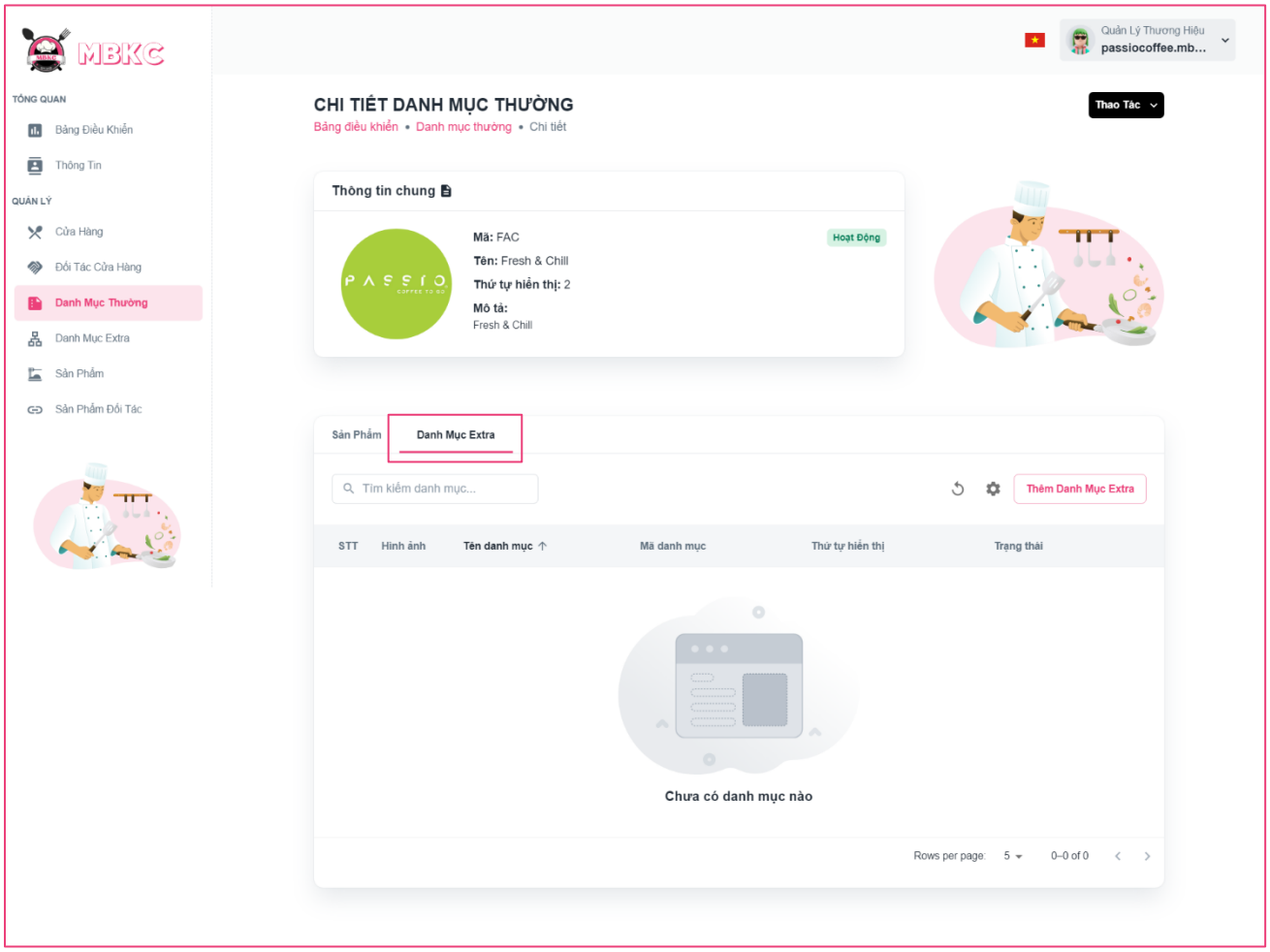


Figure 283: User Manual - View details Normal Categories - Step 5 - Case

3.2.6.4. Add Extra Category to Normal Category

3.2.6.4.1. Description

Use this guideline to add extra category to normal category.

This web application supports add extra category to normal category with account as a Brand Manager.

3.2.6.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

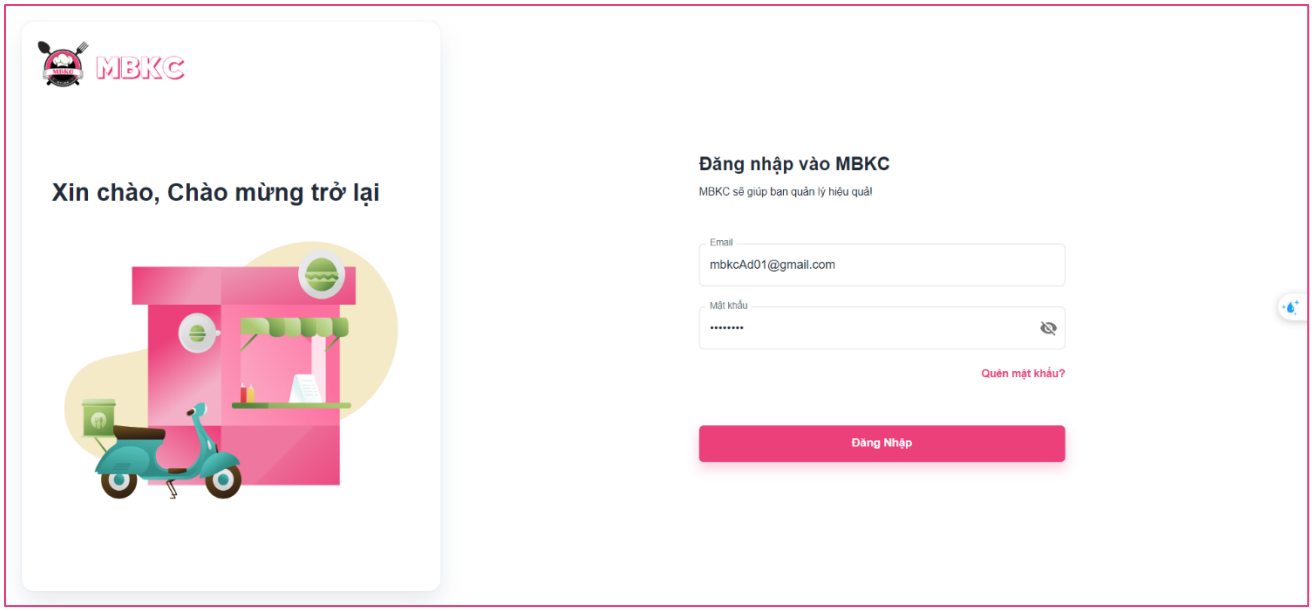


Figure 284: User Manual - View details Normal Categories - Step 1

Step 2: Click “Đăng Nhập” button.

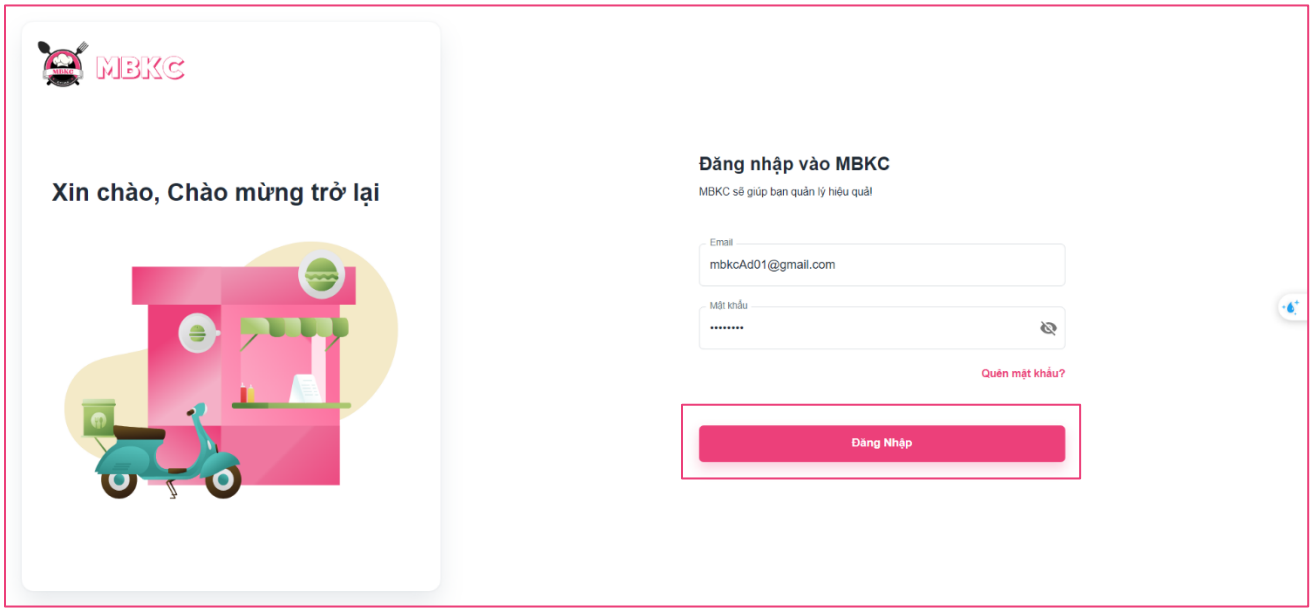


Figure 285: User Manual - View details Normal Categories - Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Thường” from the left sidebar.

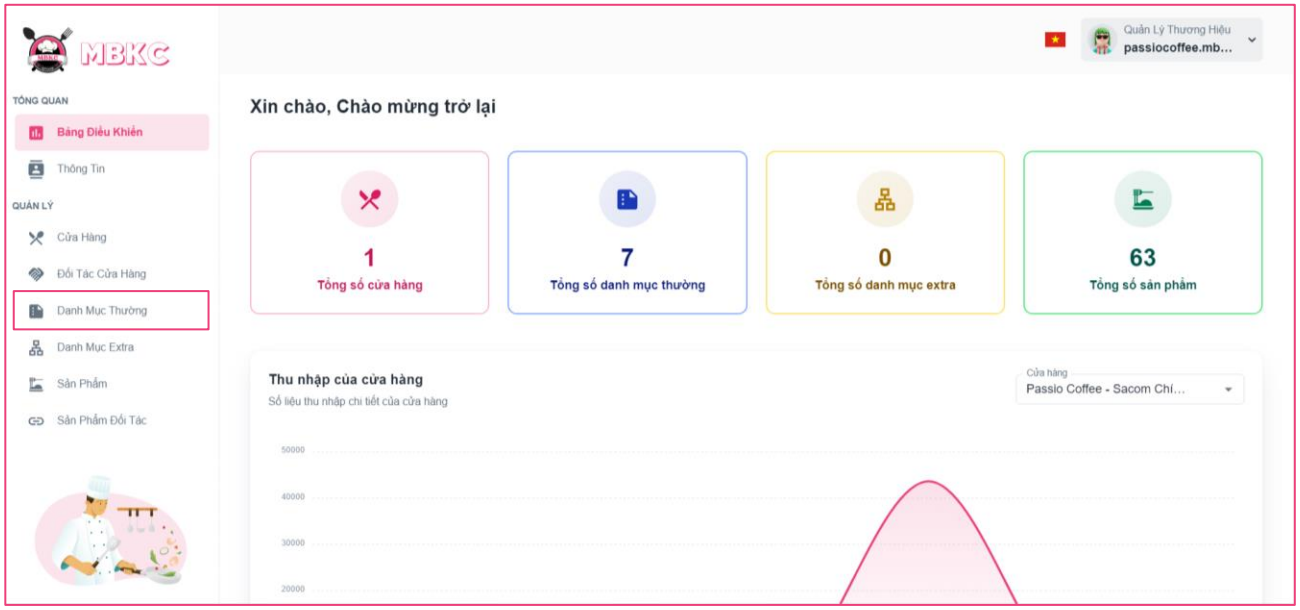


Figure 286: User Manual - View details Normal Categories - Step 3

Step 4: Click the normal category row to navigate to Normal Category Details Page.

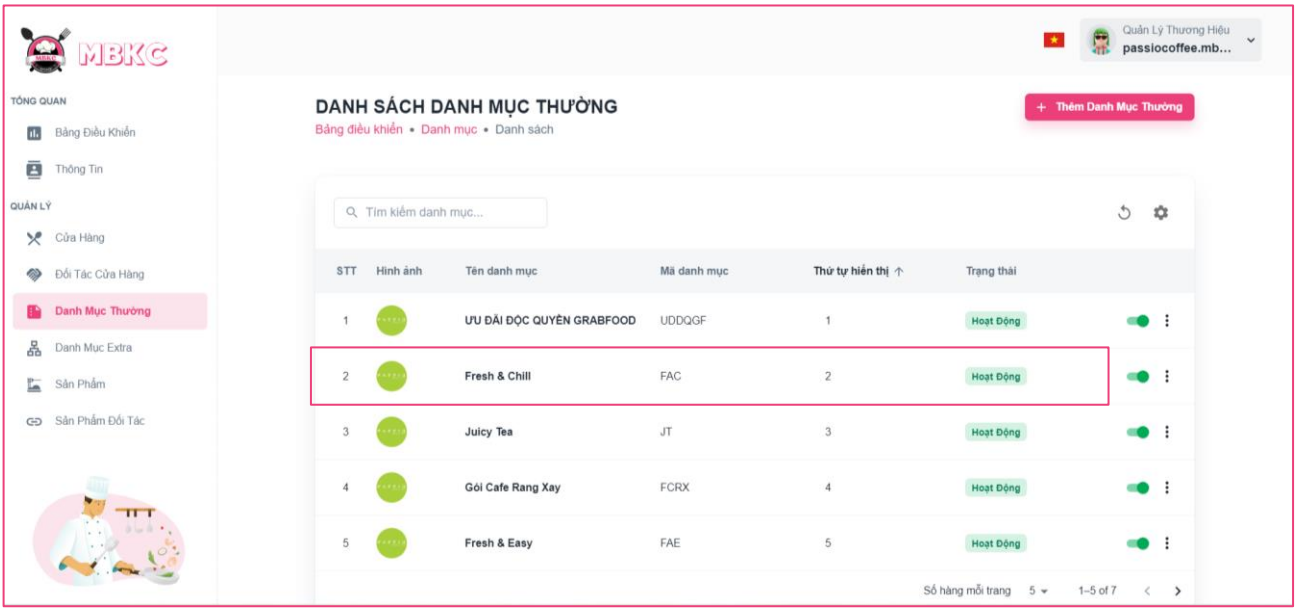


Figure 287: User Manual - View details Normal Categories - Step 4

Step 5: Navigate to Normal Category Details Page to show information details of normal category.

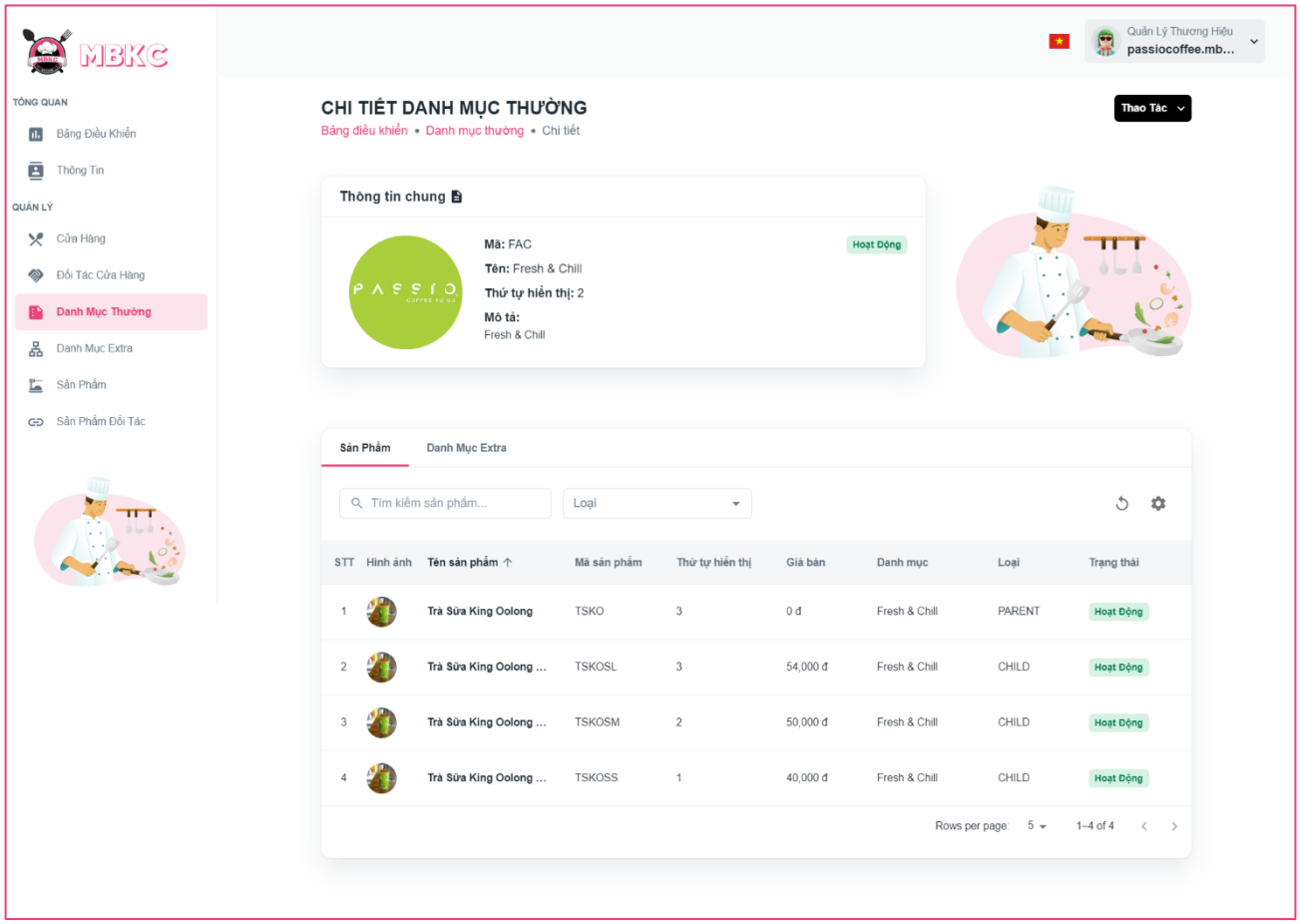


Figure 288: User Manual - View details Normal Categories - Step 5

Step 6: Click “Danh Mục Extra” tab to view the list of extra categories belonging to this normal category.

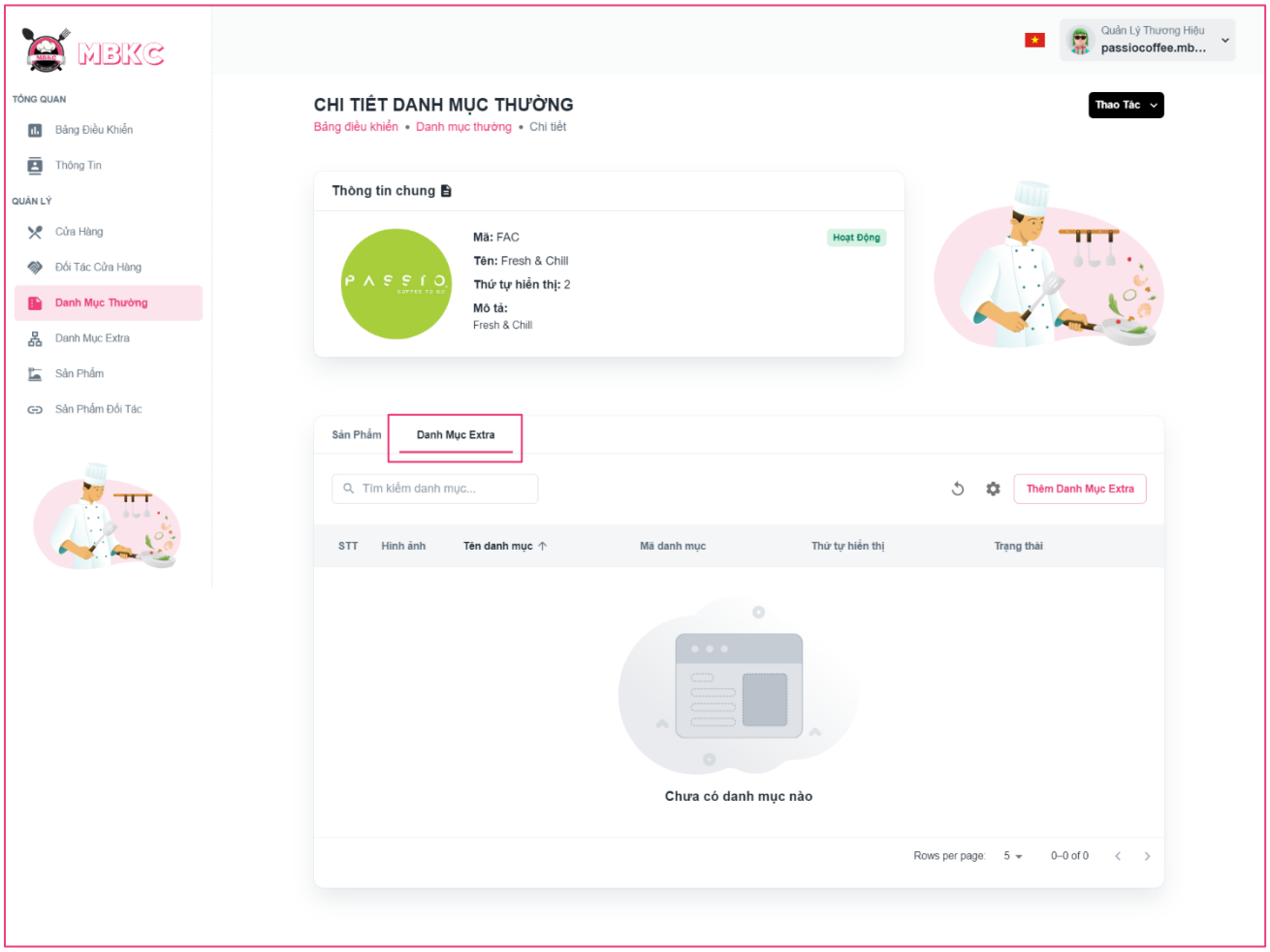


Figure 289: User Manual - View details Normal Categories - Step 6

Step 7: Click “Thêm Danh Mục Extra” button to pop up Extra List in Category Modal to add extra category to this normal category.

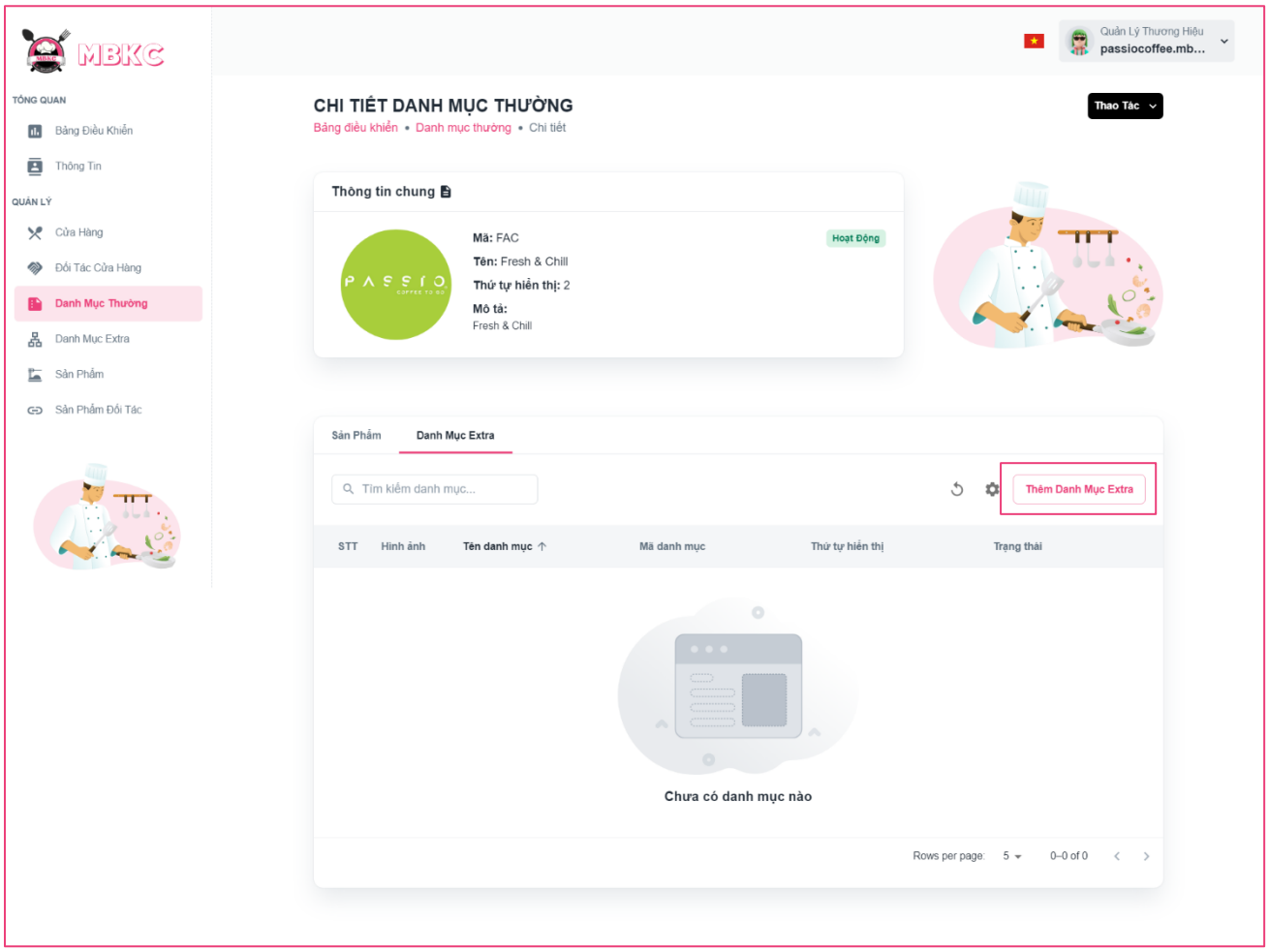


Figure 290: User Manual - View details Normal Categories - Step 7

Step 8: Check on the box of the extra category row want to add an extra category to this normal category.

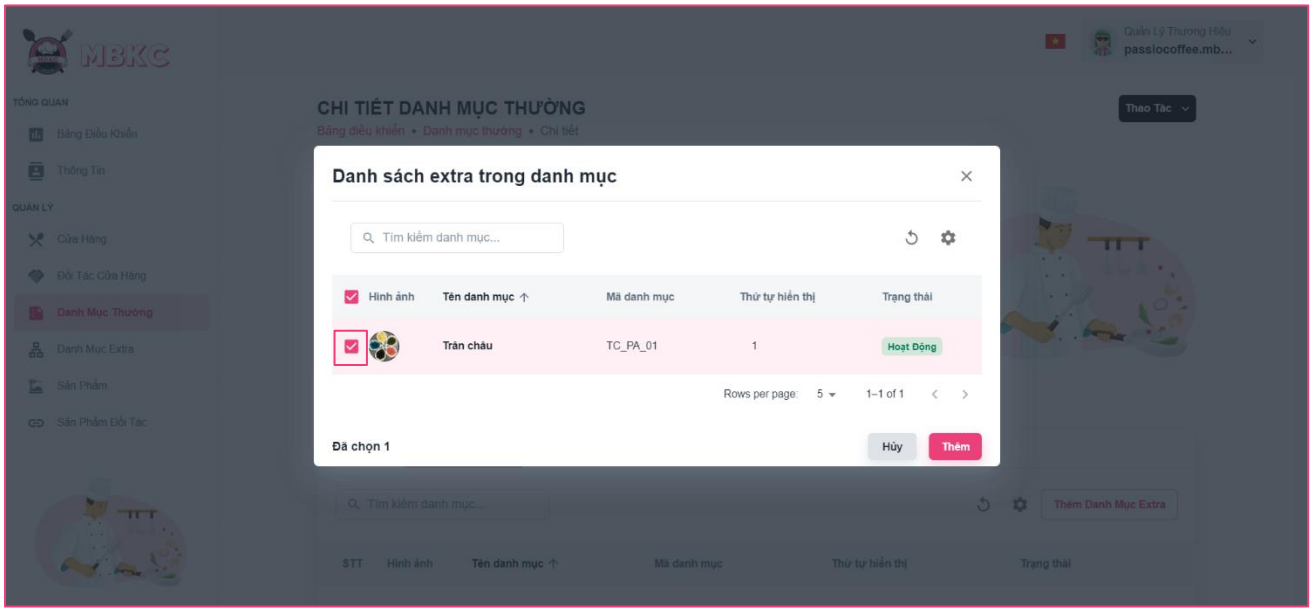


Figure 291: User Manual - View details Normal Categories - Step 8

Step 9: Click “Thêm” button.

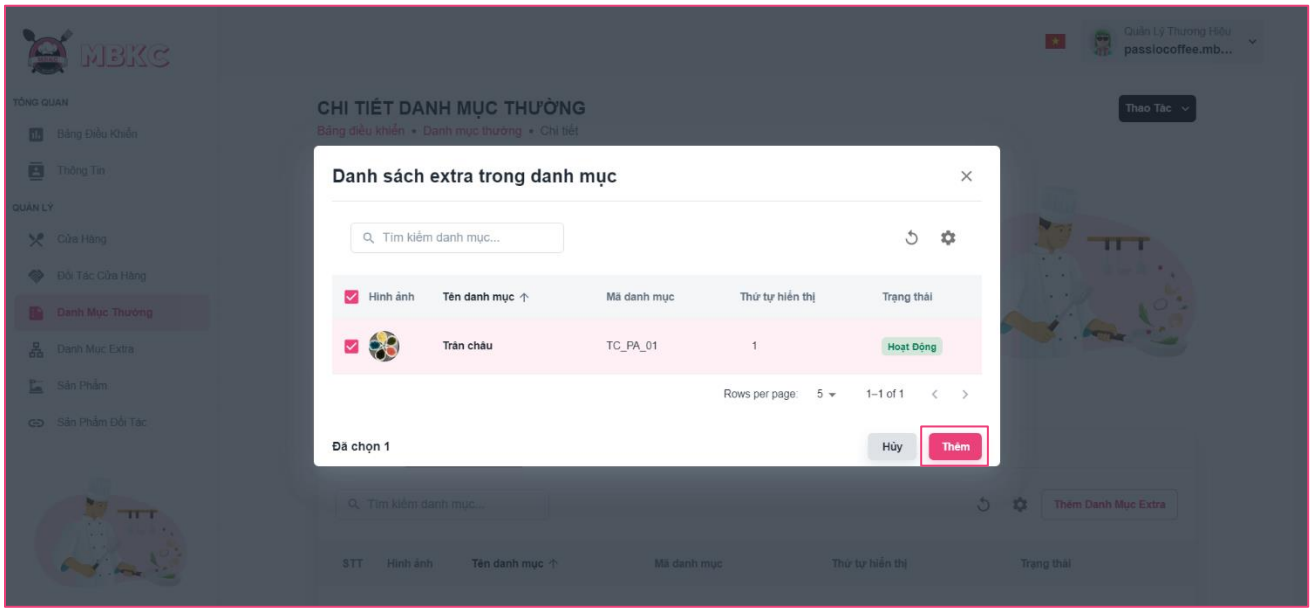


Figure 292: User Manual - View details Normal Categories – Step 9

Step 10: Normal Category Details Page will be reloaded information details of normal category and a toast message “Tạo thành công các danh mục extra cho danh mục thường”.

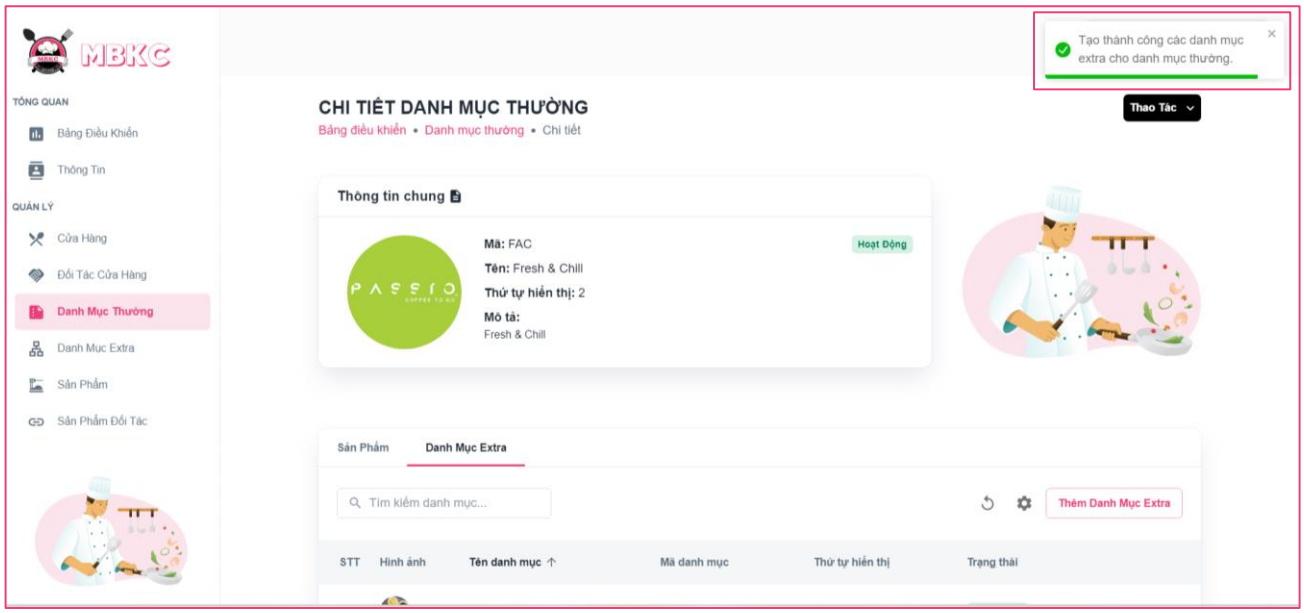


Figure 293: User Manual - View details Normal Categories – Step 10

3.2.6.5. Update Status Normal Category

3.2.6.5.1. Description

Use this guideline to update the status of the normal category.

This web application supports update the status of the normal category with account as a Brand Manager.

3.2.6.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

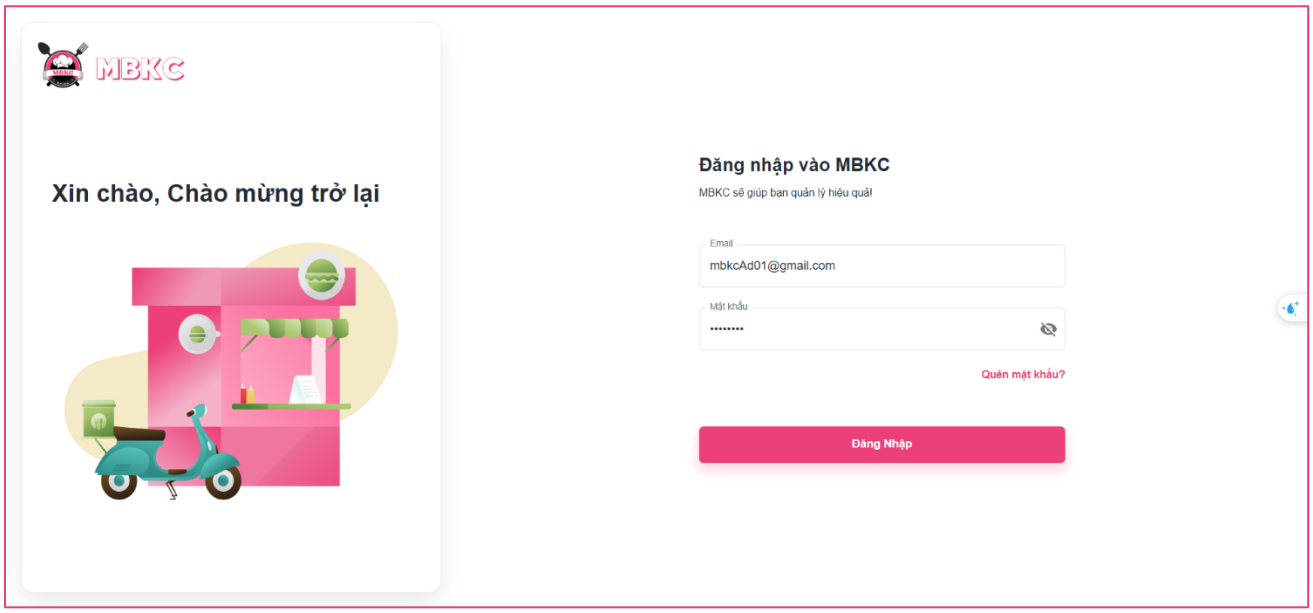


Figure 294: User Manual – Update Status Normal Category – Step 1

Step 2: Click “Đăng Nhập” button.

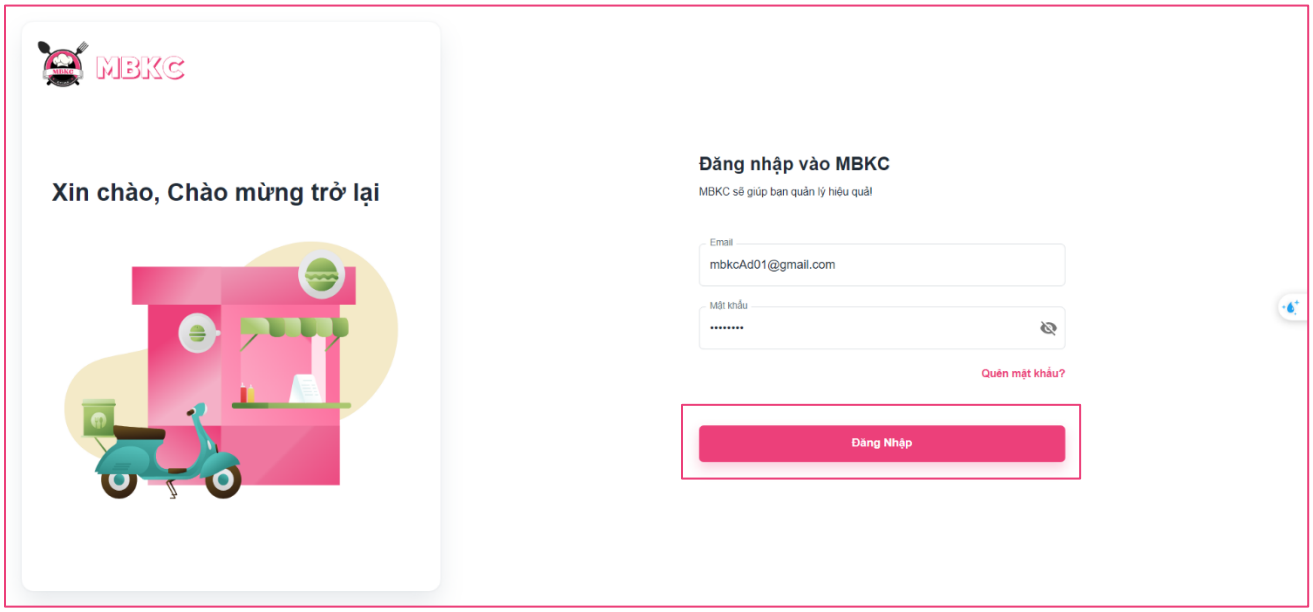


Figure 295: User Manual - Update Status Normal Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Thường” from the left sidebar.

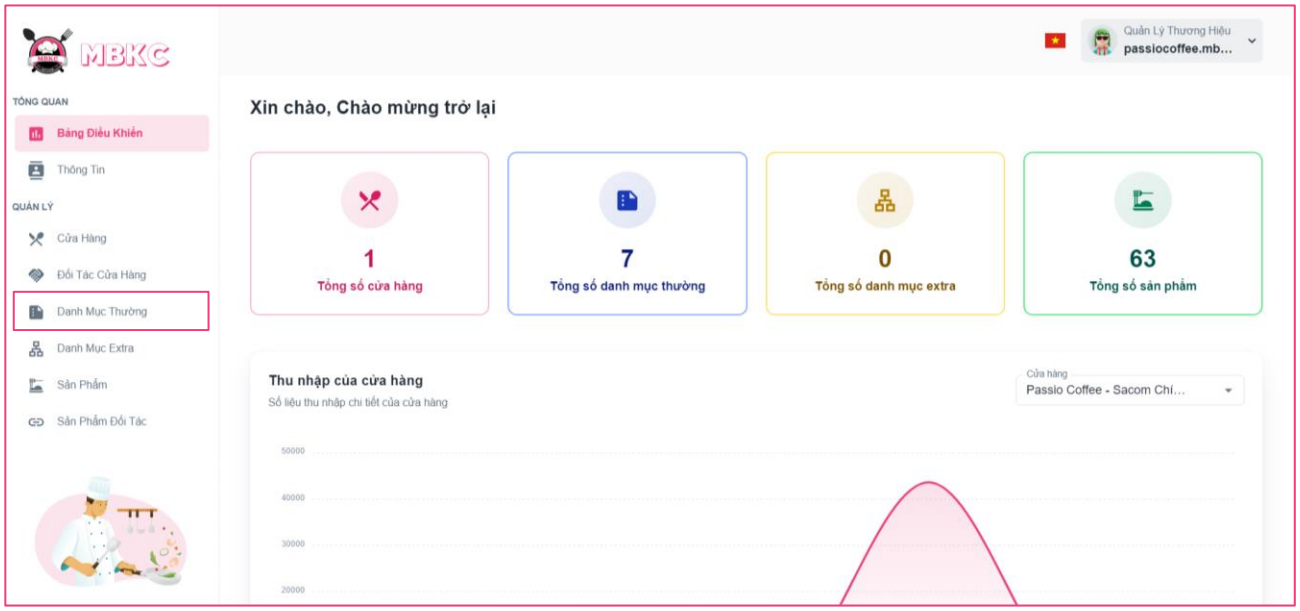


Figure 296: User Manual - Update Status Normal Category – Step 3

Step 4: Click on the switch button on the normal category row to update the status.

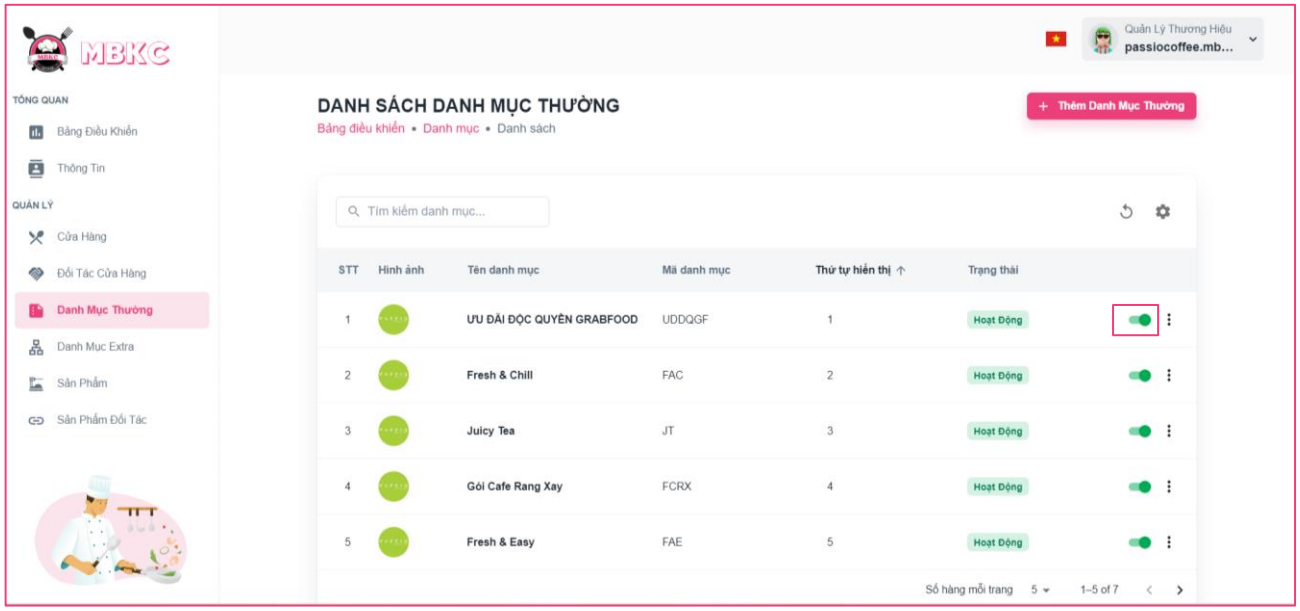


Figure …: User Manual - Update Status Normal Category - Step 3

Step 5: List of normal categories will be reloaded and a toast message “Đã cập nhật danh mục thành công”. And the status of the normal category has been updated.

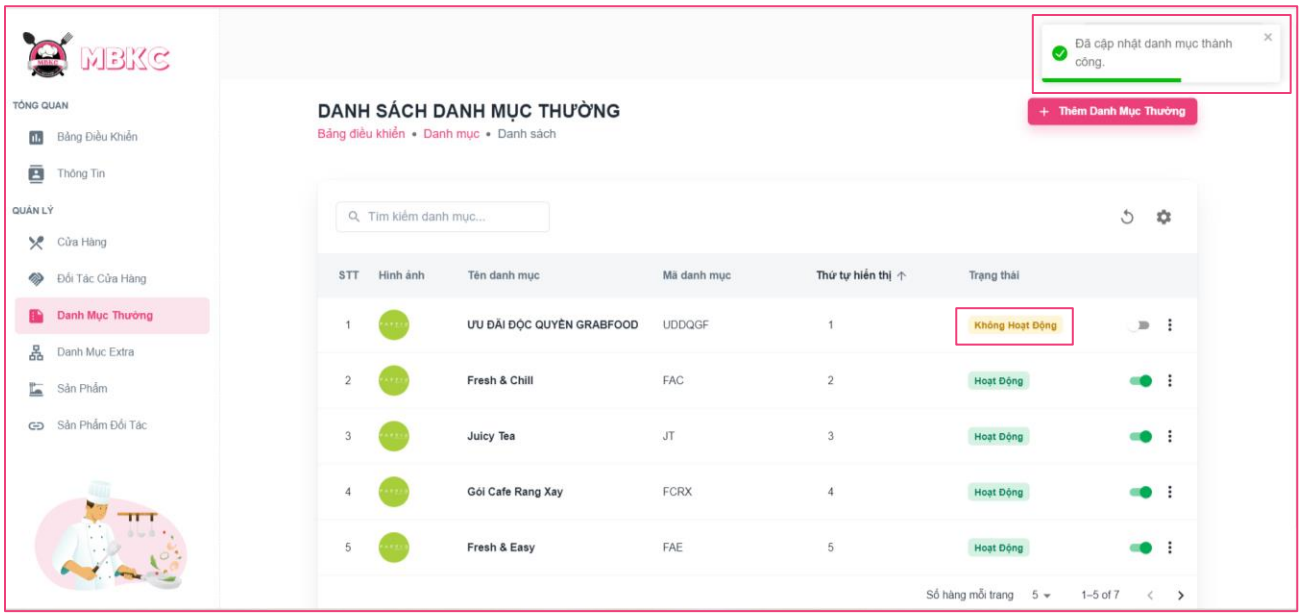


Figure 297: User Manual - Update Status Normal Category – Step 5

3.2.6.6. Update Information Normal Category

3.2.6.6.1. Description

Use this guideline to update information of the normal category.

This web application supports update information of the normal category with account as a Brand Manager.

3.2.6.6.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

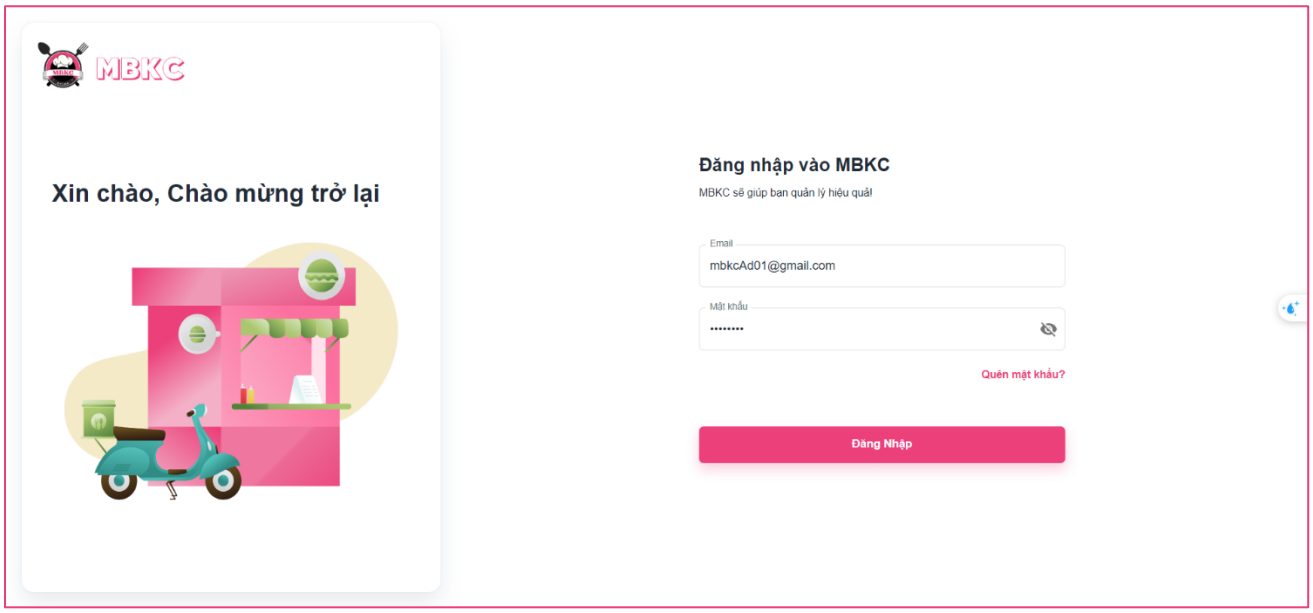


Figure 298: User Manual – Update Information Normal Category – Step 1

Step 2: Click “Đăng Nhập” button.

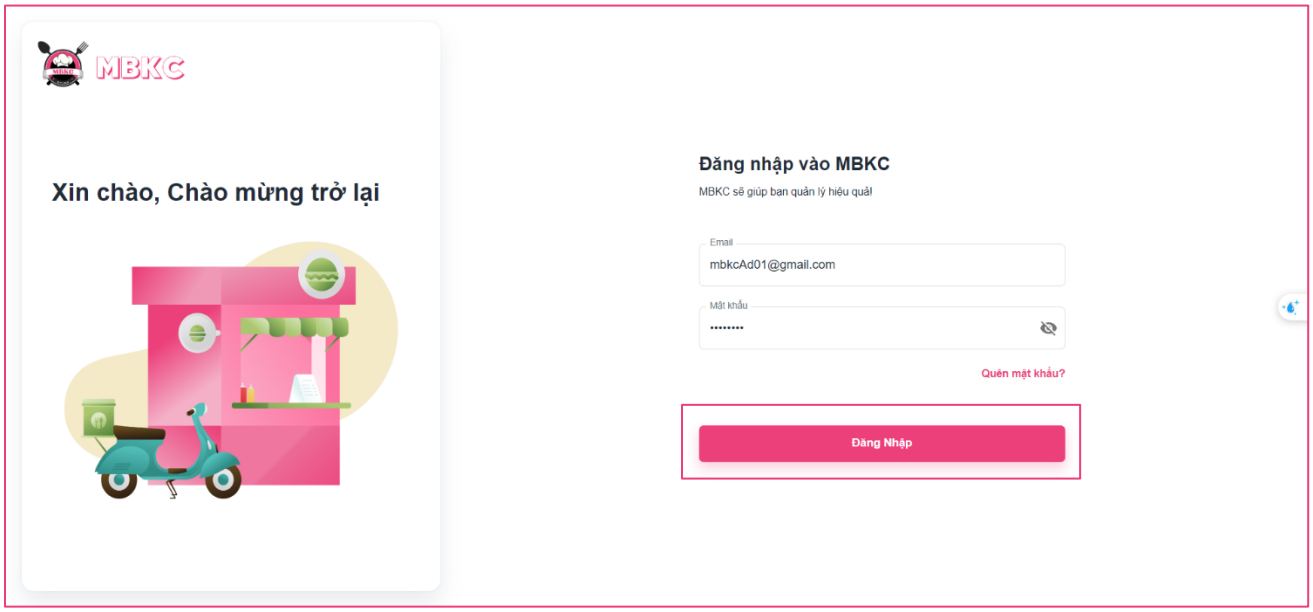


Figure 299: User Manual - Update Information Normal Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Thường” from the left sidebar.

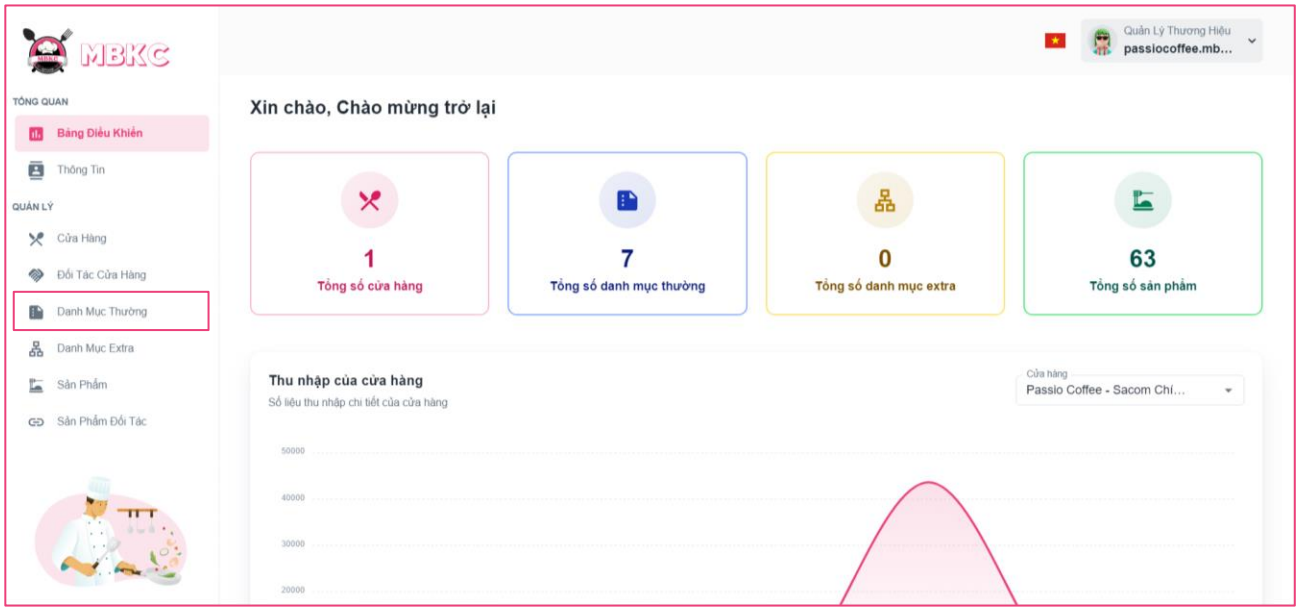


Figure 300: User Manual - Update Information Normal Category - Step 3

Step 4: Update information of the normal category.

Case 1: Update information of the normal category in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the normal category row want to navigate Update Normal Category Page.

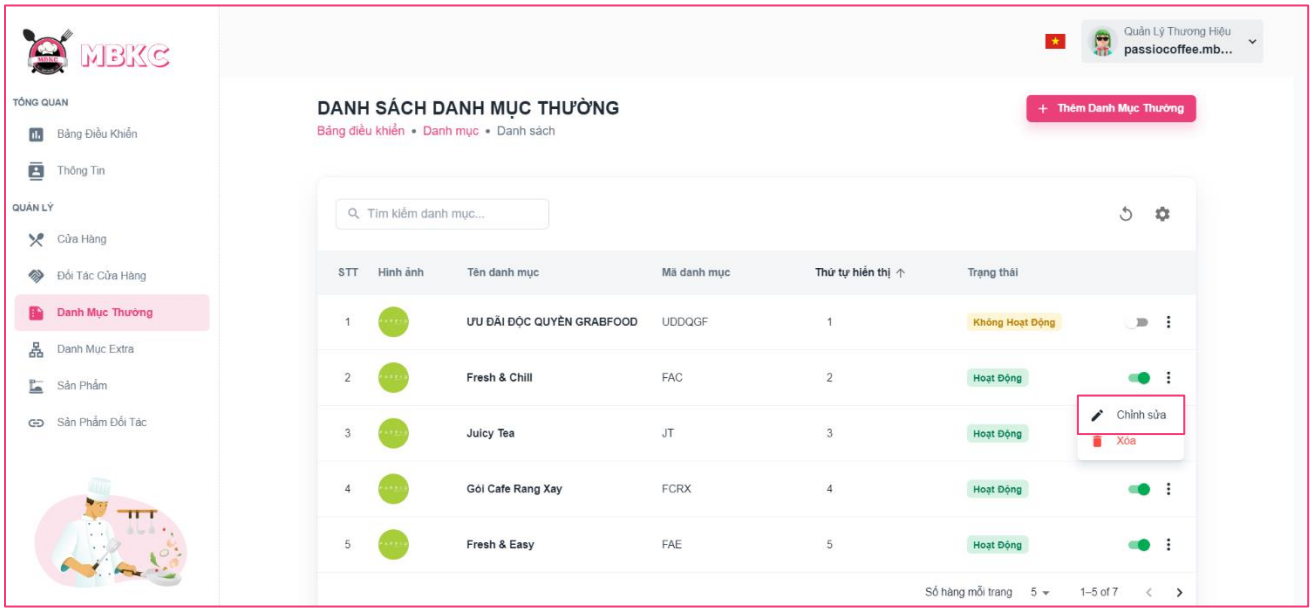


Figure 301: User Manual - Update Information Normal Category – Step 4 – Case 1

Case 2: Update information of the normal category on the details page.

1: Click the normal category row to navigate to Normal Category Details Page.

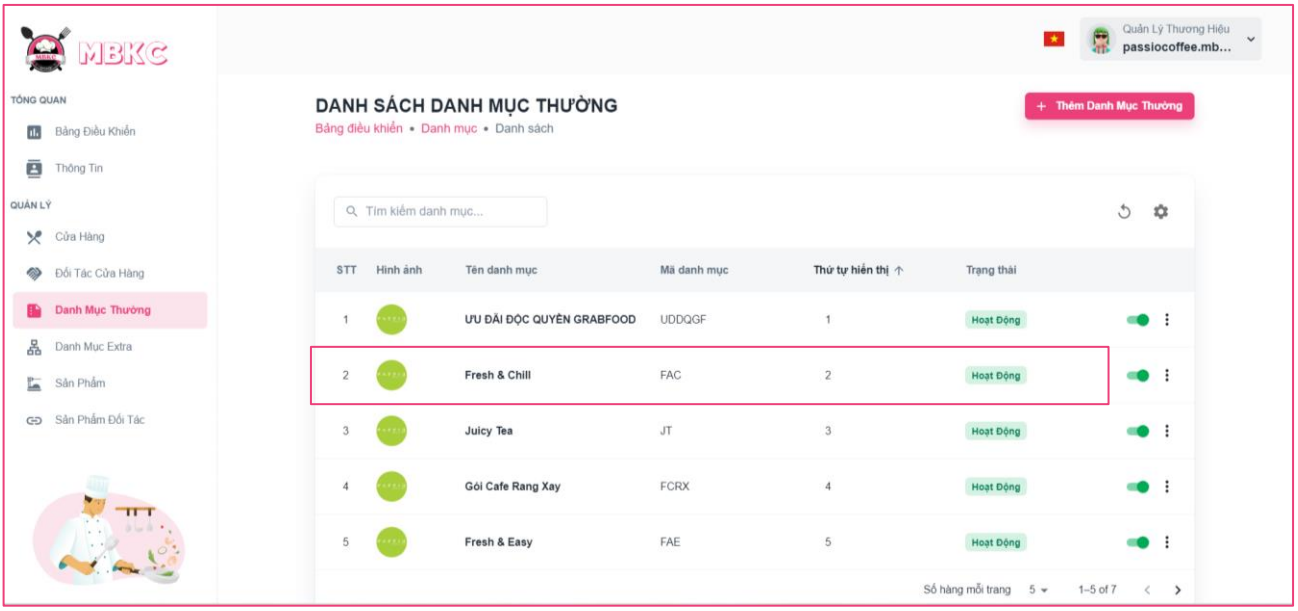


Figure ...: User Manual - Update Information Normal Category - Step 4 - Case 2 - 1

2: Click on “Thao Tác” button. Then click “Chỉnh sửa” menu item to navigate Update Normal Category Page.

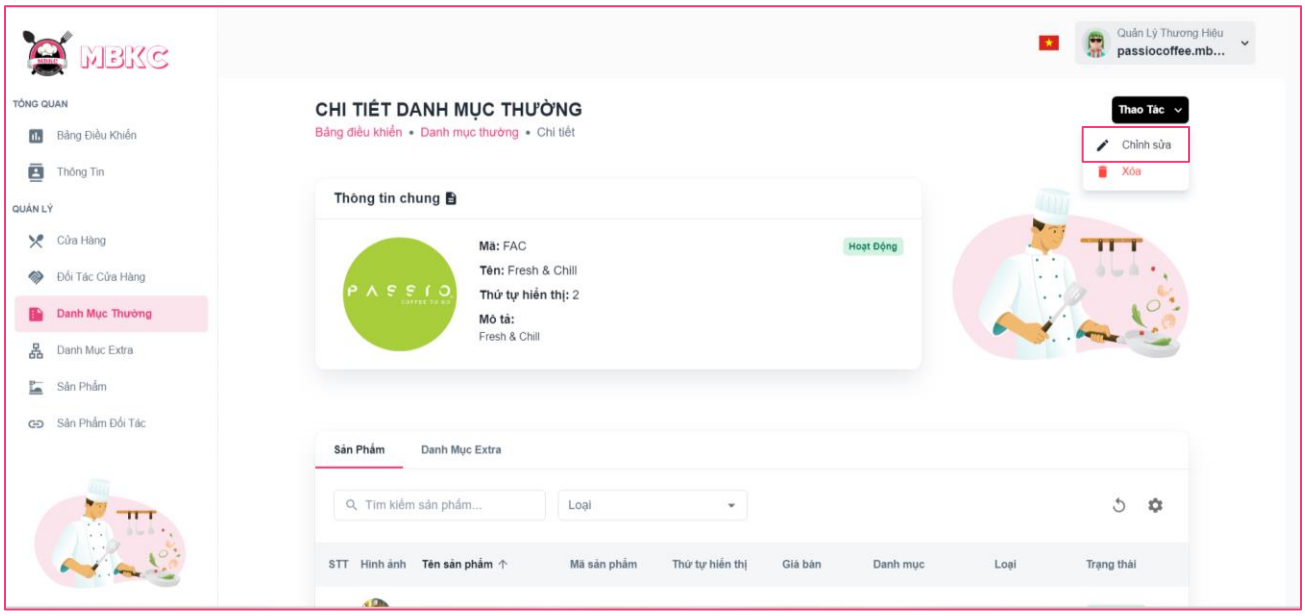


Figure 302: User Manual - Update Information Normal Category – Step 4 – Case 2 - 2

Step 5: Normal category information will be automatically filled in the form. Then edit the information as desired.

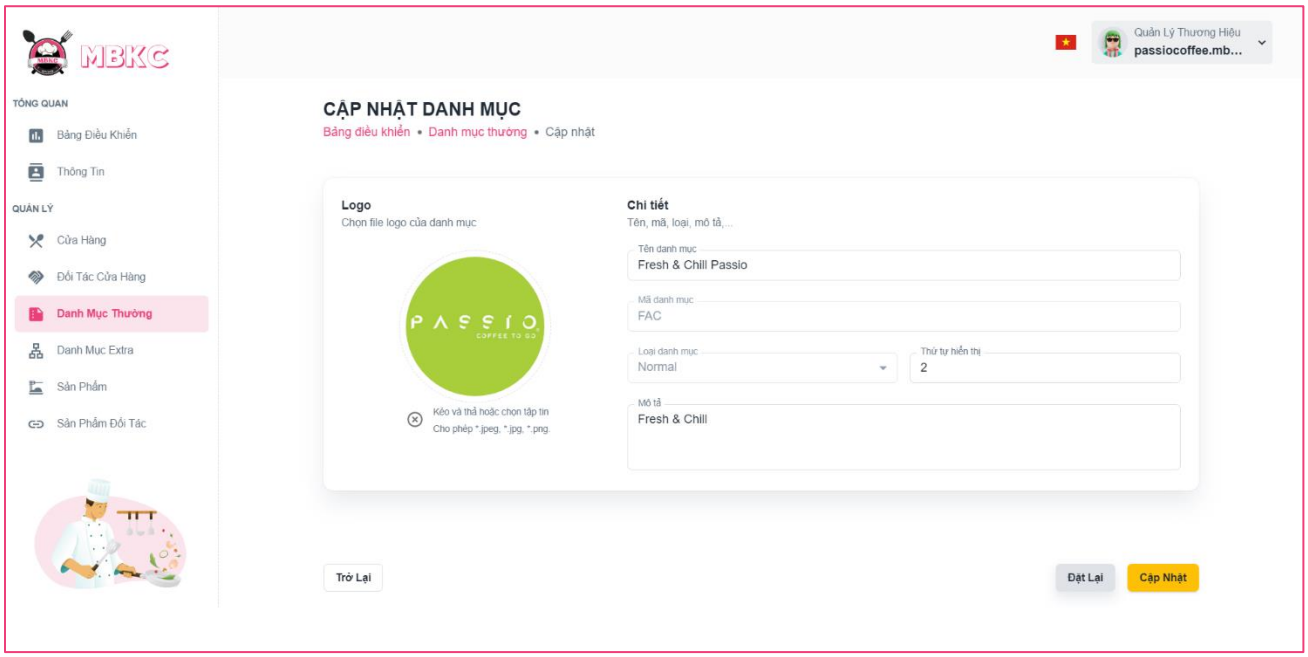


Figure 303: User Manual - Update Information Normal Category – Step 5

Step 6: Click "Cập nhật" button.

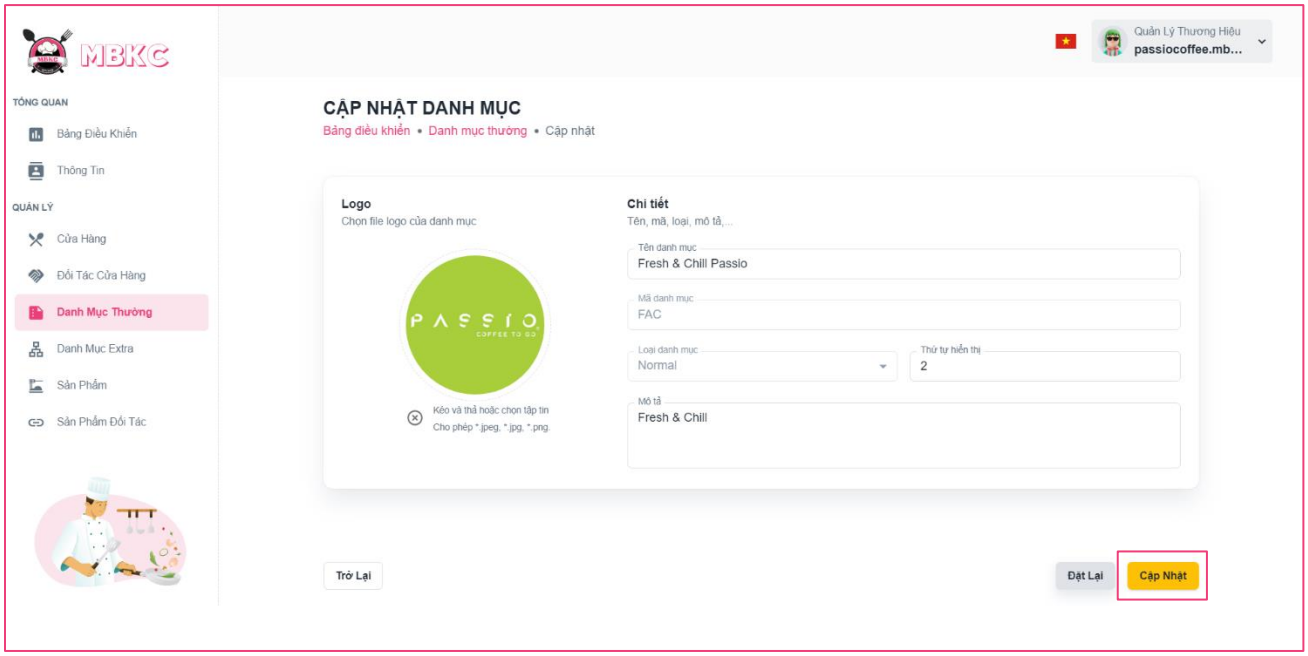


Figure 304: User Manual - Update Information Normal Category – Step 6

Step 7:

Case 1: You will be navigated to Normal Category Details Page showing the normal category details when start edit information of normal category from Normal Category Details Page and a toast message “Đã cập nhật thương hiệu thành công”.

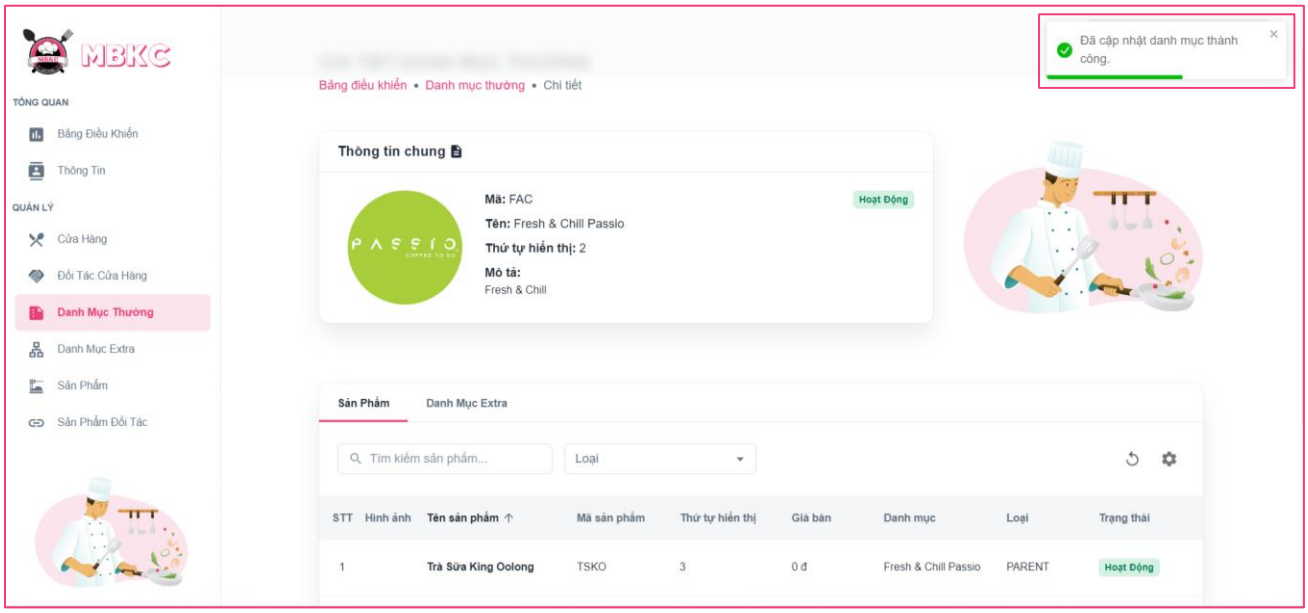


Figure 305: User Manual - Update Information Normal Category – Step 7 – Case 1

Case 2: You will be navigated to List of Normal Categories Page showing the list of normal categories when start edit information of normal category from List of Normal Categories Page and a toast message “Đã cập nhật danh mục thành công”.

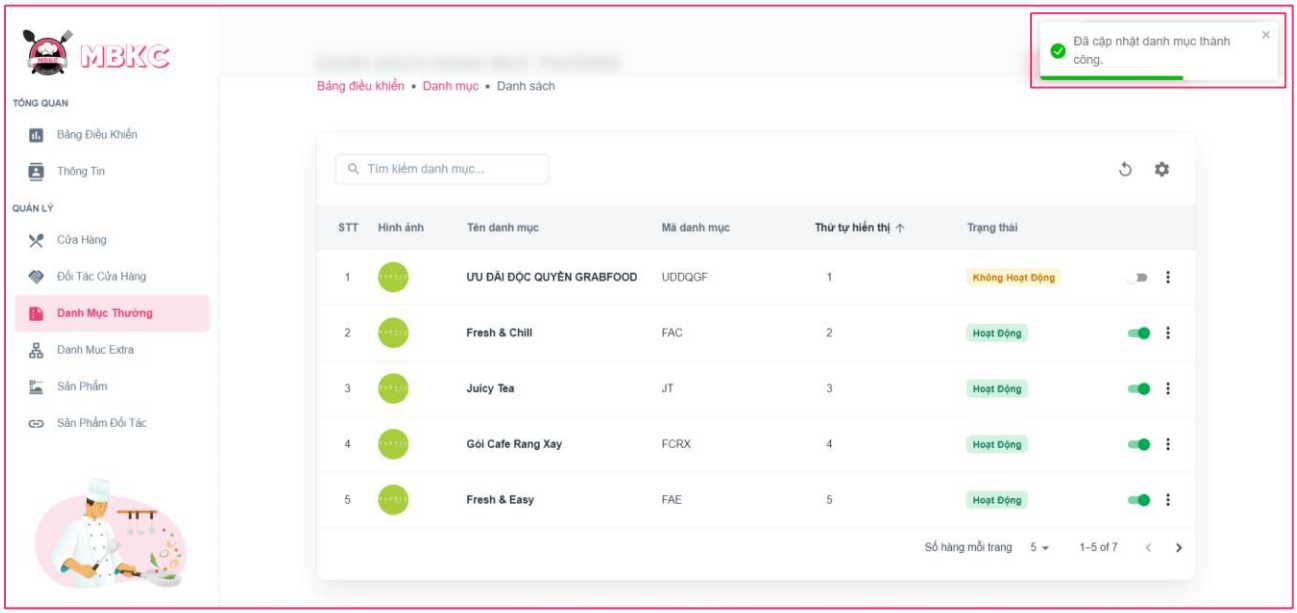


Figure 306: User Manual - Update Information Normal Category – Step 7 – Case 2

3.2.6.7. Delete Normal Category

3.2.6.7.1. Description

Use this guideline to delete normal category.

This web application supports delete normal category with account as a Brand Manager.

3.2.6.7.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

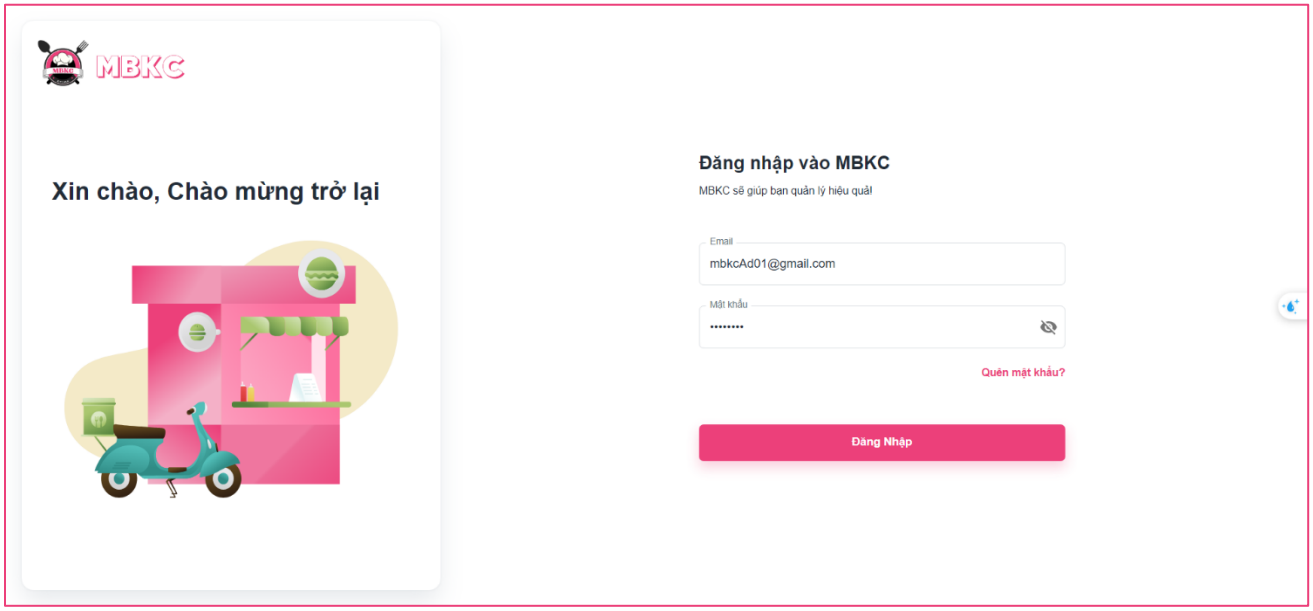


Figure 307: User Manual – Delete Normal Category – Step 1

Step 2: Click “Đăng Nhập” button.

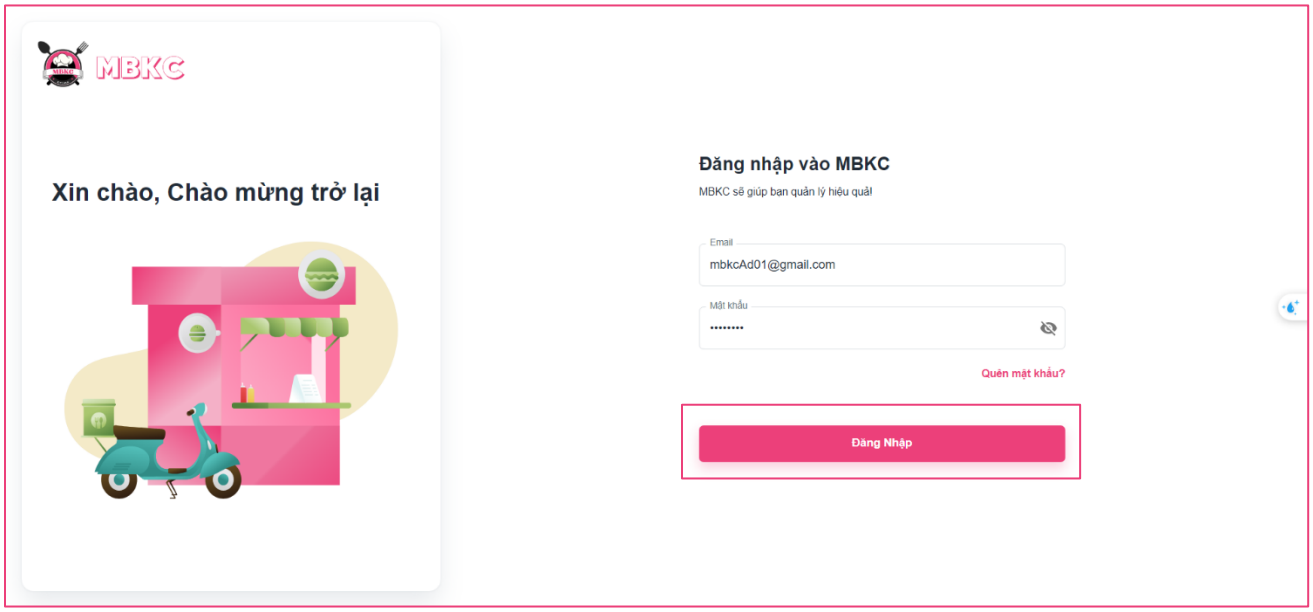


Figure 308: User Manual - Delete Normal Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Thường” from the left sidebar.

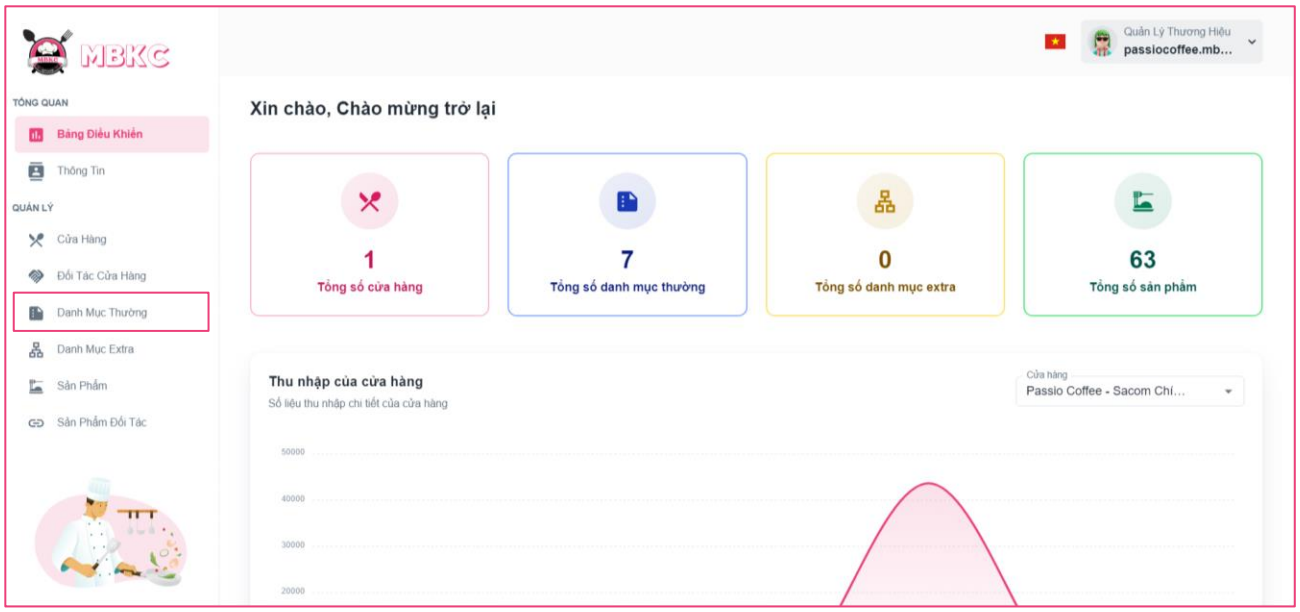


Figure 309: User Manual - Delete Normal Category – Step 3

Step 4: Delete normal category.

Case 1: Delete normal category in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the normal category row to delete.

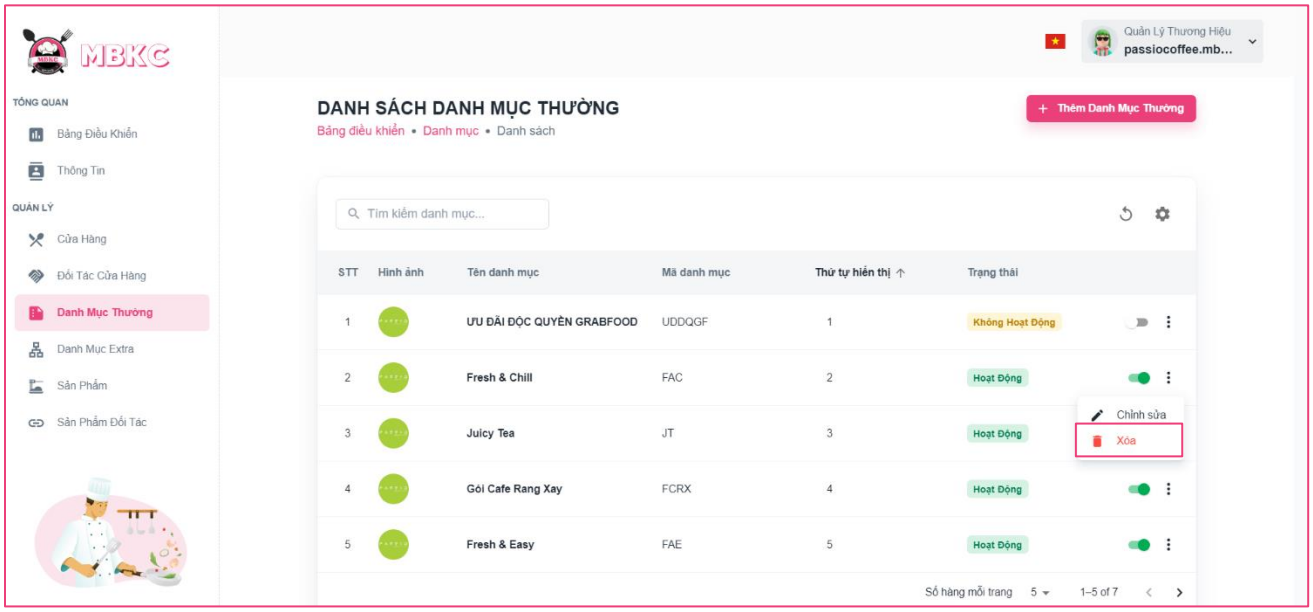


Figure 310: User Manual - Delete Normal Category – Step 4 – Case 1

Case 2: Delete normal category on the details page.

1: Click the normal category row to navigate to Normal Category Details Page.

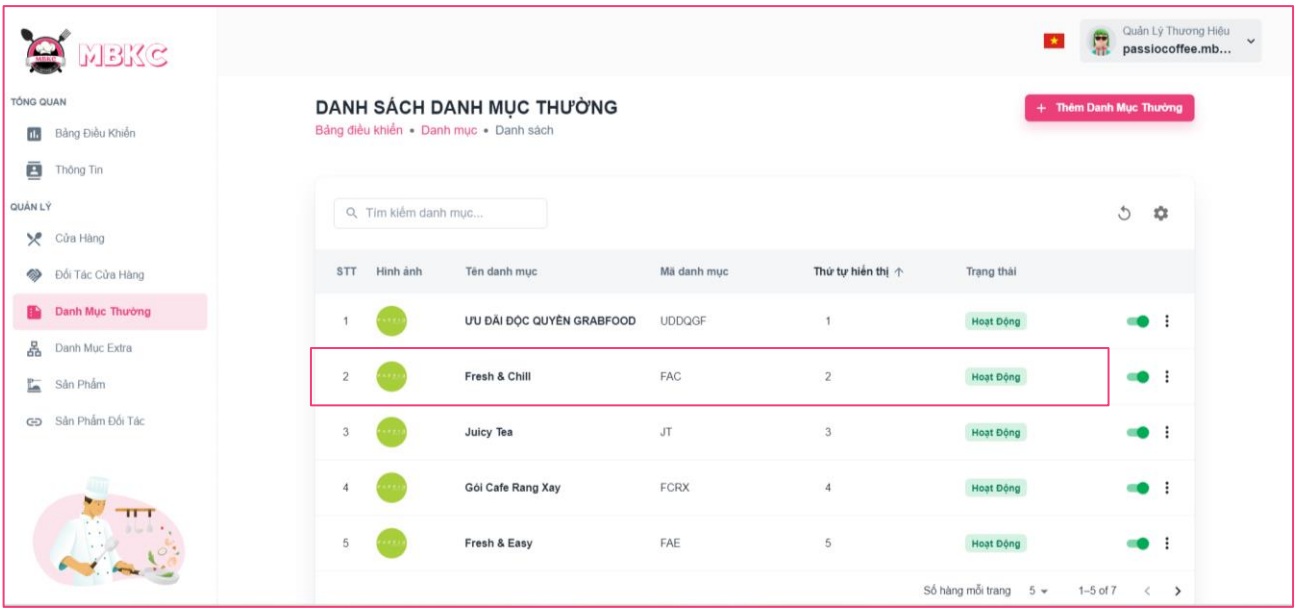


Figure 311: User Manual - Delete Normal Category – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item to delete.

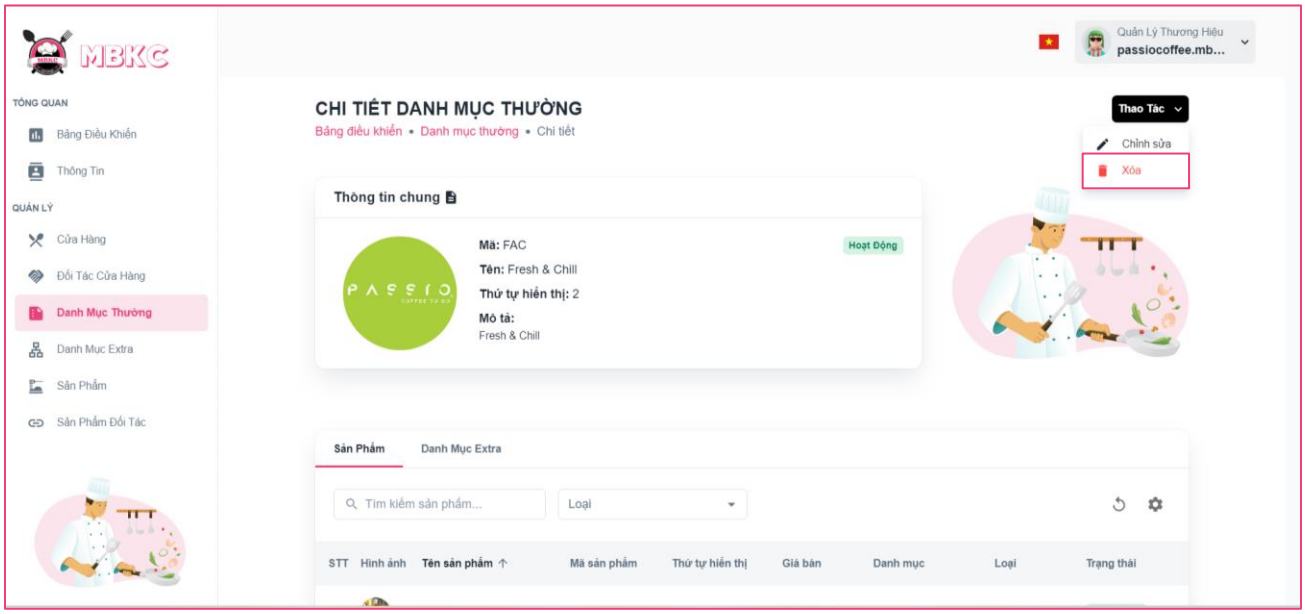


Figure 312: User Manual - Delete Normal Category - Step 4 - Case 2 - 2

Step 5: Click “Xác Nhận” button.

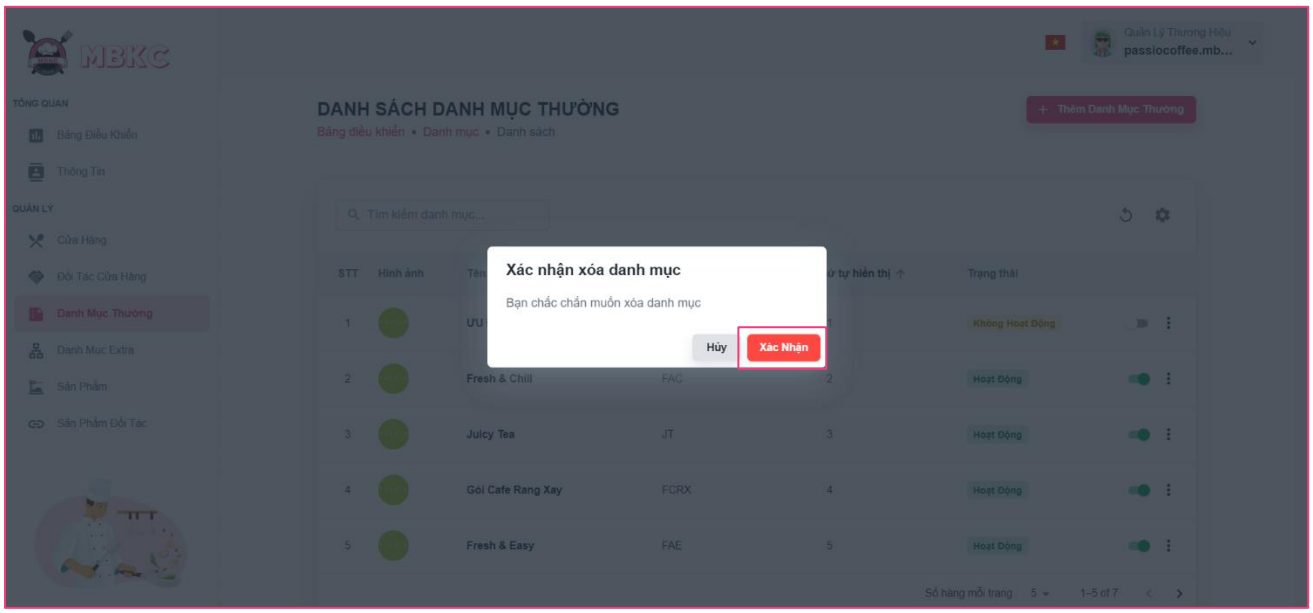


Figure 313: User Manual - Delete Normal Category – Step 5

Step 6: List of normal categories will be reloaded and a toast message “Đã xóa danh mục thành công”. And normal category has been deleted.

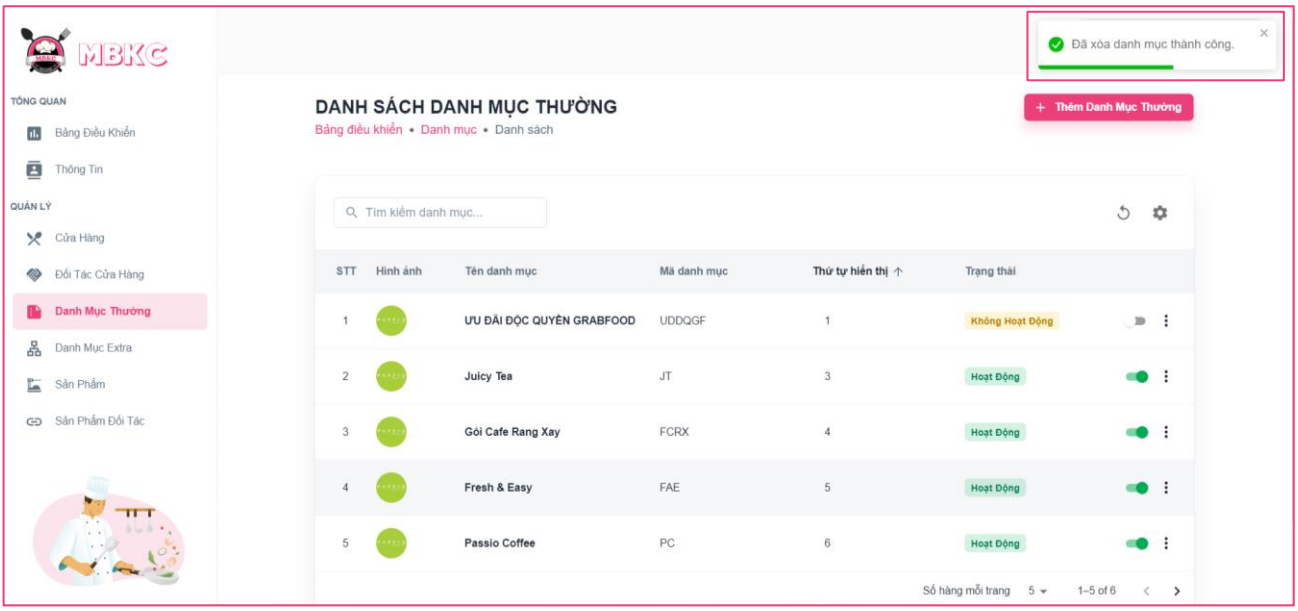


Figure 314: User Manual – Delete Normal Category – Step 6

3.2.7. Extra category management

3.2.7.1. Create new Extra Category

3.2.7.1.1. Description

Use this guideline to login into MBKC account to create a new extra category.

This web application supports create a new extra category with account as a Brand Manager.

3.2.7.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

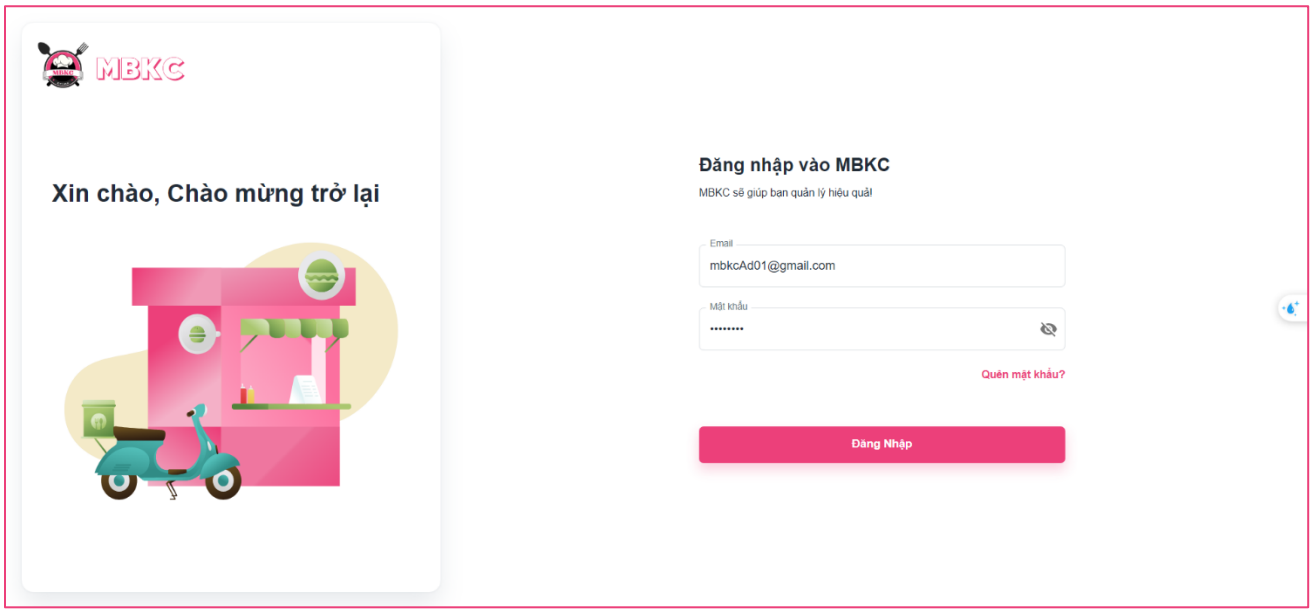


Figure 315: User Manual - Create new Extra Category – Step 1

Step 2: Click “Đăng Nhập” button.

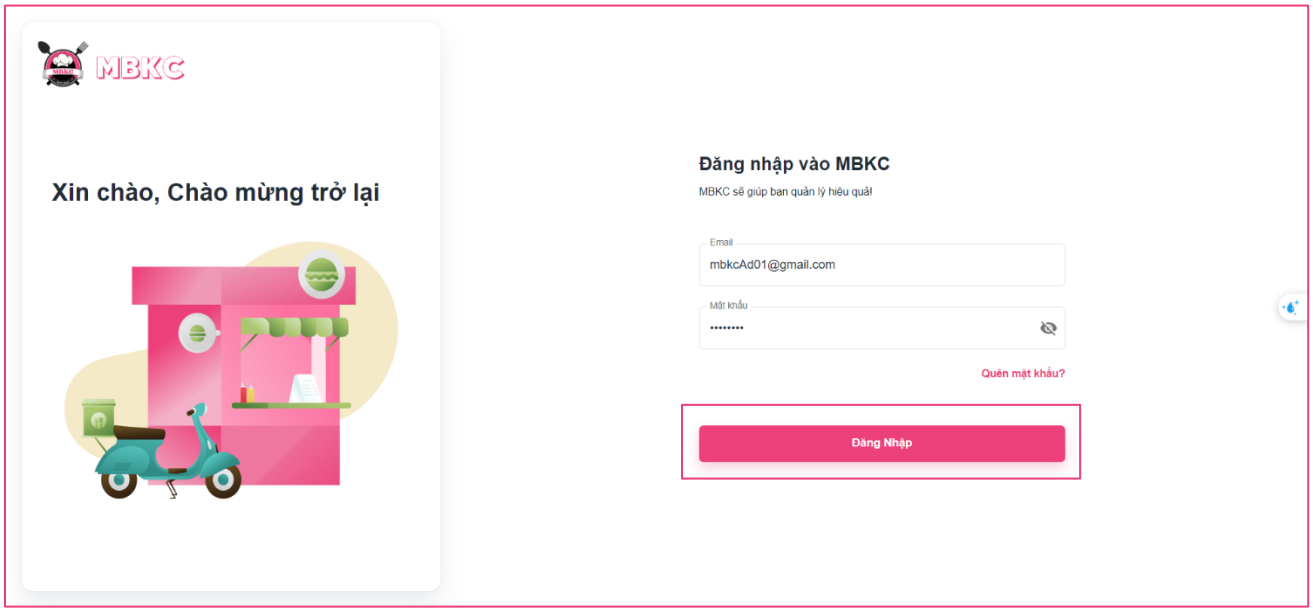


Figure 316: User Manual - Create new Extra Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Extra” from the left sidebar.

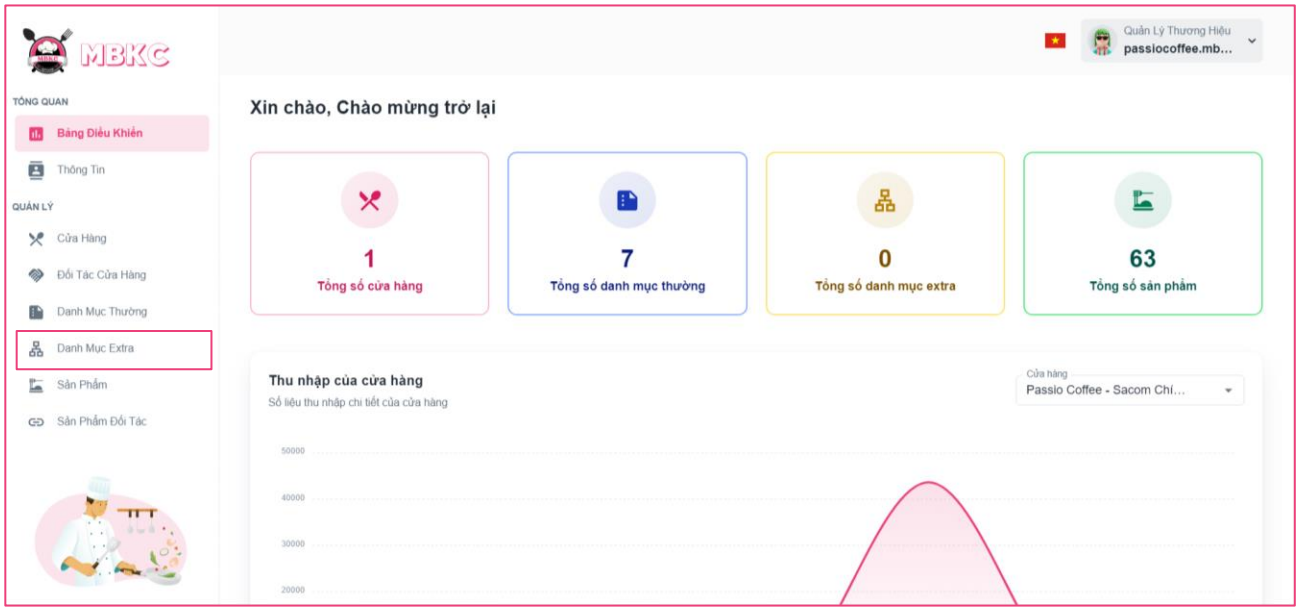


Figure 317: User Manual - Create new Extra Category – Step 3

Step 4: Click “Thêm Danh Mục Extra” button on the screen and navigate to Create New Extra Category Page to add a new extra category.

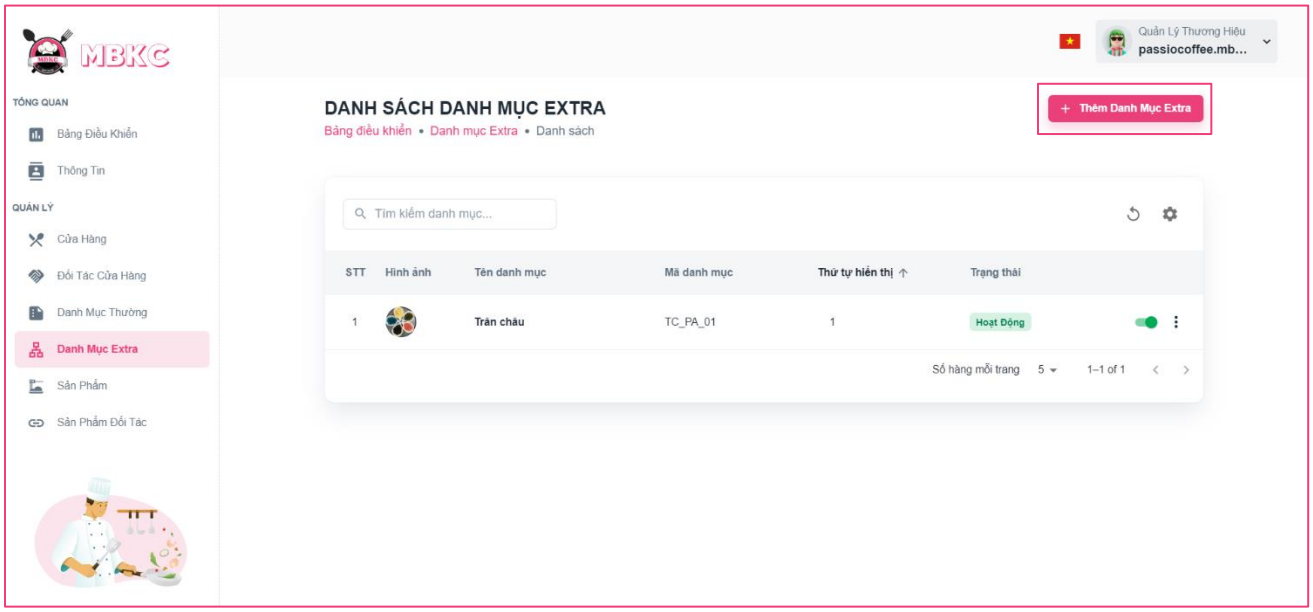


Figure 318: User Manual - Create new Extra Category – Step 4

Step 5: Filling all the required information to add a new extra category including: “Logo”, “Tên danh mục", “Mã danh mục", “Thứ tự hiển thị", “Mô tả”.

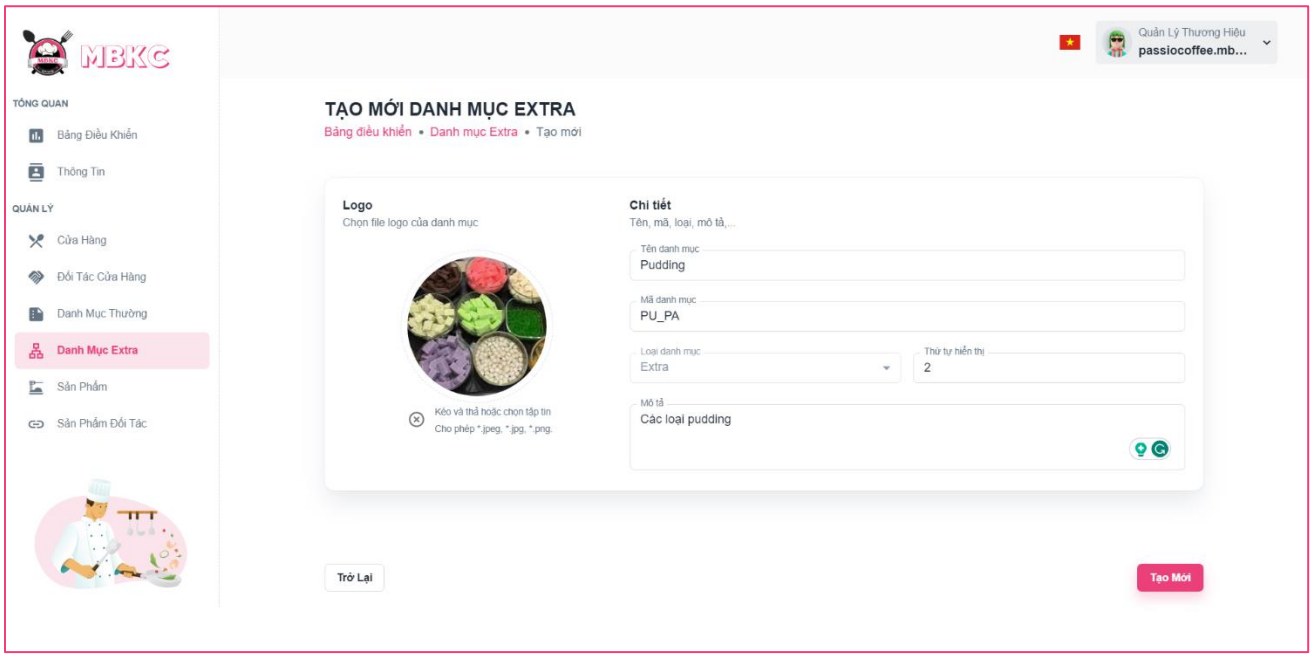


Figure 319: User Manual - Create new Extra Category – Step 5

Step 6: After filling in all required field, click “Tạo Mới” to finish creating a new extra category.

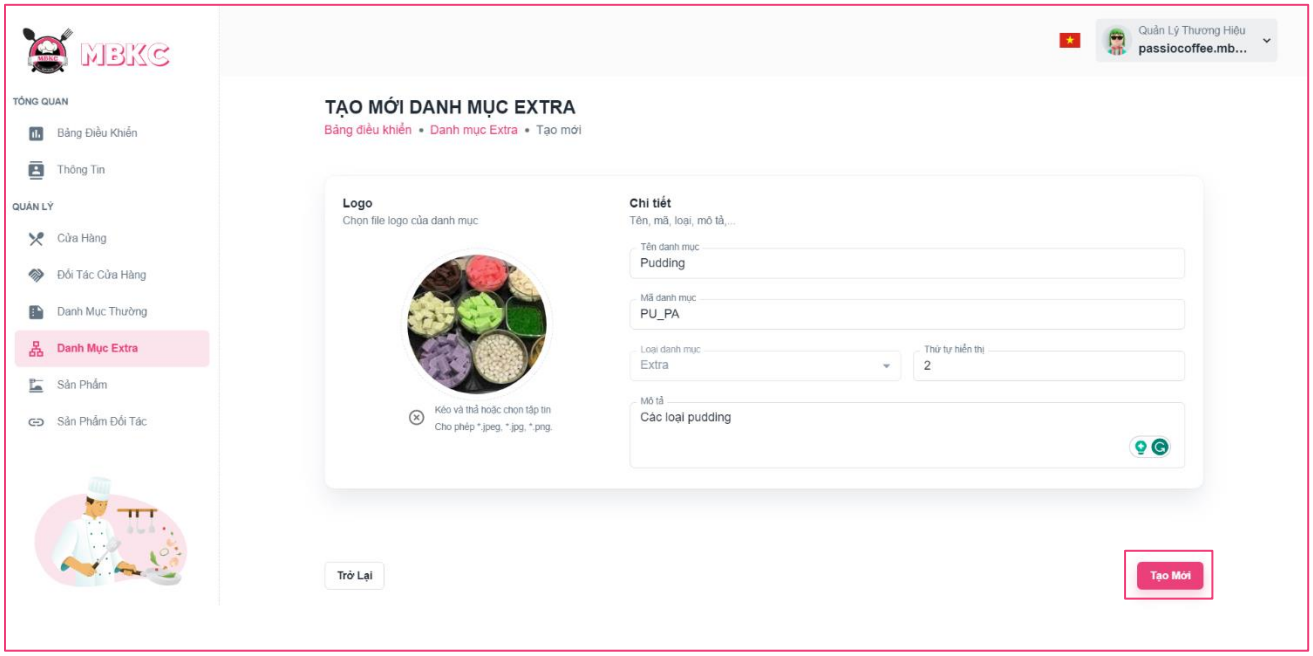


Figure 320: User Manual - Create new Extra Category – Step 6

Step 7: You will be navigated to List of Extra Categories Page showing the list of extra categories and a toast message “Đã tạo danh mục mới thành công”.

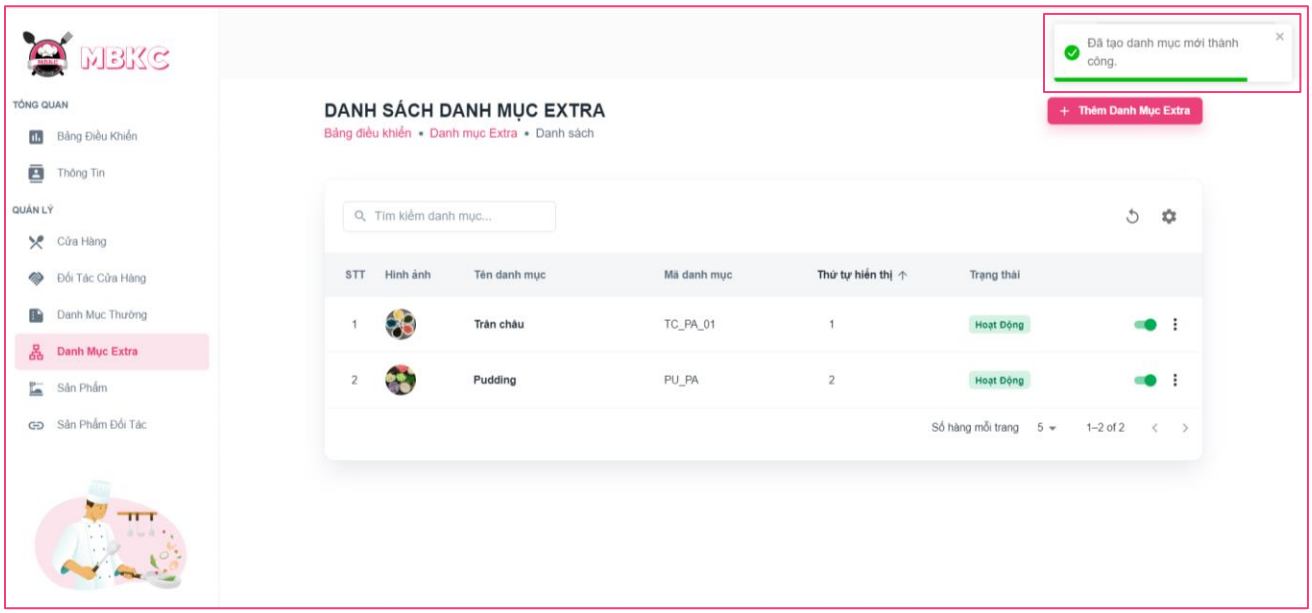


Figure 321: User Manual – Create new Extra Category – Step 7

3.2.7.2. View list of Extra Categories

3.2.7.2.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of extra categories.

This web application supports view list of extra categories with account as a Brand Manager.

3.2.7.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

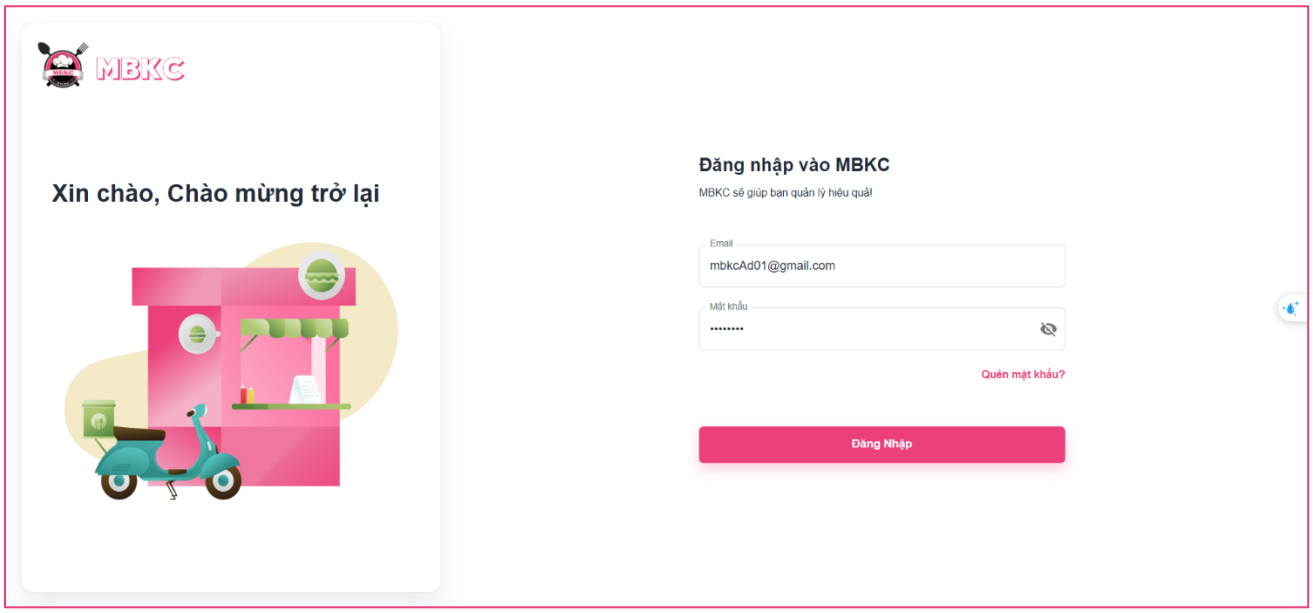


Figure 322: User Manual - View list of Extra Categories – Step 1

Step 2: Click “Đăng Nhập” button.

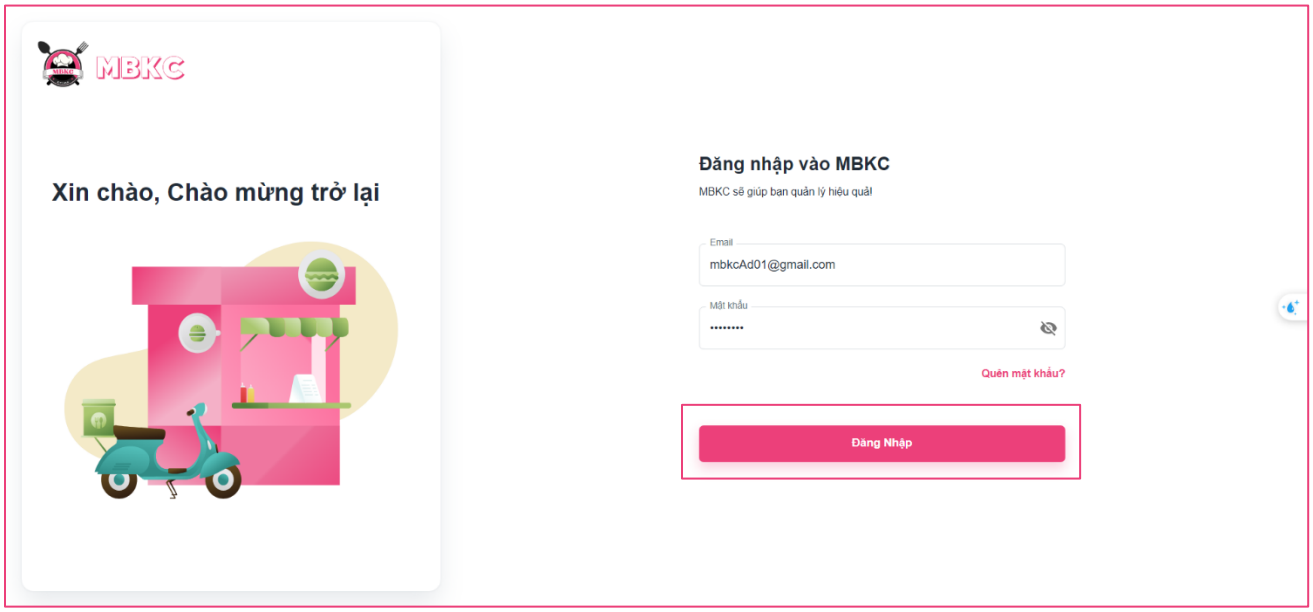


Figure 323: User Manual - View list of Extra Categories – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Extra” from the left sidebar.

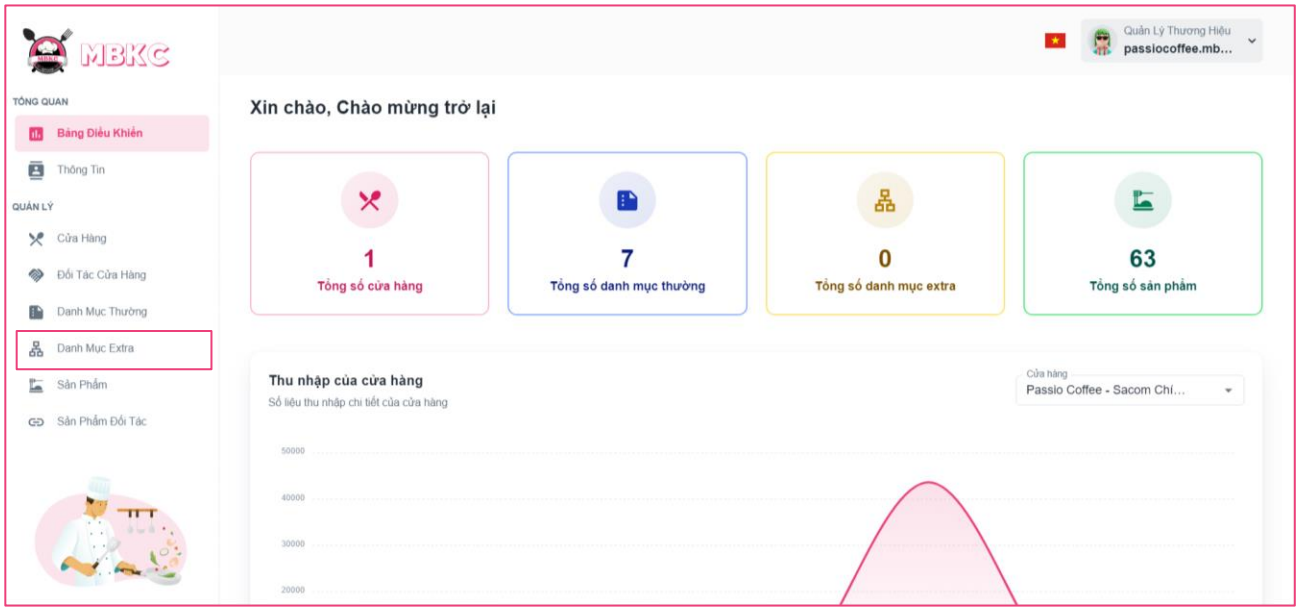


Figure 324: User Manual - View list of Extra Categories – Step 3

Step 4: Search, reload data, and show or hide columns in the table list of extra categories.

Case 1: Fill in the extra category’s name in the search box to search extra categories in the table list.

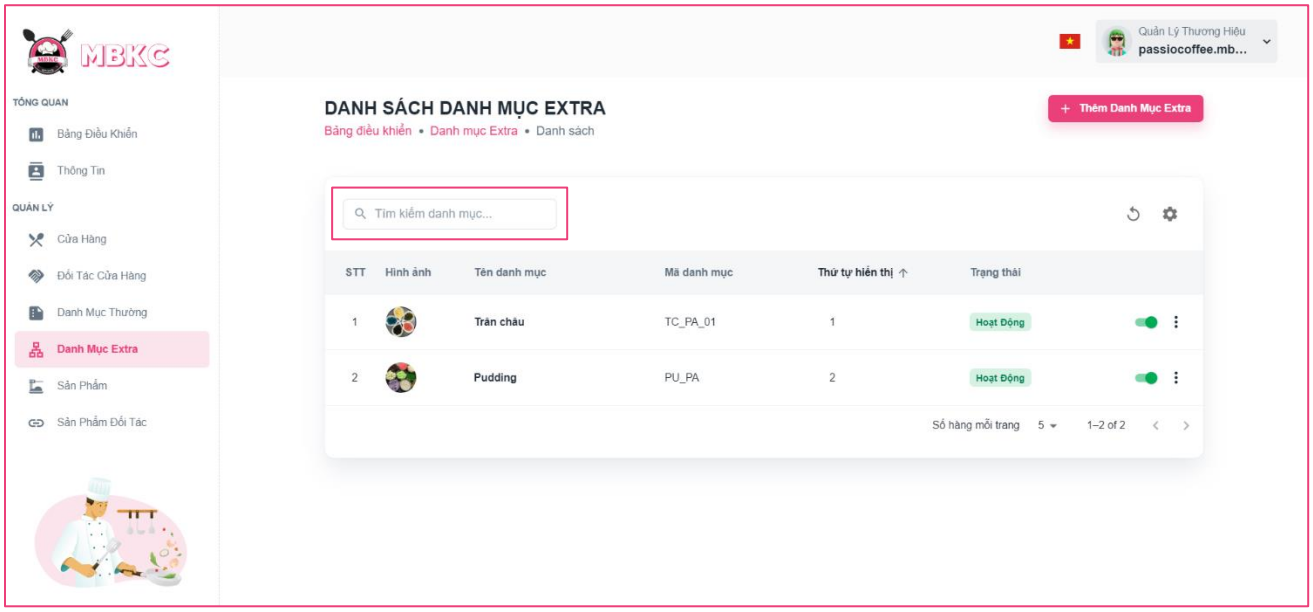


Figure 325: User Manual - View list of Extra Categories – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of extra categories.

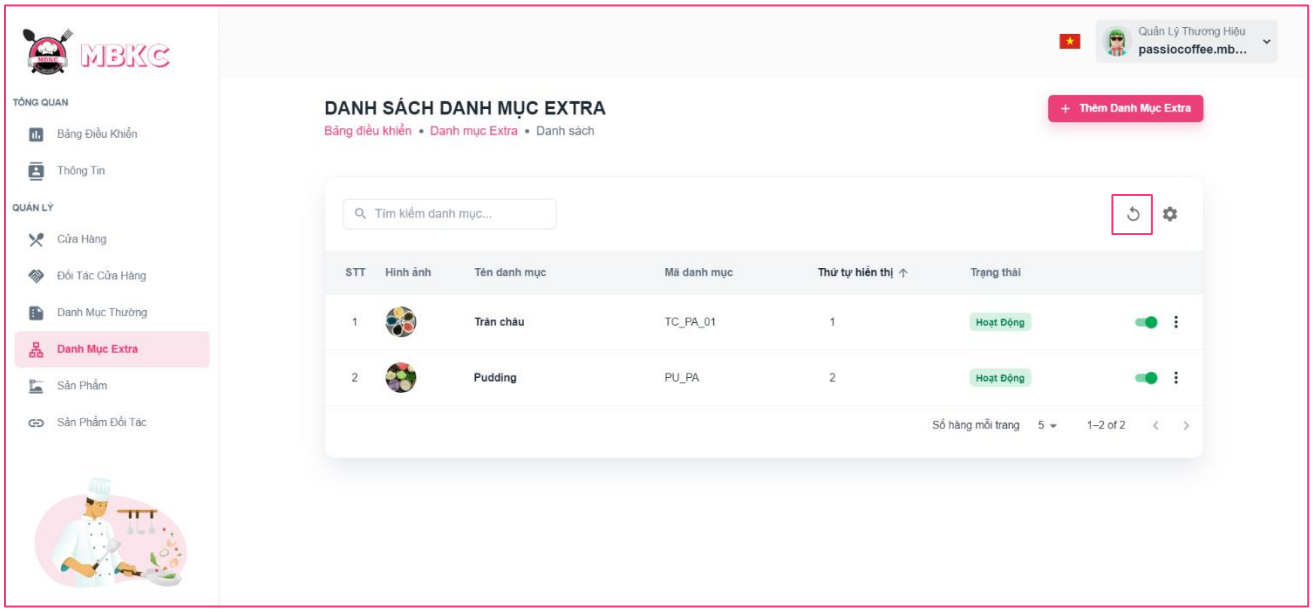


Figure 326: User Manual - View list of Extra Categories – Step 4 – Case 2

Case 3: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of extra categories.

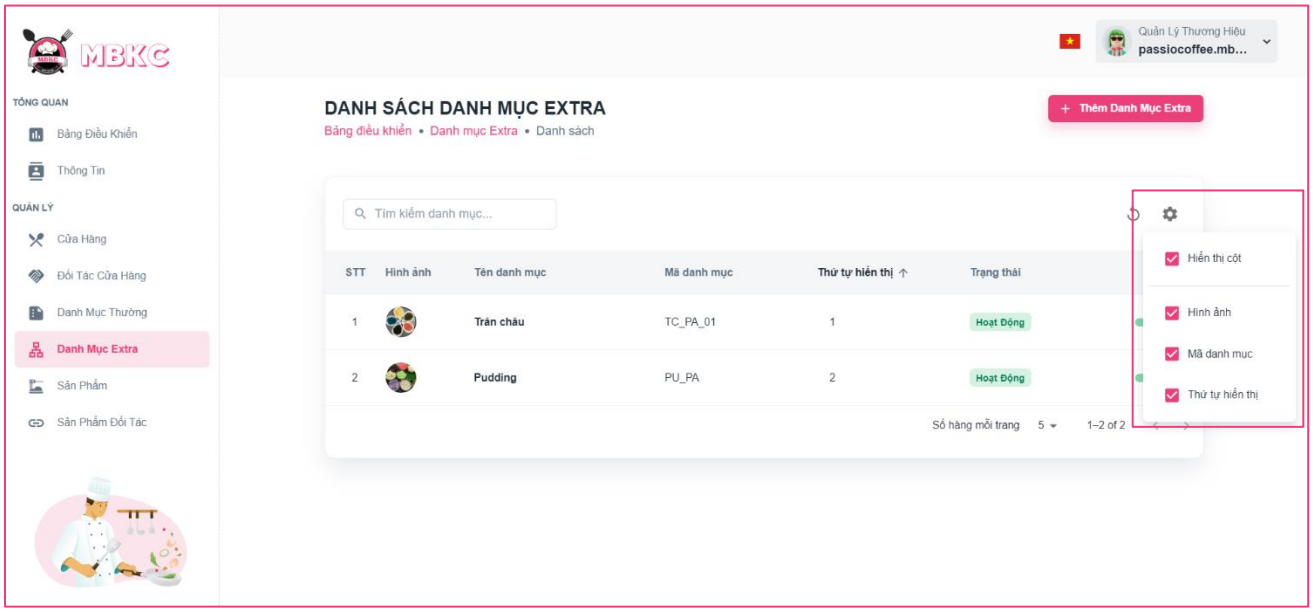


Figure 327: User Manual - View list of Extra Categories - Step 4 - Case 3

3.2.7.3. View details Extra Category

3.2.7.3.1. Description

Use this guideline to view details extra category.

This web application supports view details extra category with account as a Brand Manager.

3.2.7.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

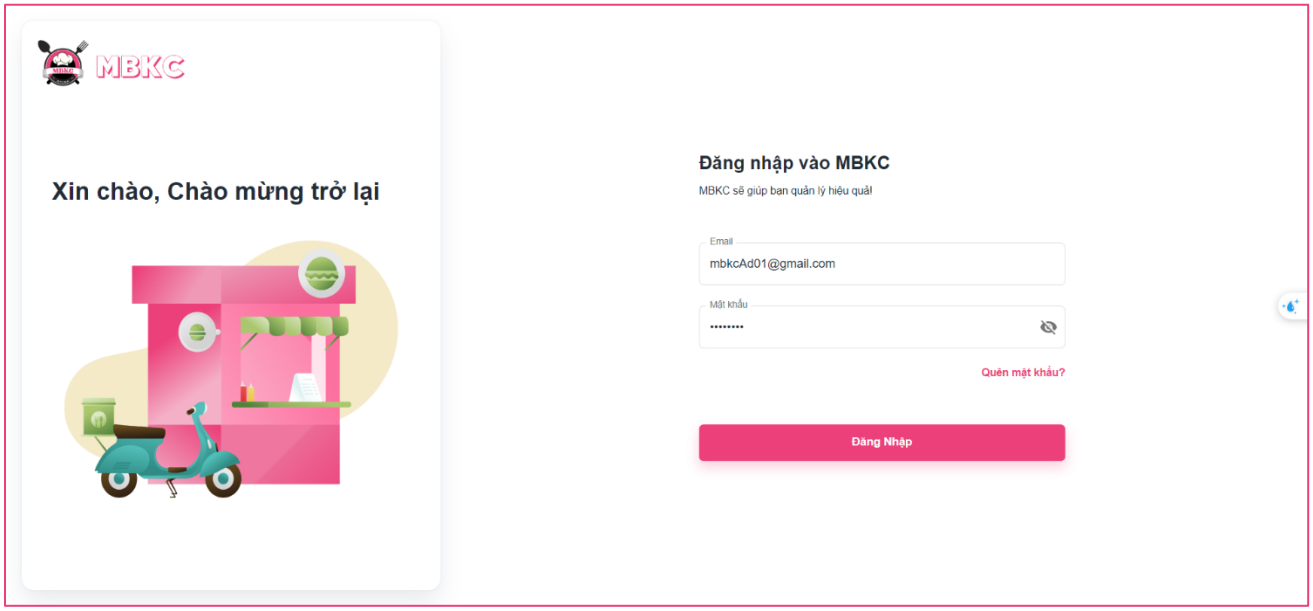


Figure 328: User Manual - View details Extra Category – Step 1

Step 2: Click “Đăng Nhập” button.

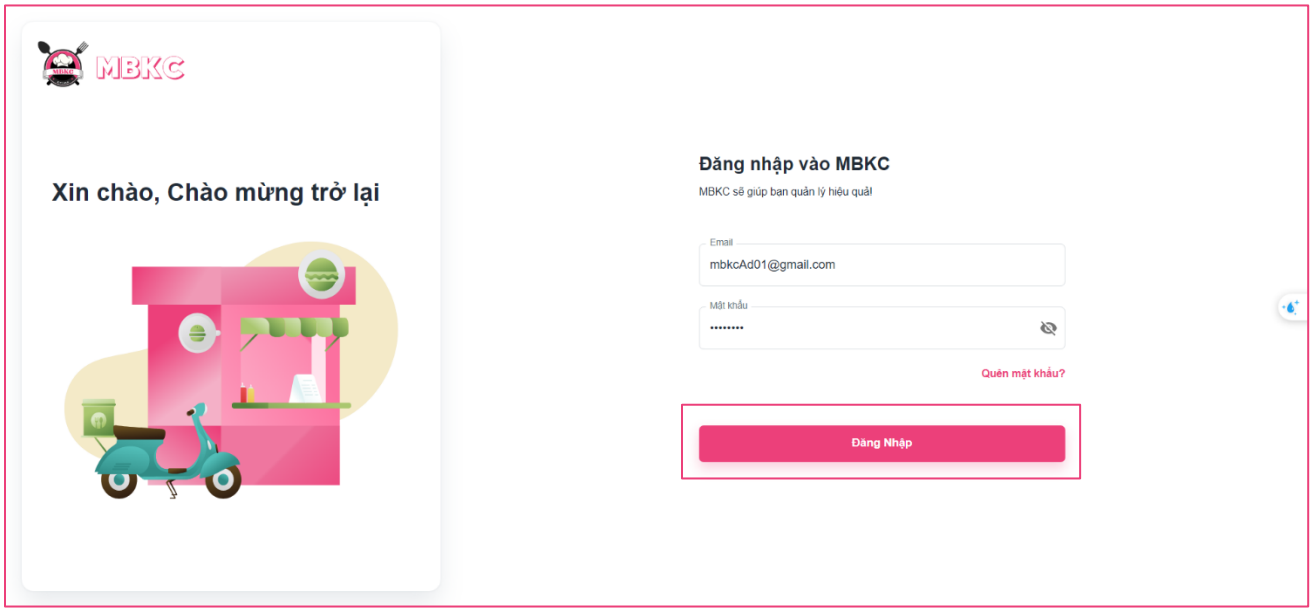


Figure 329: User Manual - View details Extra Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Extra” from the left sidebar.

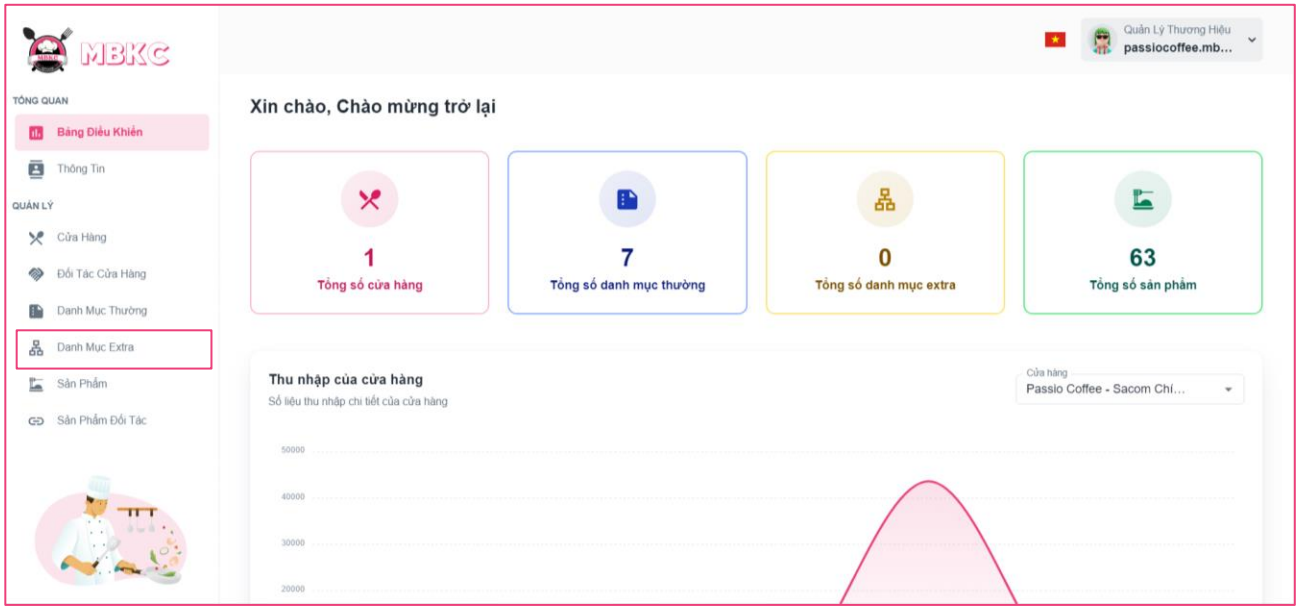


Figure 330: User Manual - View details Extra Category – Step 3

Step 4: Click the extra category row to navigate to Extra Category Details Page.

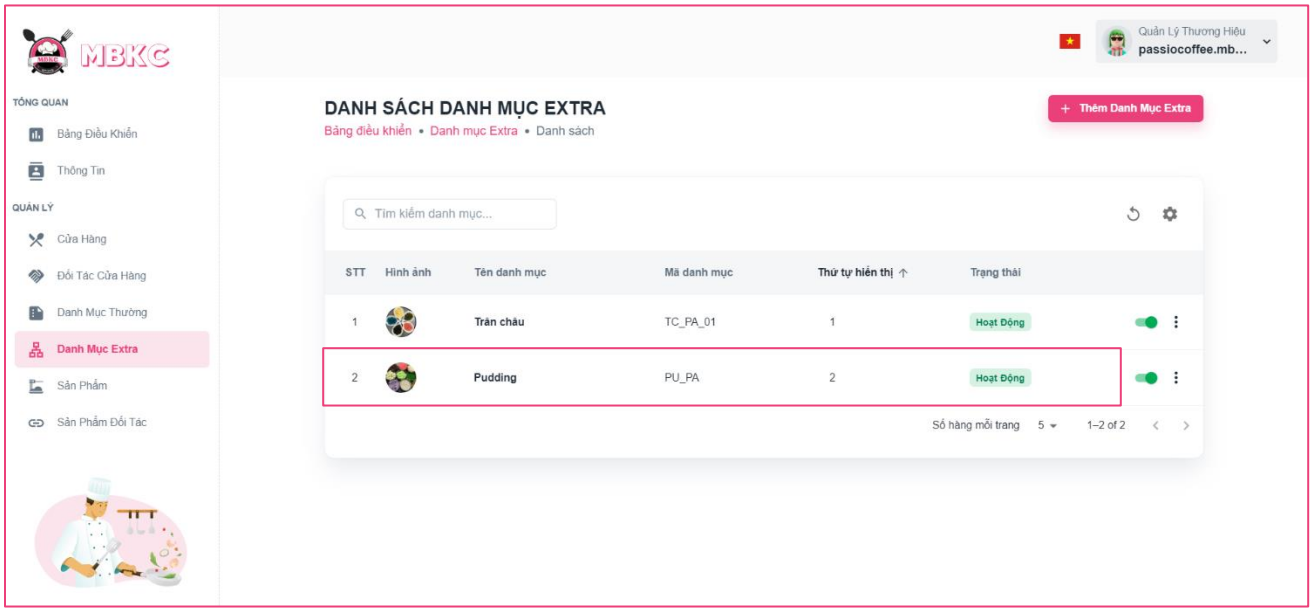


Figure 331: User Manual - View details Extra Category – Step 4

Step 5: Navigate to Extra Category Details Page to show information details of extra category.

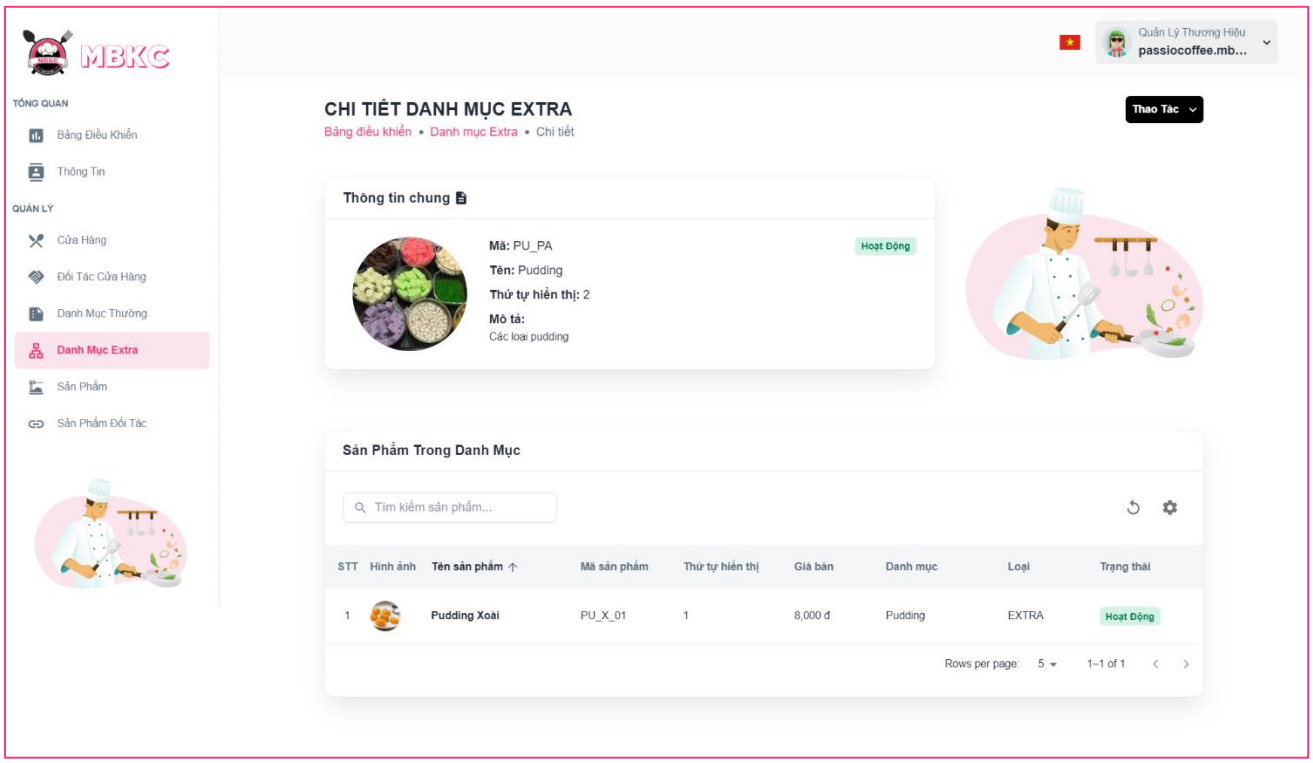


Figure 332: User Manual - View details Extra Category – Step 5

3.2.7.4. Update Status Extra Category

3.2.7.4.1. Description

Use this guideline to update the status of the extra category.

This web application supports update the status of the extra category with account as a Brand Manager.

3.2.7.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

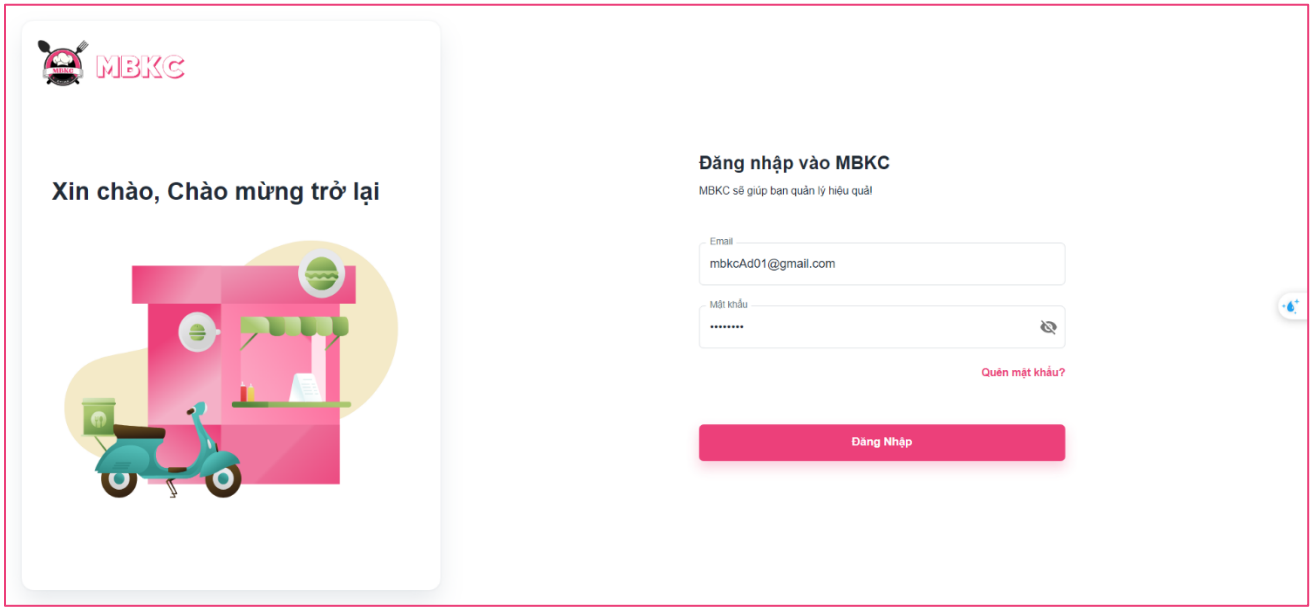


Figure 333: User Manual – Update Status Extra Category – Step 1

Step 2: Click “Đăng Nhập” button.

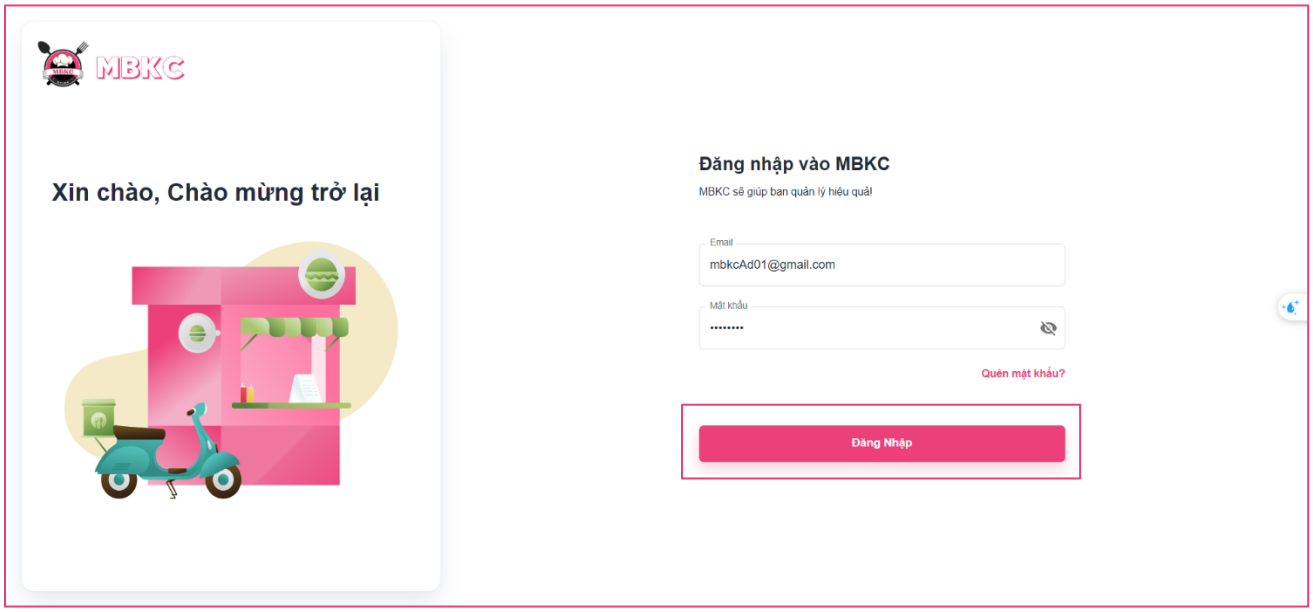


Figure 334: User Manual - Update Status Extra Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Extra” from the left sidebar.

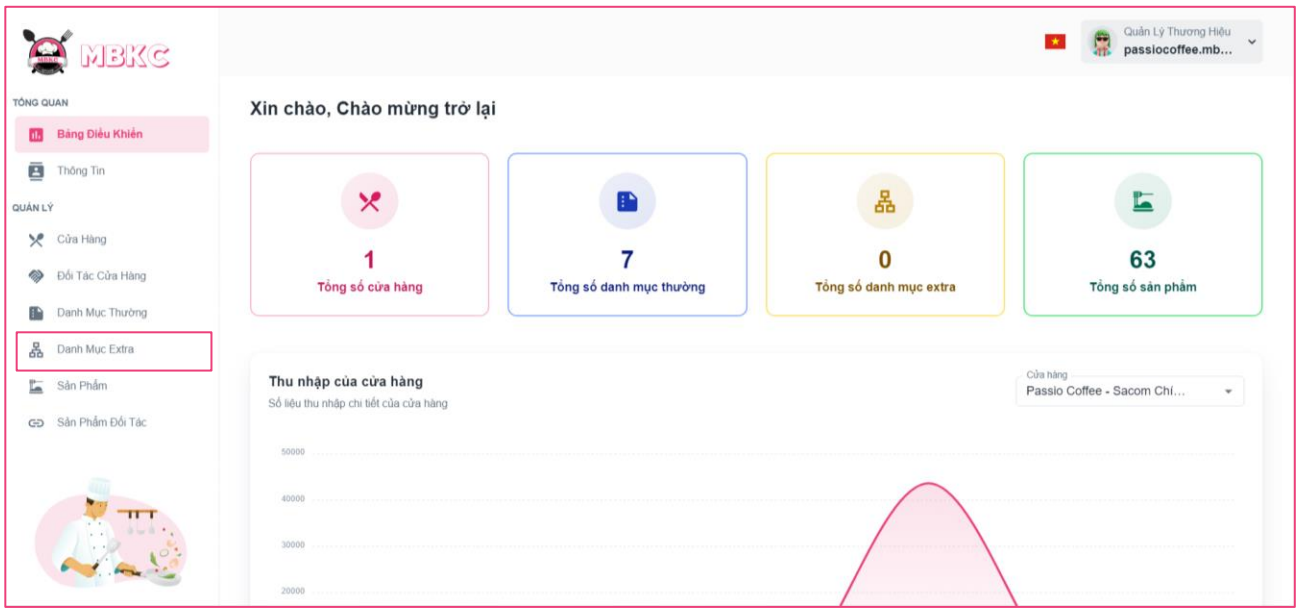


Figure 335: User Manual - Update Status Extra Category – Step 3

Step 4: Click on the switch button on the extra category row to update the status.

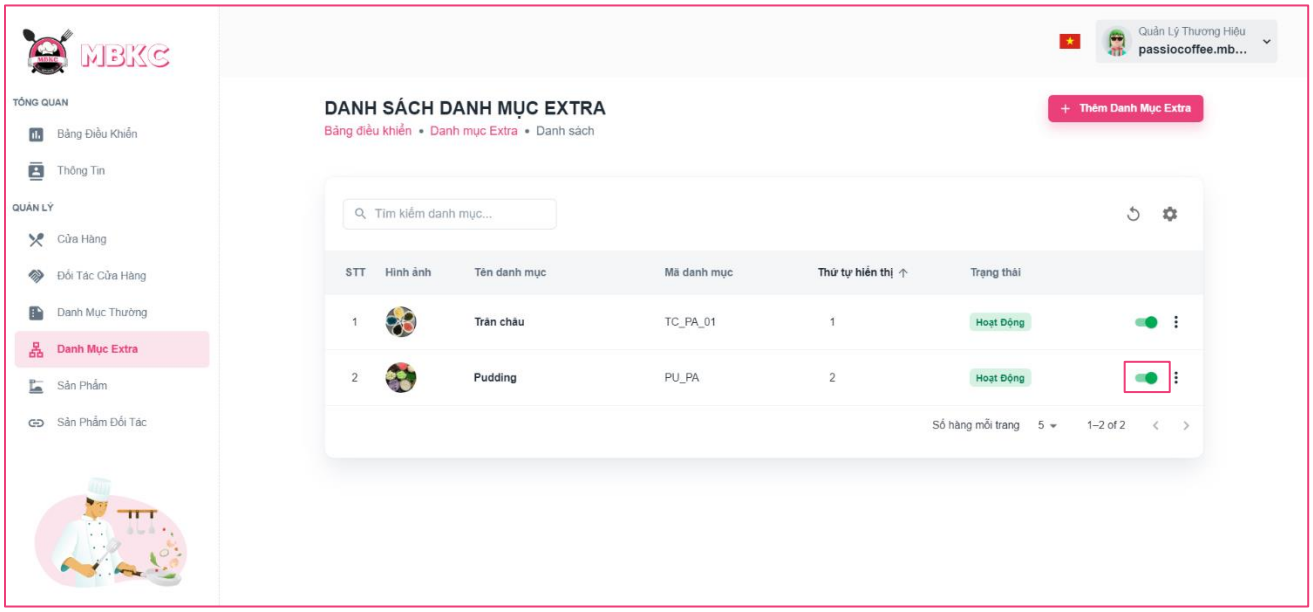


Figure 336: User Manual - Update Status Extra Category – Step 4

Step 5: List of extra categories will be reloaded and a toast message "Đã cập nhật danh mục thành công". And the status of the extra category has been updated.

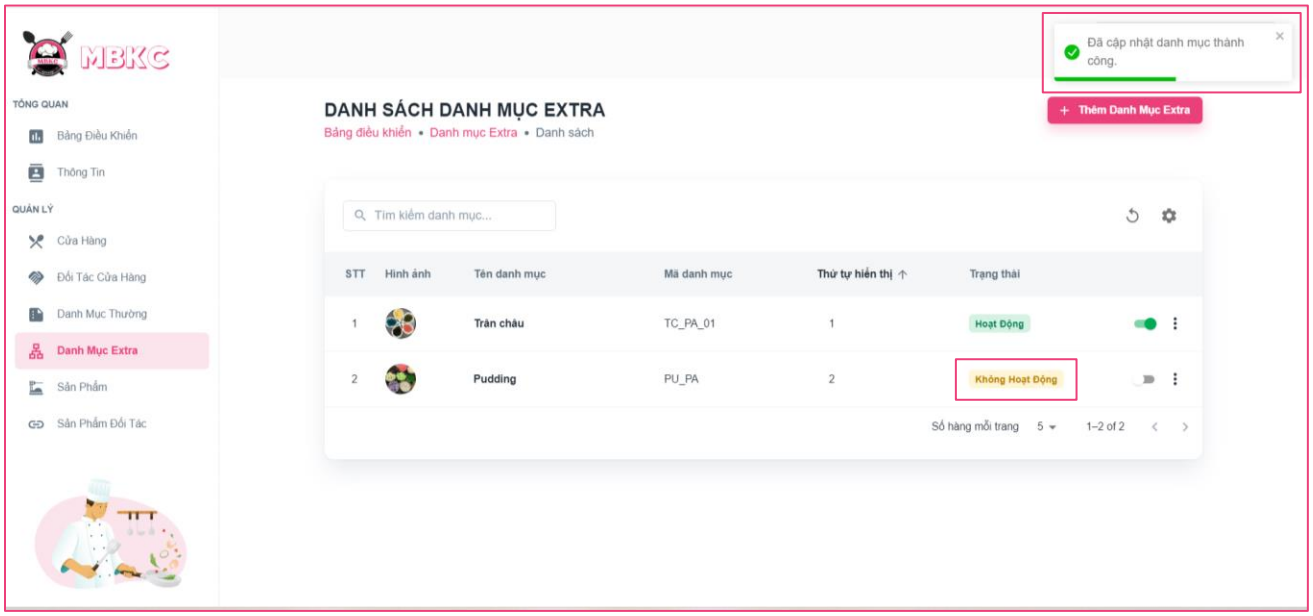


Figure 337: User Manual - Update Status Extra Category – Step 5

3.2.7.5. Update Information Extra Category

3.2.7.5.1. Description

Use this guideline to update information of the extra category.

This web application supports update information of the extra category with account as a Brand Manager.

3.2.7.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

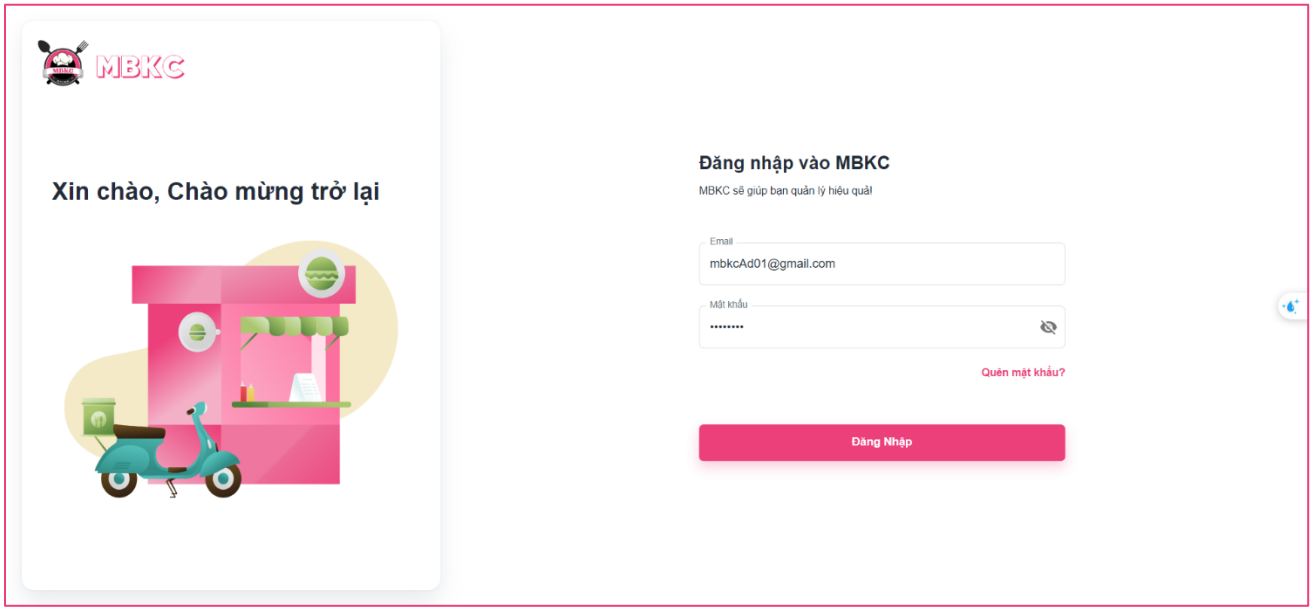


Figure 338: User Manual – Update Information Extra Category – Step 1

Step 2: Click “Đăng Nhập” button.

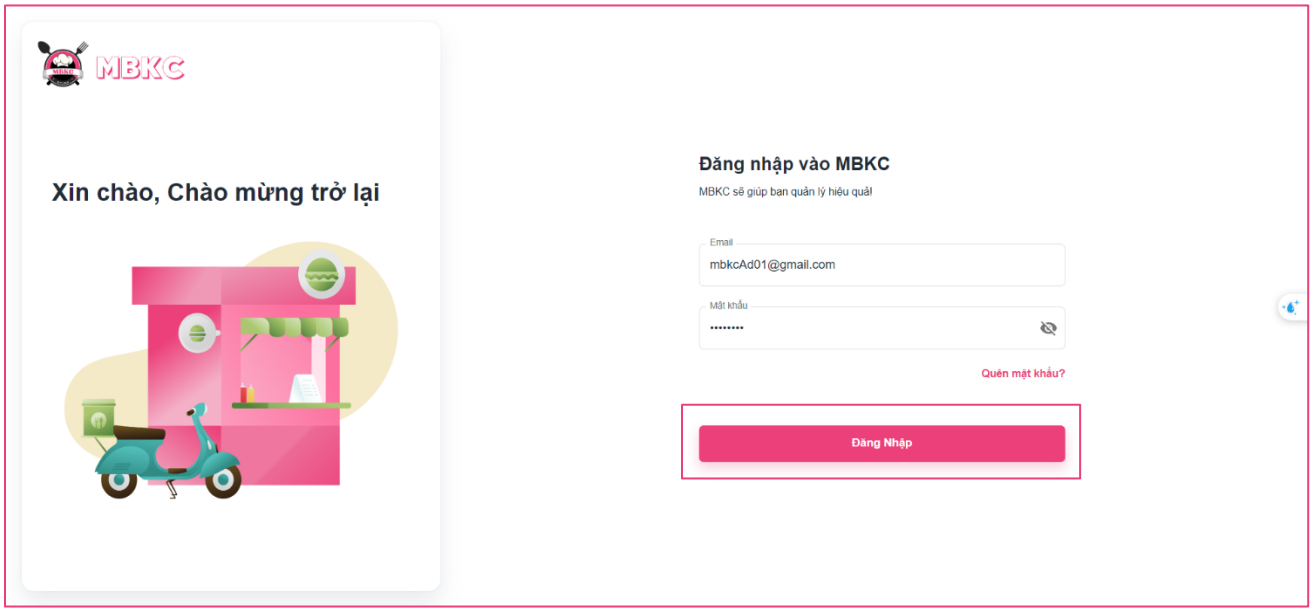


Figure 339: User Manual - Update Information Extra Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Extra” from the left sidebar.

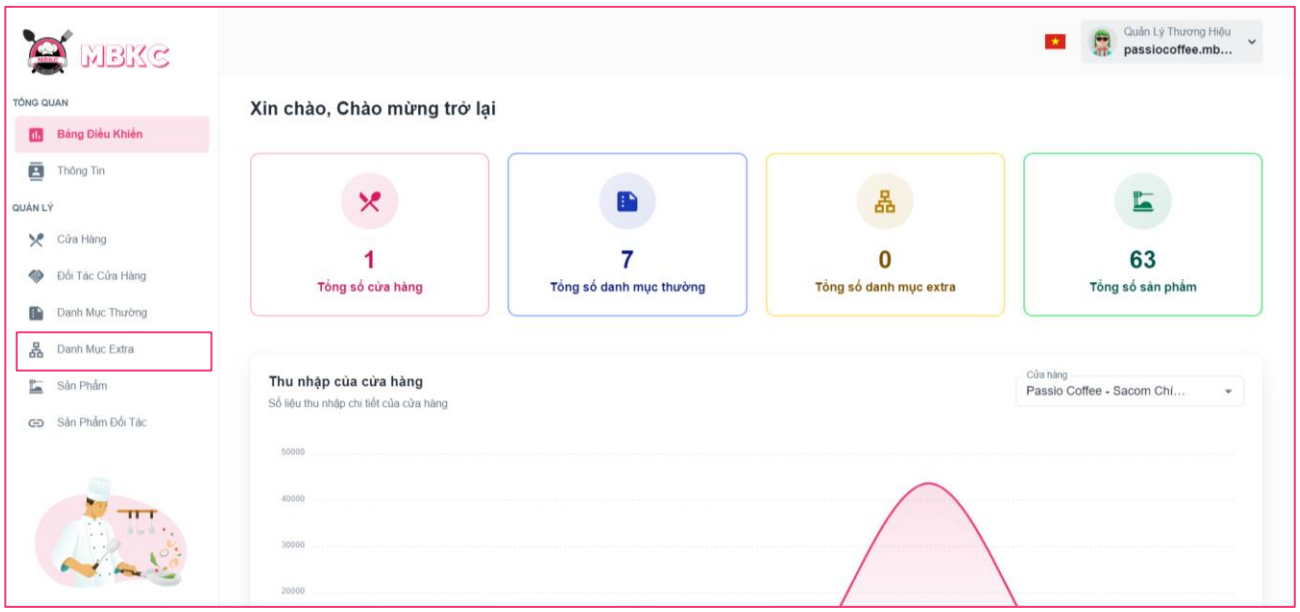


Figure 340: User Manual - Update Information Extra Category – Step 3

Step 4: Update information of the extra category.

Case 1: Update information of the extra category in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the extra category row want to navigate Update Extra Category Page.

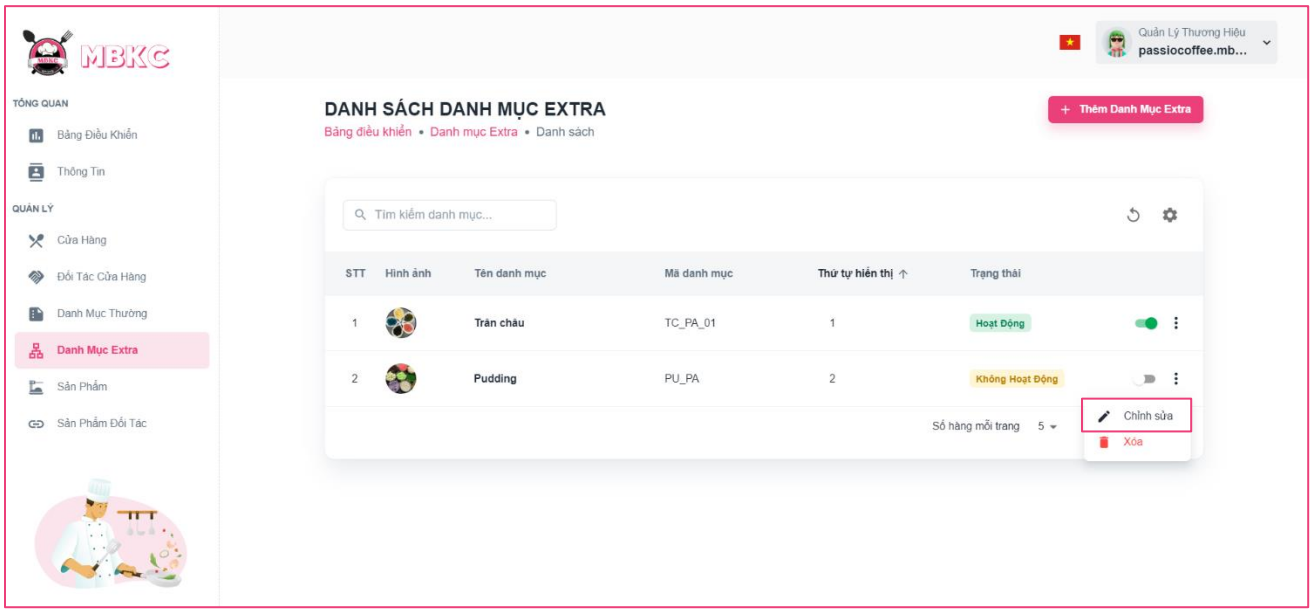


Figure 341: User Manual - Update Information Extra Category – Step 4 – Case 1

Case 2: Update information of the extra category on the details page.

1: Click the extra category row to navigate to Extra Category Details Page.

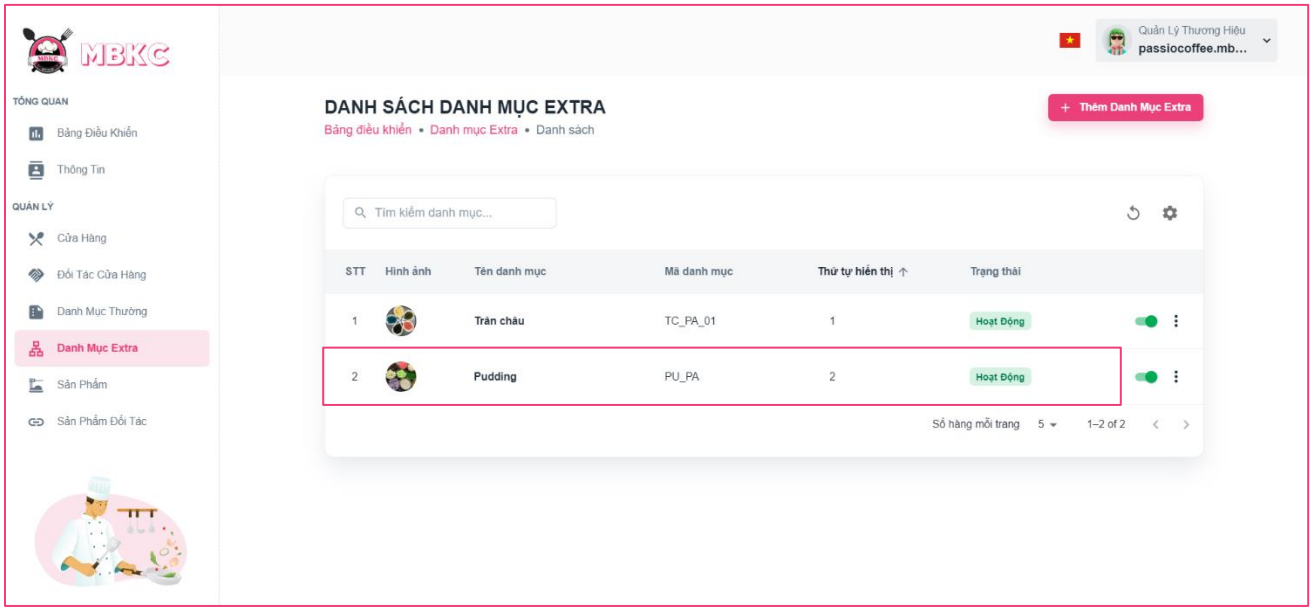


Figure 342: User Manual - Update Information Extra Category – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Chỉnh sửa” menu item to navigate Update Extra Category Page.

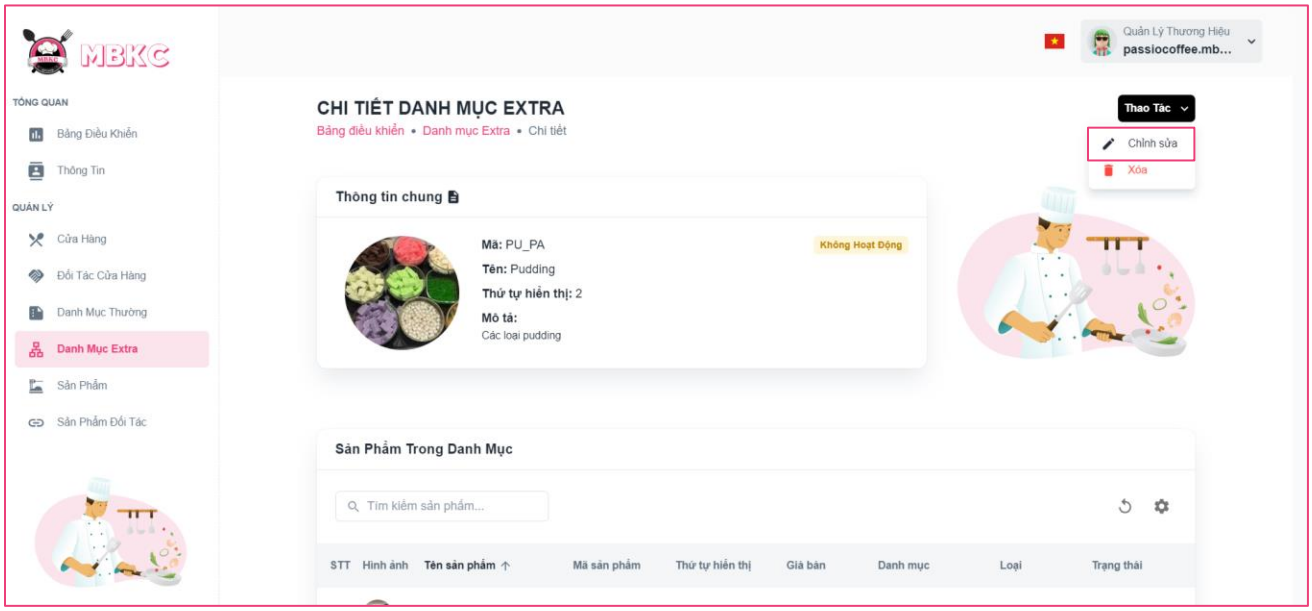


Figure 343: User Manual - Update Information Extra Category – Step 4 – Case 2 – 2

Step 5: Extra category information will be automatically filled in the form. Then edit the information as desired.

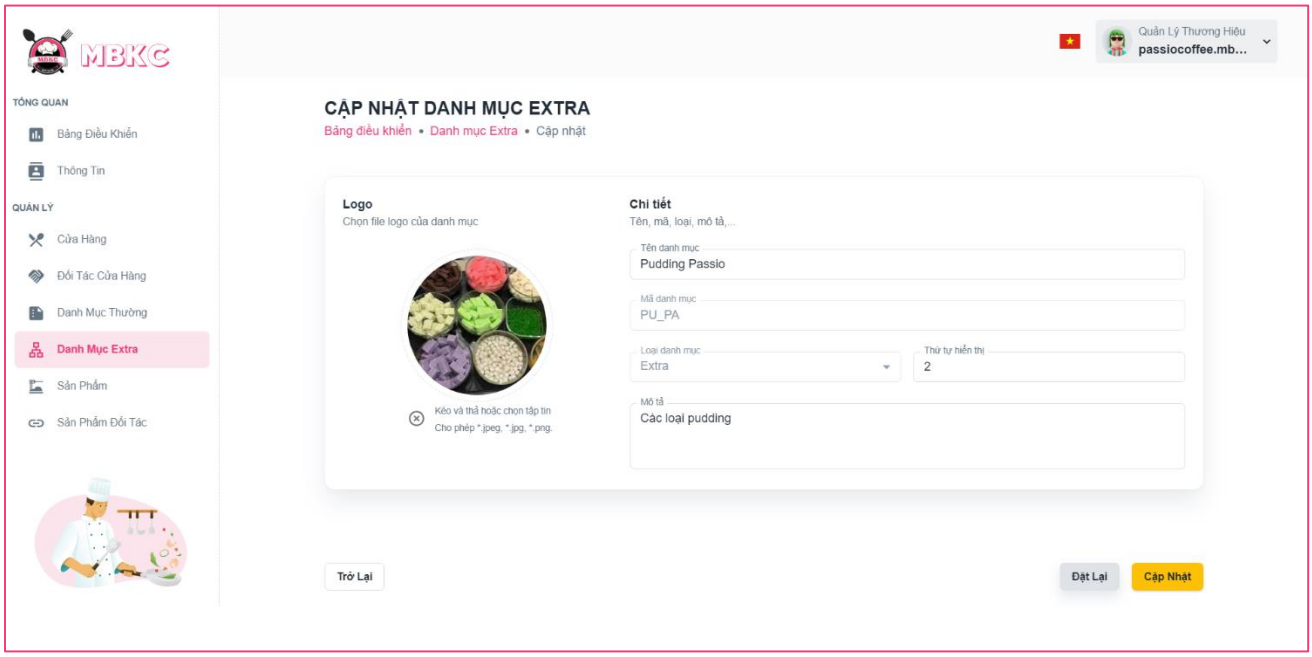


Figure 344: User Manual - Update Information Extra Category – Step 5

Step 6: Click “Cập nhật” button.

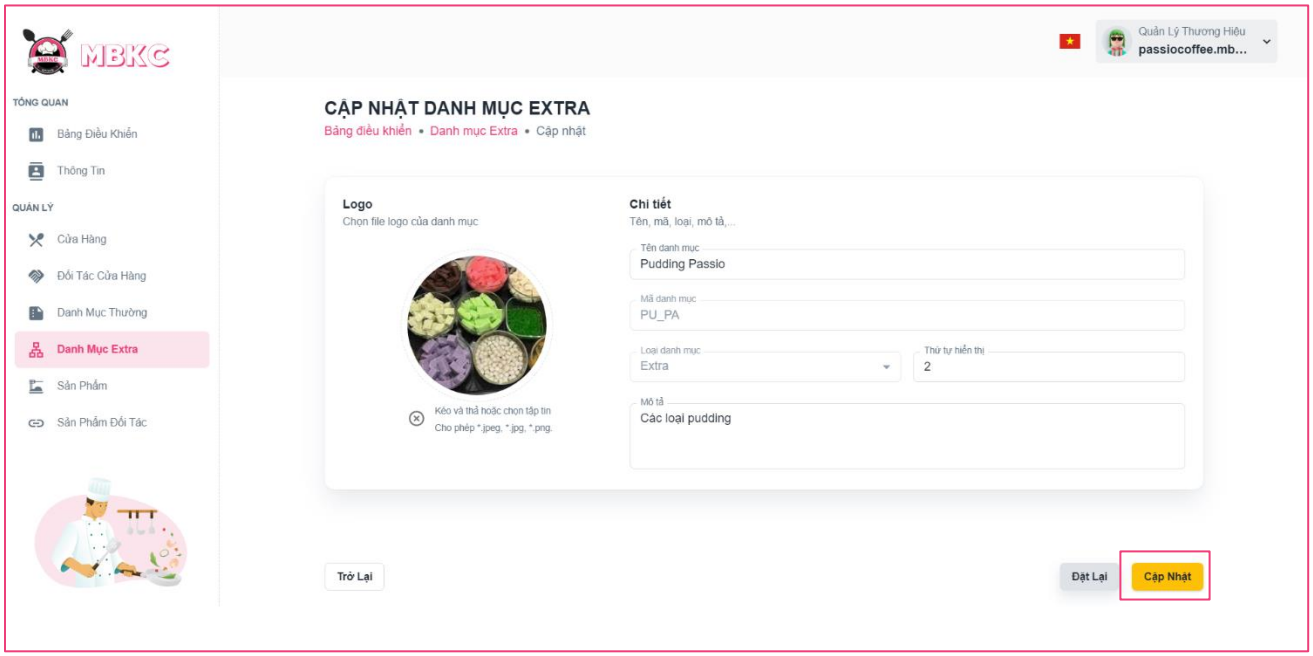


Figure 345: User Manual - Update Information Extra Category – Step 6

Step 7:
Case 1: You will be navigated to Extra Category Details Page showing the extra category details when start edit information of extra category from Extra Category Details Page and a toast message “Cập nhật thương hiệu thành công”.

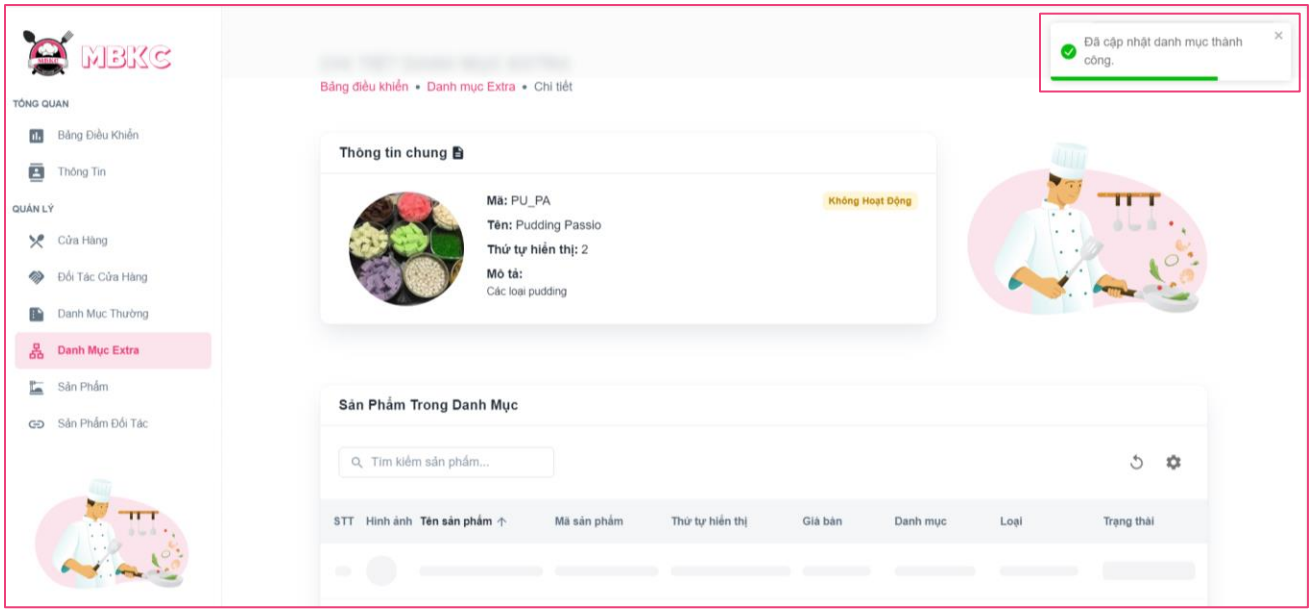


Figure 346: User Manual - Update Information Extra Category – Step 7 – Case 1

Case 2: You will be navigated to List of Extra Categories Page showing the list of extra categories when start edit information of extra category from List of Extra Categories Page and a toast message “Đã cập nhật danh mục thành công”.

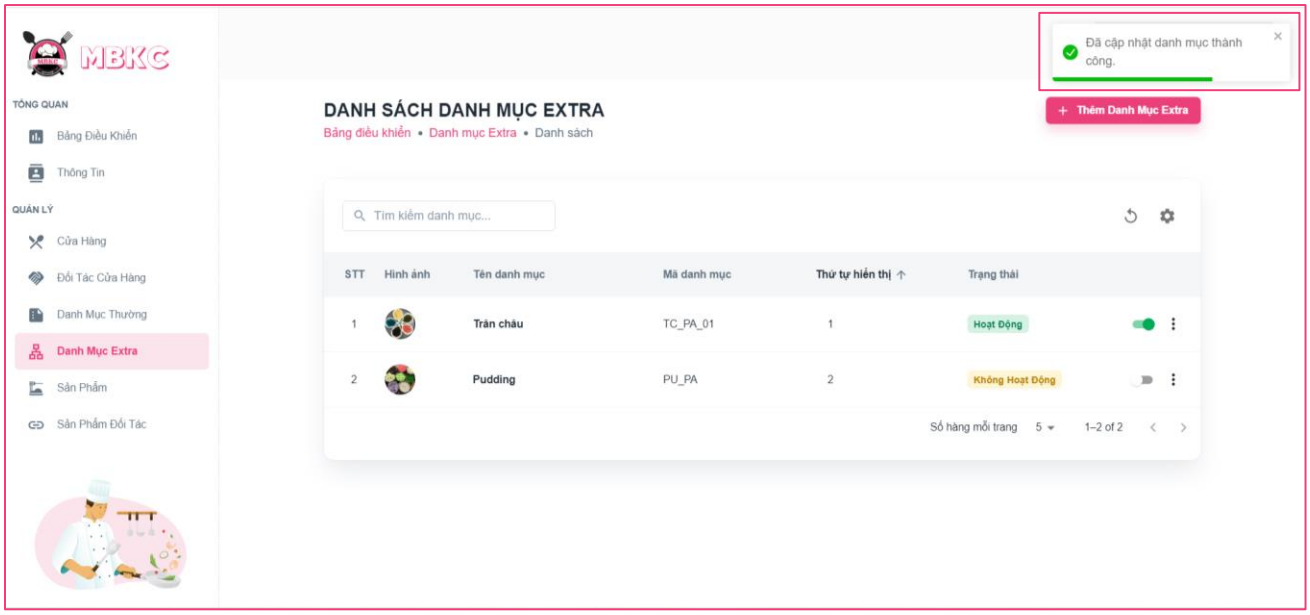


Figure 347: User Manual - Update Information Extra Category – Step 7 – Case 2

3.2.7.6. Delete Extra Category

3.2.7.6.1. Description
Use this guideline to delete extra category.

This web application supports delete extra category with account as a Brand Manager.

3.2.7.6.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

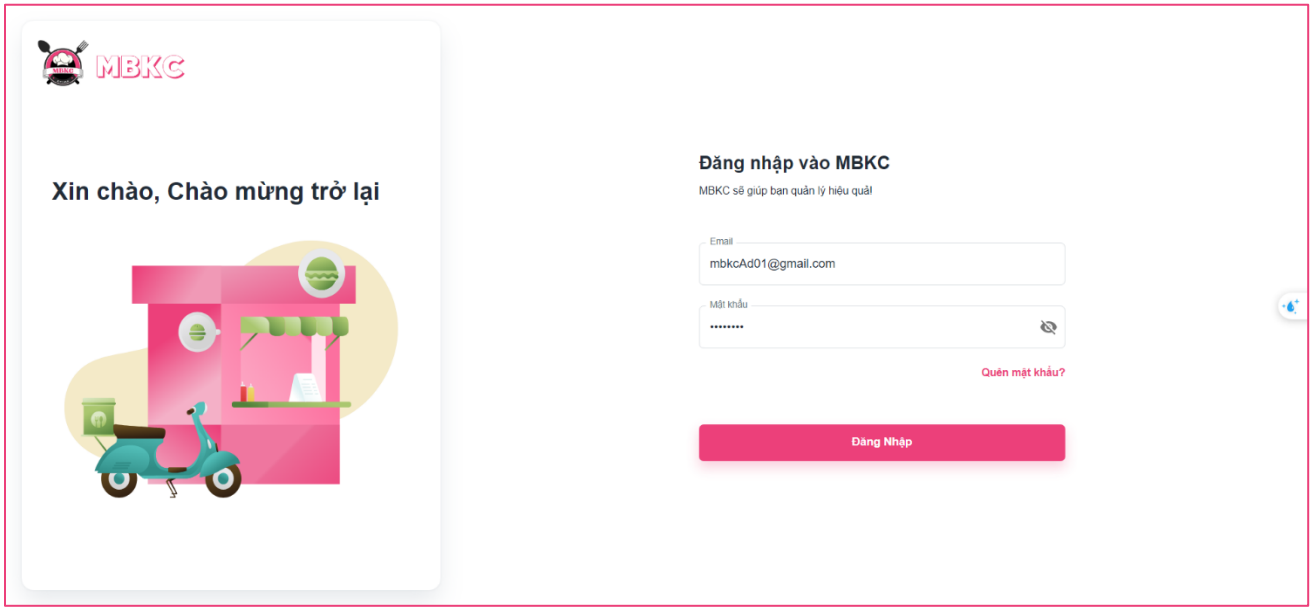


Figure 348: User Manual – Delete Extra Category – Step 1

Step 2: Click “Đăng Nhập” button.

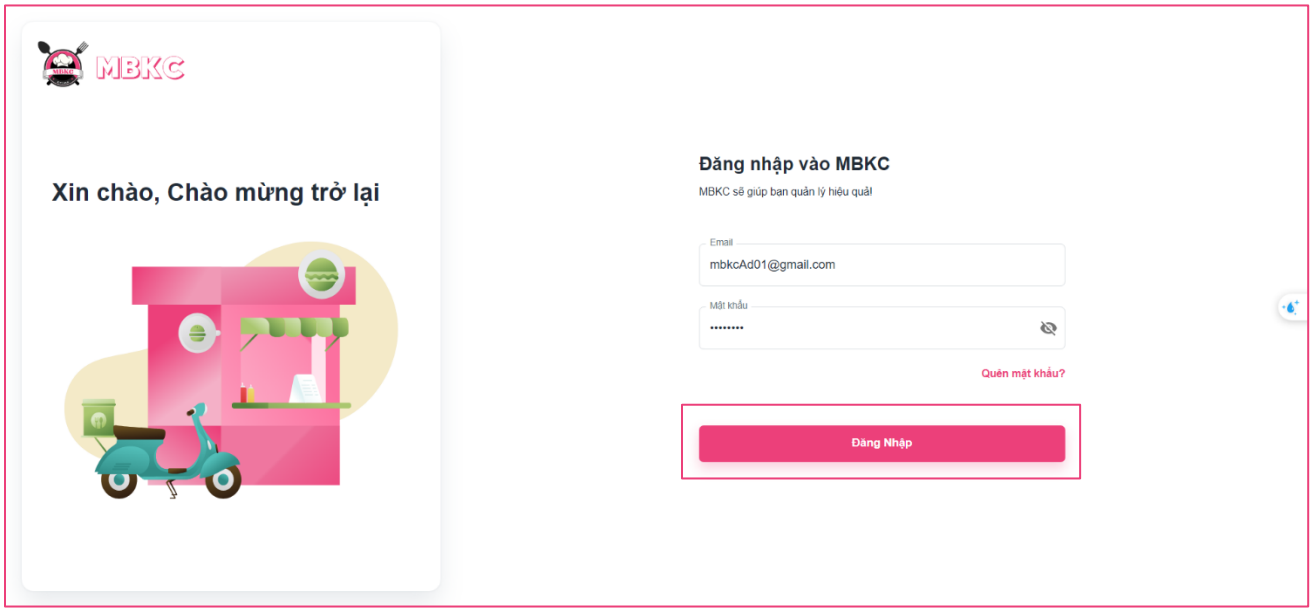


Figure 349: User Manual - Delete Extra Category – Step 2

Step 3: From Brand Dashboard Page, choose “Danh Mục Extra” from the left sidebar.

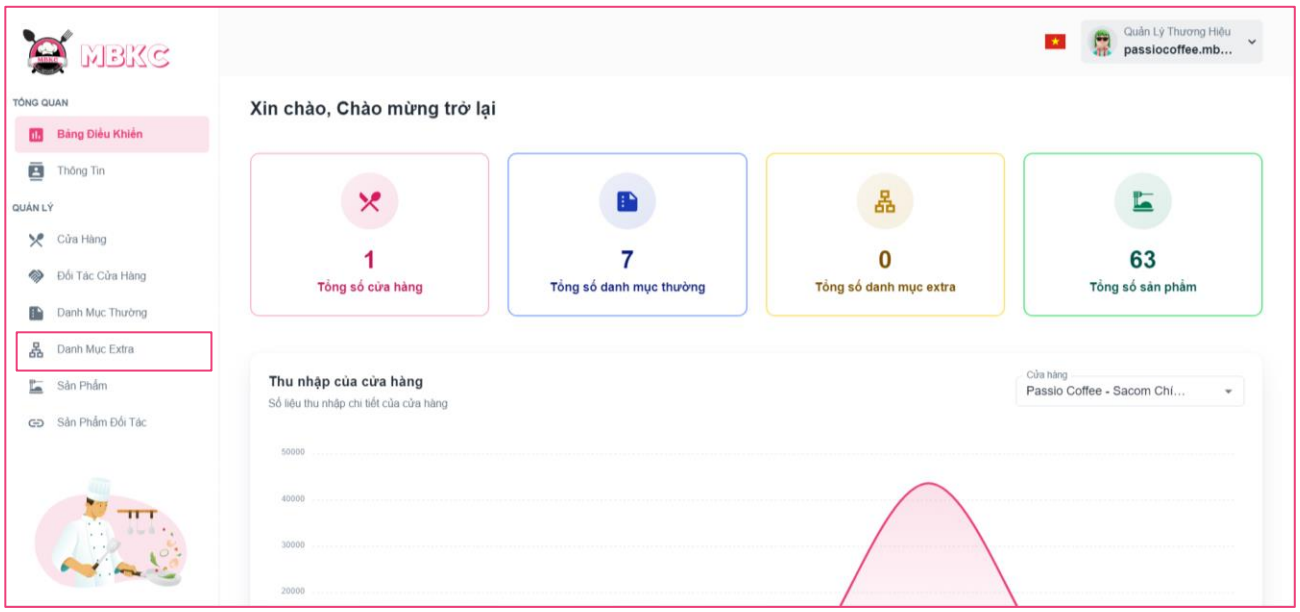


Figure 350: User Manual - Delete Extra Category – Step 3

Step 4: Delete extra category.

Case 1: Delete extra category in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the extra category row to delete.

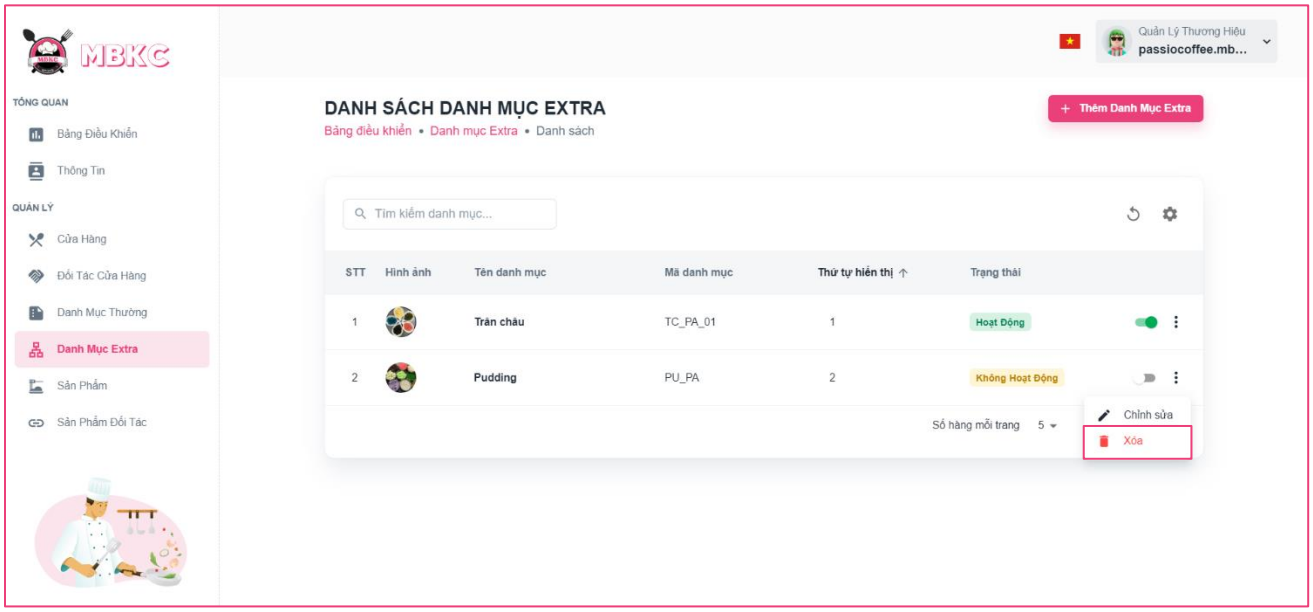


Figure 351: User Manual - Delete Extra Category – Step 4 – Case 1

Case 2: Delete extra category on the details page.

1: Click the extra category row to navigate to Extra Category Details Page.

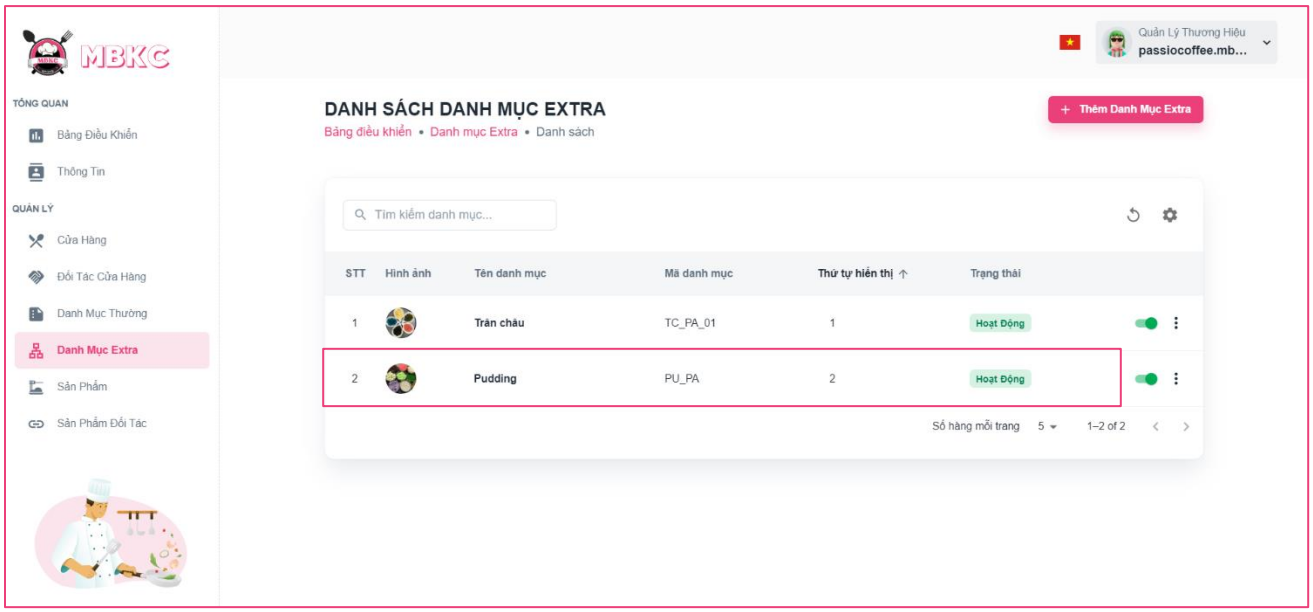


Figure 352: User Manual - Delete Extra Category – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item to delete.

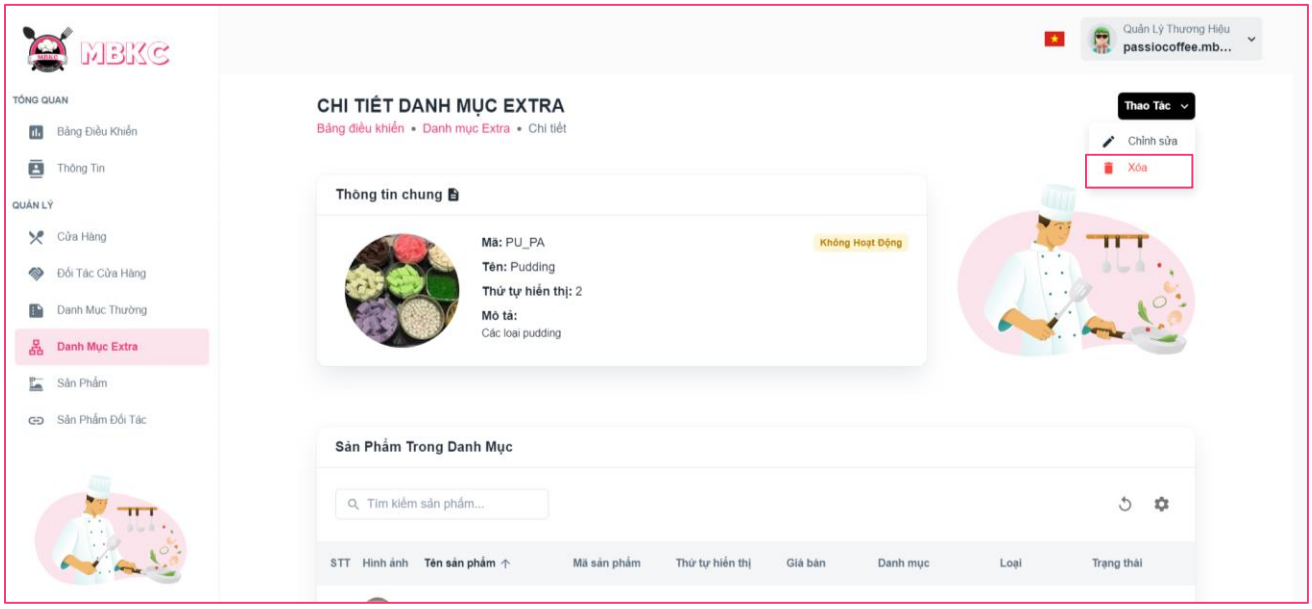


Figure 353: User Manual - Delete Extra Category – Step 4 – Case 2 - 2

Step 5: Click “Xác Nhận” button.

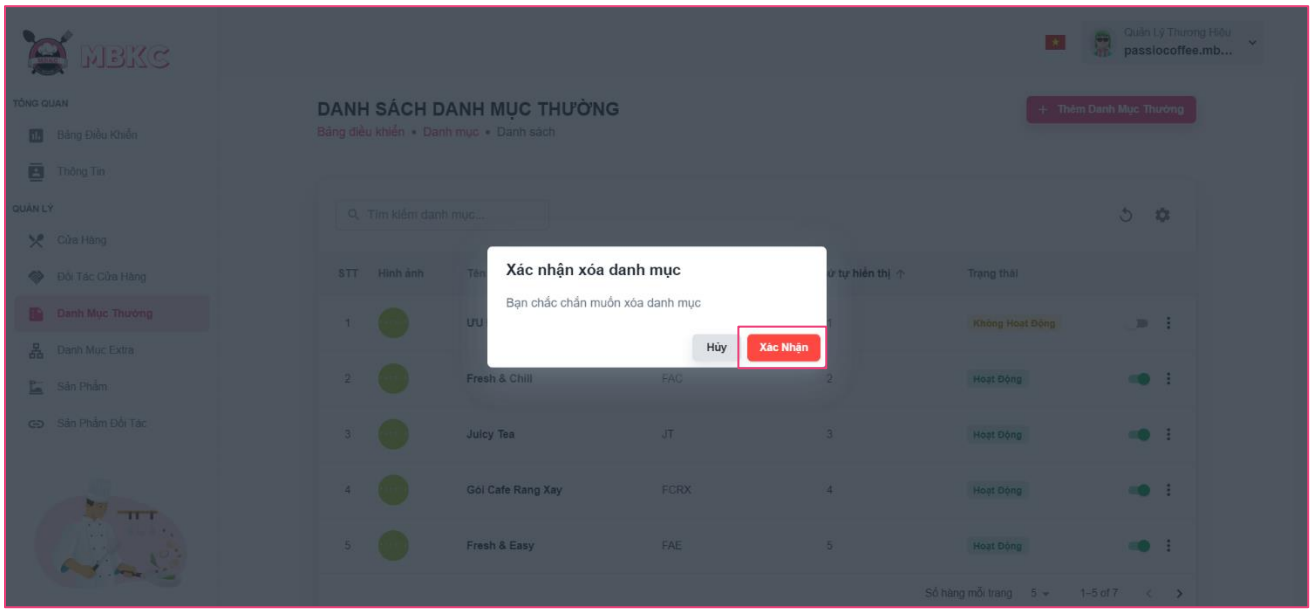


Figure 354: User Manual - Delete Extra Category – Step 5

Step 6: List of extra categories will be reloaded and a toast message “Đã xóa danh mục thành công”. And extra category has been deleted.

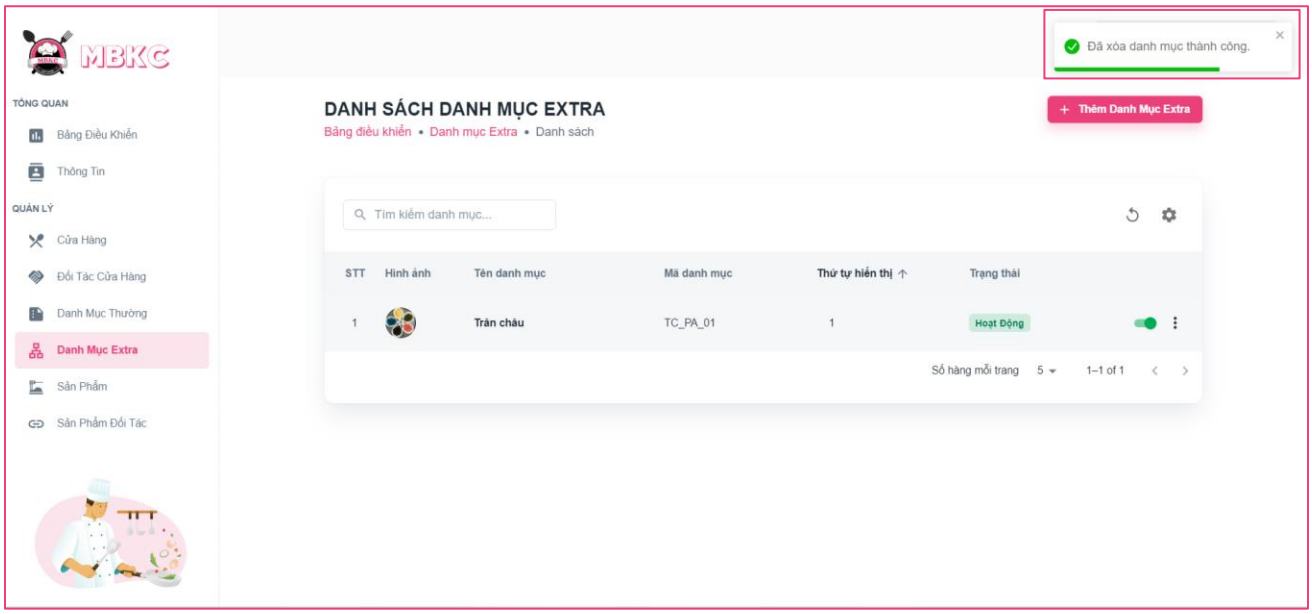


Figure 355: User Manual – Delete Extra Category – Step 6

3.2.8. Product management

3.2.8.1. Create new Product

3.2.8.1.1. Description

Use this guideline to login into MBKC account to create a new product.

This web application supports create a new product with account as a Brand Manager.

3.2.8.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

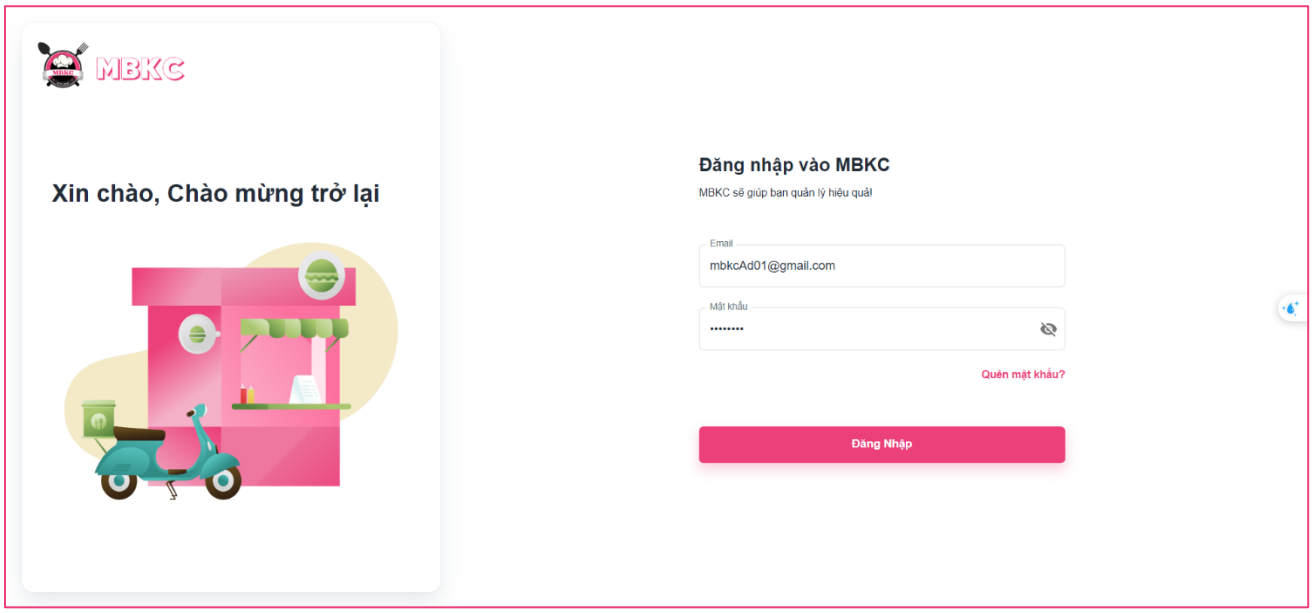


Figure 356: User Manual - Create new Product – Step 1

Step 2: Click “Đăng Nhập” button.

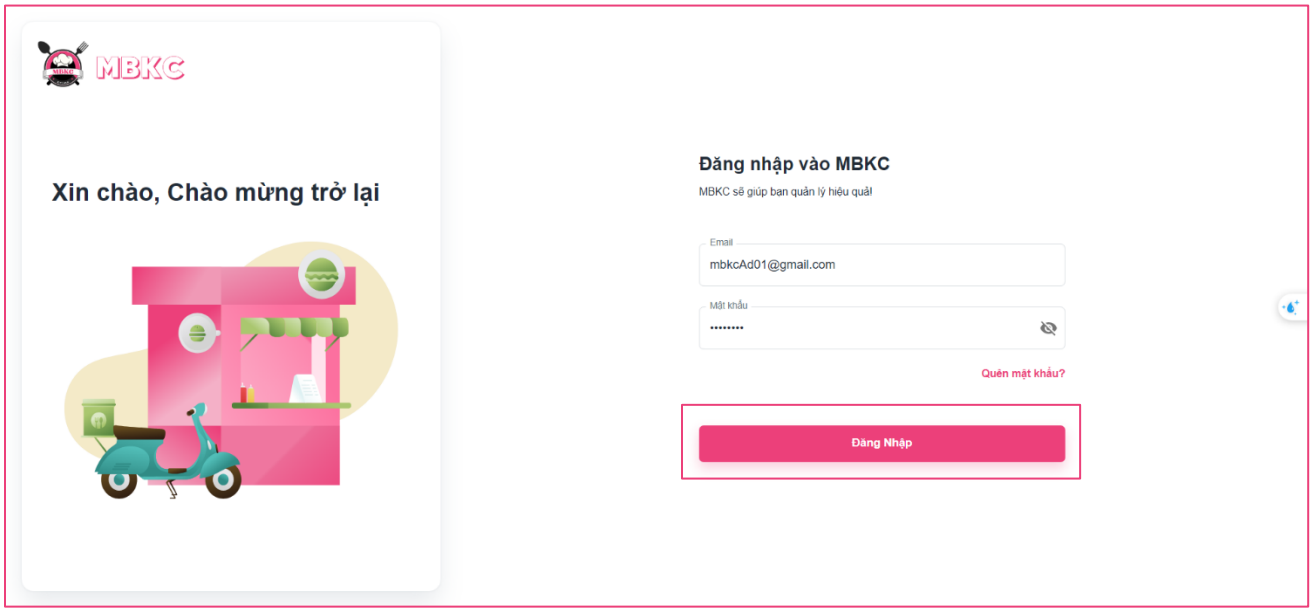


Figure 357: User Manual - Create new Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm” from the left sidebar.

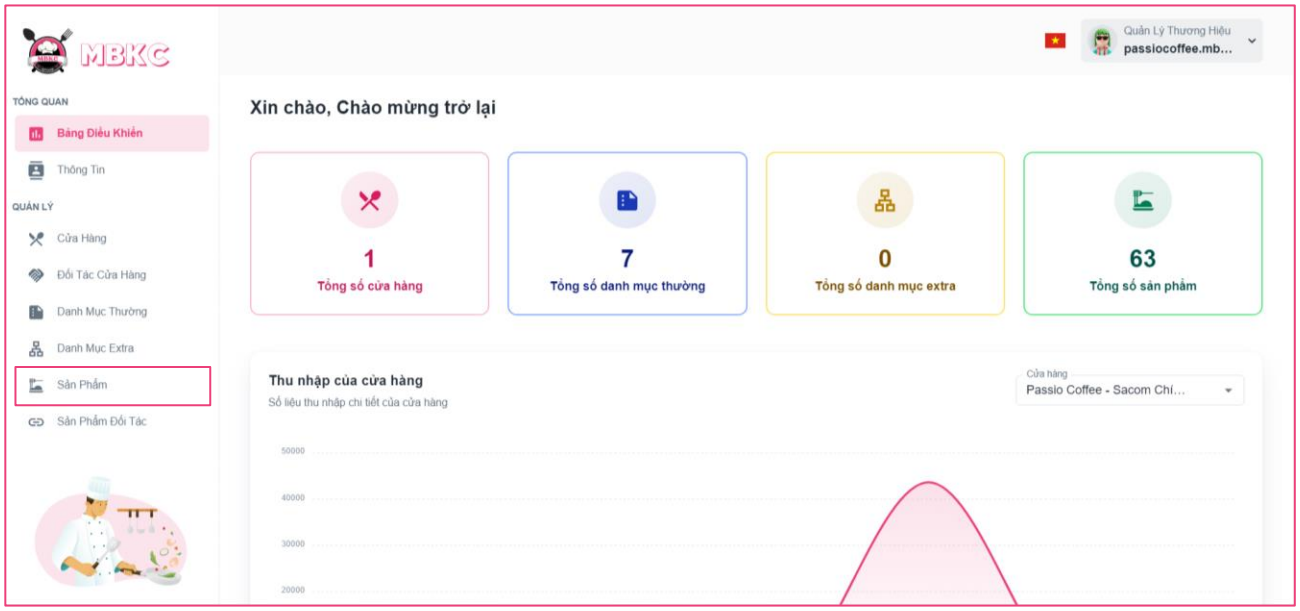


Figure 358: User Manual - Create new Product – Step 3

Step 4: Click “Thêm Sản Phẩm” button on the screen and navigate to Create New Product Page to add a new product.

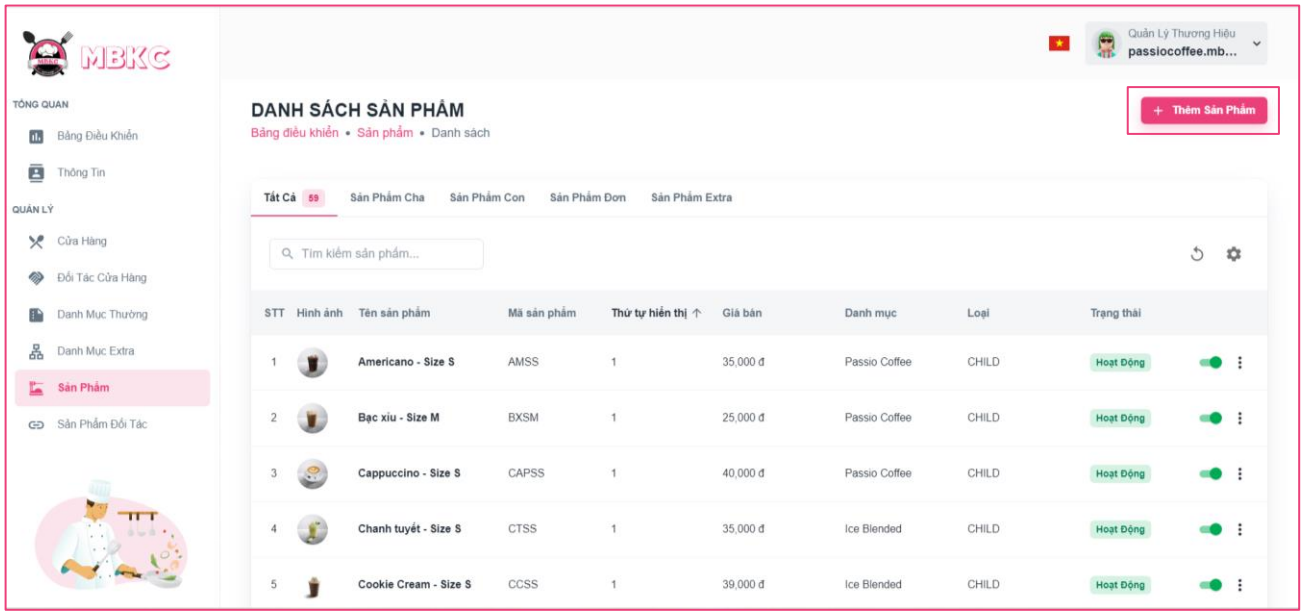


Figure 359: User Manual - Create new Product – Step 4

Step 5: Filling all the required information to add a new product.

Case 1: If you want to create a product with type "Single" – “Sản phẩm đơn” or "Extra" – “Sản phẩm extra”, filling all the required including: “Hình ảnh”, “Loại sản phẩm", “Tên sản phẩm”, “Mã sản phẩm", “Thứ tự hiển thị", “Giá bán”, “Giảm giá”, “Giá gốc”, “Danh mục”, “Mô tả”.

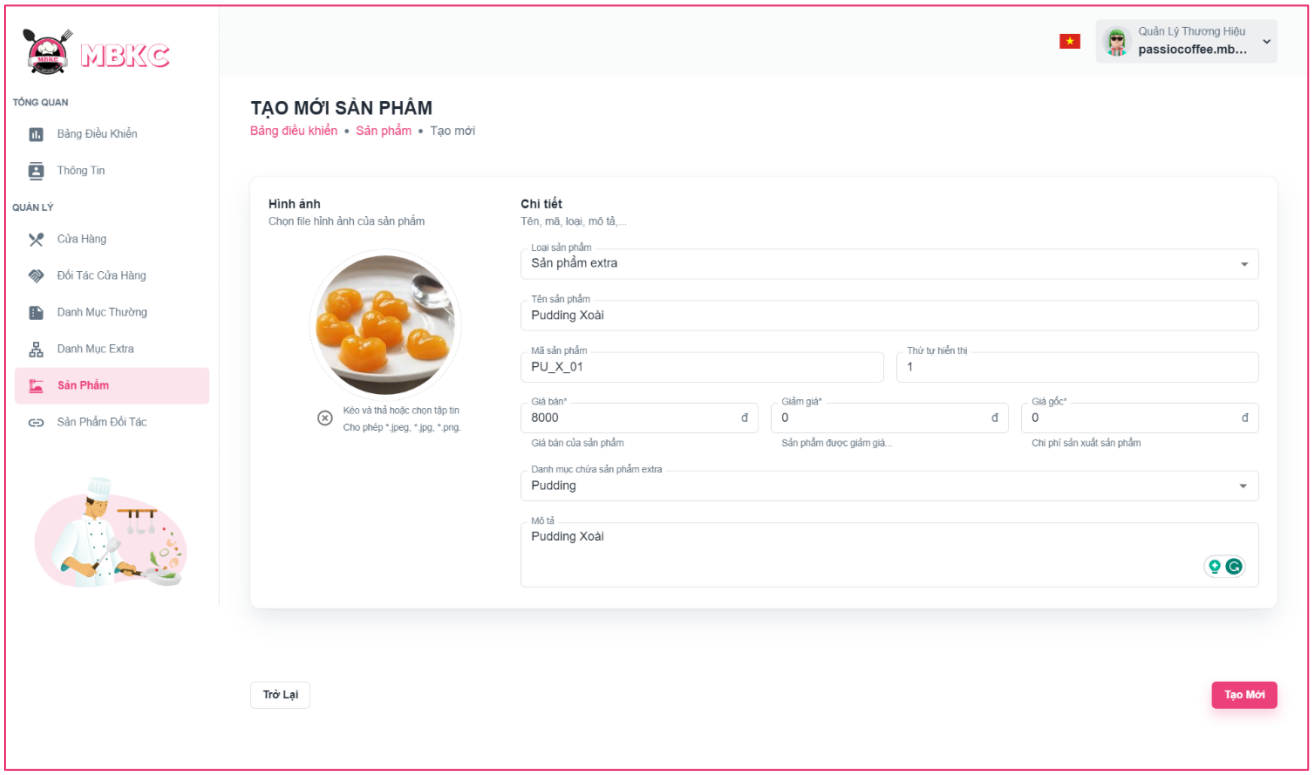


Figure 360: User Manual - Create new Product – Step 5 – Case 1

Case 2: If you want to create a product with type "Parent" – “Sản phẩm cha”, filling all the required including: “Hình ảnh”, “Loại sản phẩm", “Tên sản phẩm”, “Mã sản phẩm", “Thứ tự hiển thị", “Danh mục”, “Mô tả”.

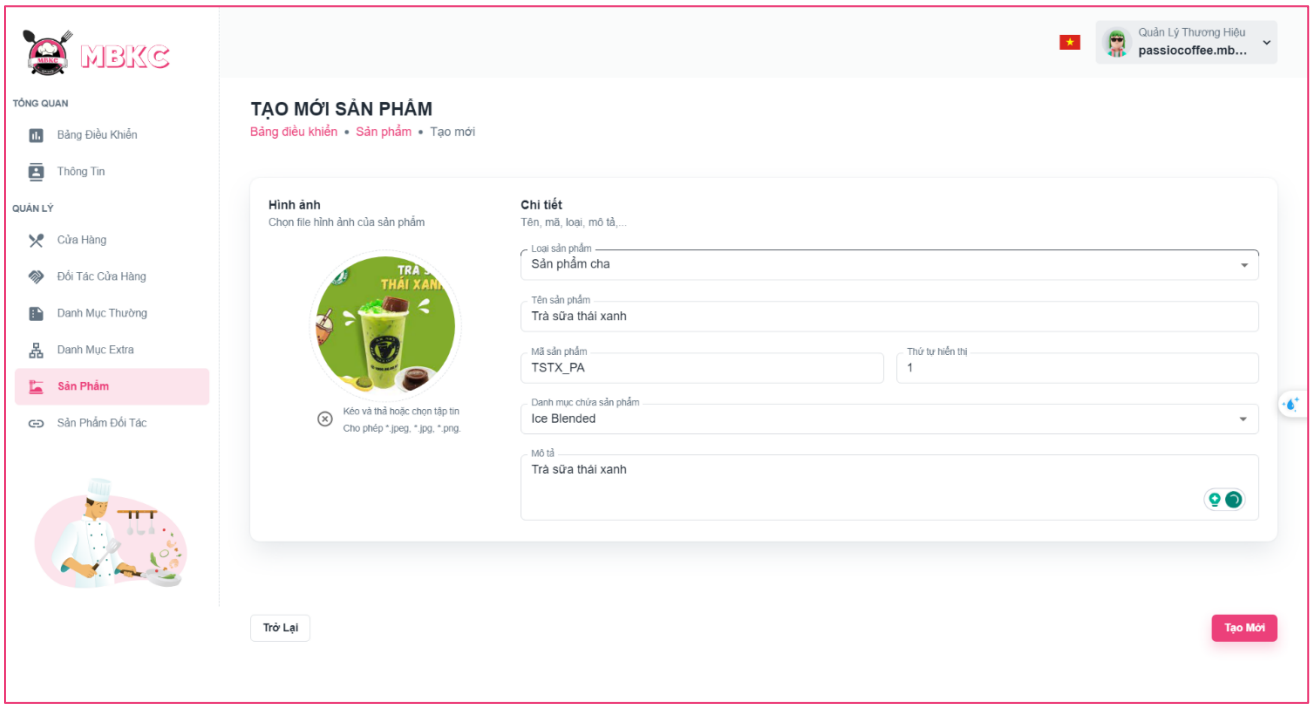


Figure 361: User Manual - Create new Product – Step 5 – Case 2

Case 3: If you want to create a product with type "Child" – “Sản phẩm con”, filling all the required including: “Hình ảnh”, “Loại sản phẩm", “Mã sản phẩm", “Sản phẩm cha”, “Kích thước”, “Thứ tự hiển thị", “Giá bán”, “Giảm giá”, “Giá gốc”, “Mô tả”.

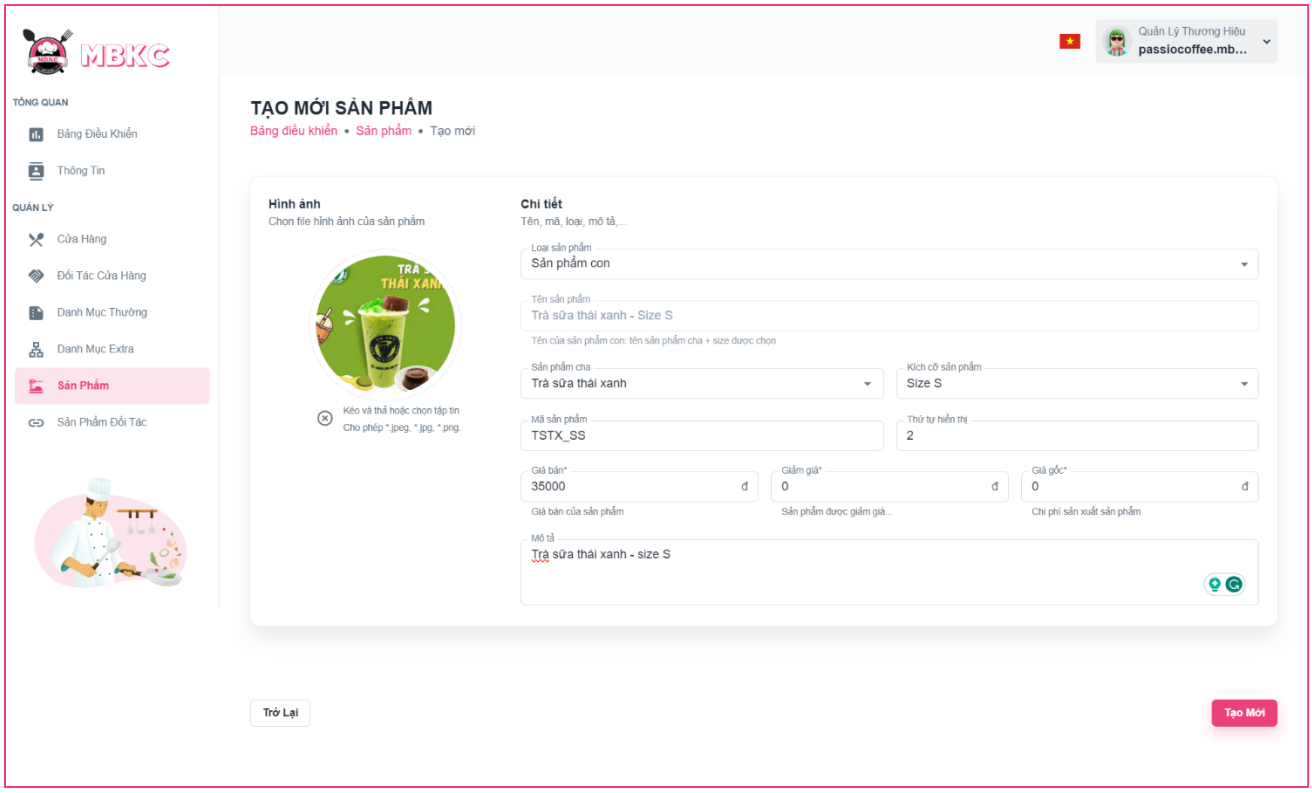


Figure 362: User Manual - Create new Product – Step 5 – Case 3

Step 6: After filling in all required field, click “Tạo Mới” to finish creating a new product.

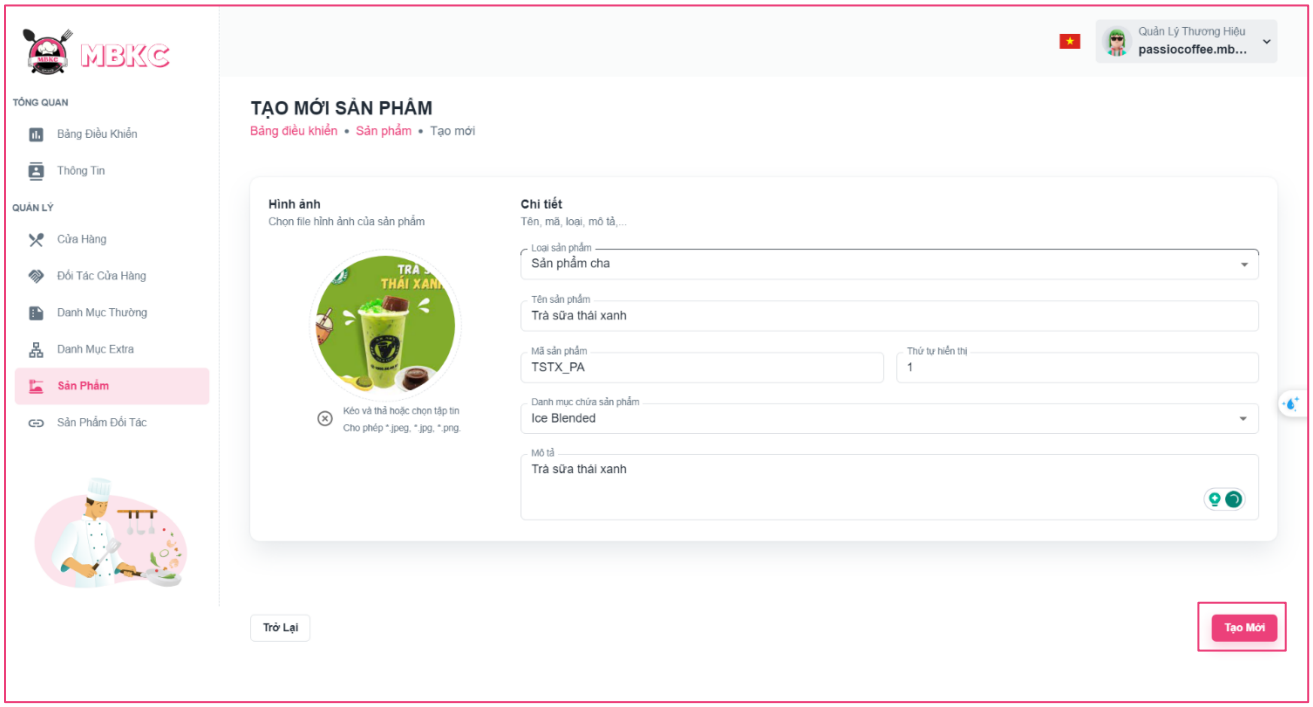


Figure 363: User Manual - Create new Product – Step 6

Step 7: You will be navigated to List of Products Page showing the list of products and a toast message “Tạo mới sản phẩm thành công”.

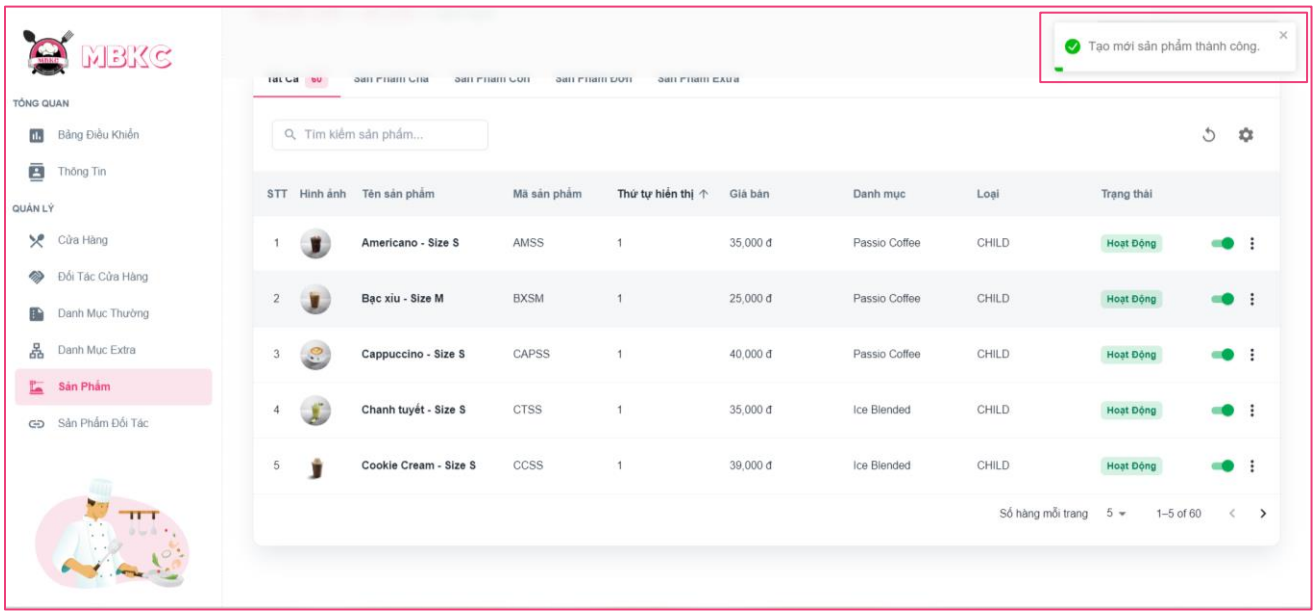


Figure 364: User Manual – Create new Product – Step 7

3.2.8.2. View list of Products

3.2.8.2.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of products.

This web application supports view list of products with account as a Brand Manager.

3.2.8.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

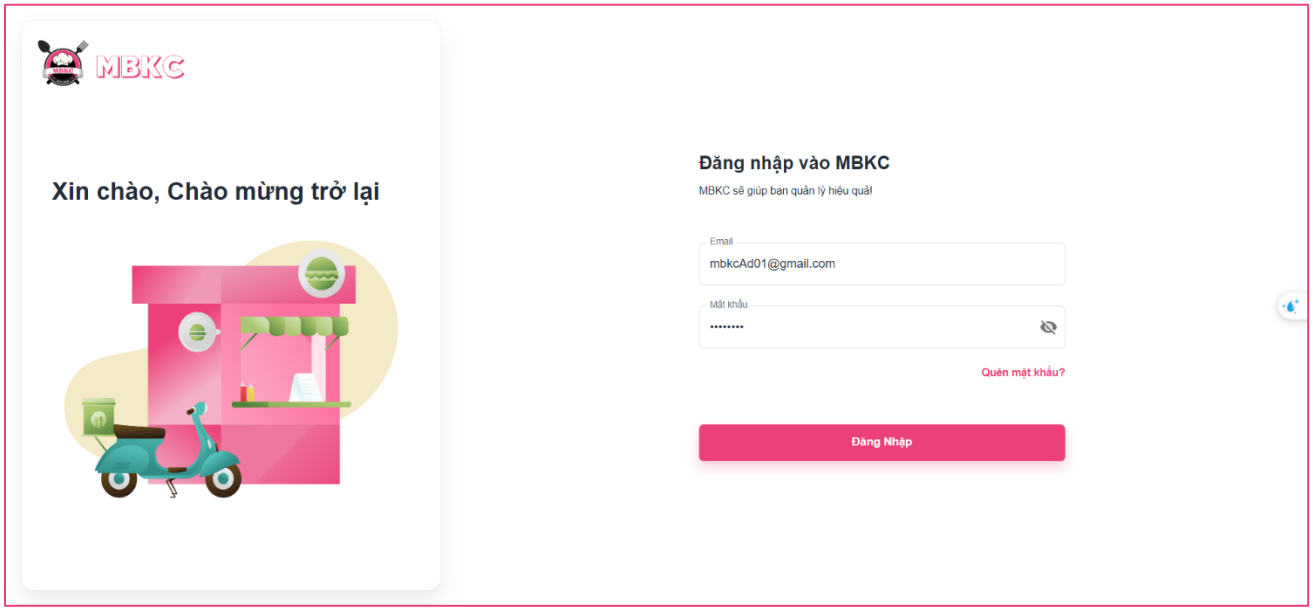


Figure 365: User Manual - View list of Products – Step 1

Step 2: Click “Đăng Nhập” button.

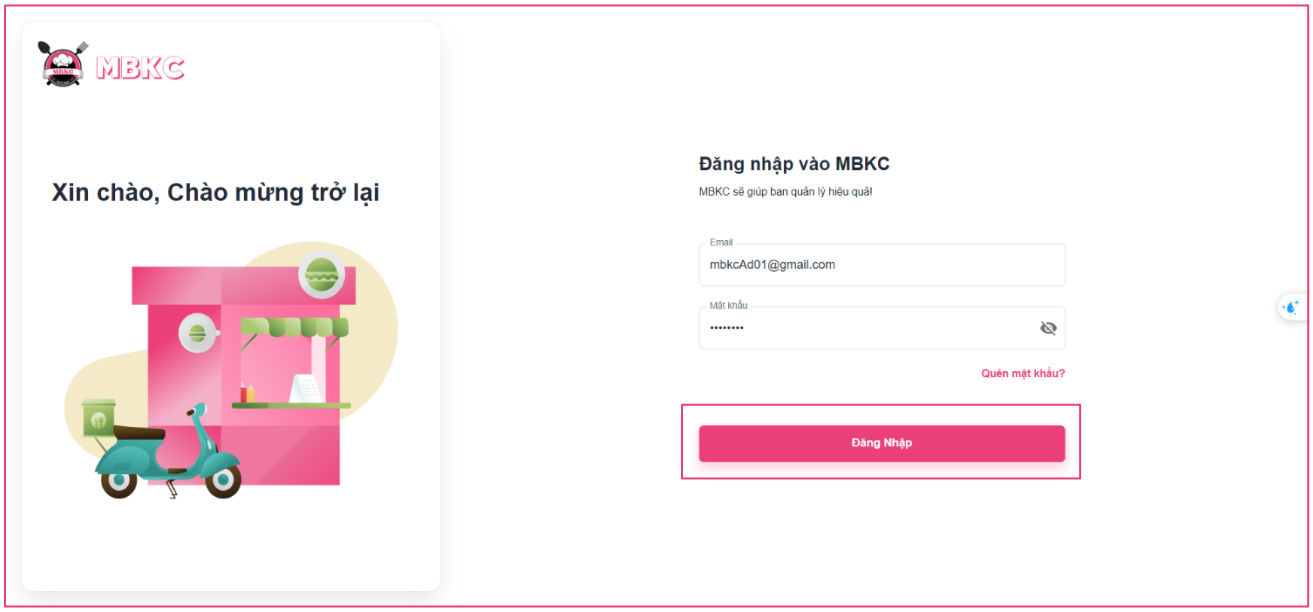


Figure 366: User Manual - View list of Products – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm” from the left sidebar.

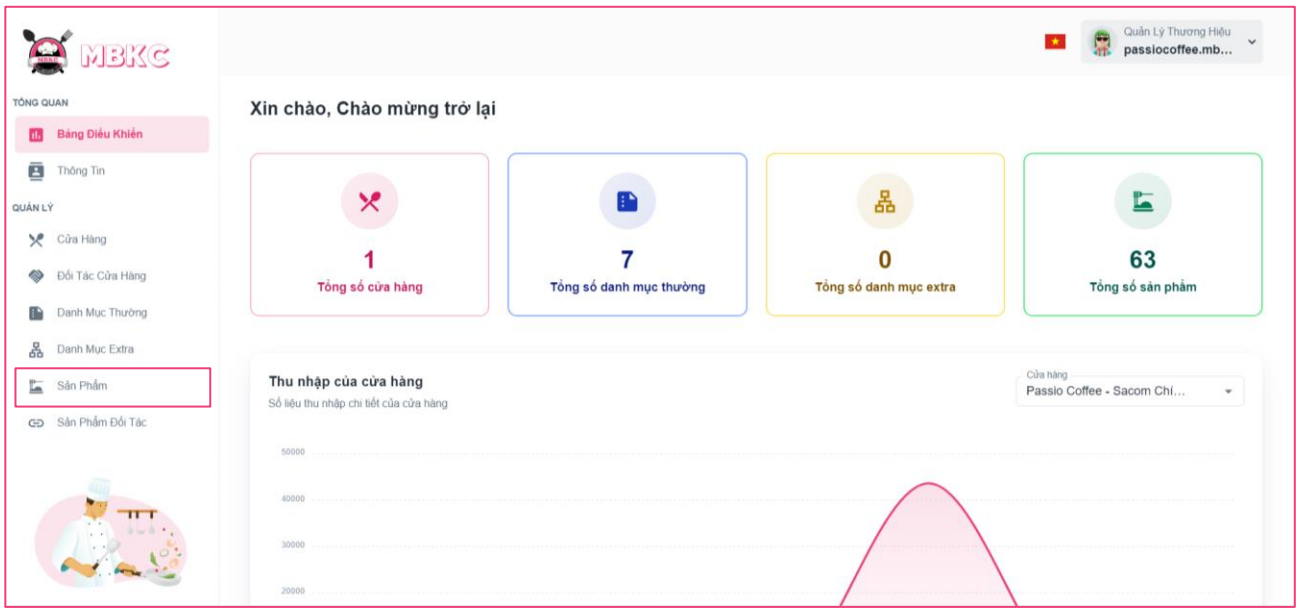


Figure 367: User Manual - View list of Products – Step 3

Step 4: Search, reload data, and show or hide columns in the table list of products.

Case 1: Click on the tab item to select filter by product type to get the list of products corresponding product type selected.

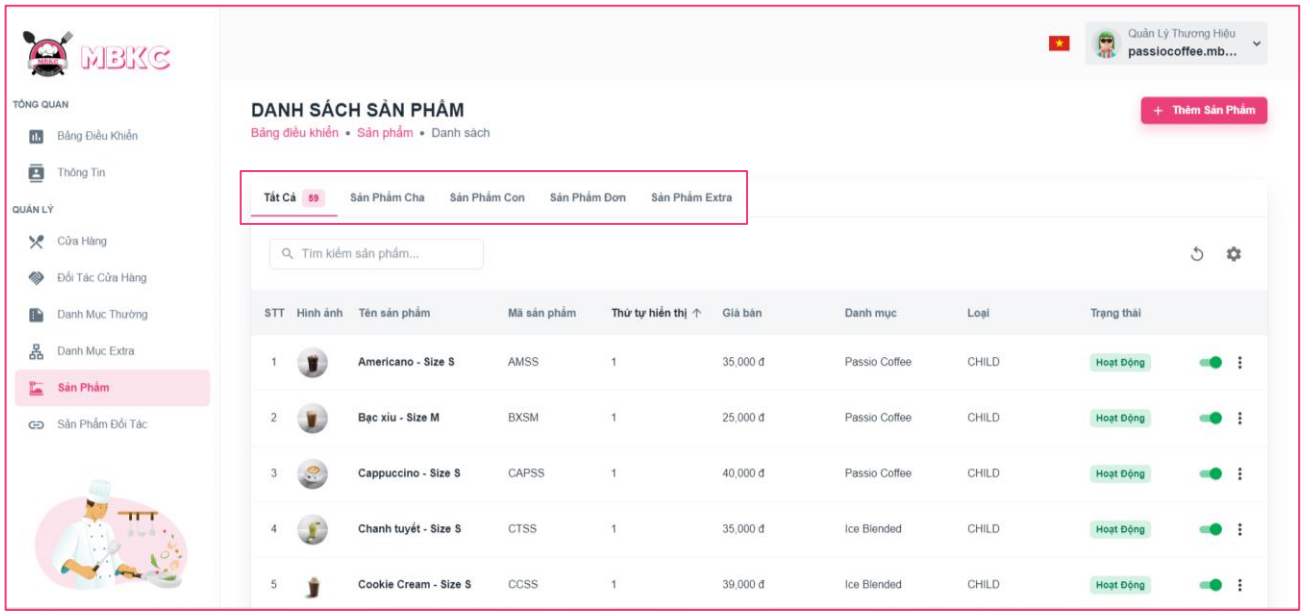


Figure 368: User Manual - View list of Products – Step 4 – Case 1

Case 2: Fill in the product's name in the search box to search products in the table list.

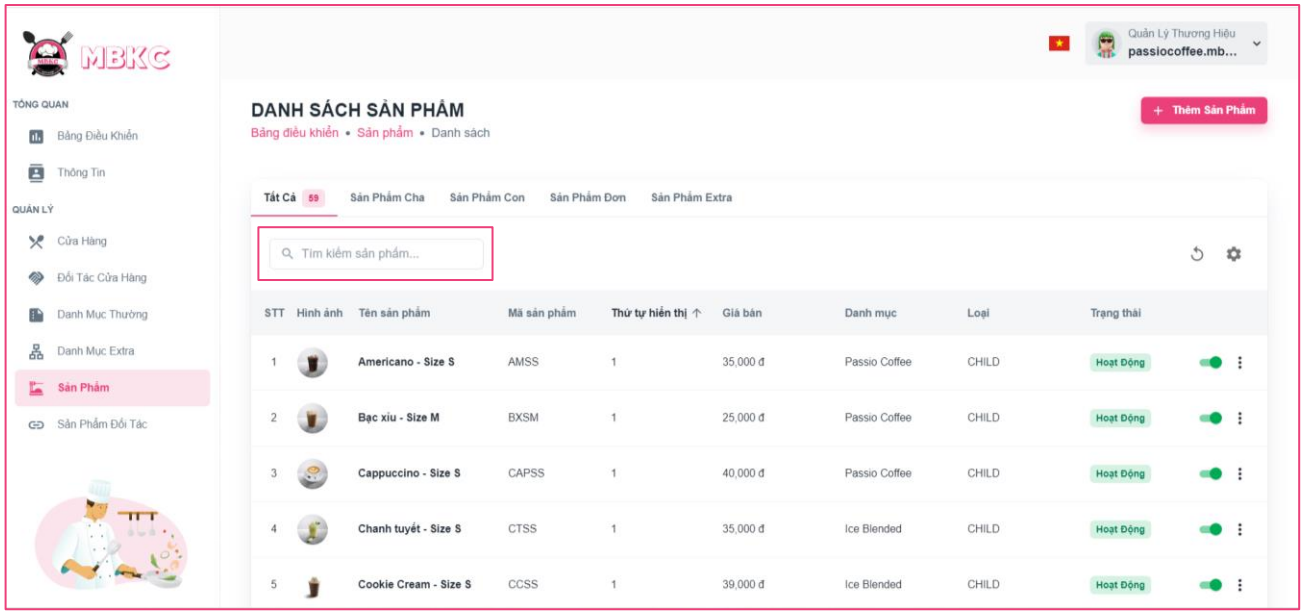


Figure 369: User Manual - View list of Products – Step 4 – Case 2

Case 3: Click the Reload icon button to reload the list of products.

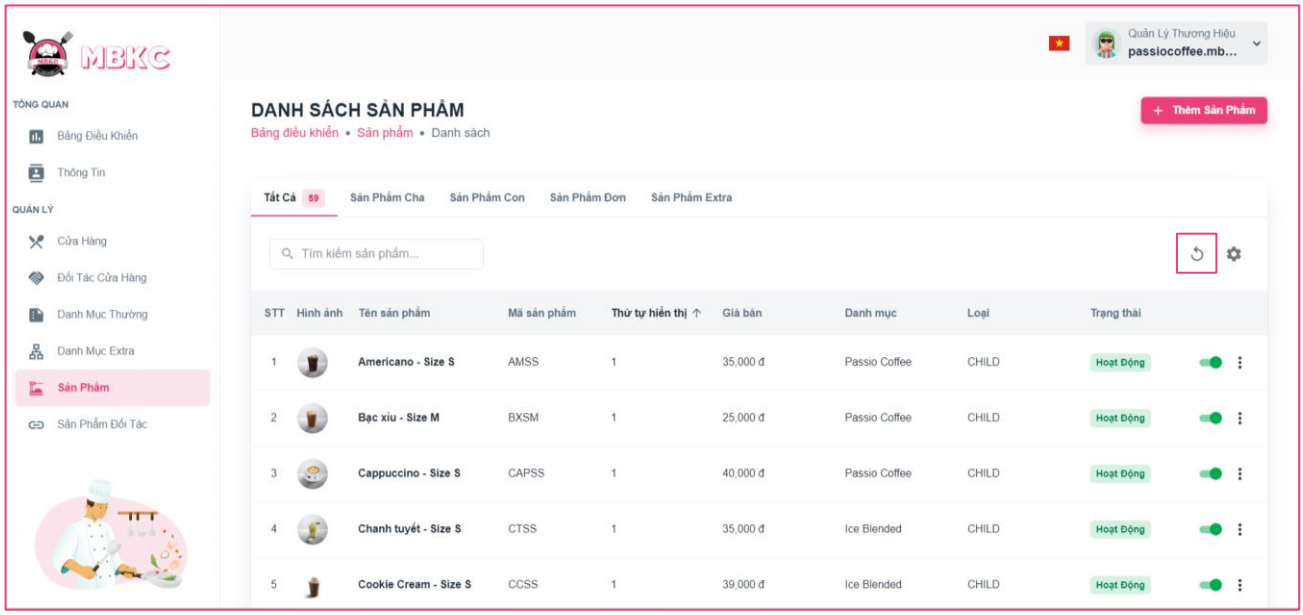


Figure 370: User Manual - View list of Products – Step 4 – Case 3

Case 4: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of products.

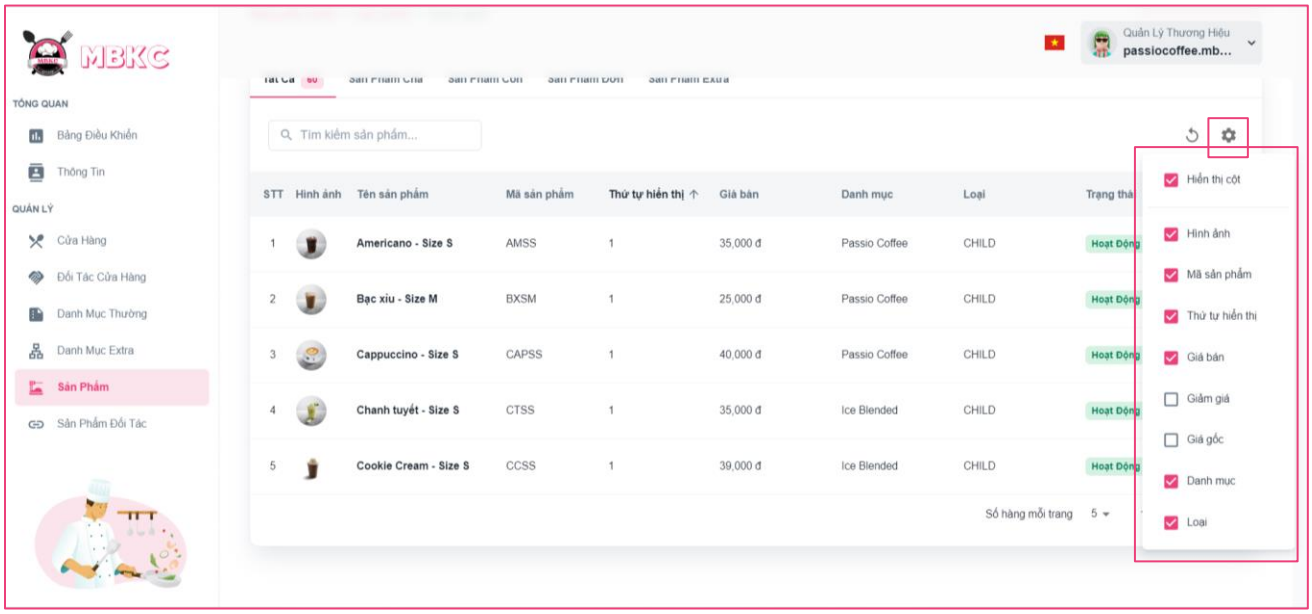


Figure 371: User Manual - View list of Products – Step 4 – Case 4

3.2.8.3. View details Product

3.2.8.3.1. Description

Use this guideline to view details product.

This web application supports view details product with account as a Brand Manager.

3.2.8.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

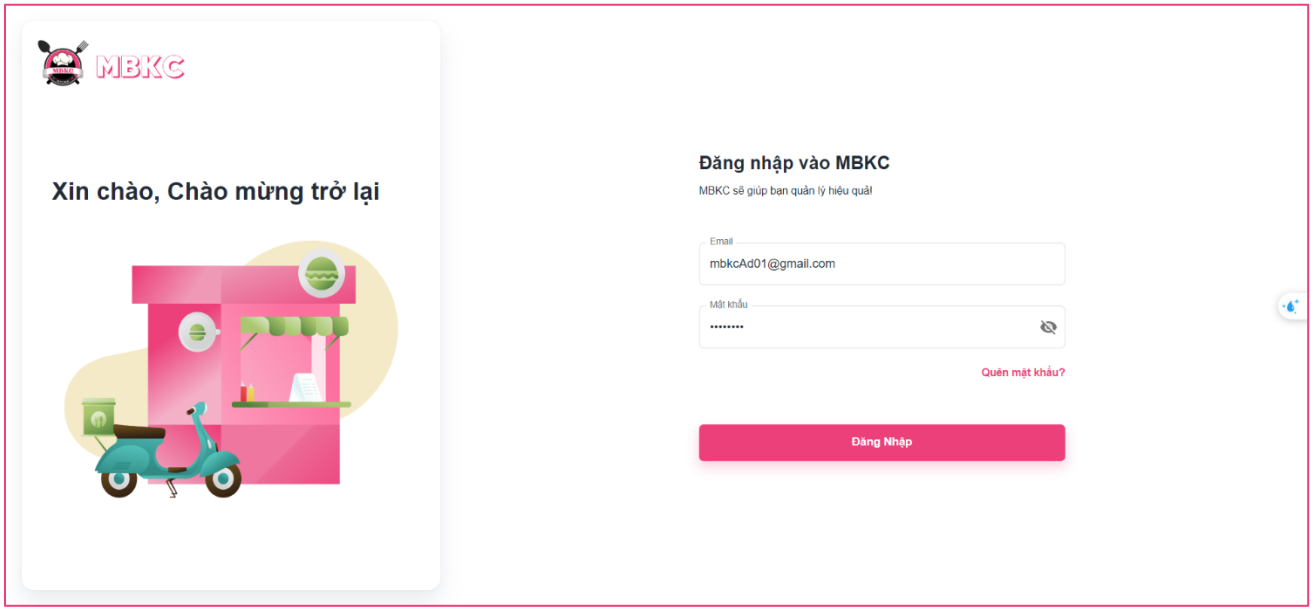


Figure 372: User Manual - View details Product – Step 1

Step 2: Click “Đăng Nhập” button.

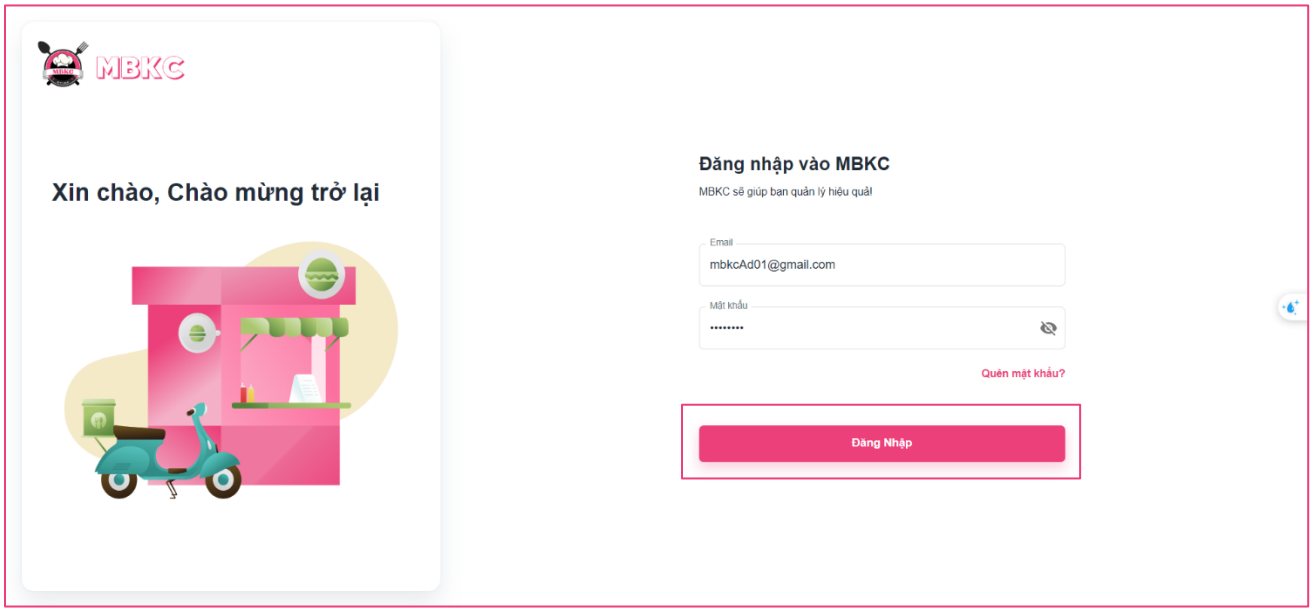


Figure 373: User Manual - View details Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm” from the left sidebar.

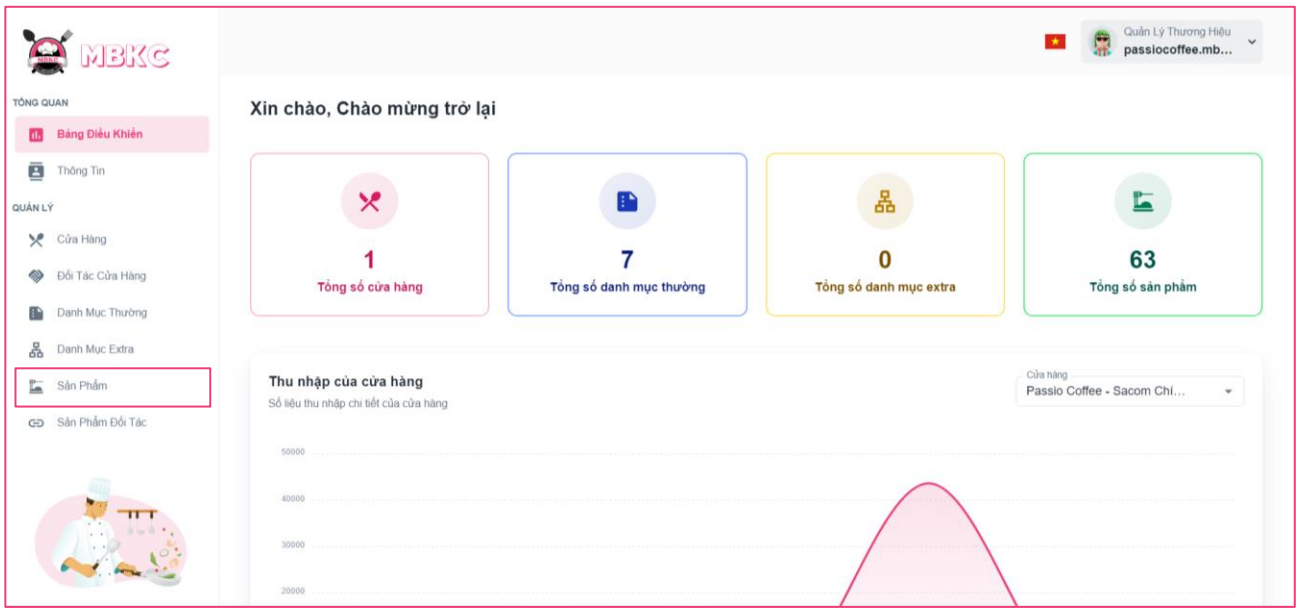


Figure 374: User Manual - View details Product – Step 3

Step 4: Click the product row to navigate to Product Details Page.

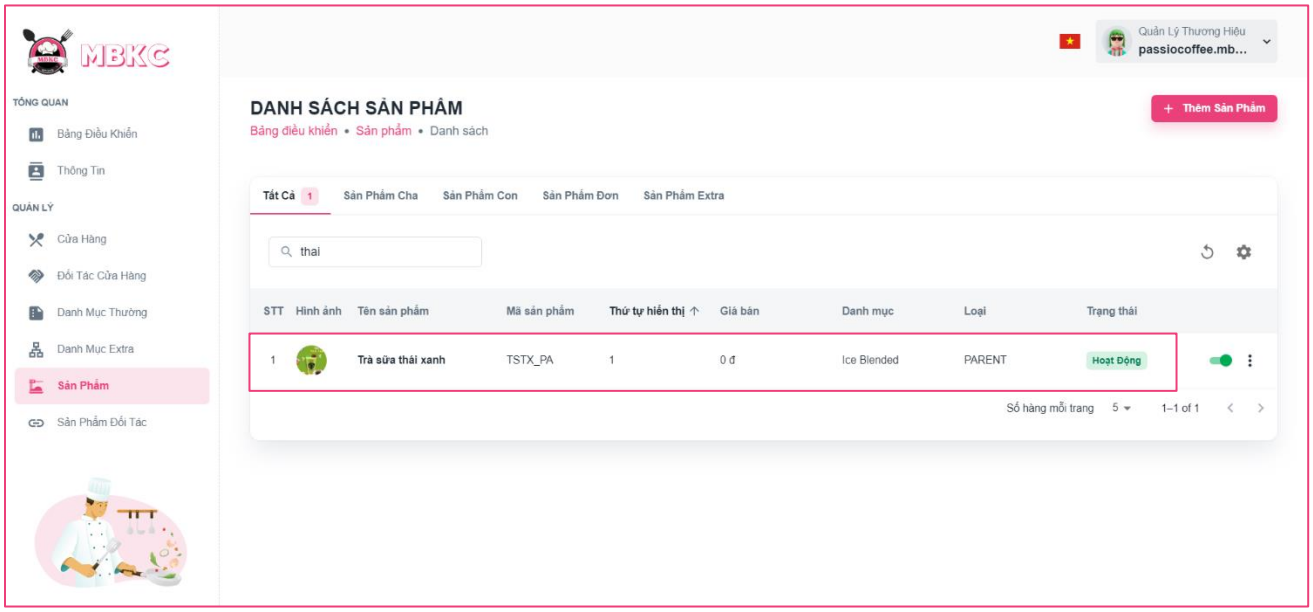


Figure 375: User Manual - View details Product – Step 4

Step 5: Navigate to Product Details Page to show information details of product.

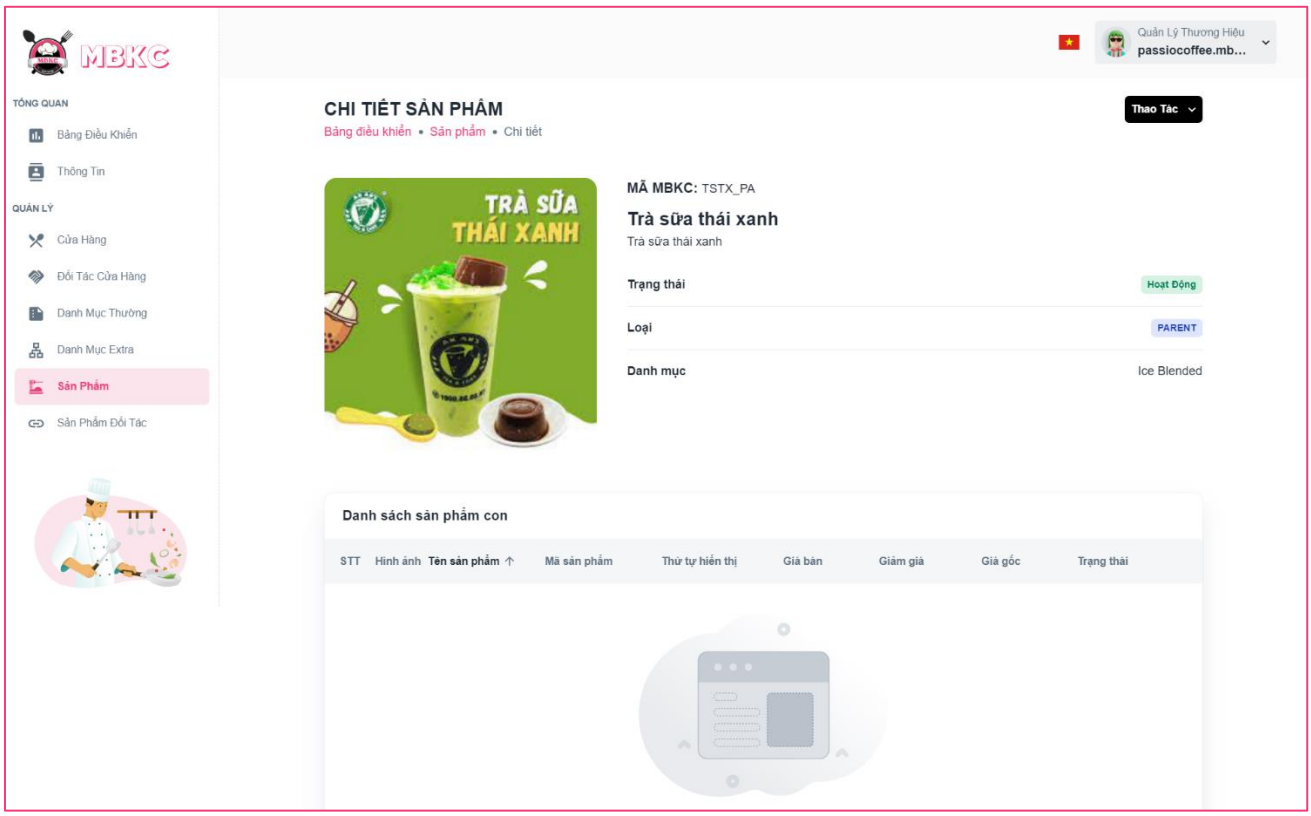


Figure 376: User Manual - View details Product – Step 5

3.2.8.4. Update Status Product

3.2.8.4.1. Description

Use this guideline to update the status of the product.

This web application supports update the status of the product with account as a Brand Manager.

3.2.8.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

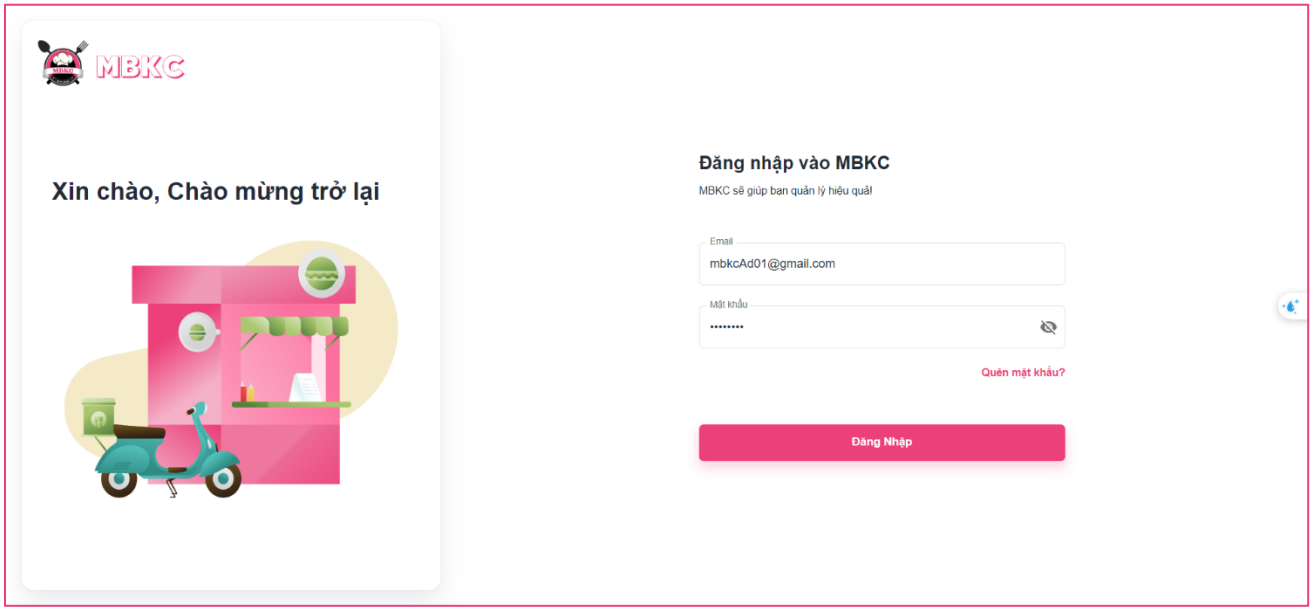


Figure 377: User Manual – Update Status Product – Step 1

Step 2: Click “Đăng Nhập” button.

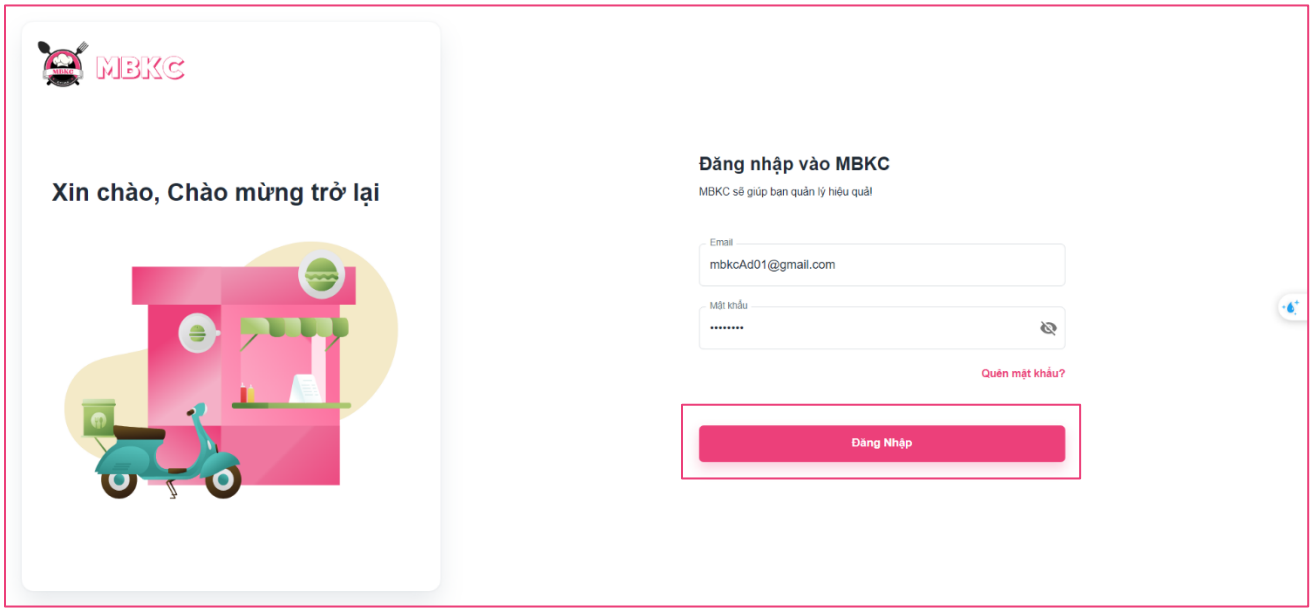


Figure 378: User Manual - Update Status Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm” from the left sidebar.

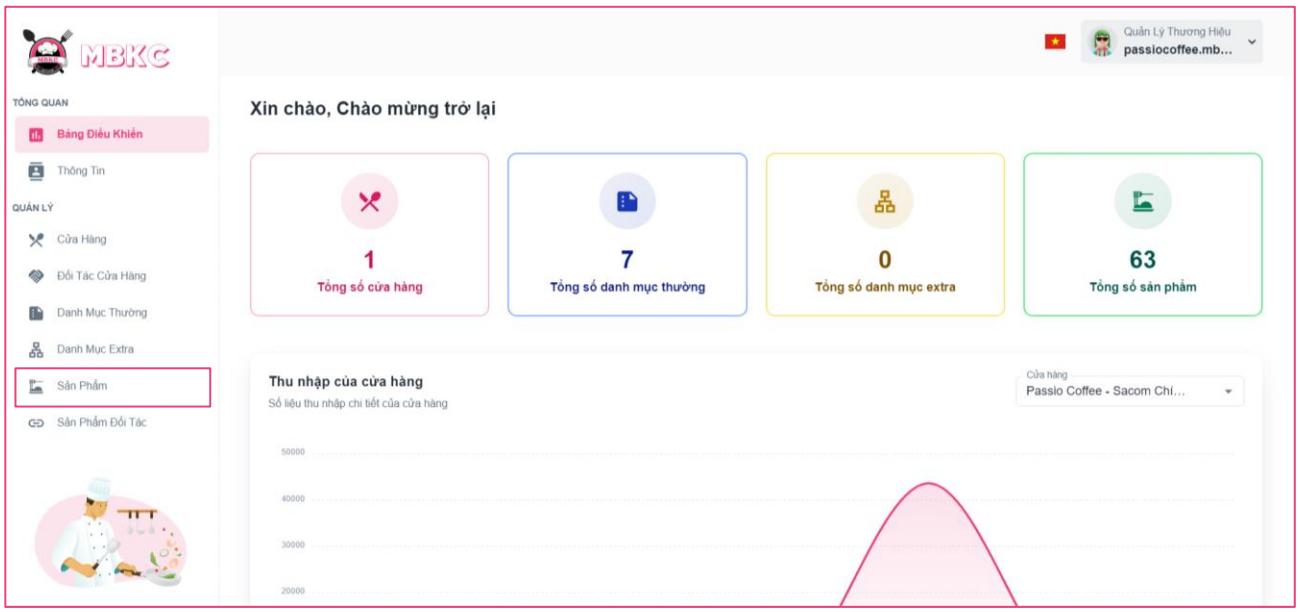


Figure 379: User Manual - Update Status Product – Step 3

Step 4: Click on the switch button on the product row to update the status.

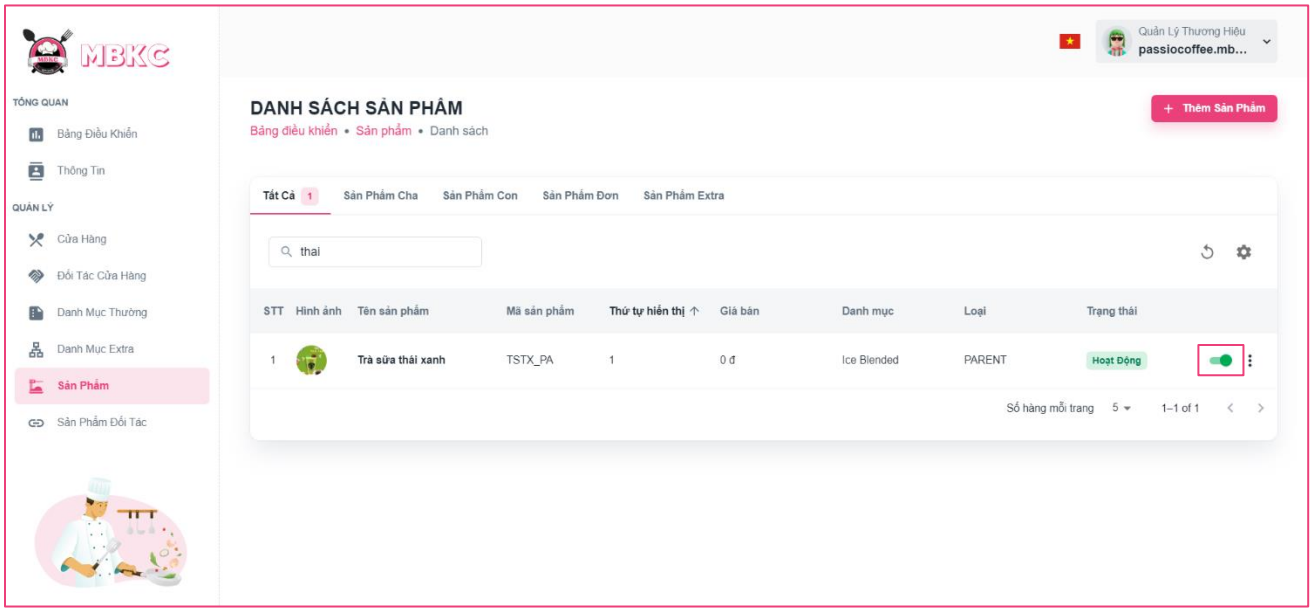


Figure 380: User Manual - Update Status Product – Step 4

Step 5: List of products will be reloaded and a toast message "Cập nhật trạng thái sản phẩm thành công". And the status of the product has been updated.

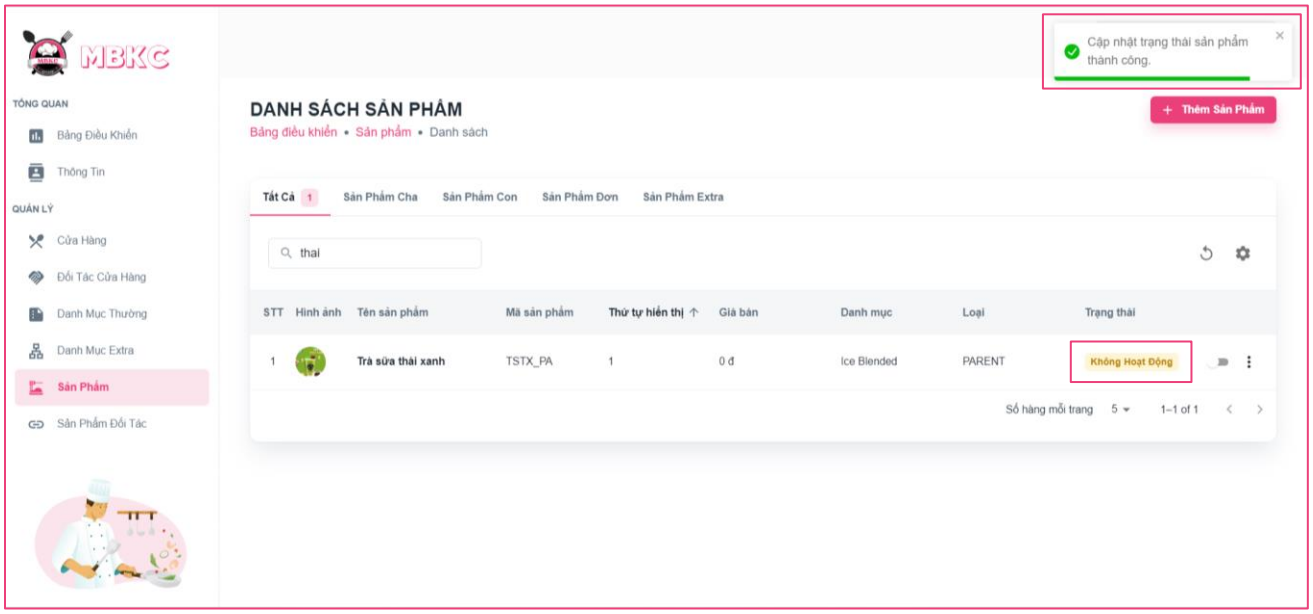


Figure 381: User Manual - Update Status Product – Step 5

3.2.8.5. Update Information Product

3.2.8.5.1. Description

Use this guideline to update information of the product.

This web application supports update information of the product with account as a Brand Manager.

3.2.8.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

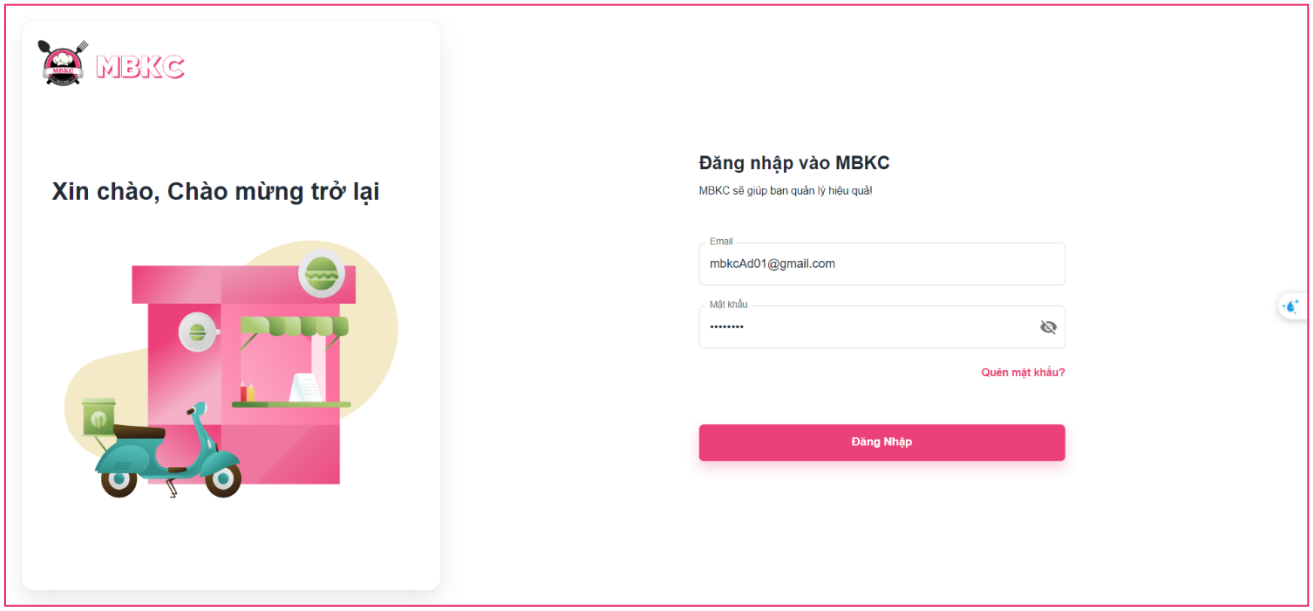


Figure 382: User Manual – Update Information Product – Step 1

Step 2: Click “Đăng Nhập” button.

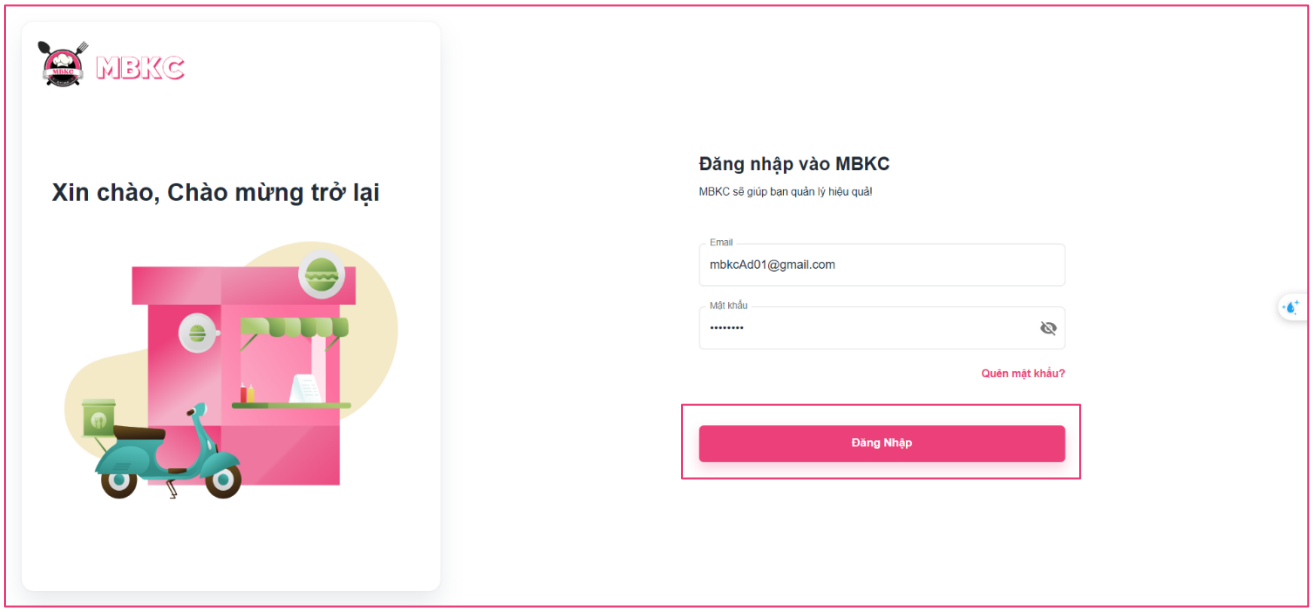


Figure 383: User Manual - Update Information Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm” from the left sidebar.

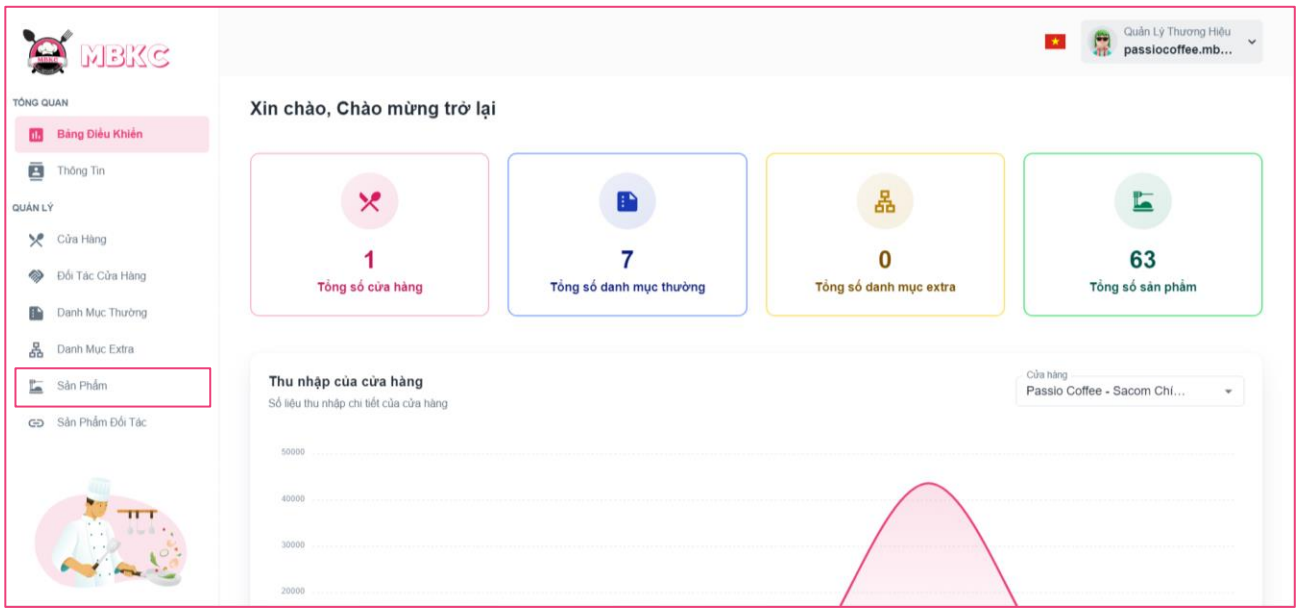


Figure 384: User Manual - Update Information Product – Step 3

Step 4: Update information of the product.

Case 1: Update information of the product in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the product row want to navigate Update Product Page.

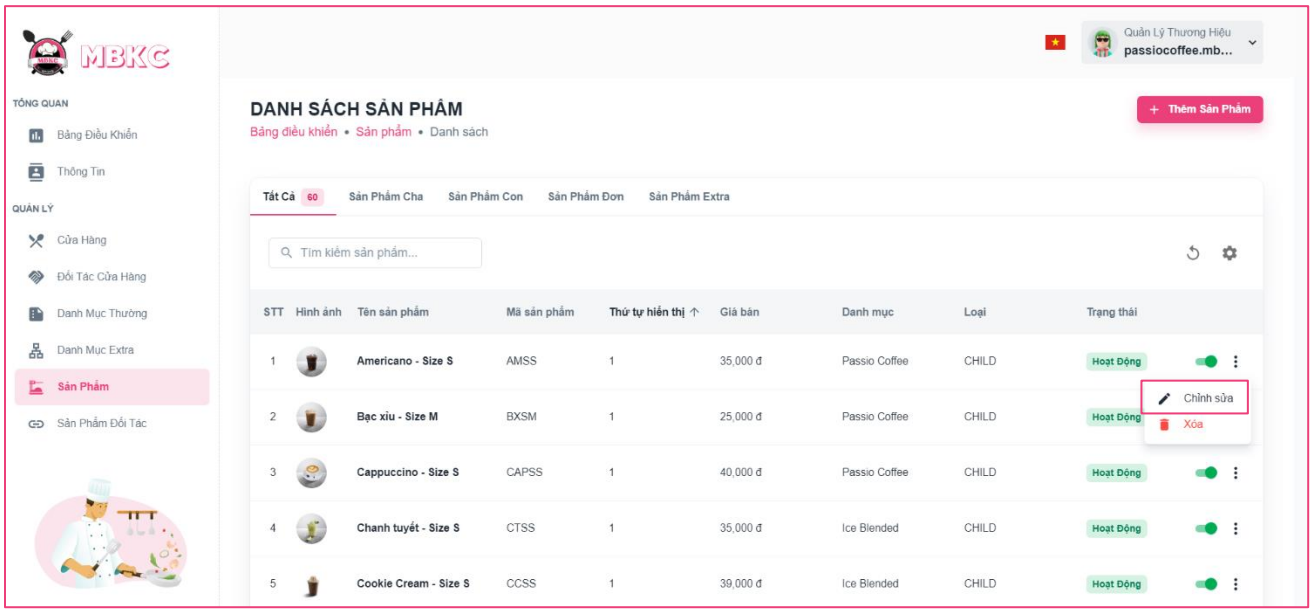


Figure 385: User Manual - Update Information Product – Step 4 – Case 1

Case 2: Update information of the product on the details page.

1: Click the product row to navigate to Product Details Page.

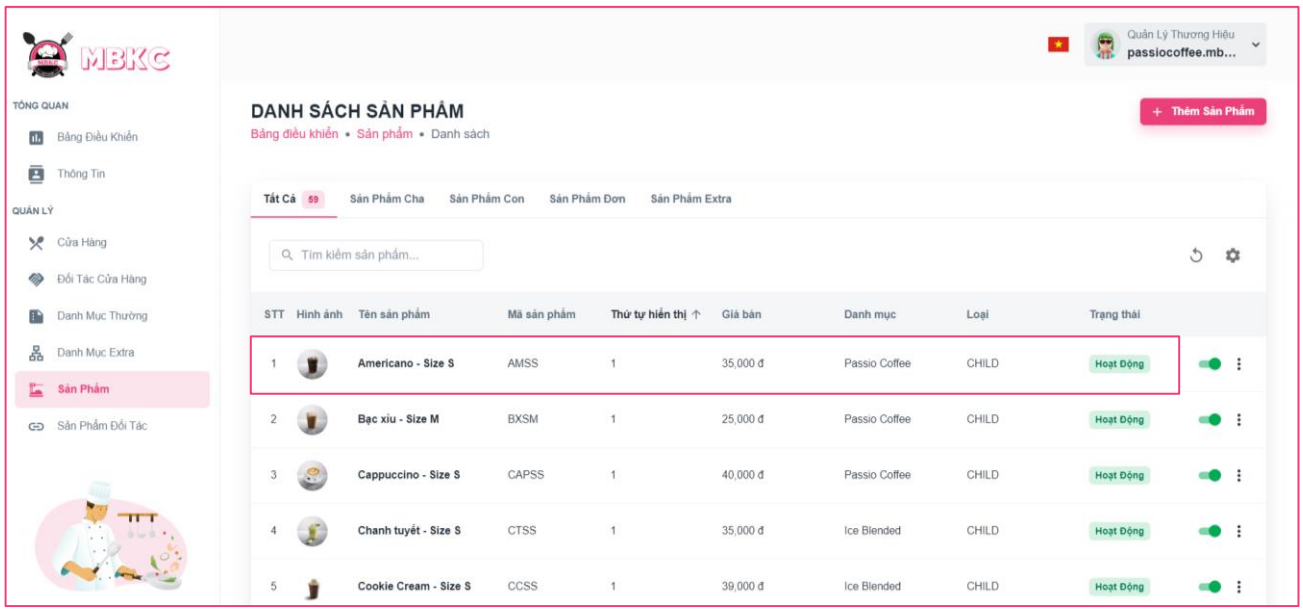


Figure 386: User Manual - Update Information Product – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Chỉnh sửa” menu item to navigate Update Product Page.

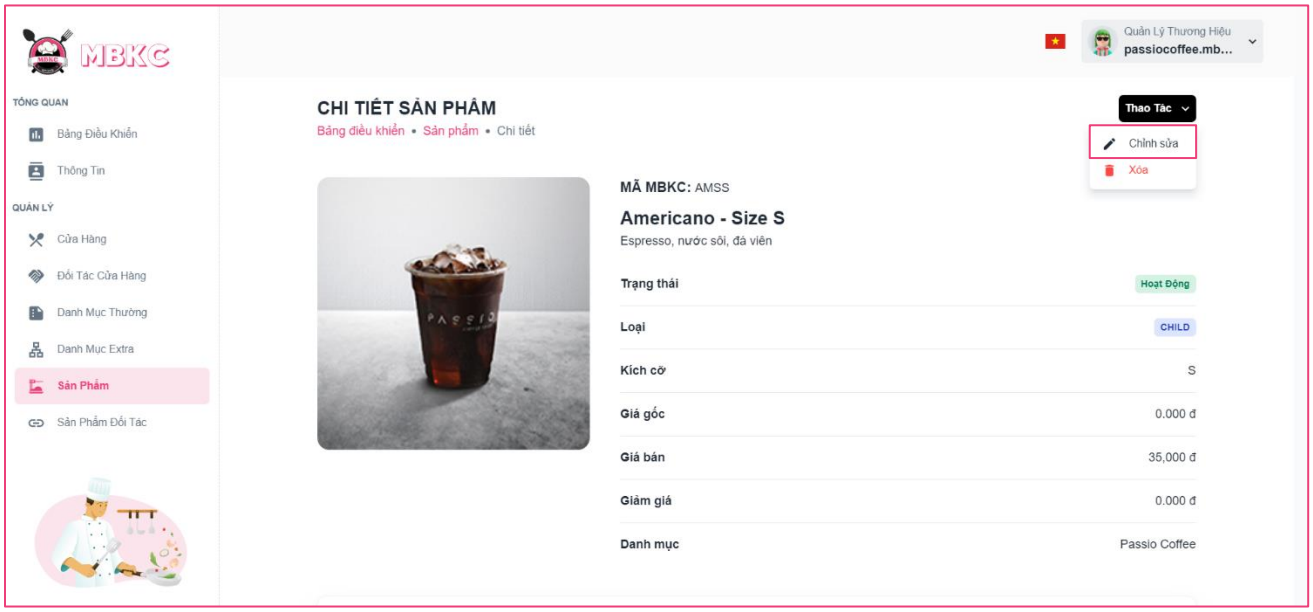


Figure 387: User Manual - Update Information Product – Step 4 – Case 2 – 2

Step 5: Product information will be automatically filled in the form. Then edit the information as desired.

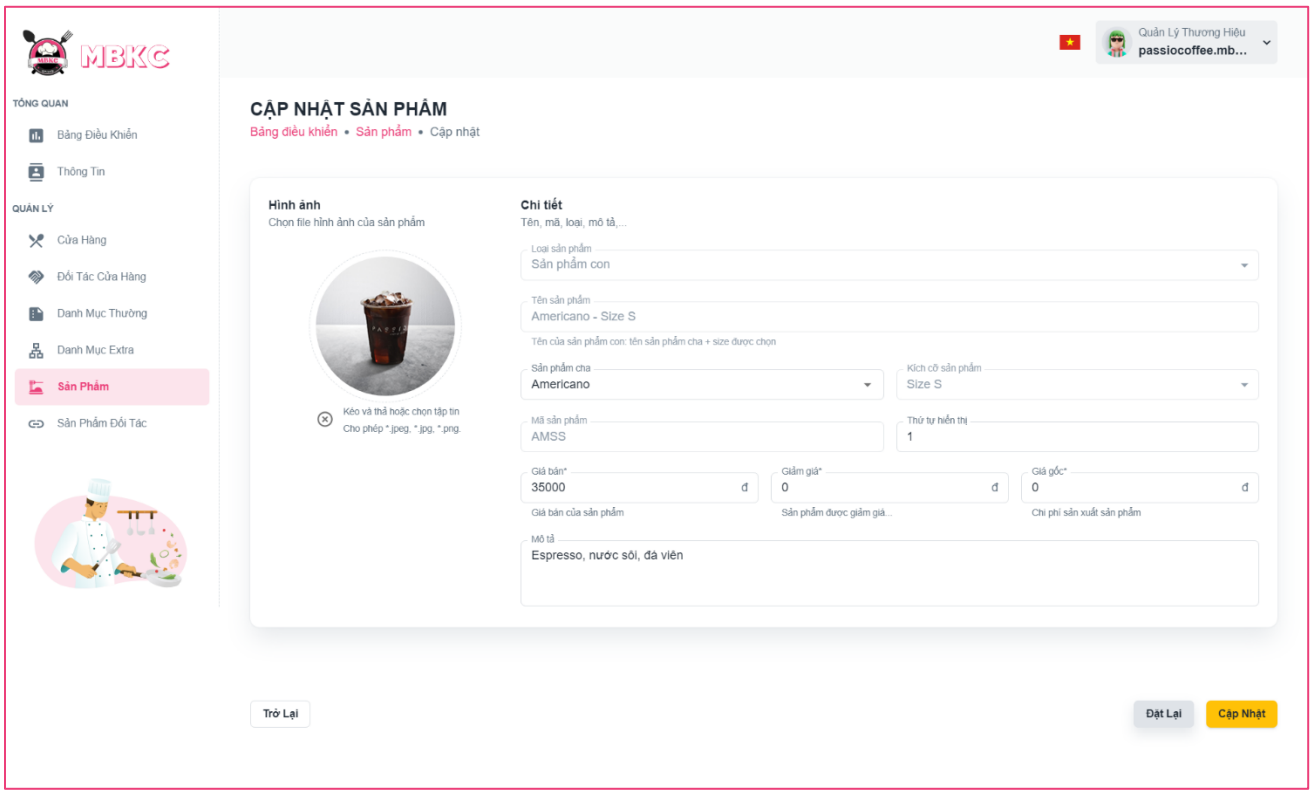


Figure 388: User Manual - Update Information Product – Step 5

Step 6: Click “Cập nhật” button.

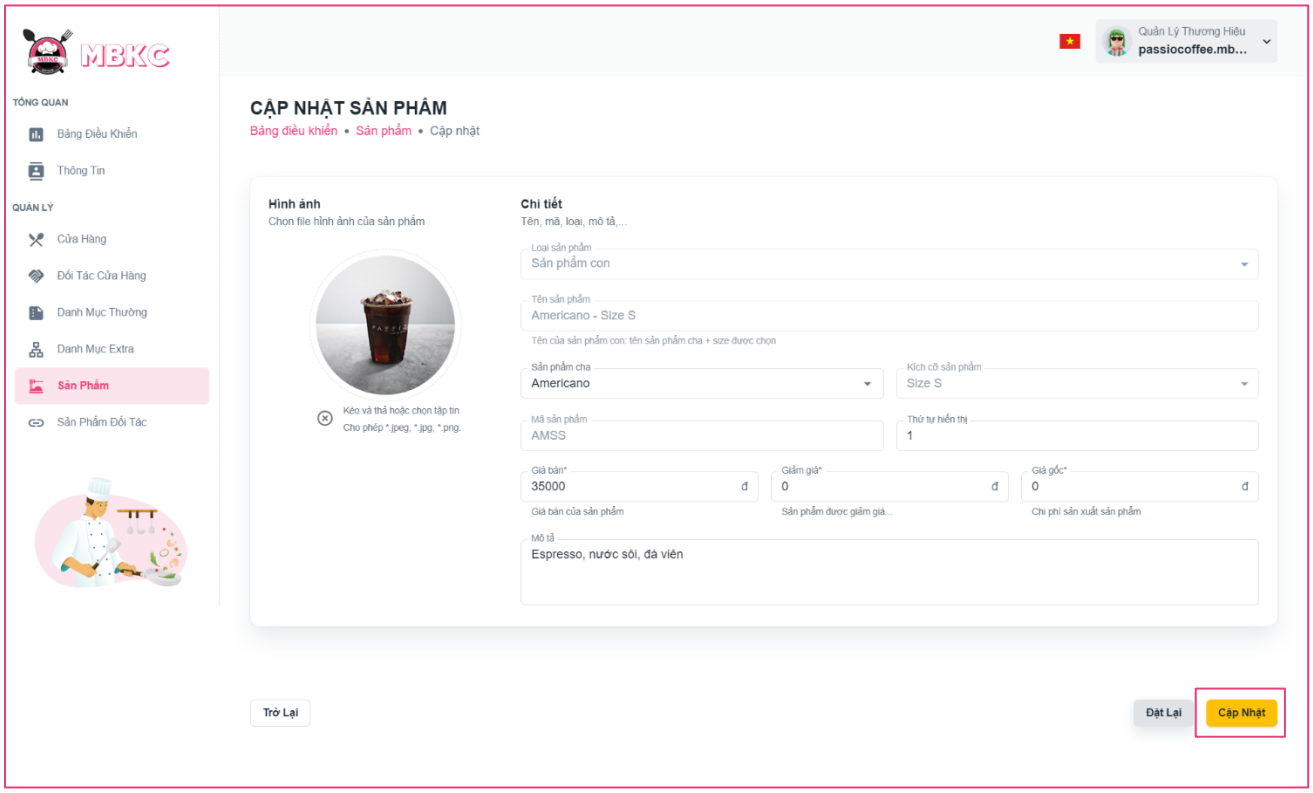


Figure 389: User Manual - Update Information Product – Step 6

Step 7:
Case 1: You will be navigated to Product Details Page showing the product details when start edit information of product from Product Details Page and a toast message “Cập nhật sản phẩm thành công”.

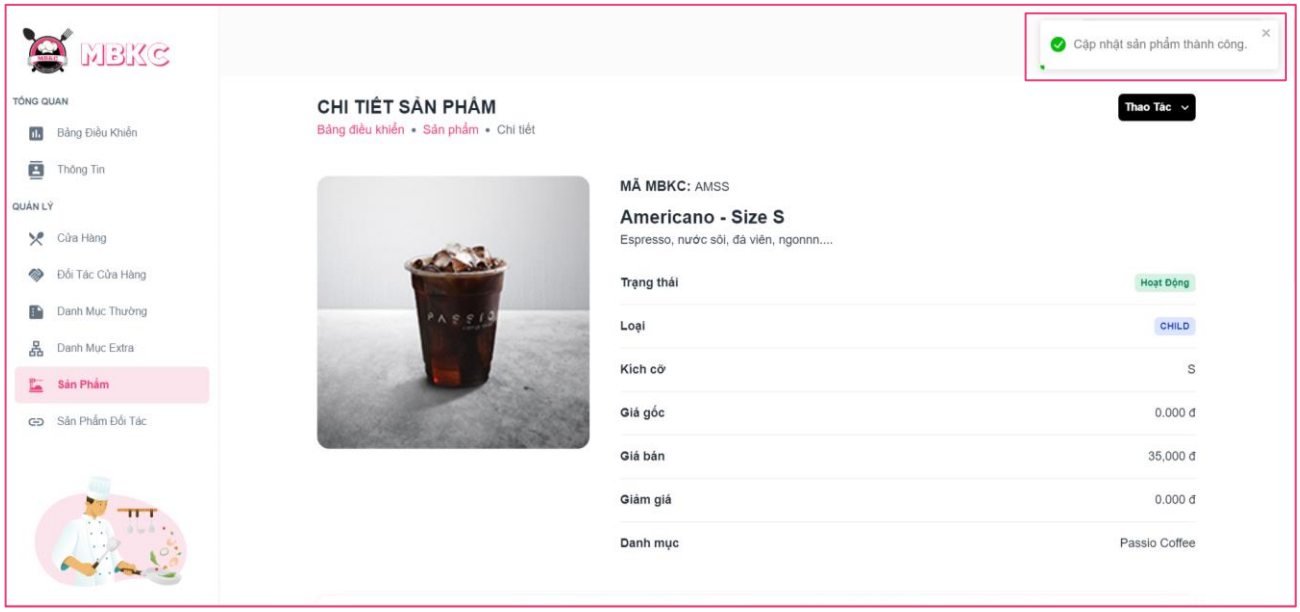


Figure 390: User Manual - Update Information Product – Step 7 – Case 1

Case 2: You will be navigated to List of Products Page showing the list of products when start edit information of product from List of Products Page and a toast message “Cập nhật sản phẩm thành công”.

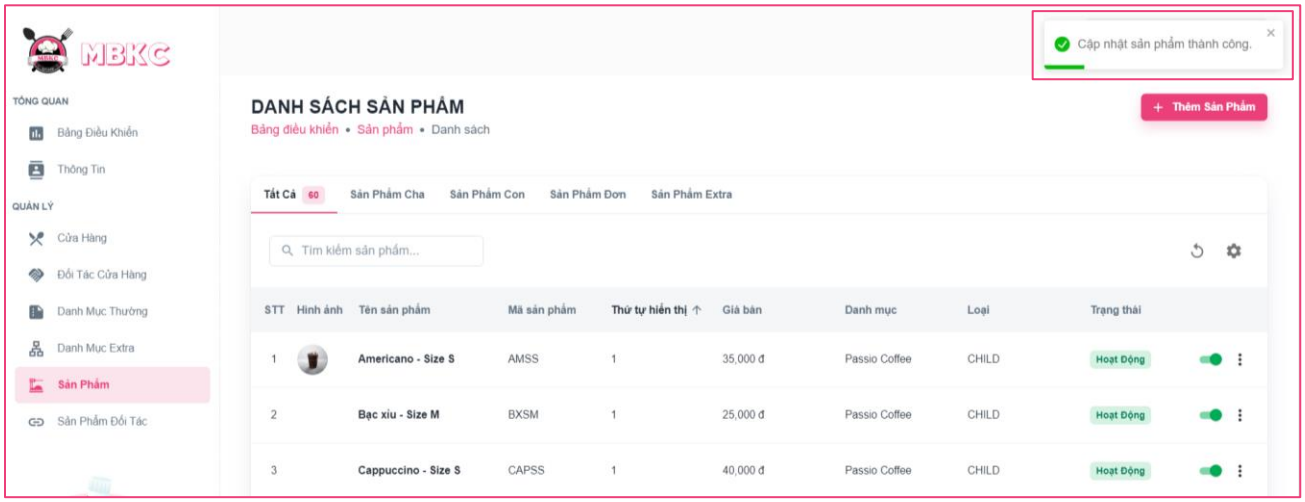


Figure 391: User Manual - Update Information Product – Step 7 – Case 2

3.2.8.6. Delete Product

3.2.8.6.1. Description

Use this guideline to delete product.

This web application supports update information of the product with account as a Brand Manager.

3.2.8.6.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

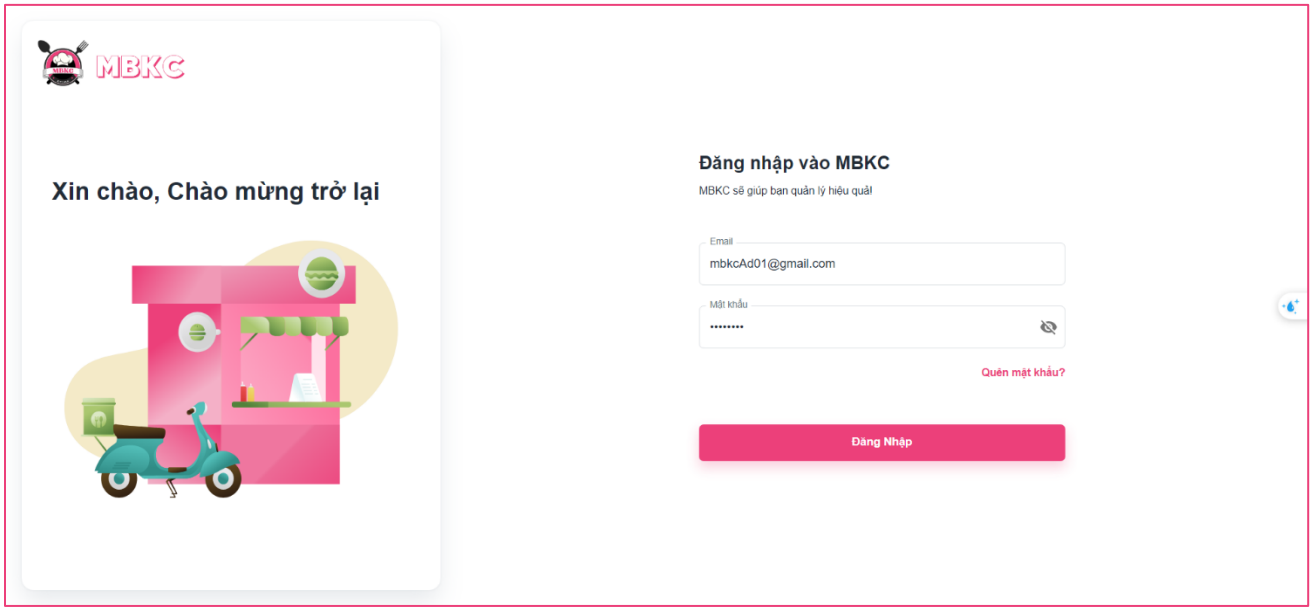


Figure 392: User Manual – Delete Product – Step 1

Step 2: Click “Đăng Nhập” button.

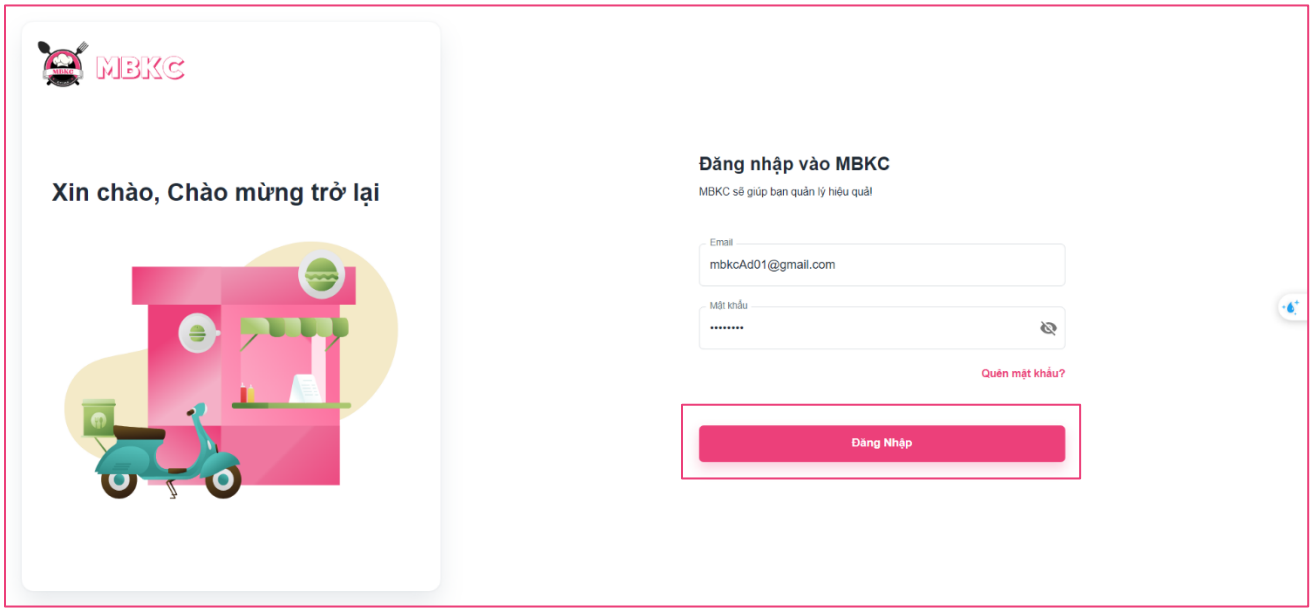


Figure 393: User Manual - Delete Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm” from the left sidebar.

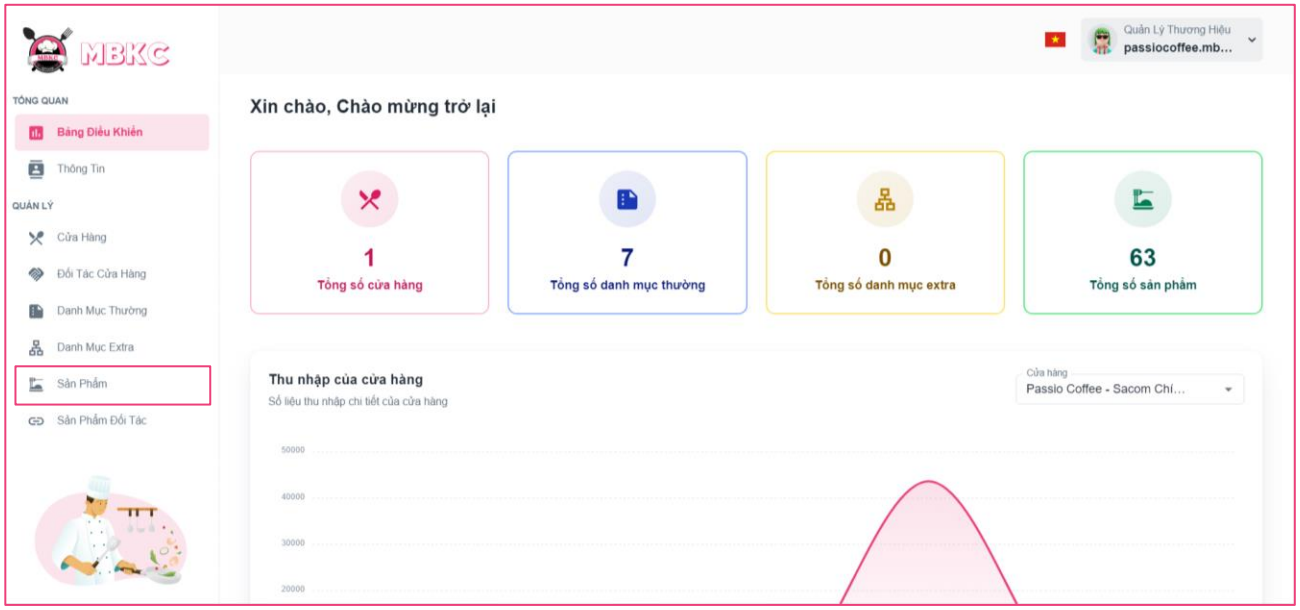


Figure 394: User Manual - Delete Product – Step 3

Step 4: Delete product.

Case 1: Delete product in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the product row to delete.

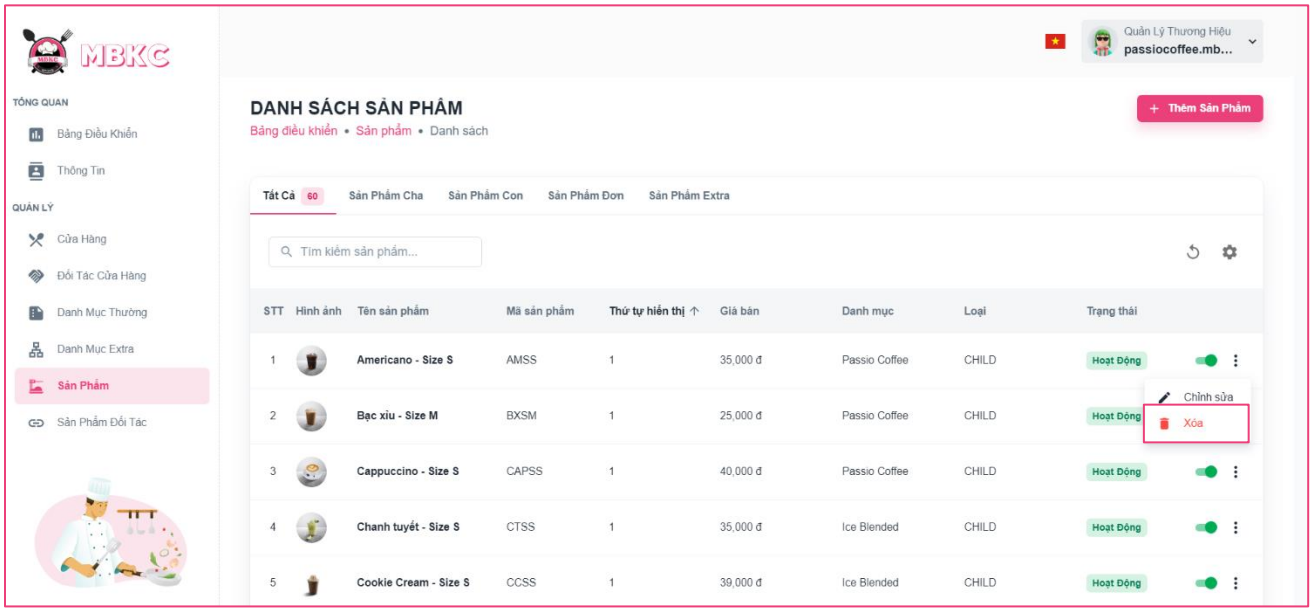


Figure 395: User Manual - Delete Product – Step 4 – Case 1

Case 2: Delete product on the details page.

1: Click the product row to navigate to Product Details Page.

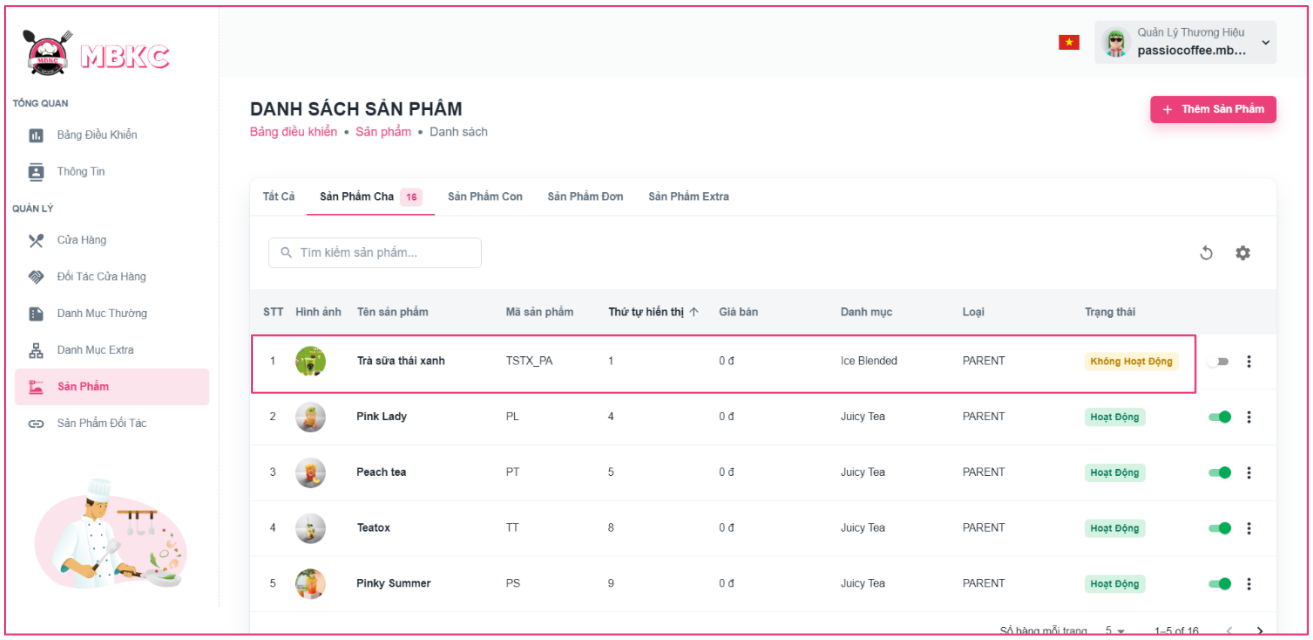


Figure 396: User Manual - Delete Product – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item to delete.

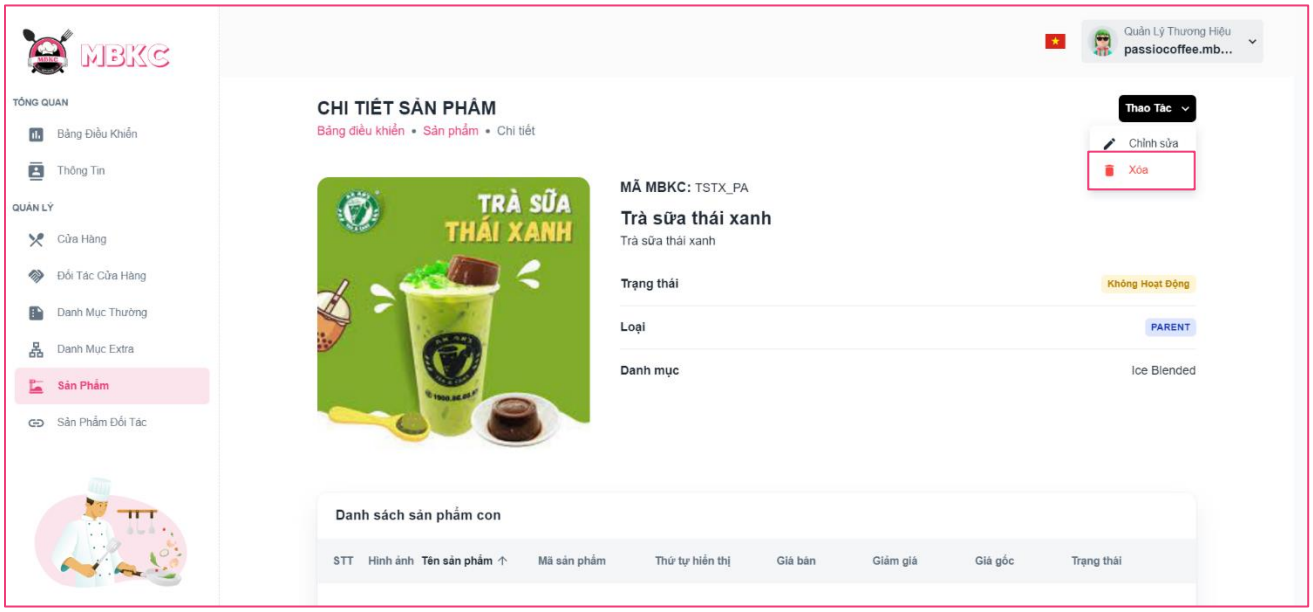


Figure 397: User Manual - Delete Product – Step 4 – Case 2 - 2

Step 5: Click “Xác Nhận” button.

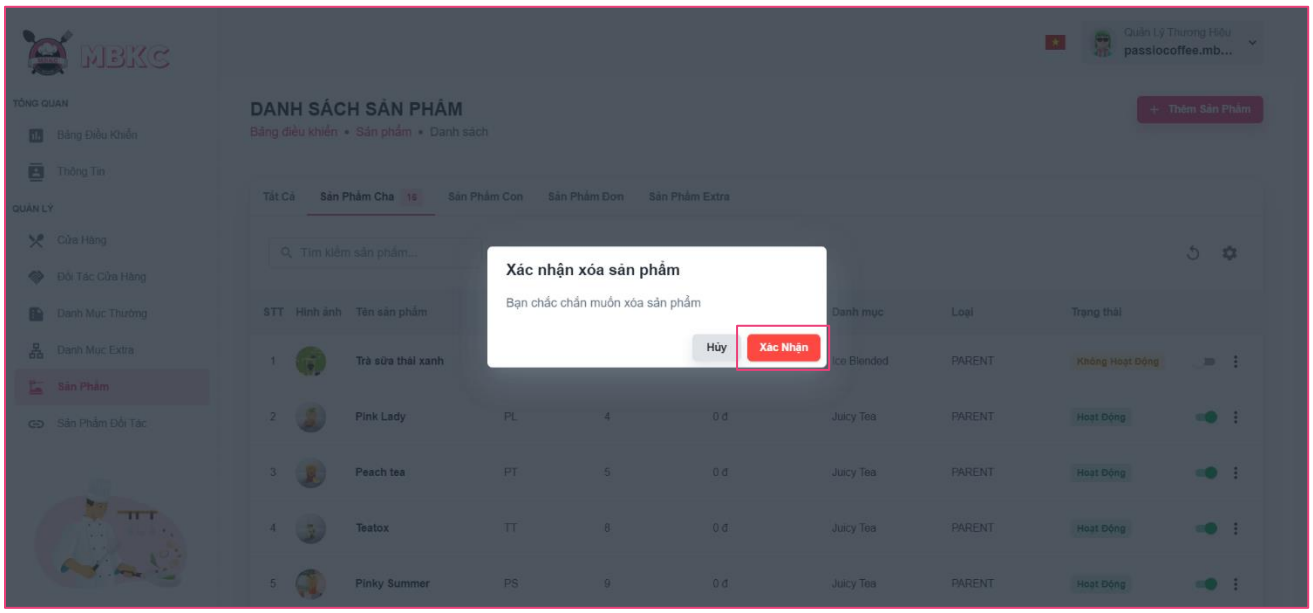


Figure 398: User Manual - Delete Product – Step 5

Step 6: List of products will be reloaded and a toast message “Đã xóa sản phẩm thành công”. And product has been deleted.

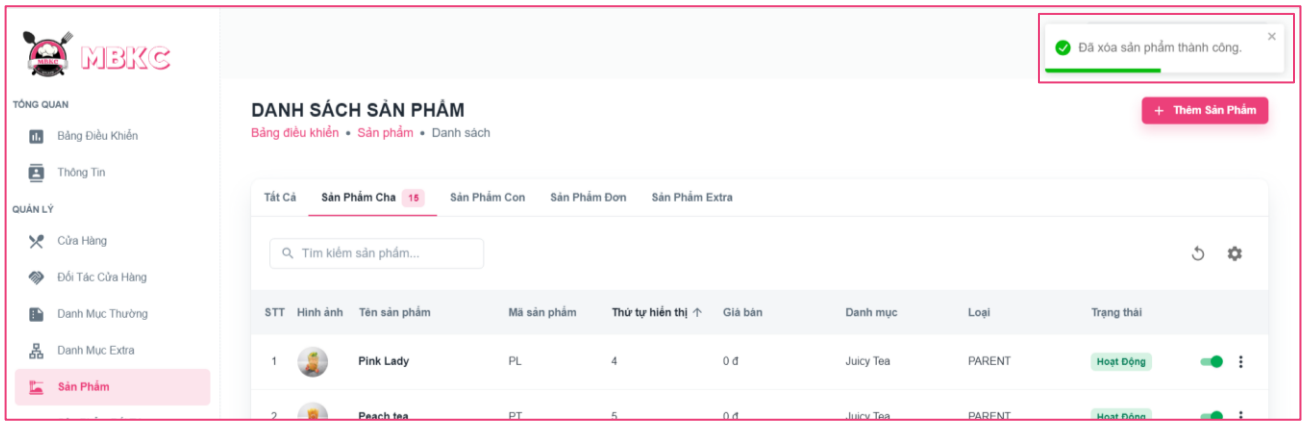


Figure 399: User Manual – Delete Product – Step 6

3.2.9. Store Partner management

3.2.9.1. Create new Store Partner

3.2.9.1.1. Description

Use this guideline to create a new store partner.

This web application supports create a new store partner with account as a Brand Manager.

3.2.9.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

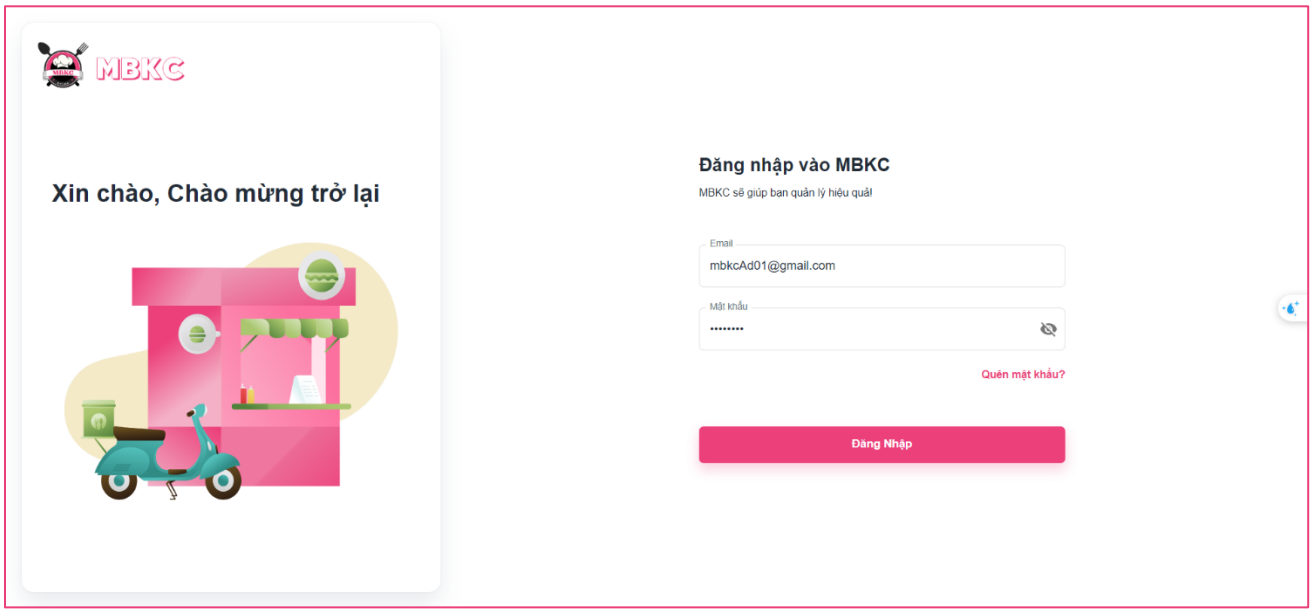


Figure 400: User Manual - Create new Store Partner – Step 1

Step 2: Click “Đăng Nhập” button.

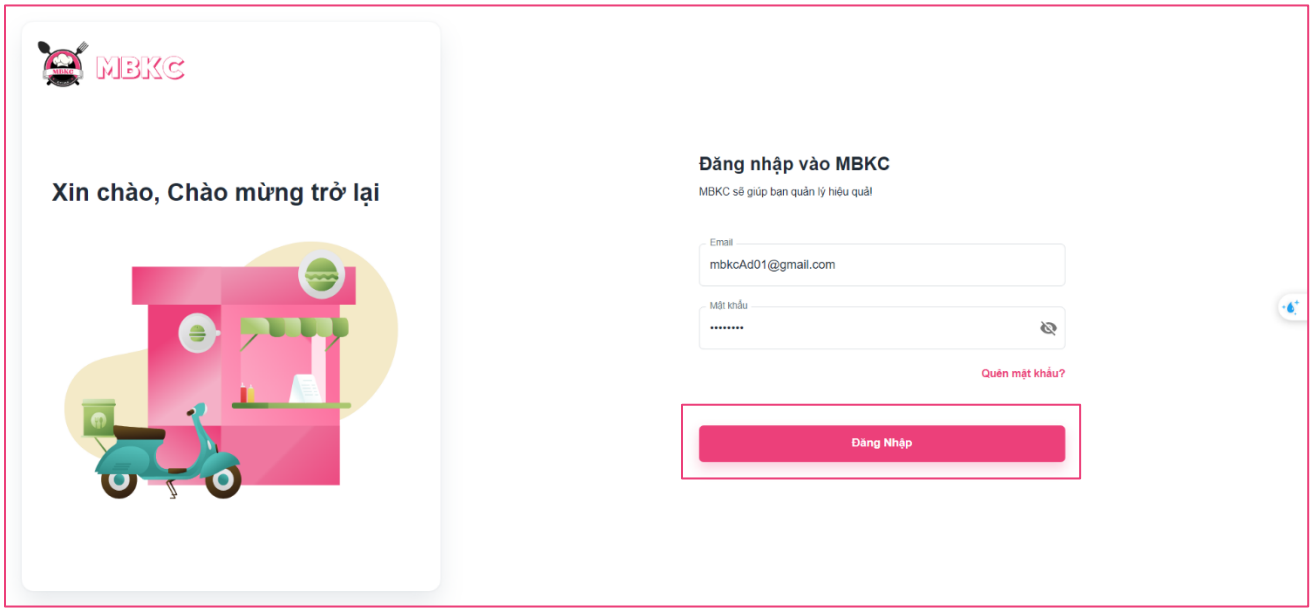


Figure 401: User Manual - Create new Store Partner – Step 2

Step 3: From Brand Dashboard Page, choose “Đối Tác Cửa Hàng” from the left sidebar.

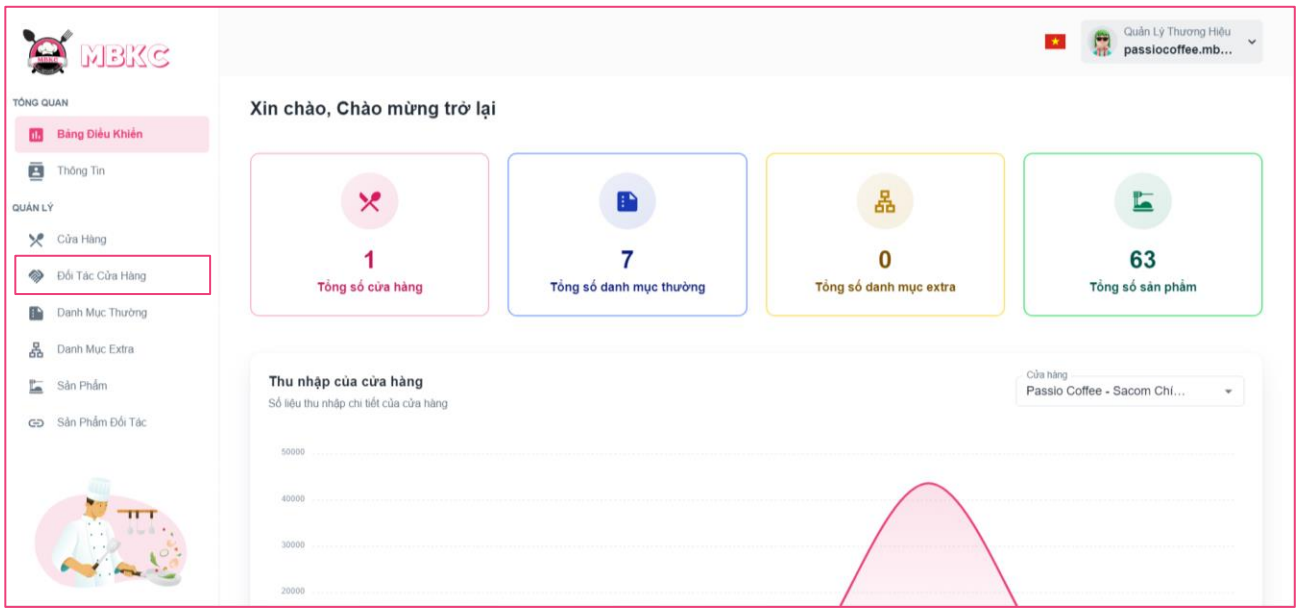


Figure 402: User Manual - Create new Store Partner – Step 3

Step 4: Click “Thêm Đối Tác Cửa Hàng” button on the screen and navigate to Create New Store Partner Page to create a new store partner.

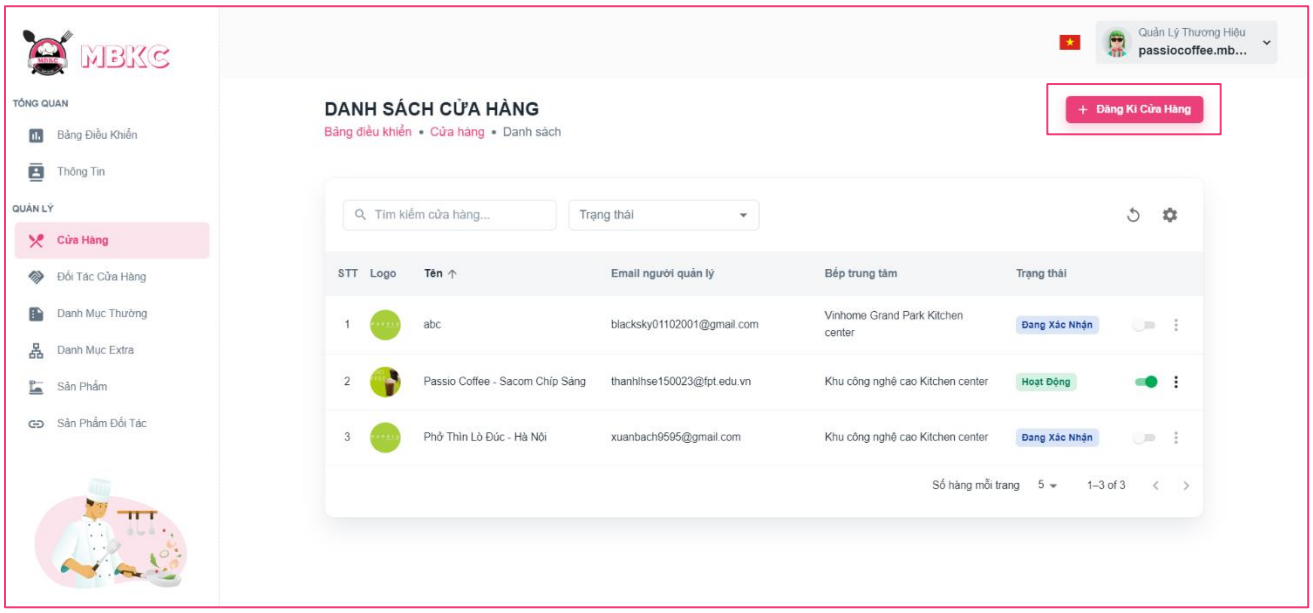


Figure 403: User Manual - Create new Store Partner - Step 4

Step 5: Filling all the required information to create a new store partner including: “Cửa hàng", “Đối tác", “Tên tài khoản", “Mật khẩu”, “Phầm trắm hoa hồng”.

Case 1: Create a new store partner between one store with one partner.

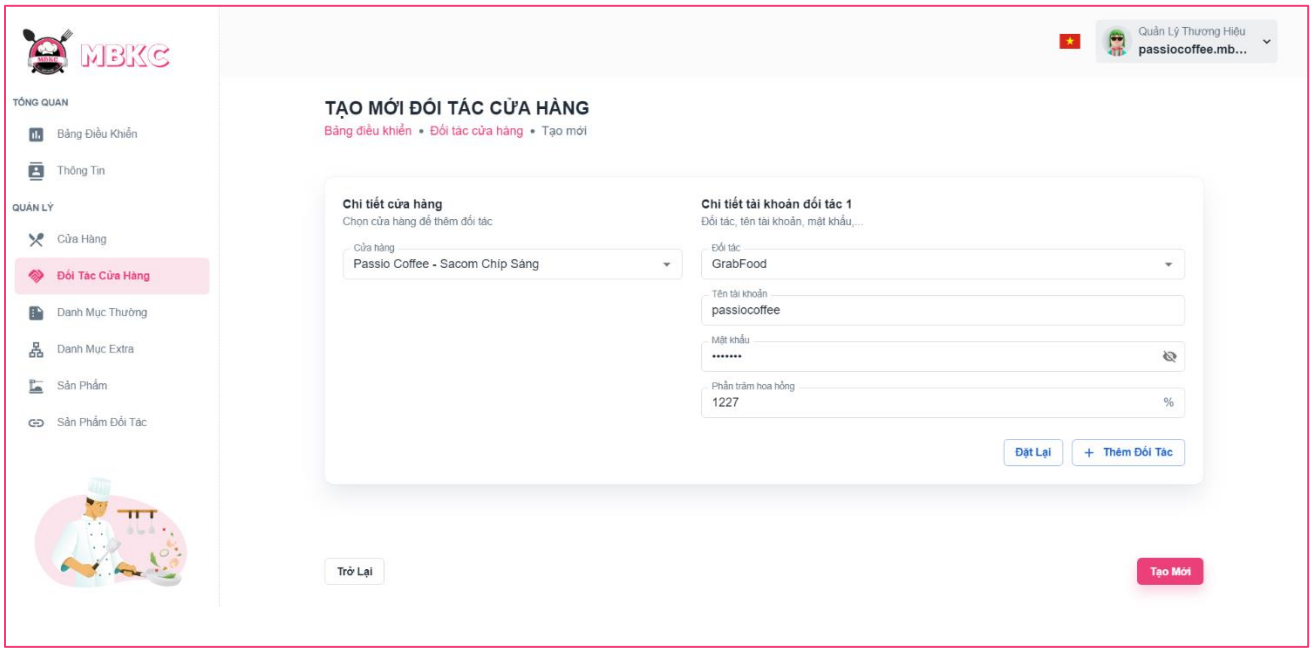


Figure 404: User Manual - Create new Store Partner - Step 5 – Case 1

Case 2: If you want to create a new store partner between one store with two or more partner. Click “Thêm Đối Tác” button to add new form for partner.

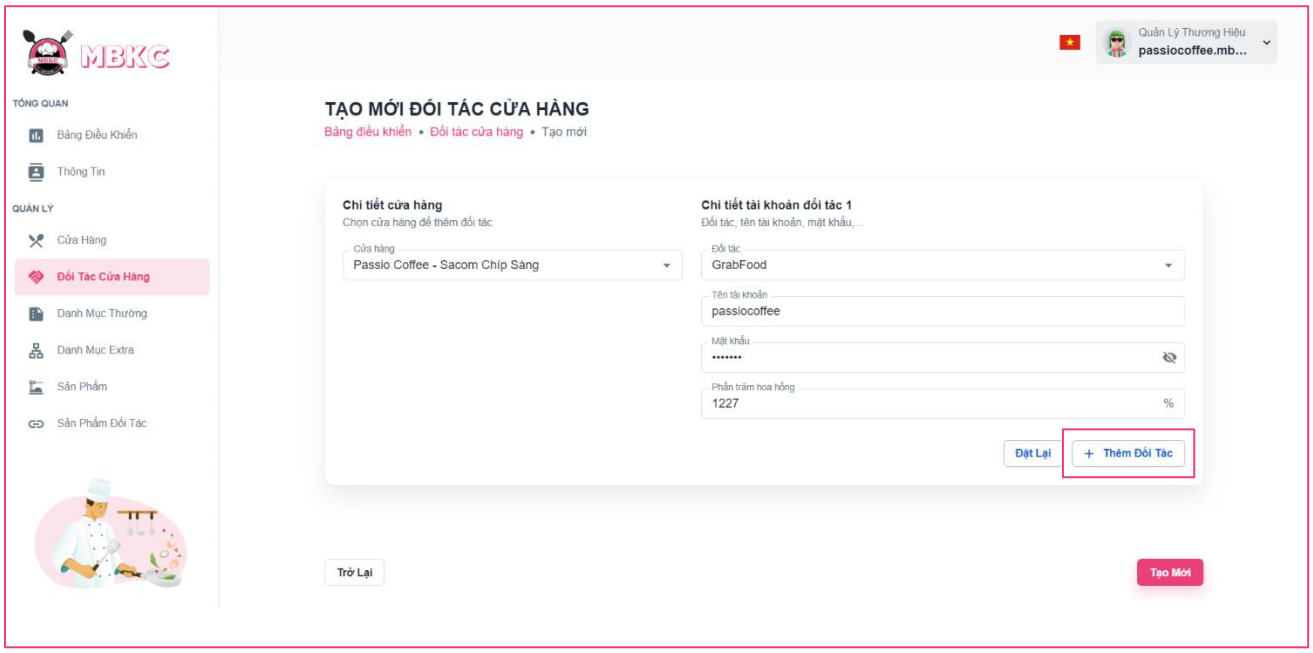


Figure 405: User Manual - Create new Store Partner - Step 5 – Case 2 -1

New form for partners was added.

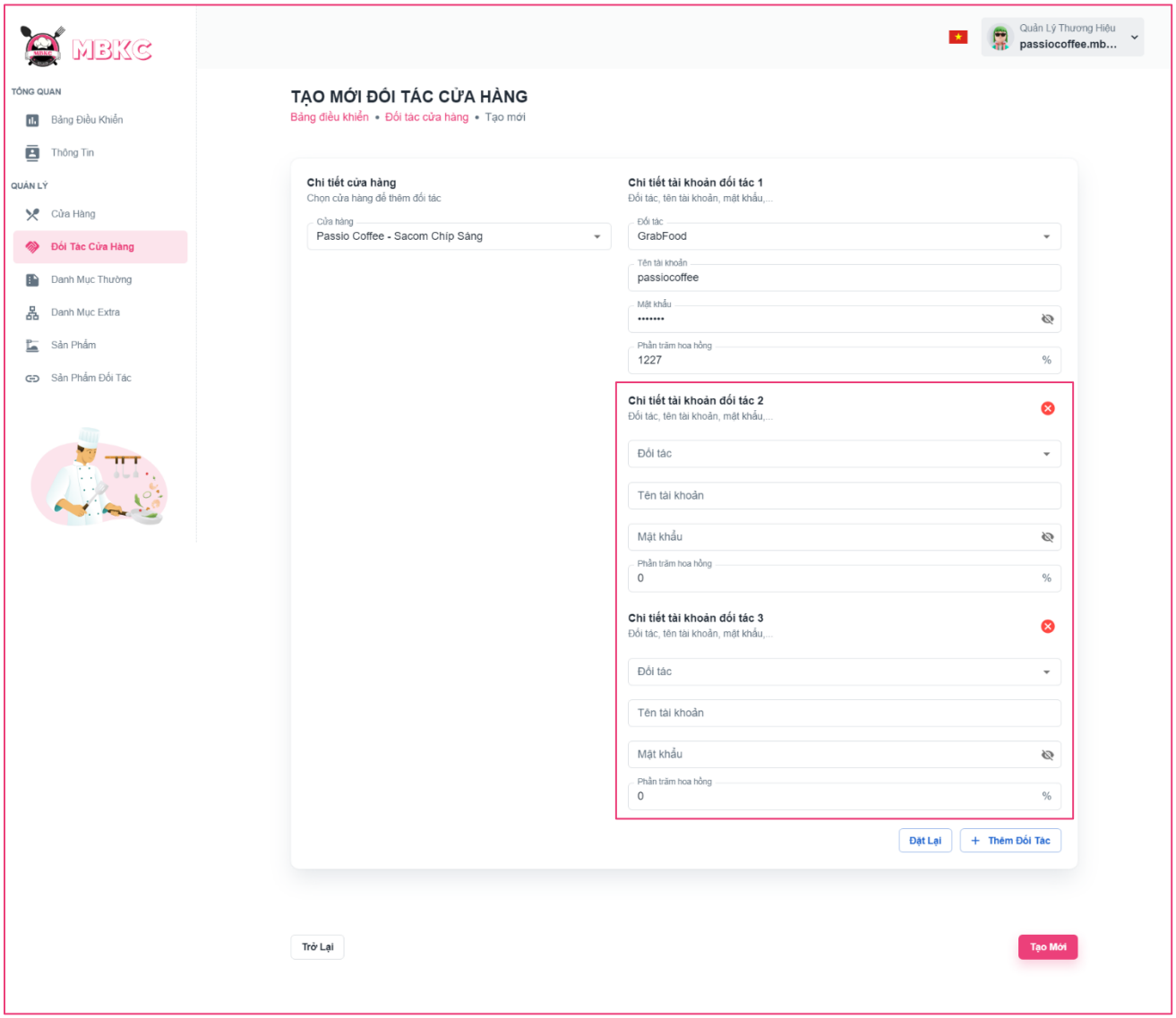


Figure 406: User Manual - Create new Store Partner - Step 5 - Case 2 – 2

Case 3: If you want to create a new store partner between one store with one partner.

+ Click “Đặt Lại” button to reset form for create store partner.

+ Or click Cancel icon button to delete form for partner you want.

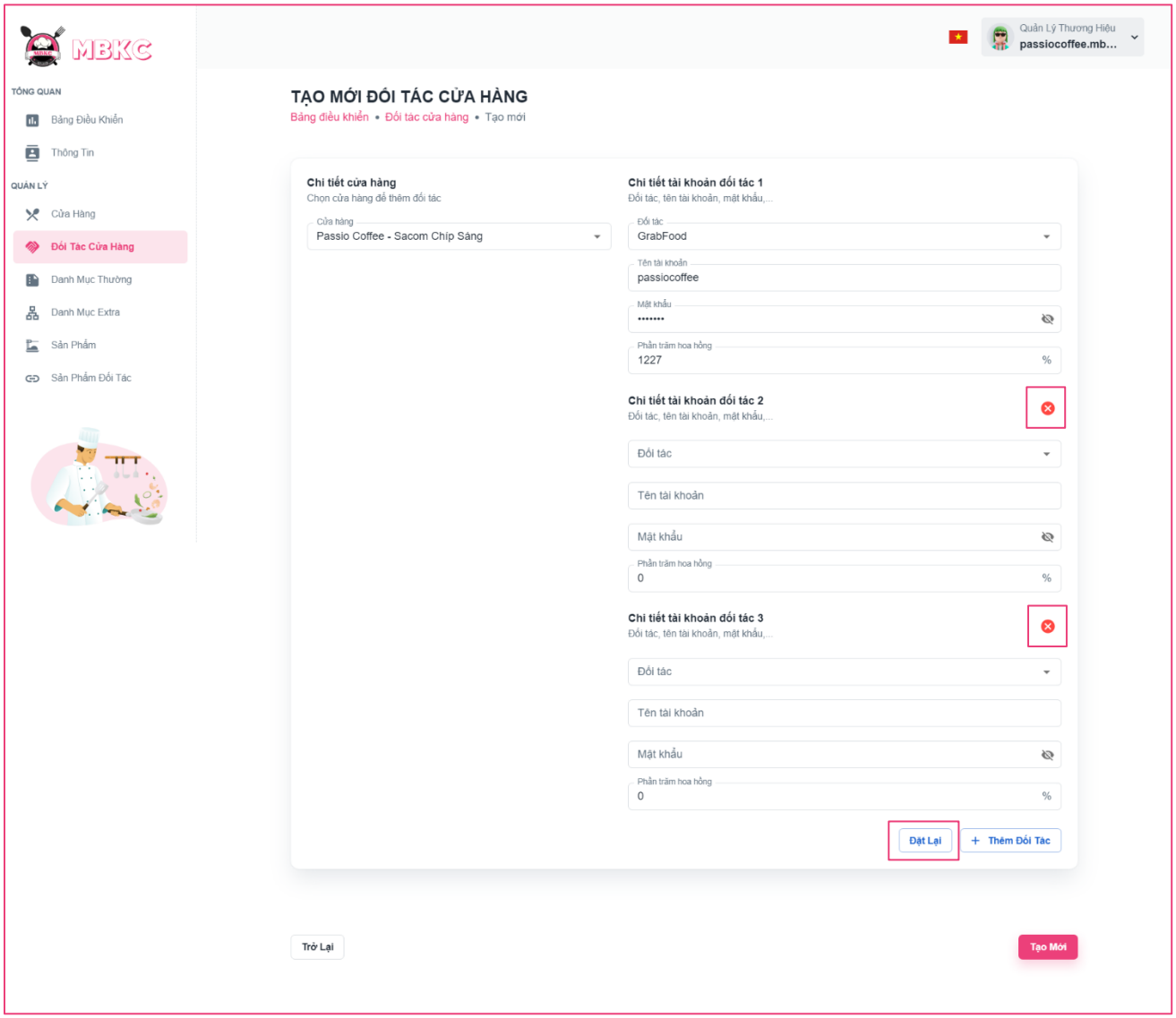


Figure 407: User Manual - Create new Store Partner - Step 5 - Case 3

Step 6: After filling in all required field, click “Đăng Kí” to finish register a new store.

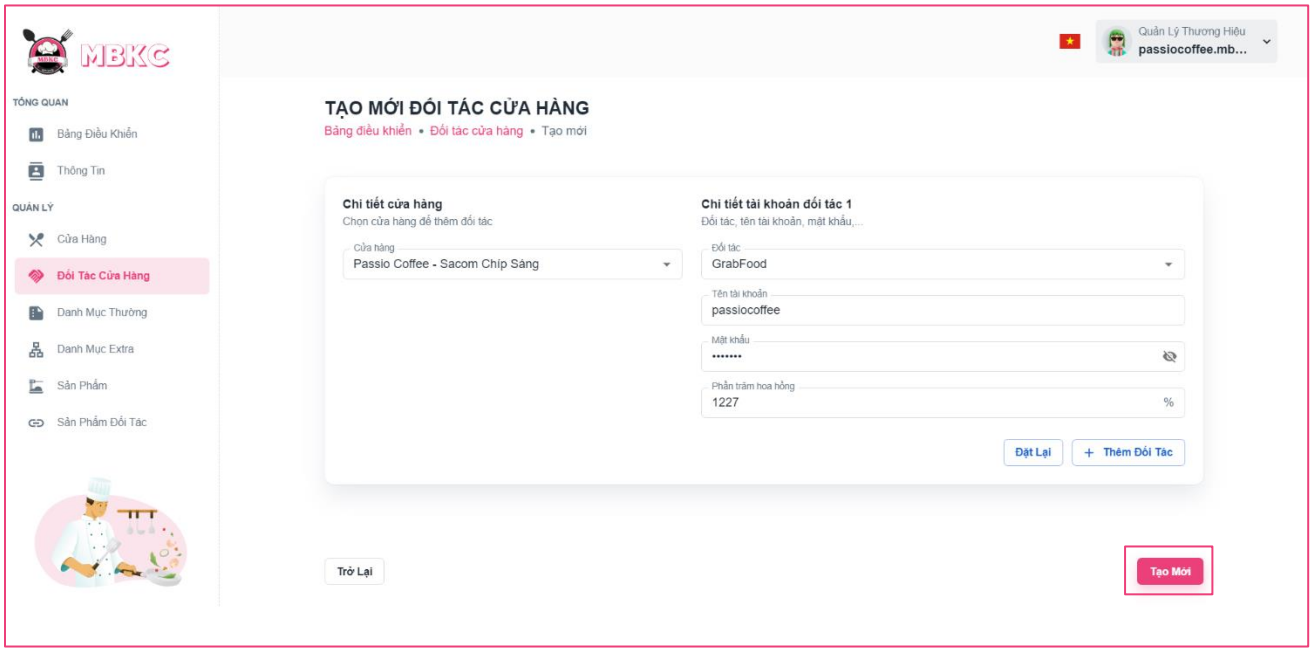


Figure 408: User Manual - Create new Store Partner - Step 6

You will be navigated to List of Store Partner Page showing the list of store partners and a toast message “Tạo mới đối tác cửa hàng thành công”.

3.2.9.2. View list Store Partner

3.2.9.2.1. Description

Use this guideline to search, reload data in the table list of store partners.

This web application supports view list of store partners with account as a Brand Manager.

3.2.9.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

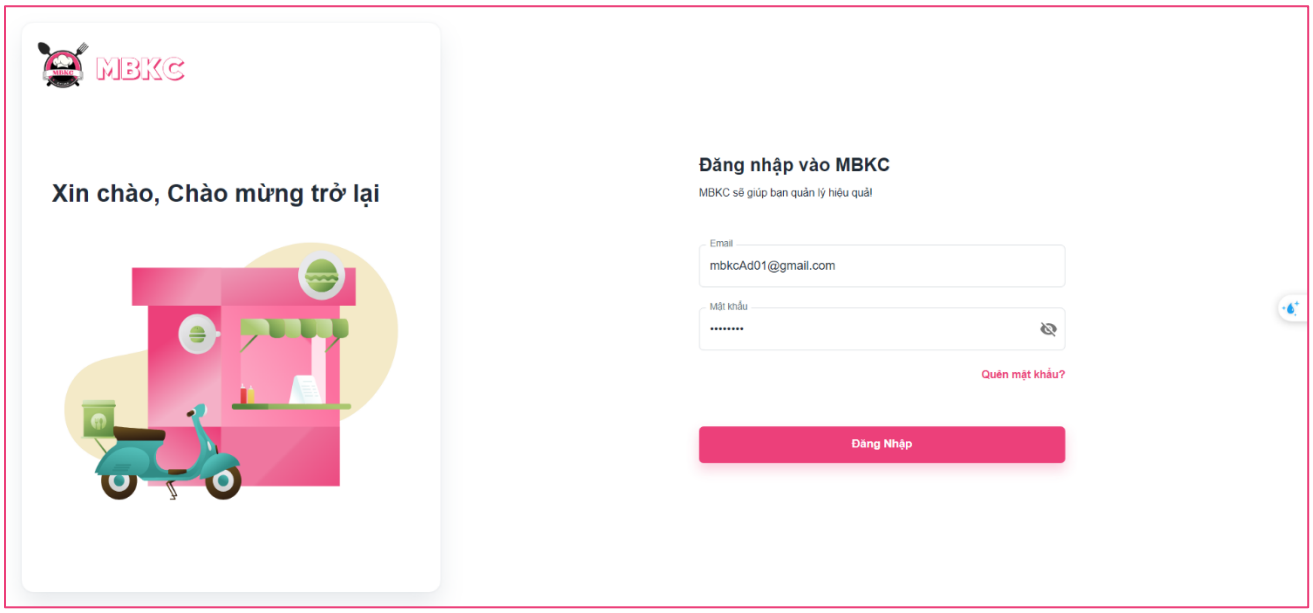


Figure 409: User Manual - View list of Store Partners - Step 1

Step 2: Click “Đăng Nhập” button.

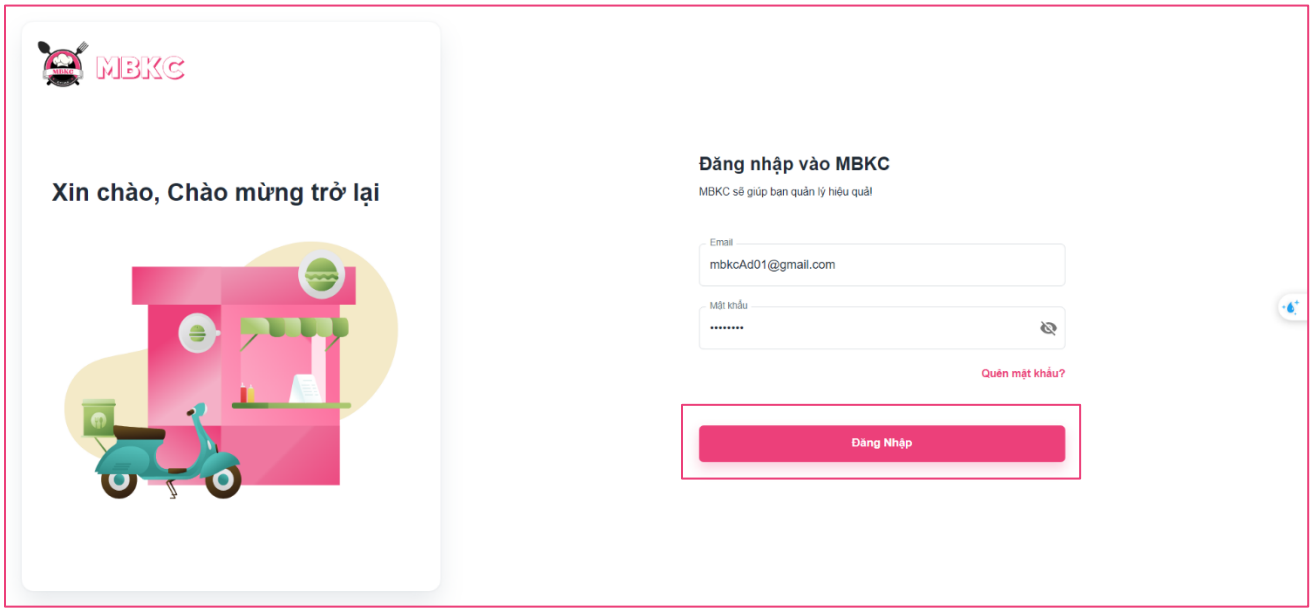


Figure 410: User Manual - View list of Store Partners - Step 2

Step 3: From Brand Dashboard Page, choose “Đối Tác Cửa Hàng” from the left sidebar.

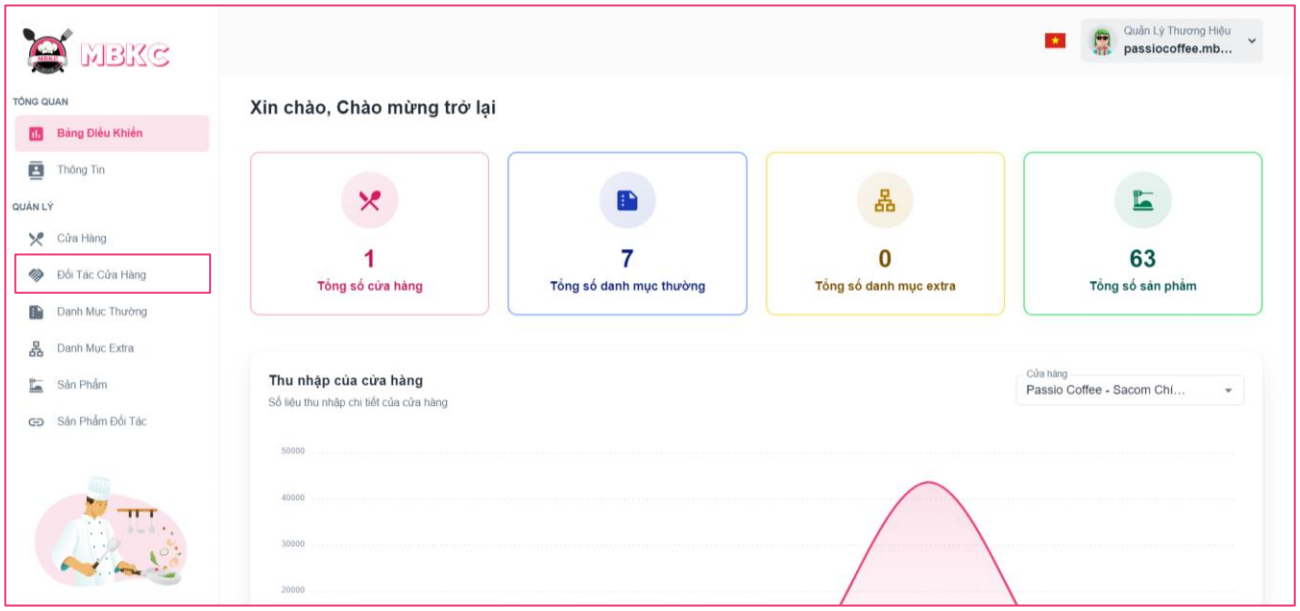


Figure 411: User Manual - View list of Store Partners - Step 3

Step 4: Search, filter store status, reload data in the table list of stores.

Case 1: Fill in the store name in the search box to search stores in the table of store partner list.

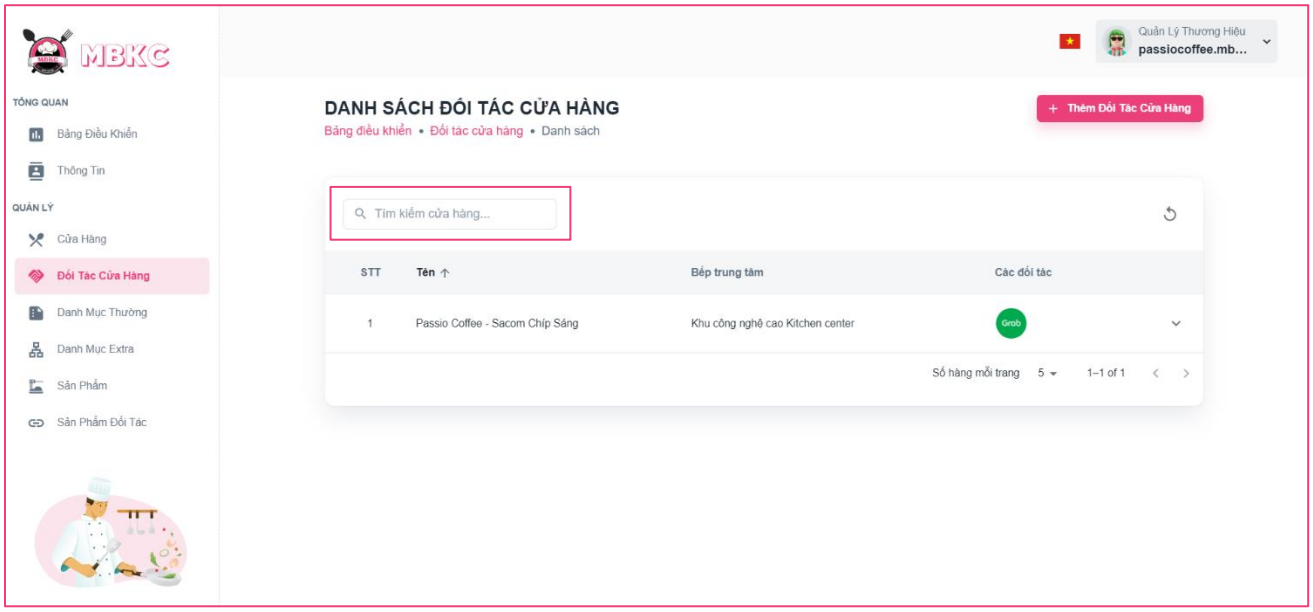


Figure 412: User Manual - View list of Store Partners - Step 4 - Case 1

Case 2: Click the Reload icon button to reload the list of store partners.

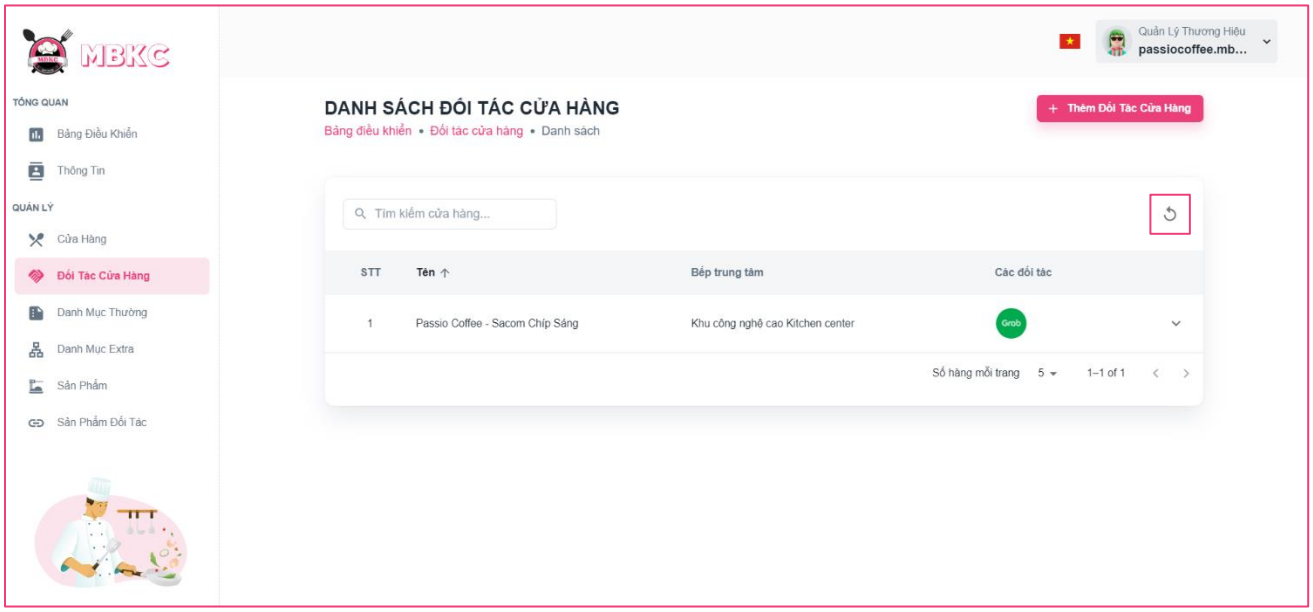


Figure 413: User Manual - View list of Store Partners - Step 4 - Case 2

Case 3: You want view information of partner, click the Arrow down icon button to show partner row.

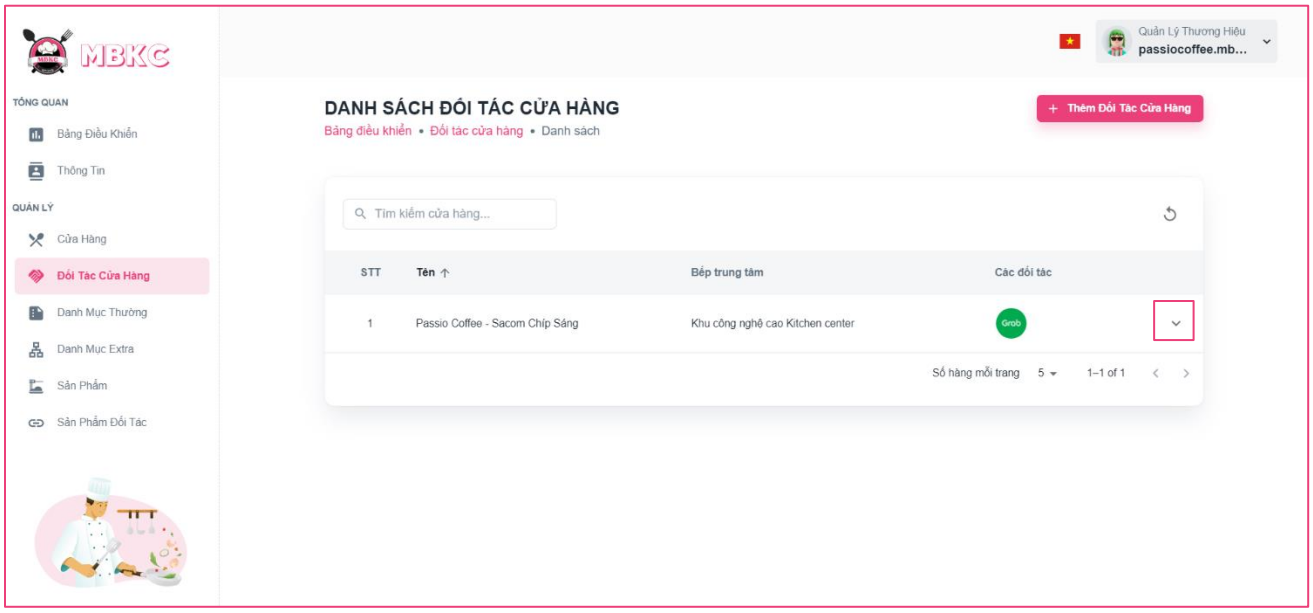


Figure 414: User Manual - View list of Store Partners - Step 4 - Case 3 - 1

View partner’s information in list of store partners.

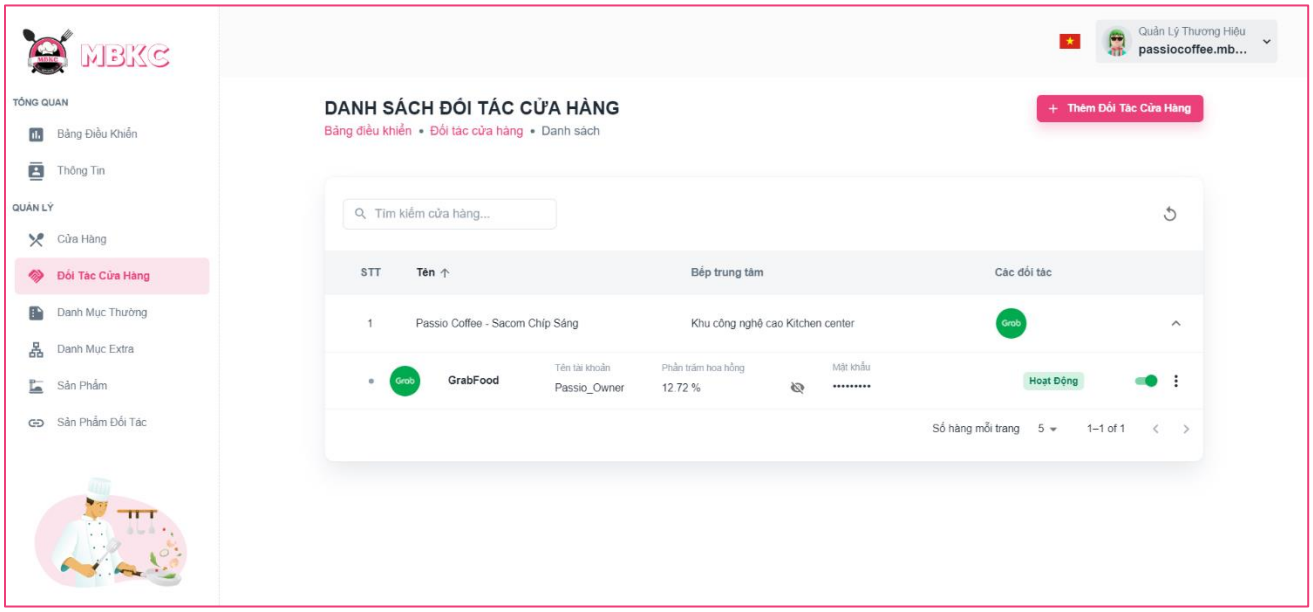


Figure 415: User Manual - View list of Store Partners - Step 4 - Case 3 – 2

3.2.9.3. Update Status Store Partner

3.2.9.3.1. Description

Use this guideline to update the status of the store partner.

This web application supports update the status of the store partner with account as a Brand Manager.

3.2.9.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Band Manager.

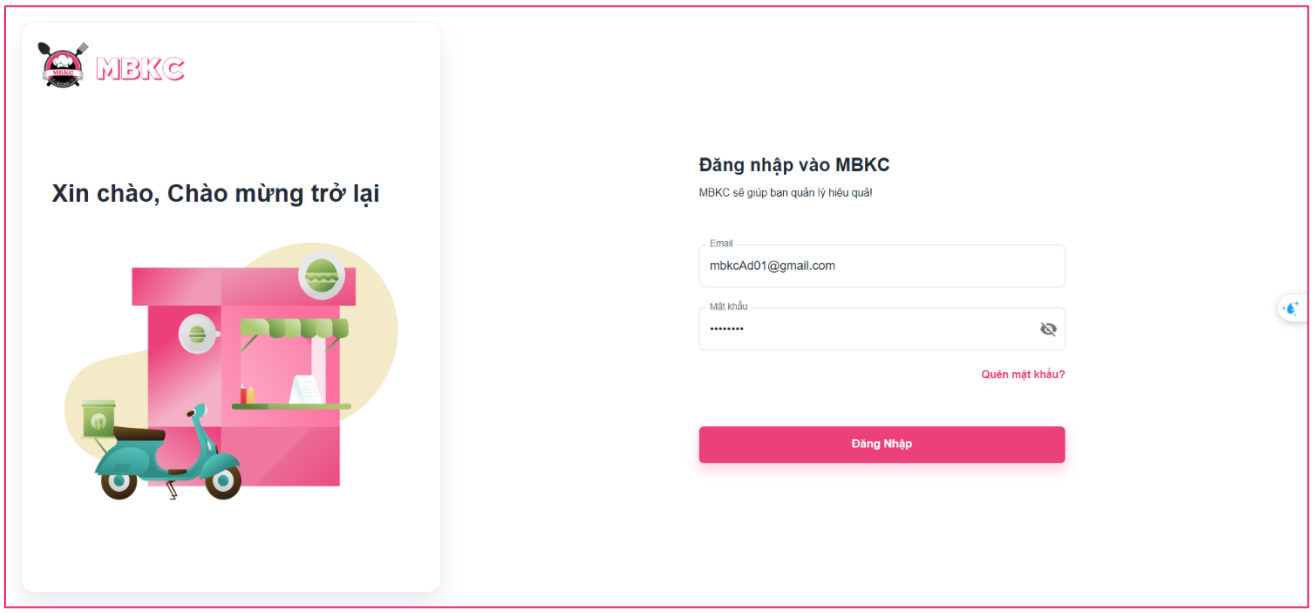


Figure 416: User Manual – Update Status Store Partner – Step 1

Step 2: Click “Đăng Nhập” button.

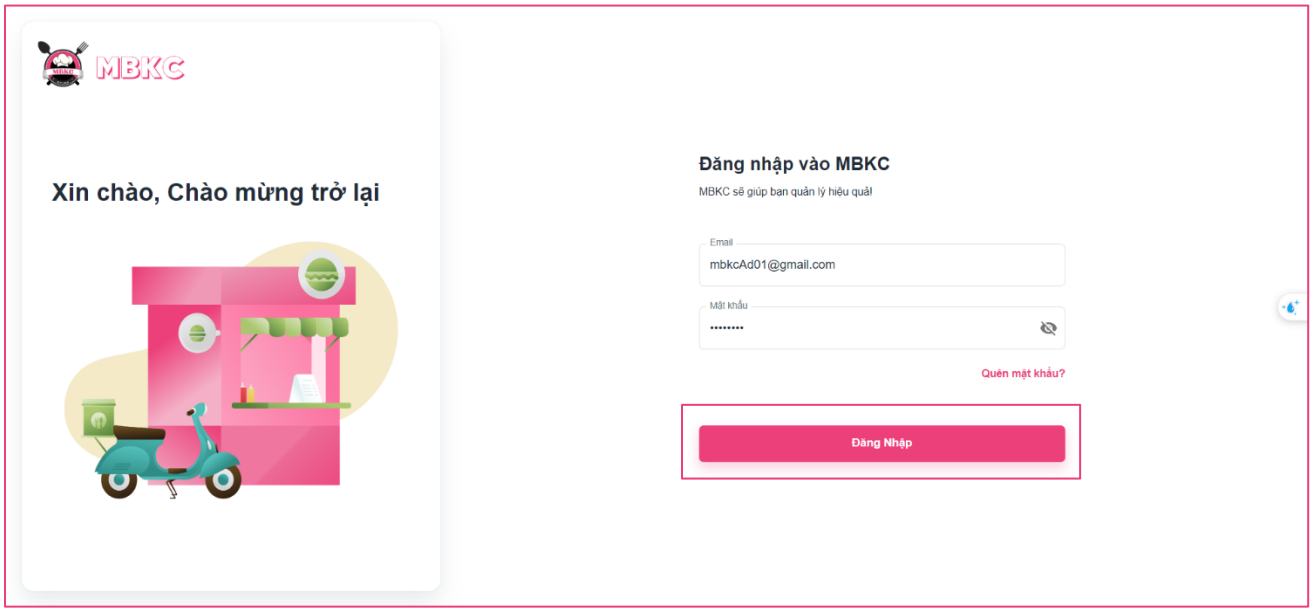


Figure 417: User Manual - Update Status Store Partner – Step 2

Step 3: From Brand Dashboard Page, choose “Đối Tác Cửa Hàng” from the left sidebar.

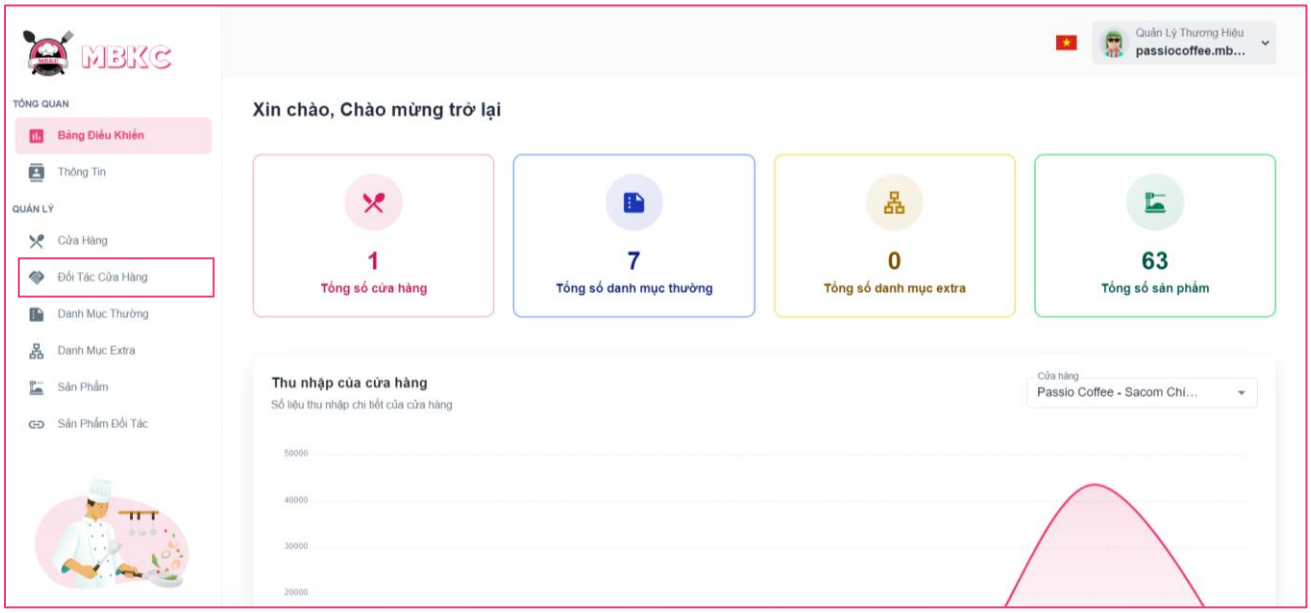


Figure 418: User Manual - Update Status Store Partner – Step 3

Case 1: Update status store partner in table list of store partners.

Step 4: Click on Arrow down icon button to show list of partners of store.

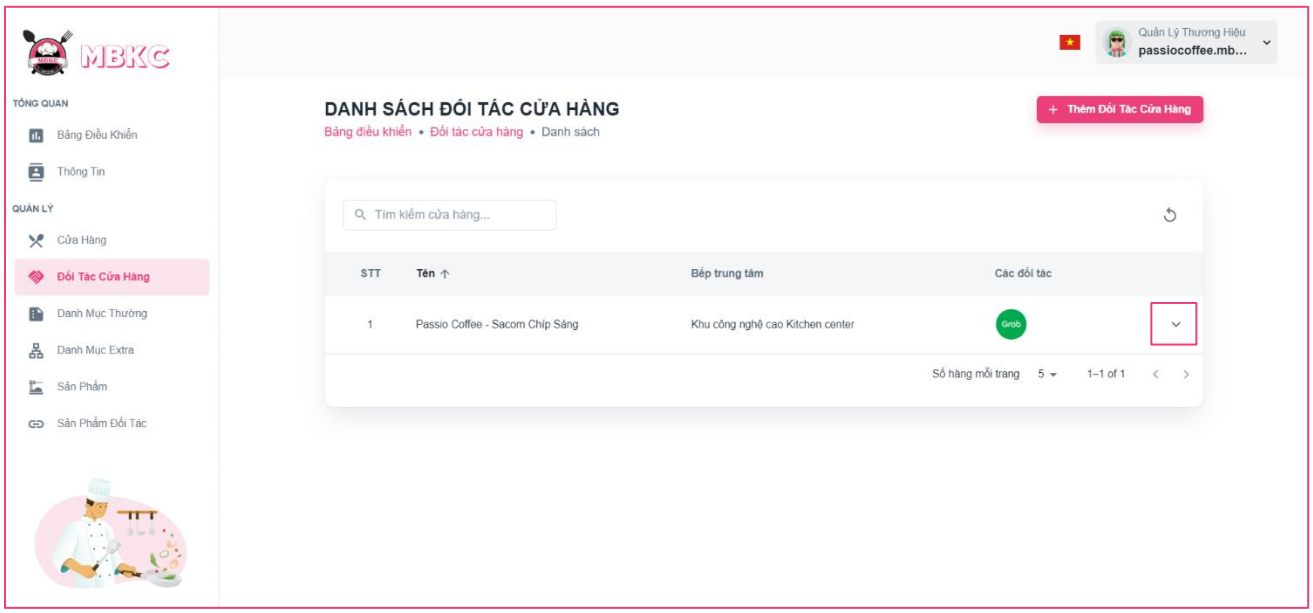


Figure 419: User Manual - Update Status Store Partner - Case 1 - Step 4

Step 5: Click on the switch button on the partner row to update the status.

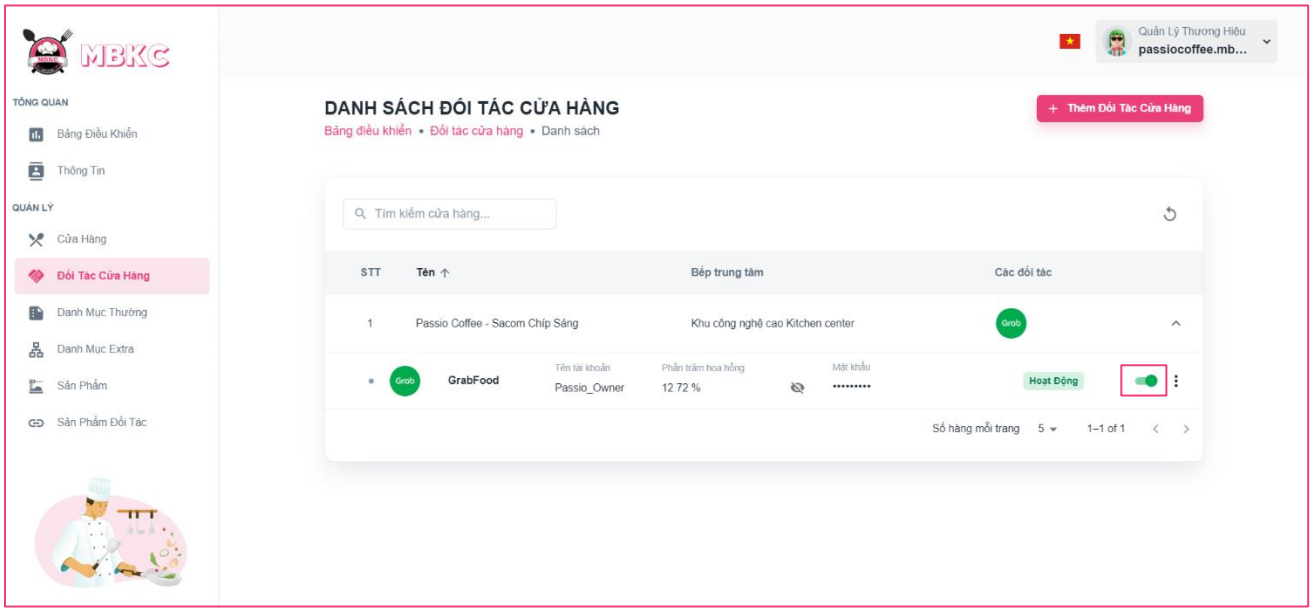


Figure 420: User Manual - Update Status Store Partner – Case 1 - Step 5

Step 6: List of store partners will be reloaded and a toast message "Đã cập nhật trạng thái đối tác cửa hàng thành công". And the status of the store partner has been updated.

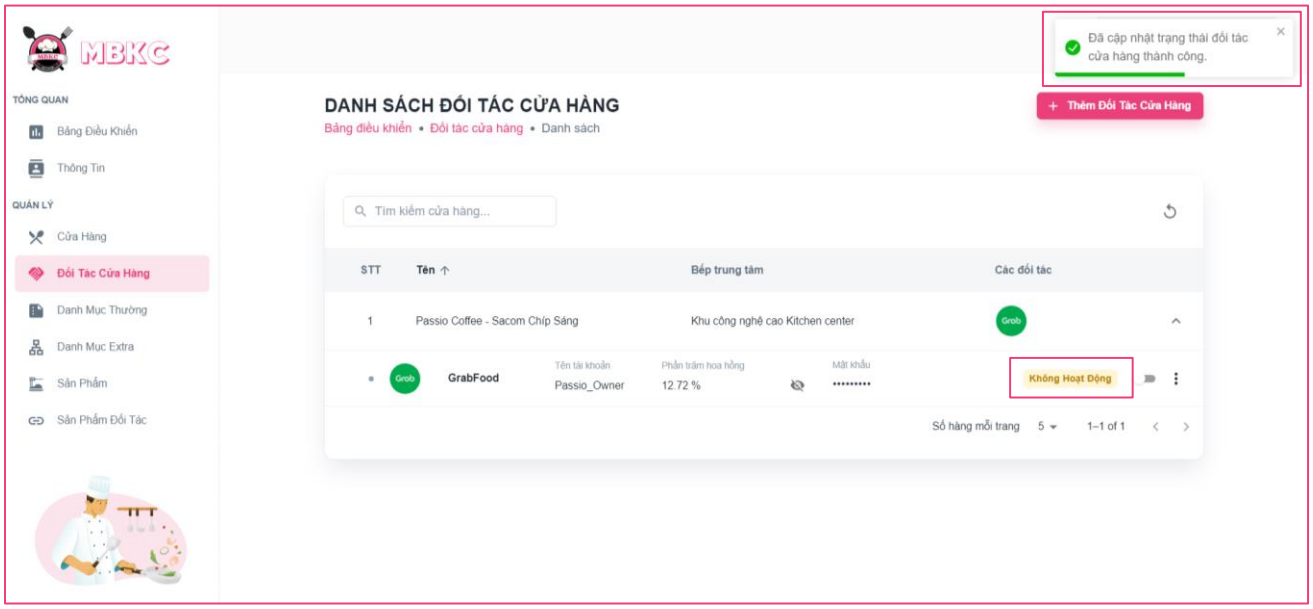


Figure 421: User Manual - Update Status Store Partner - Case 1 - Step 6

Case 2: Update status of store partner in Store Detail Page

Step 4: Click the store partner row to navigate to Store Details Page.

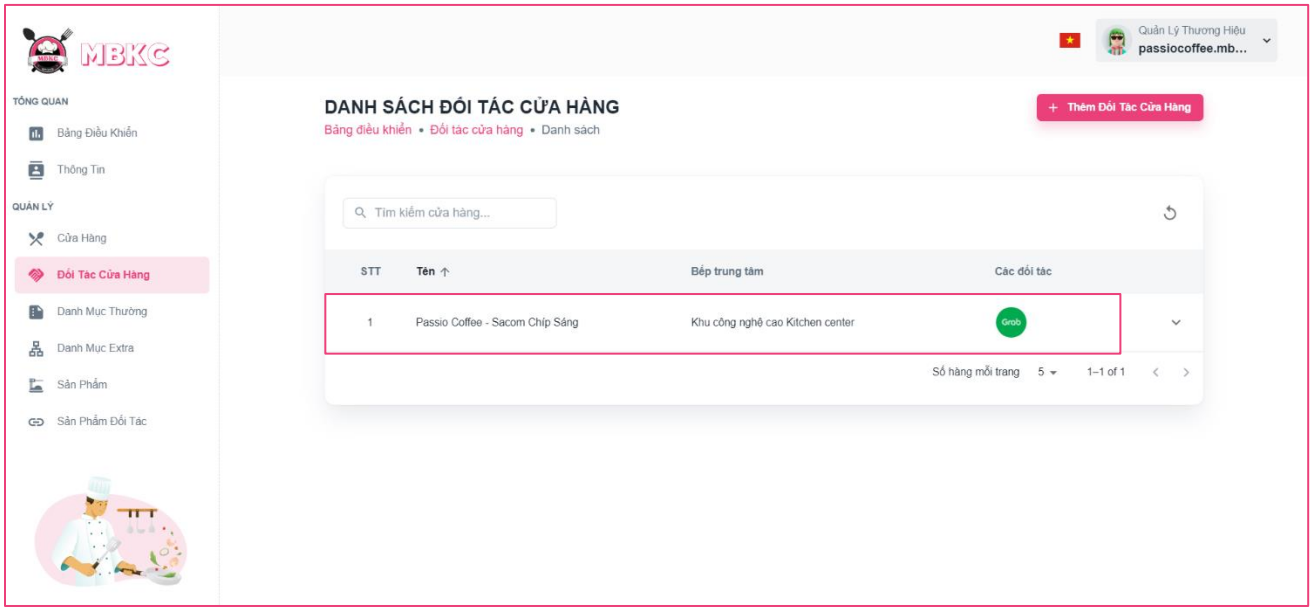


Figure 422: User Manual - Update Status Store Partner - Case 2 - Step 4

Step 5: Click on the switch button on the partner row to update the status.

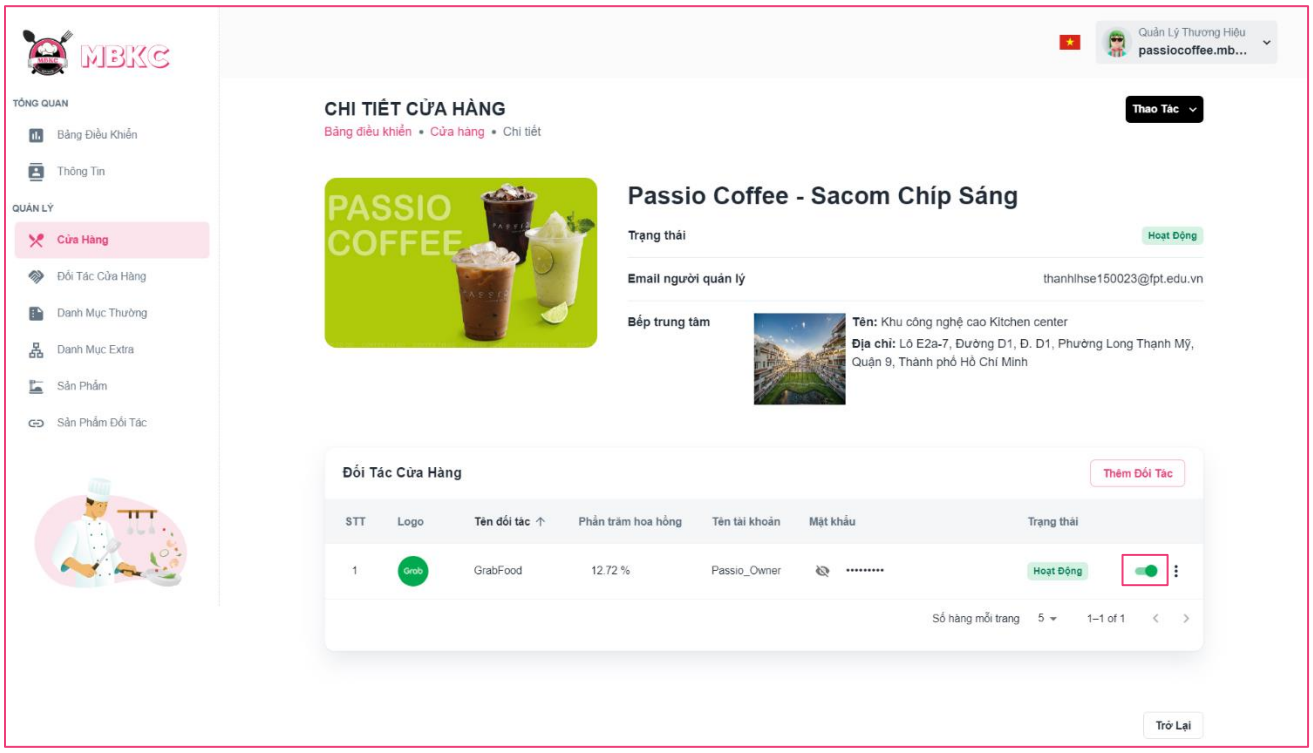


Figure 423: User Manual - Update Status Store Partner - Case 2 - Step 5

Step 6: Store details information will be reloaded and a toast message “Đã cập nhật trạng thái đối tác cửa hàng thành công”. And the status of the store partner has been updated.

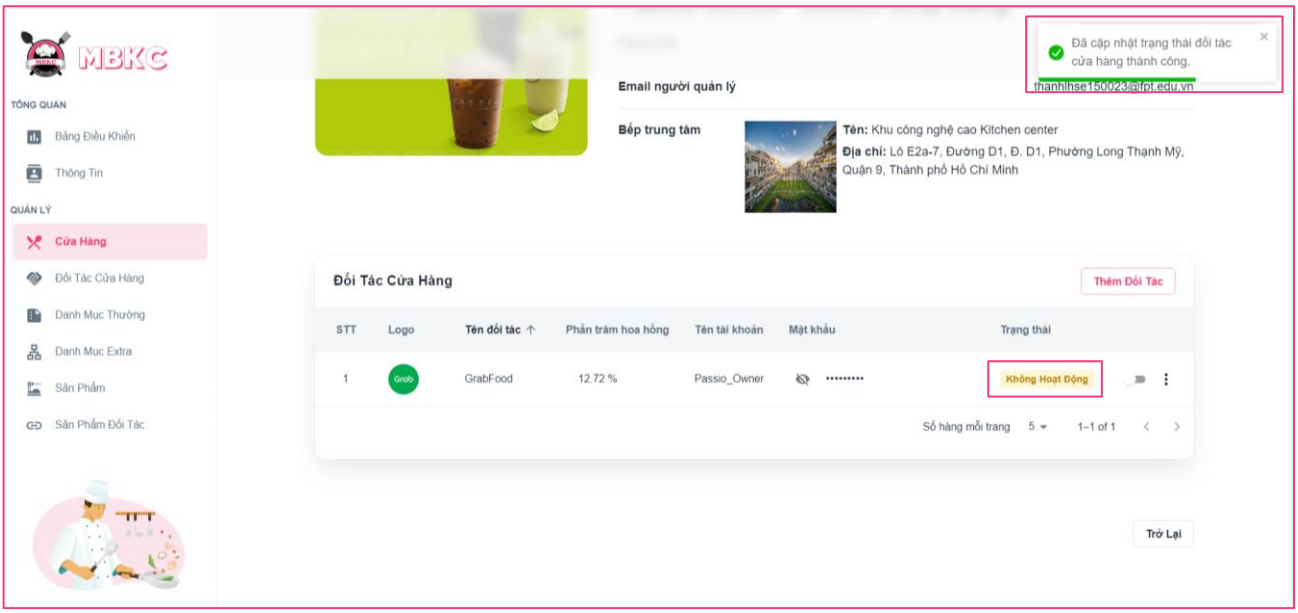


Figure 424: User Manual - Update Status Store Partner - Case 2 - Step 6

3.2.9.4. Update Information Store Partner

3.2.9.4.1. Description

Use this guideline to update information of the store partner.

This web application supports update information of the store partner with account as a Brand Manager.

3.2.9.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Band Manager.

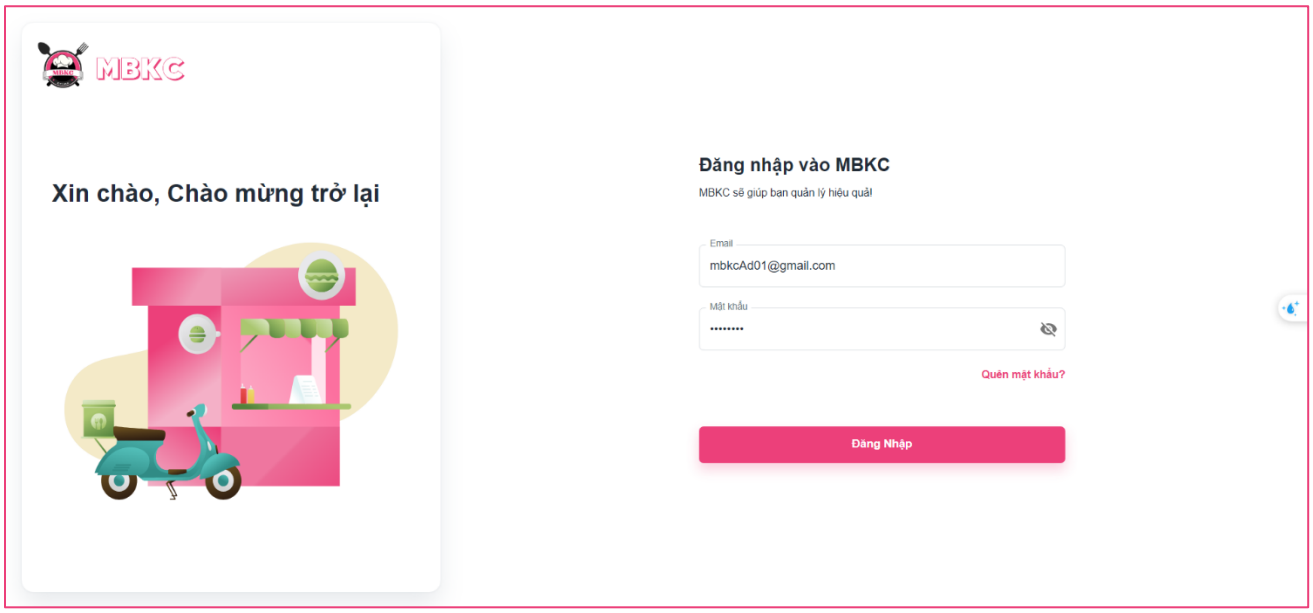


Figure 425: User Manual – Update Information Store Partner – Step 1

Step 2: Click “Đăng Nhập” button.

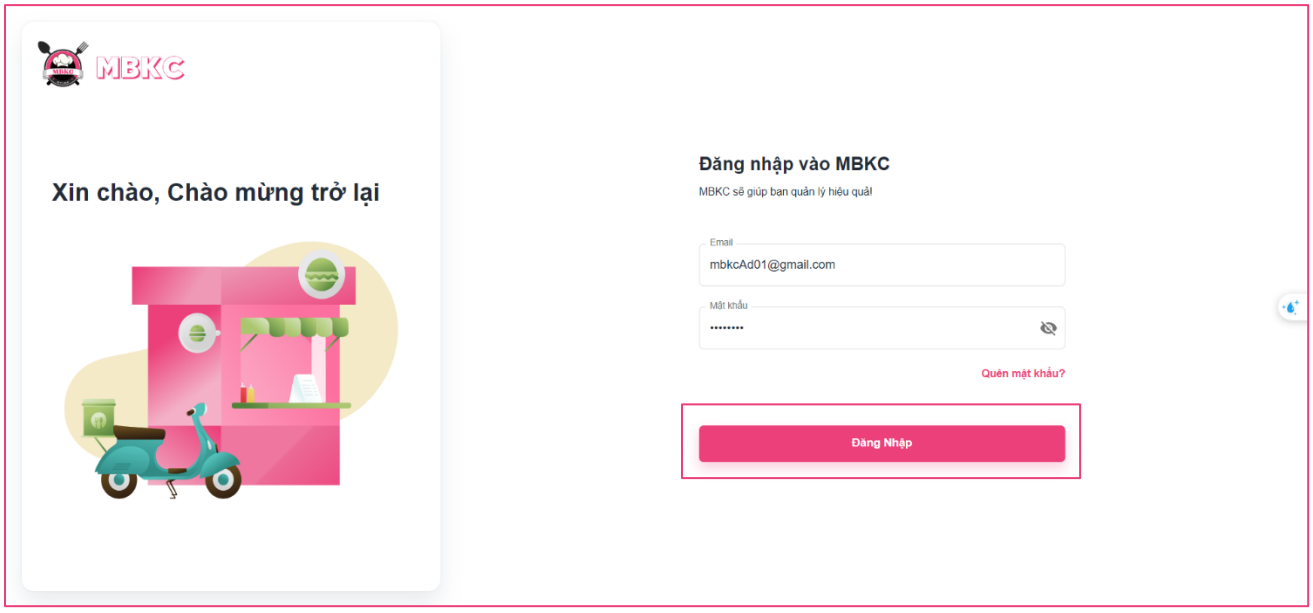


Figure 426: User Manual - Update Information Store Partner – Step 2

Step 3: From Brand Dashboard Page, choose “Đối Tác Cửa Hàng” from the left sidebar.

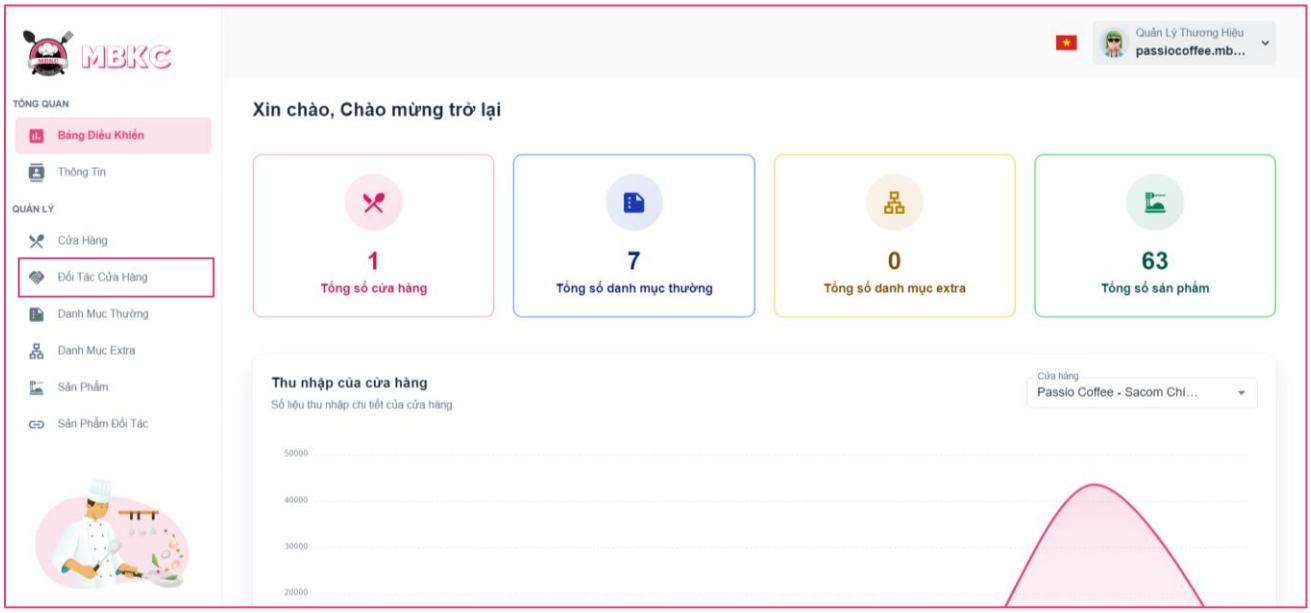


Figure 427: User Manual - Update Information Store Partner – Step 3

Step 4: Click on Arrow down icon button to show list of partners of store.

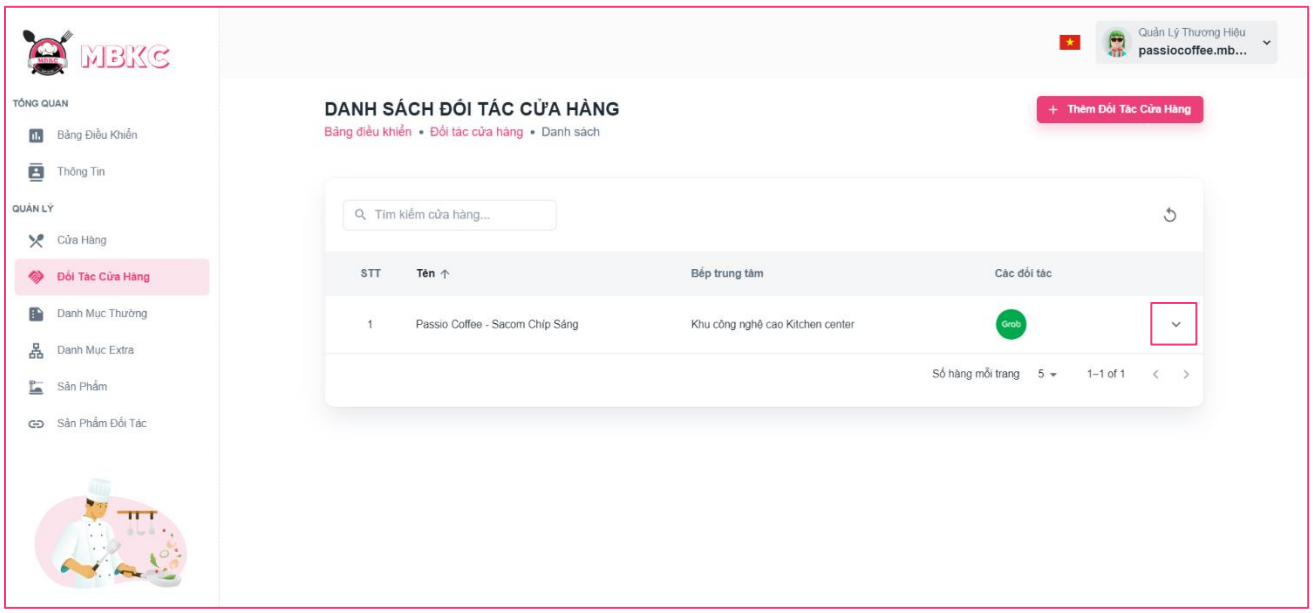


Figure 428: User Manual - Update Information Store Partner – Step 4

Step 5: Update information of the store partner.

Case 1: Update information of the store partner in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the store row want to navigate Update Store Page.

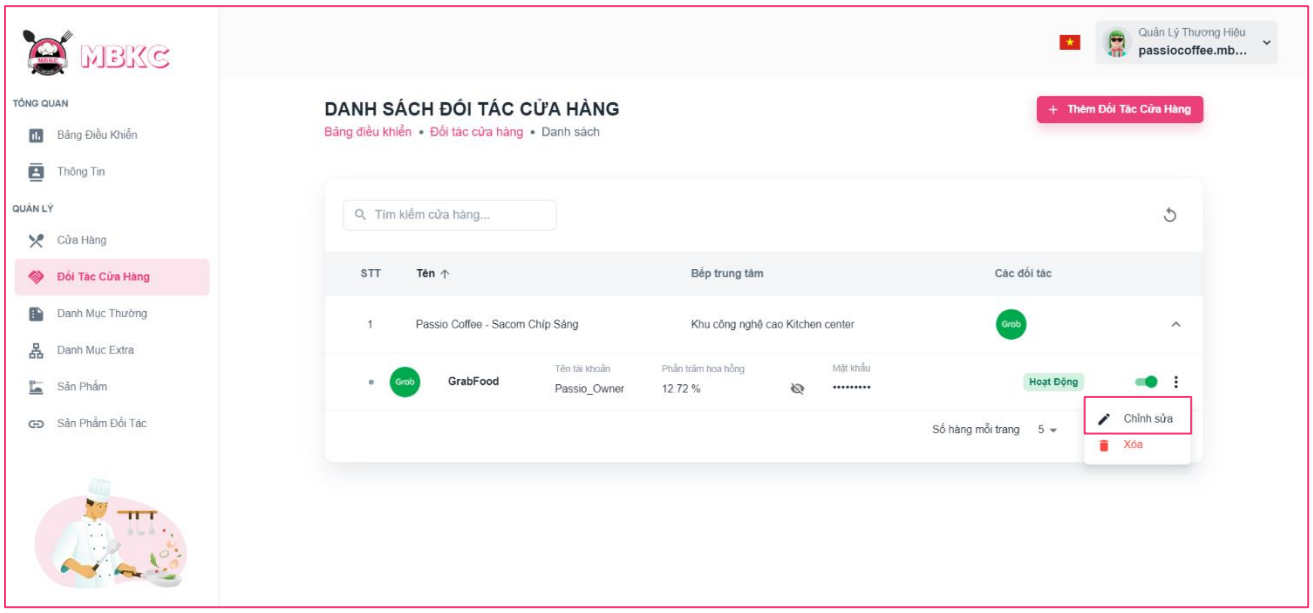


Figure 429: User Manual - Update Information Store Partner – Step 5 – Case 1

Case 2: Update information of the store partner on the Store Details Page.

1: Click the store partner row to navigate to Store Details Page.

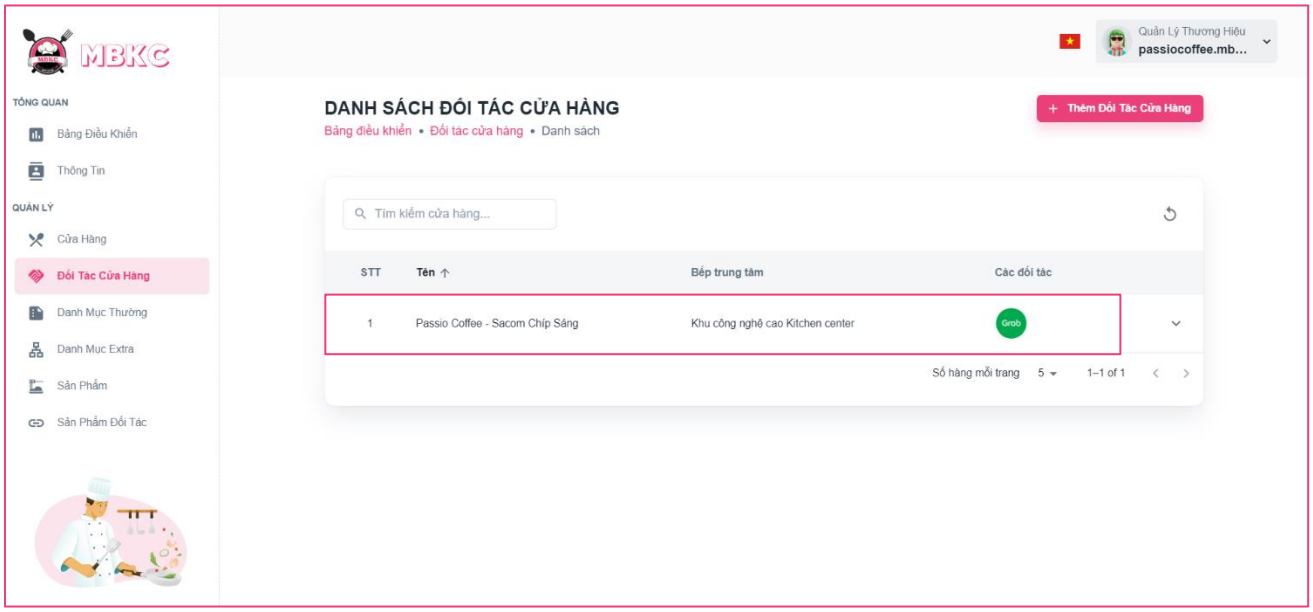


Figure 430: User Manual - Update Information Store Partner – Step 5 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Chỉnh sửa” menu item in partner row in table list of partners of store.

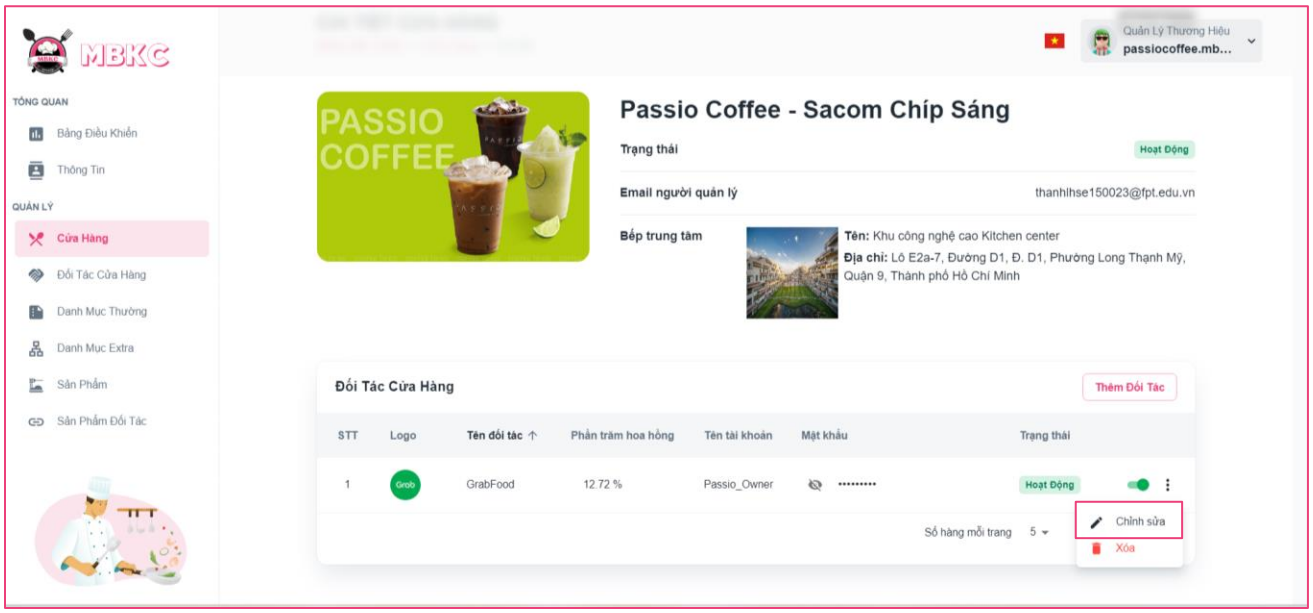


Figure 431: User Manual - Update Information Store Partner – Step 5 – Case 2 – 2

Step 5: Modal Update Partner Information will be showed, partner information will be automatically filled in the form. Then edit the information as desired.

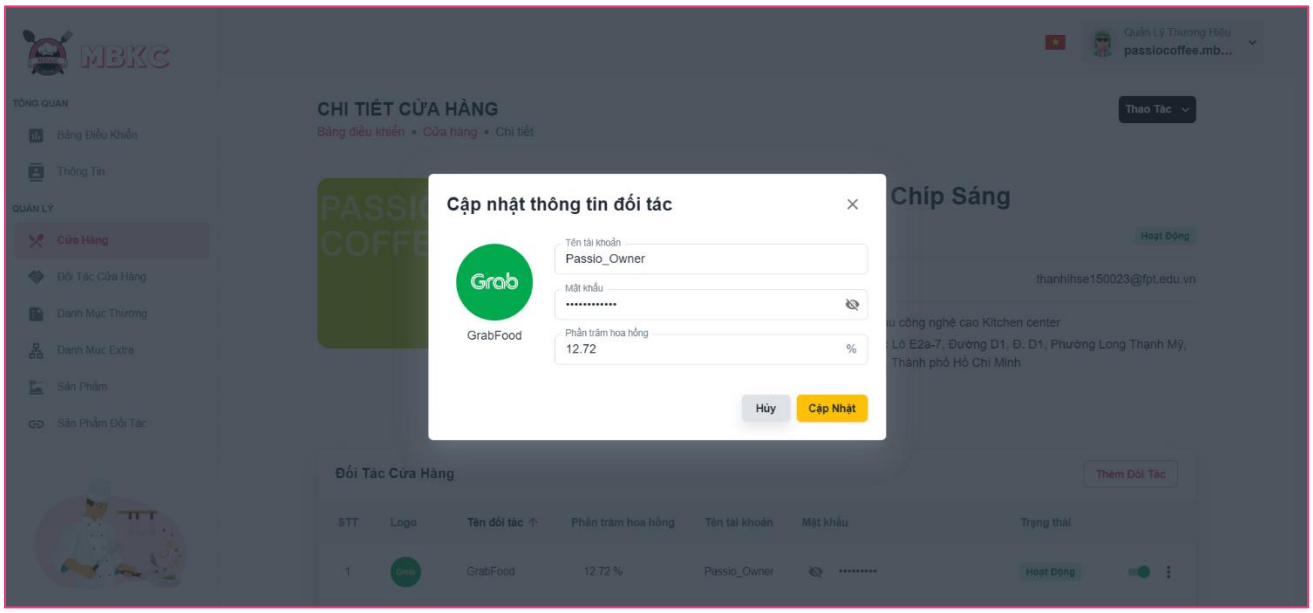


Figure 432: User Manual - Update Information Store Partner – Step 5

Step 6: Click “Cập nhật” button.

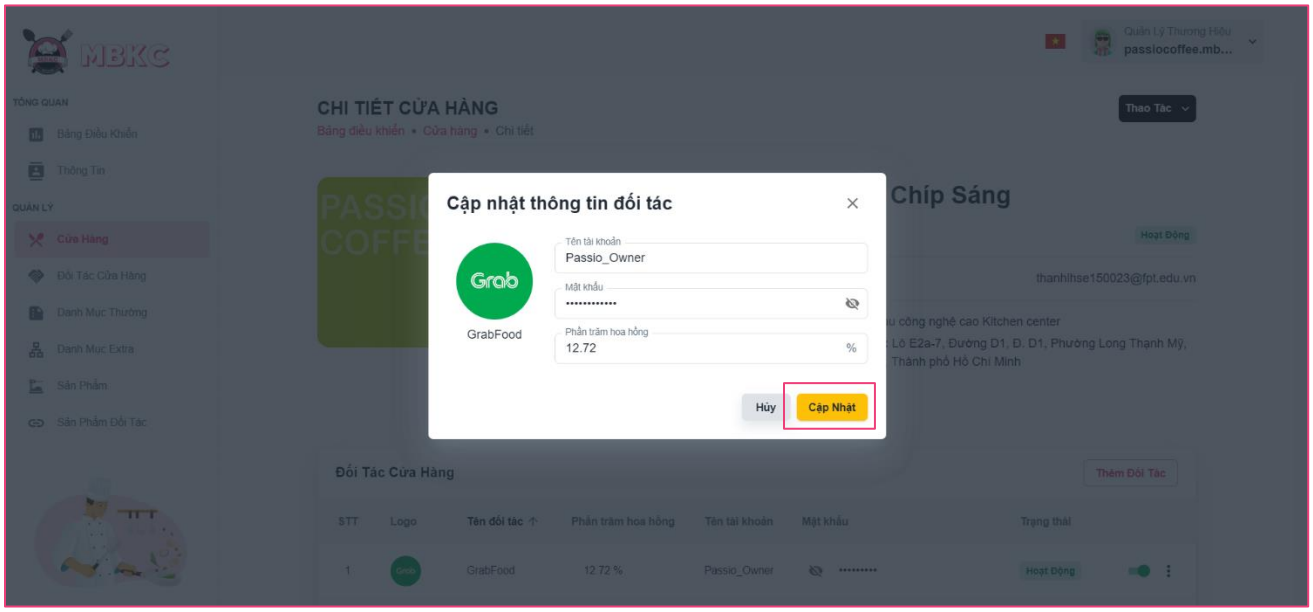


Figure 433: User Manual - Update Information Store Partner – Step 6

Step 7:
Case 1: Store Detail Page will be reloaded to showing the store details when start edit information of part in Store Detail Page and a toast message “Cập nhật thông tin đối tác cửa hàng thành công”.

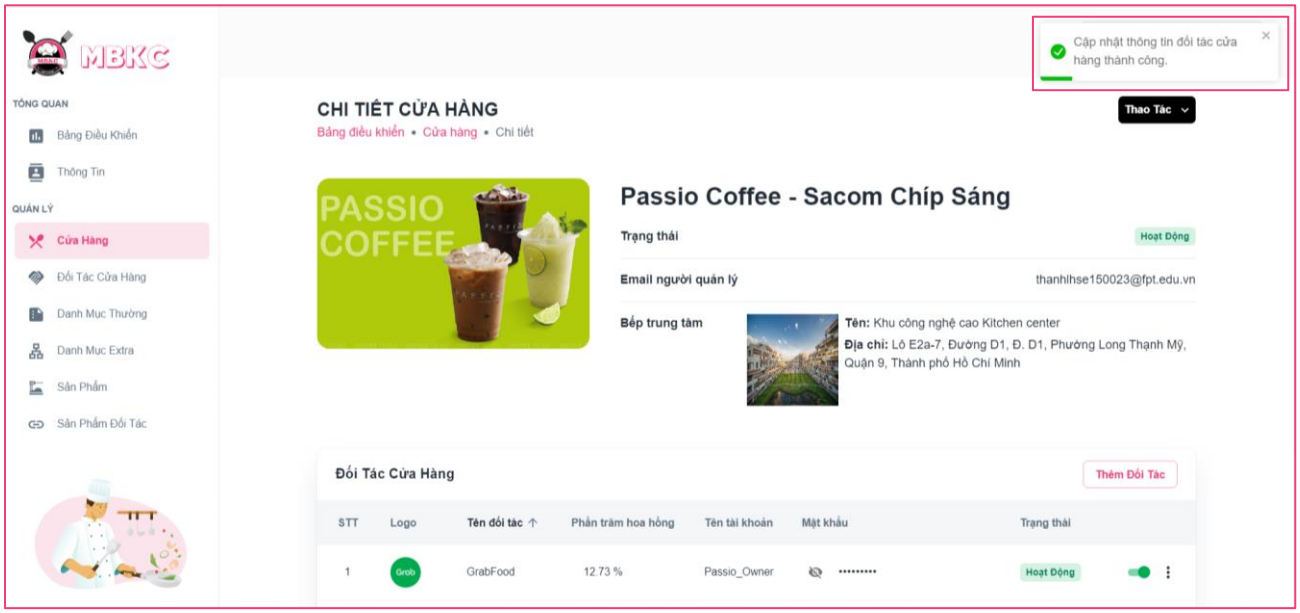


Figure 434: User Manual - Update Information Store Partner – Step 7 – Case 1

Case 2: List of Store Partner Page will be reloaded to showing the list of stores when start edit information of partner from List of Store Partner Page and a toast message “Cập nhật thông tin đối tác cửa hàng thành công”.

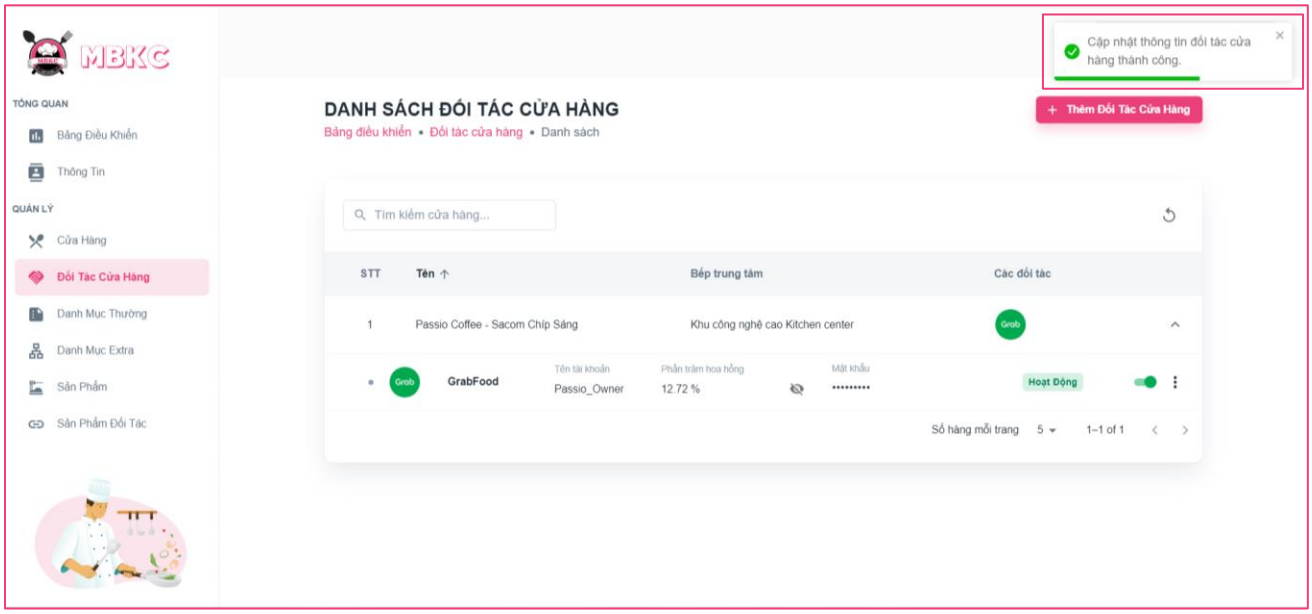


Figure 435: User Manual - Update Information Store Partner – Step 7 – Case 2

3.2.9.5. Delete Store Partner

3.2.9.5.1. Description

Use this guideline to delete store partner.

This web application supports update information of the store with account as a Brand Manager.

3.2.9.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

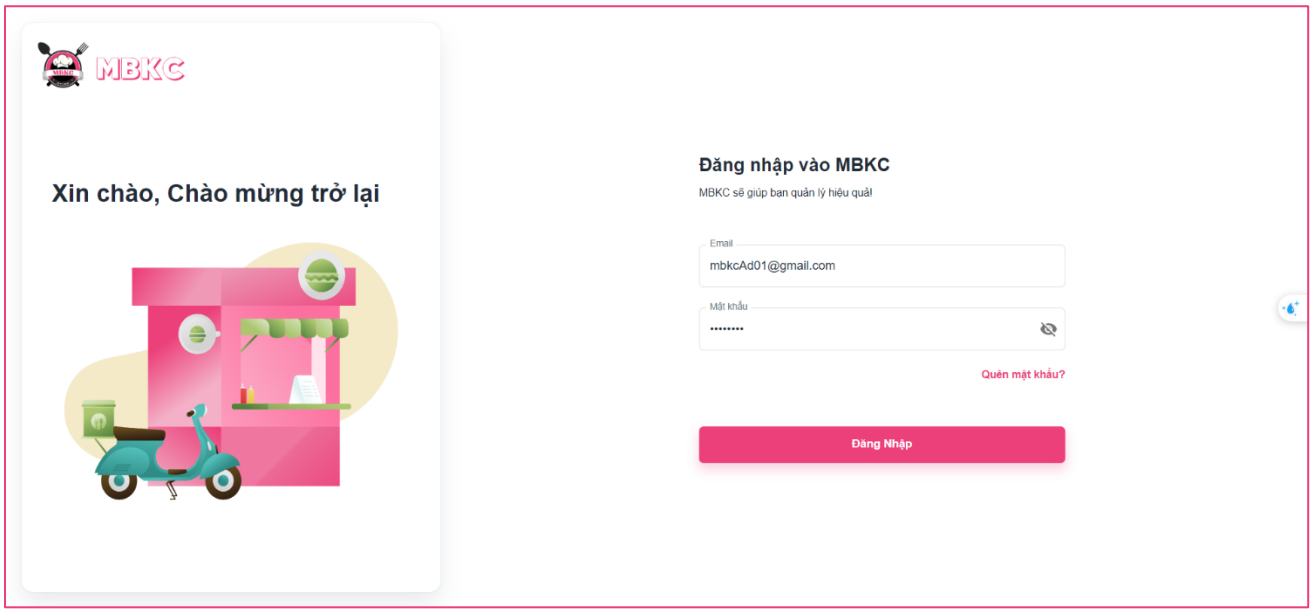


Figure 436: User Manual – Delete Store Partner – Step 1

Step 2: Click “Đăng Nhập” button.

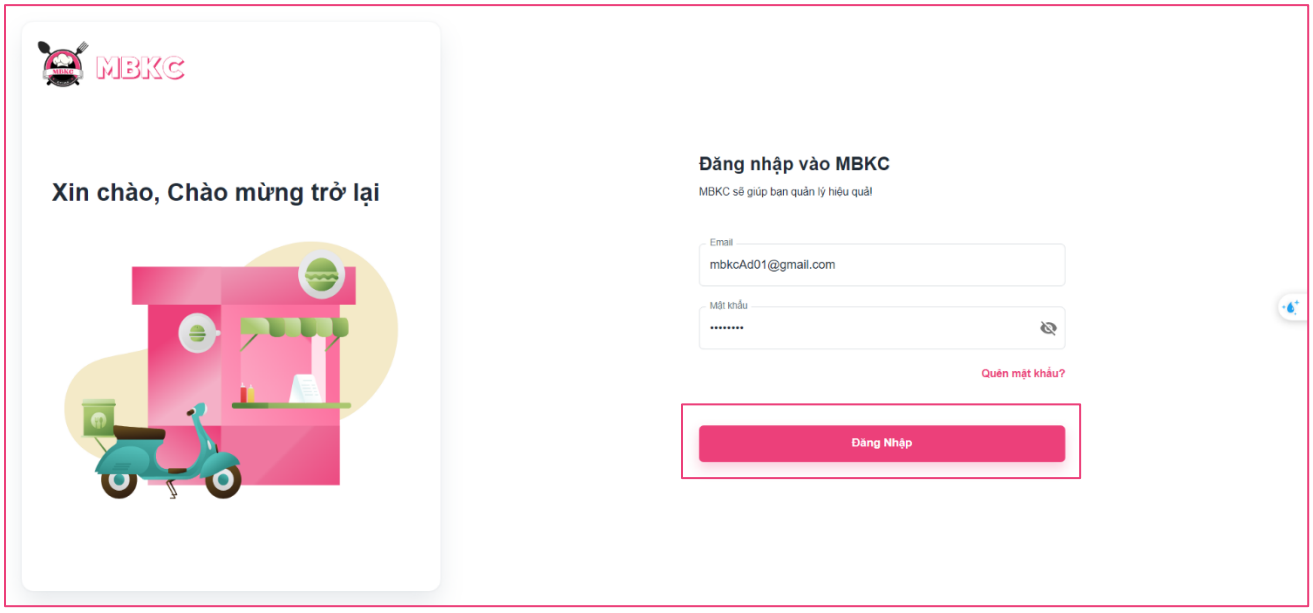


Figure 437: User Manual - Delete Store Partner – Step 2

Step 3: From Brand Dashboard Page, choose “Đối Tác Cửa Hàng” from the left sidebar.

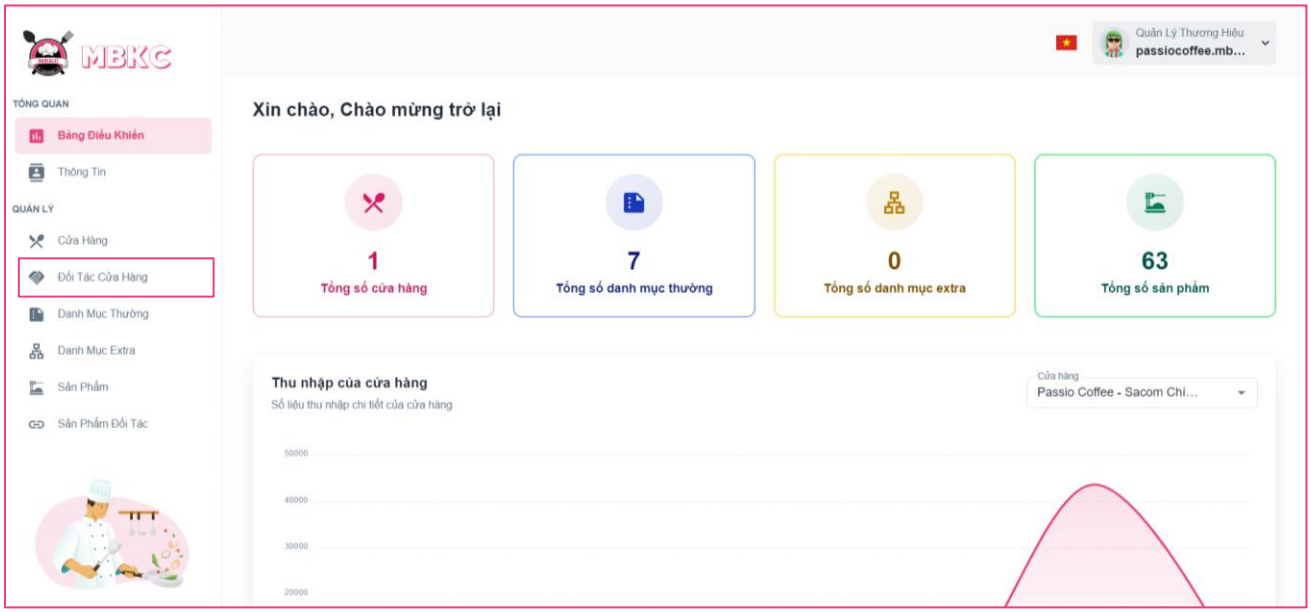


Figure 438: User Manual - Delete Store Partner – Step 3

Step 4: Click on Arrow down icon button to show list of partners of store.

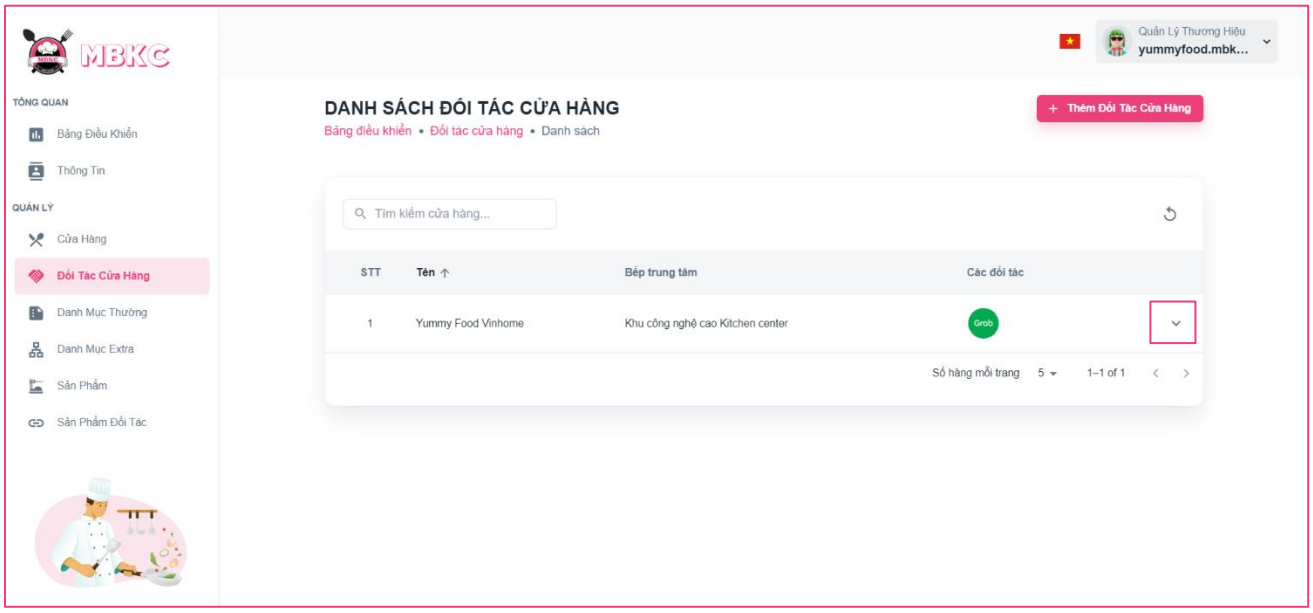


Figure 439: User Manual - Delete Store Partner – Step 4

Step 5: Delete store partner.

Case 1: Delete store partner in table list.

Click on the Menu icon button. Then click “Xóa” menu item to delete.

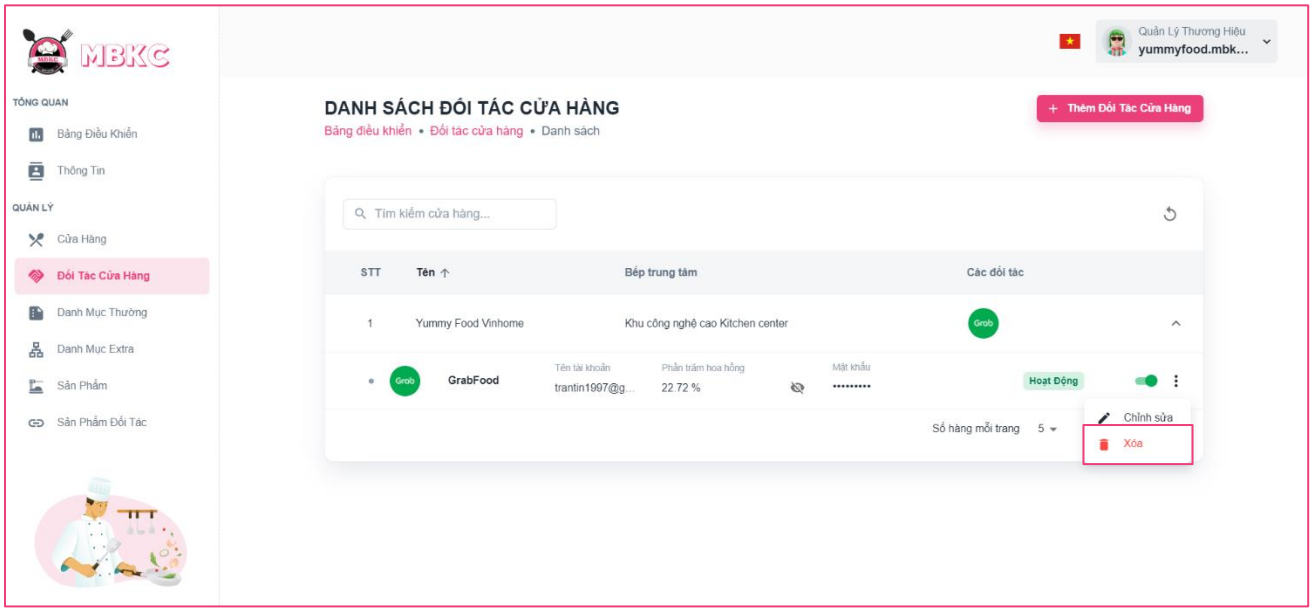


Figure 440: User Manual - Delete Store Partner – Step 5 – Case 1

Case 2: Delete partner on the Store Details Page.

1: Click the store partner row to navigate to Store Details Page.

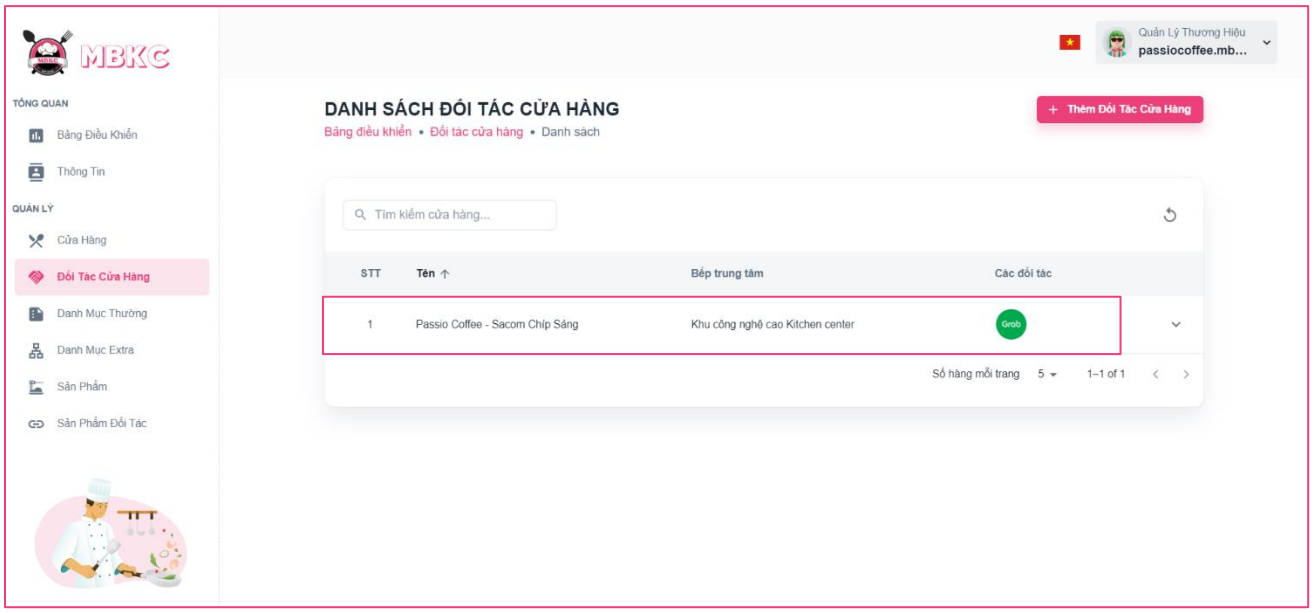


Figure 441: User Manual - Delete Store Partner – Step 5 – Case 2 – 1

2: Click on Menu icon button. Then click “Xóa” menu item to delete.

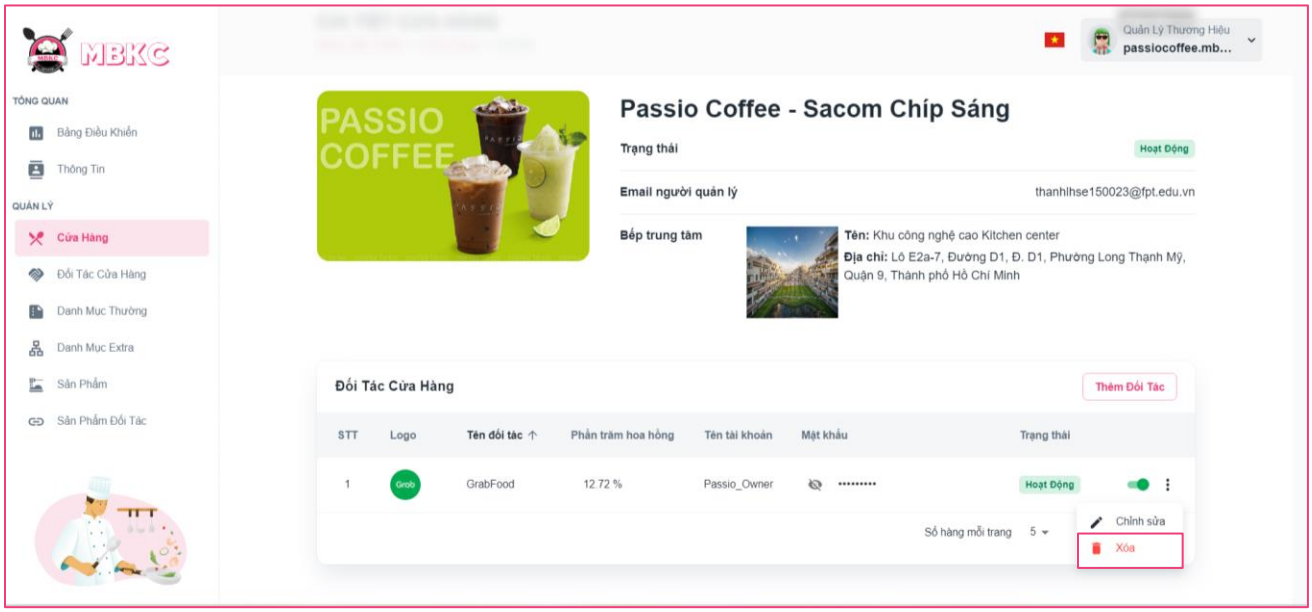


Figure 442: User Manual - Delete Store Partner – Step 5 – Case 2 – 2

Step 6: Click “Xác Nhận” button.

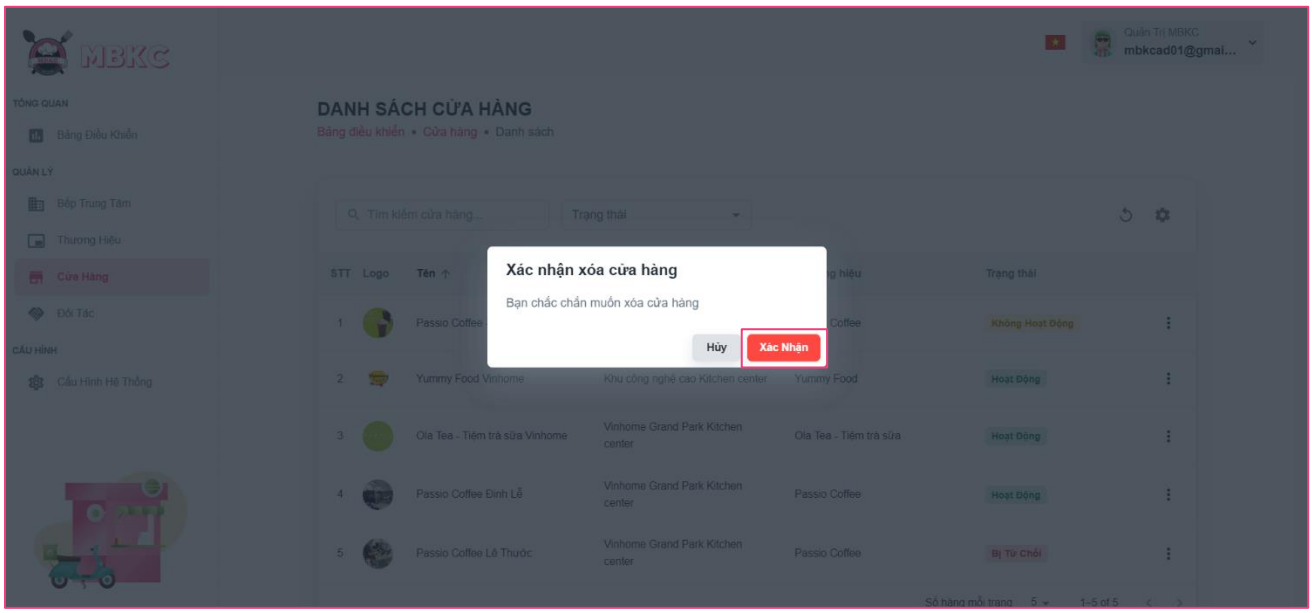


Figure 443: User Manual - Delete Store Partner – Step 6

Step 7:

Case 1: List of stores will be reloaded and a toast message “Đã xóa đối tác cửa hàng thành công”. And store partner has been deleted.

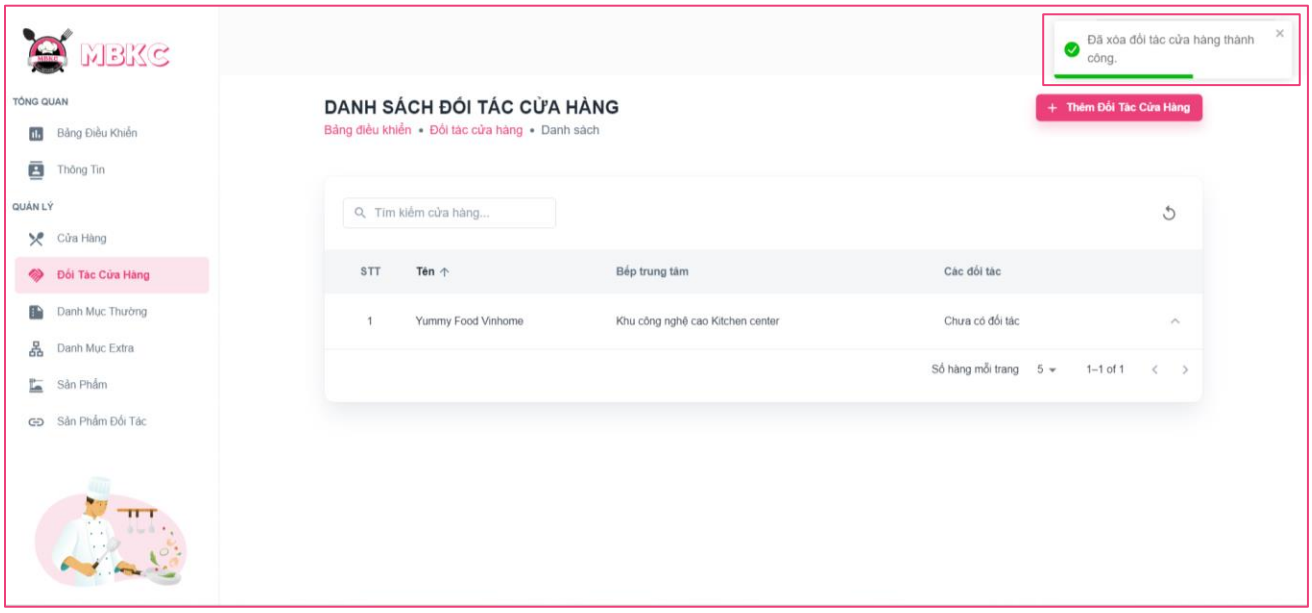


Figure 444: User Manual – Delete Store Partner – Step 7 – Case 1

Case 2: List of stores will be reloaded and a toast message “Đã xóa đối tác cửa hàng thành công”. And store partner has been deleted.

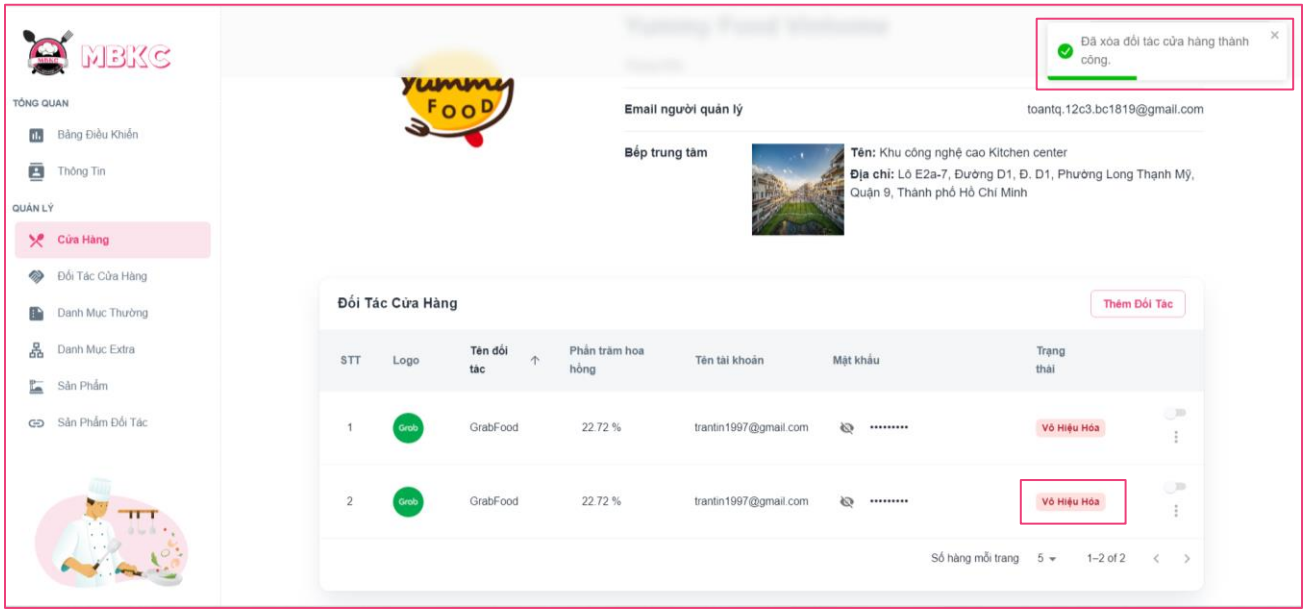


Figure 445: User Manual – Delete Store Partner – Step 7 – Case 2

3.2.10. Partner Product management

3.2.10.1. Create new Partner Product

3.2.10.1.1. Description

Use this guideline to login into MBKC account to create a new partner product.

This web application supports create a new partner product with account as a Brand Manager.

3.2.10.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

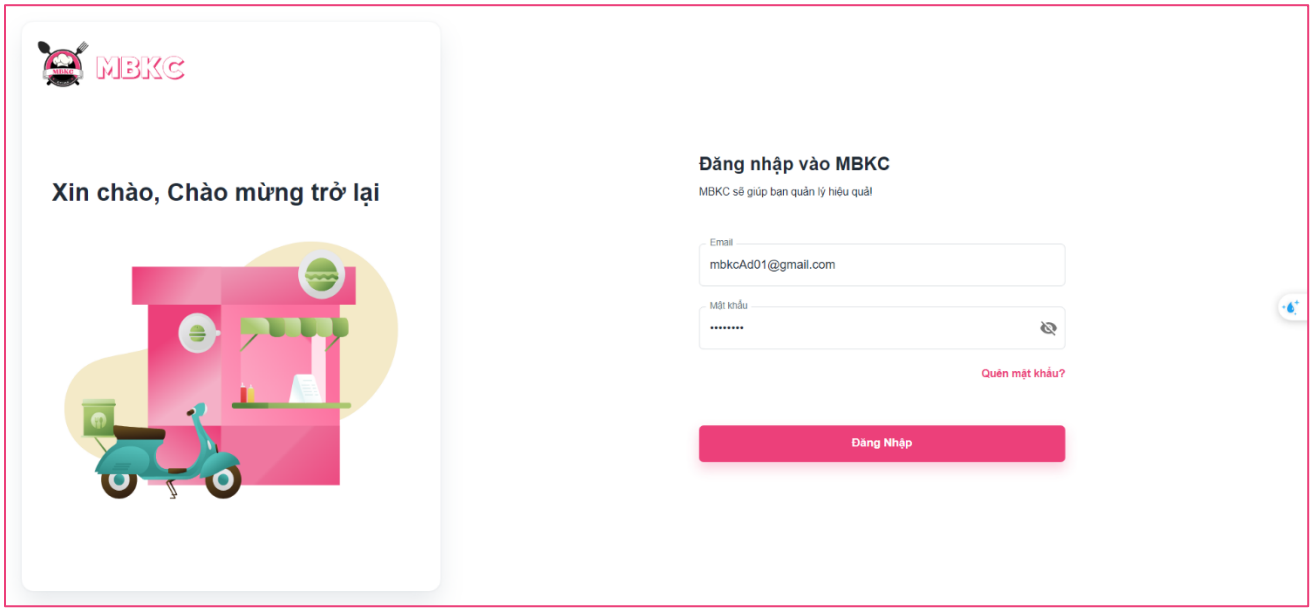


Figure 446: User Manual - Create new Partner Product – Step 1

Step 2: Click “Đăng Nhập” button.

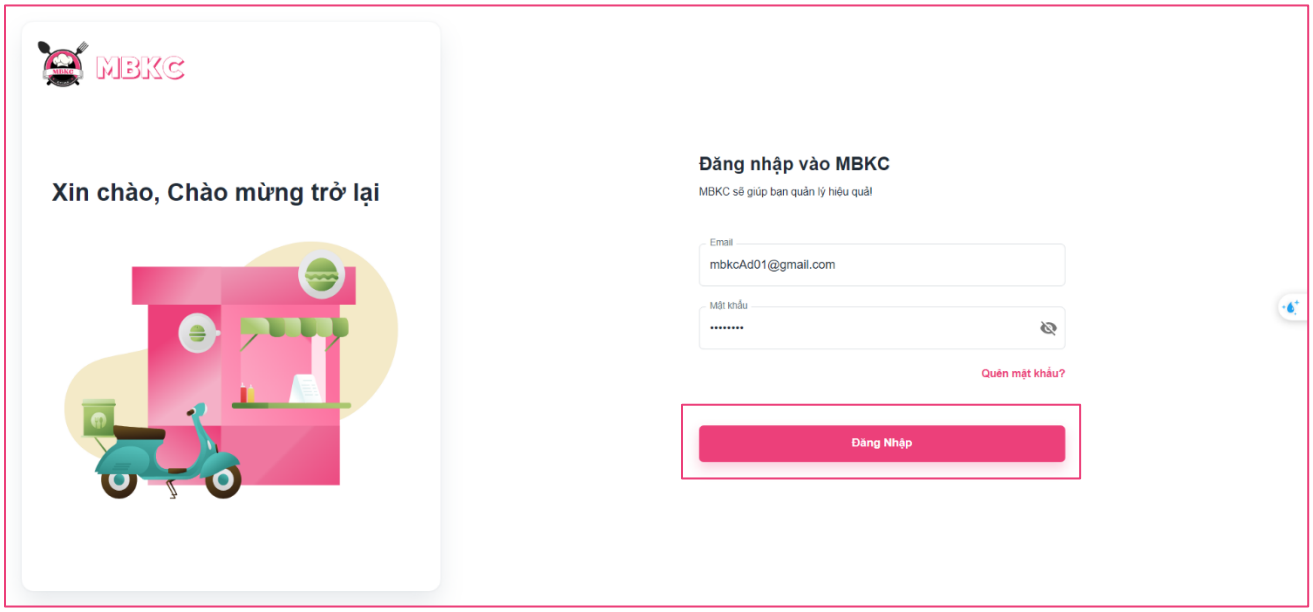


Figure 447: User Manual - Create new Partner Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm Đối Tác” from the left sidebar.

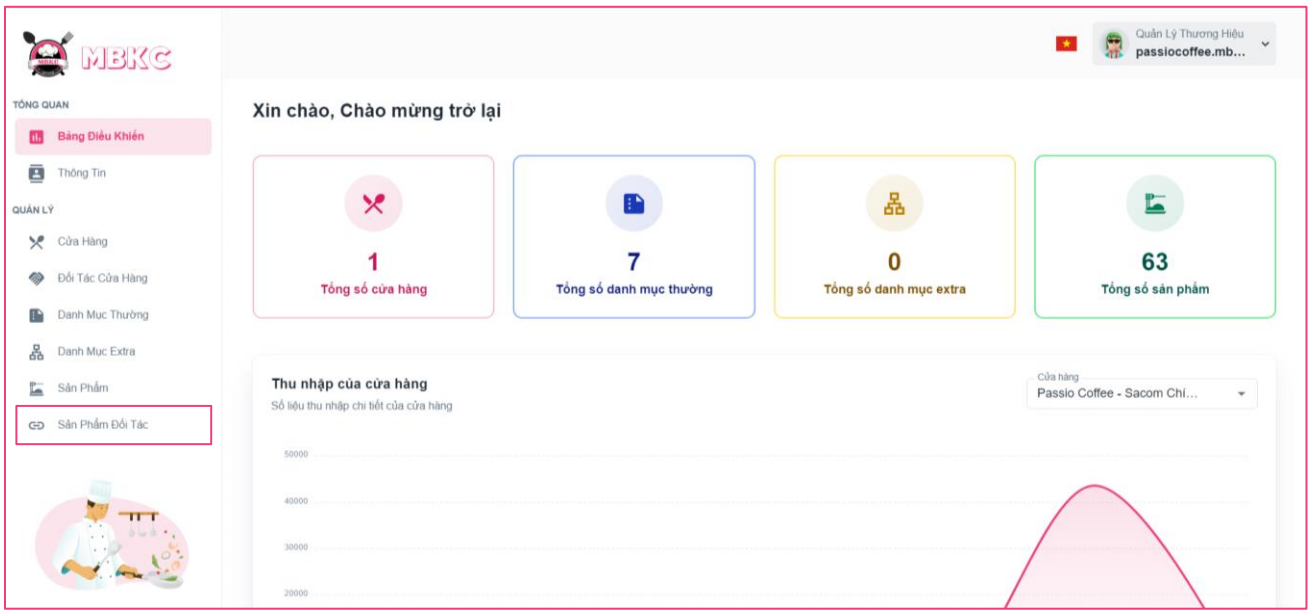


Figure 448: User Manual - Create new Partner Product – Step 3

Step 4: Click “Thêm Sản Phẩm Đối Tác” button on the screen and pop up Create New Partner Product Modal to add a new partner product.

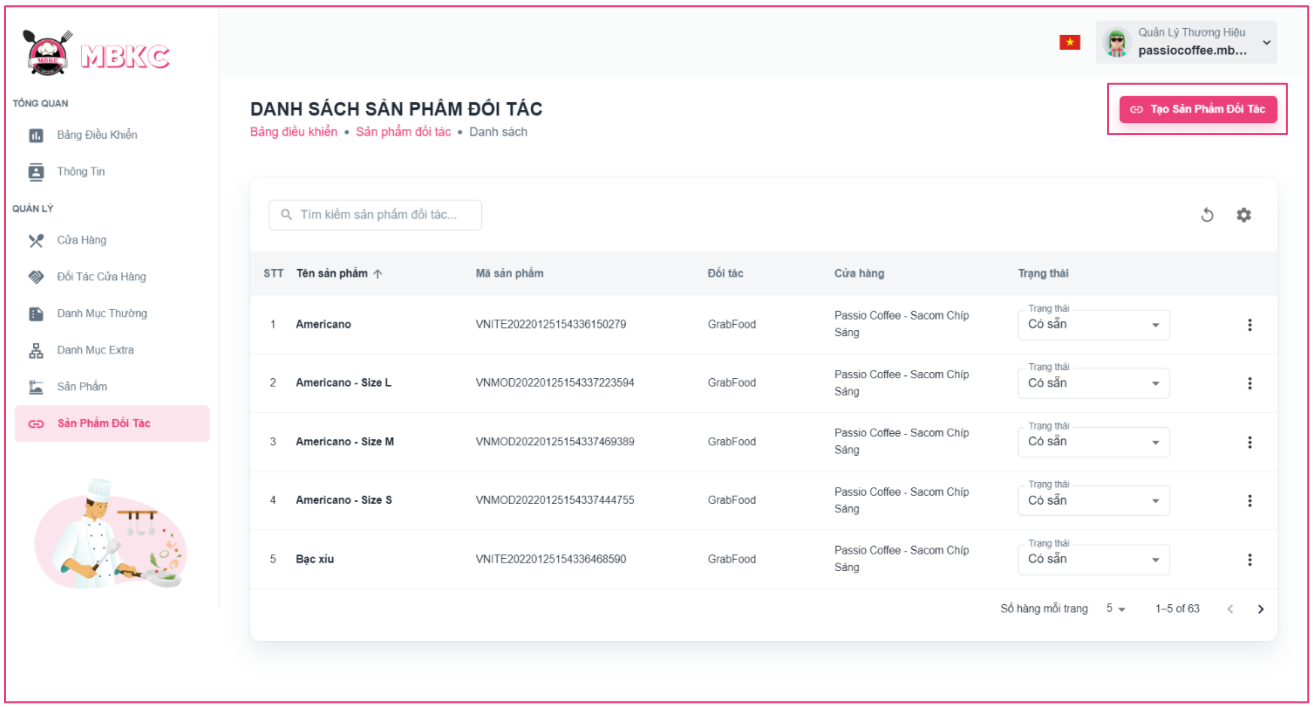


Figure 449: User Manual - Create new Partner Product – Step 4

Step 5: Filling all the required information to add a new partner product including: “Sản phẩm", “Cửa hàng", “Đối tác", “Mã Sản Phẩm”, “Trạng thái sản phẩm đối tác”, “Giá sản phẩm”.

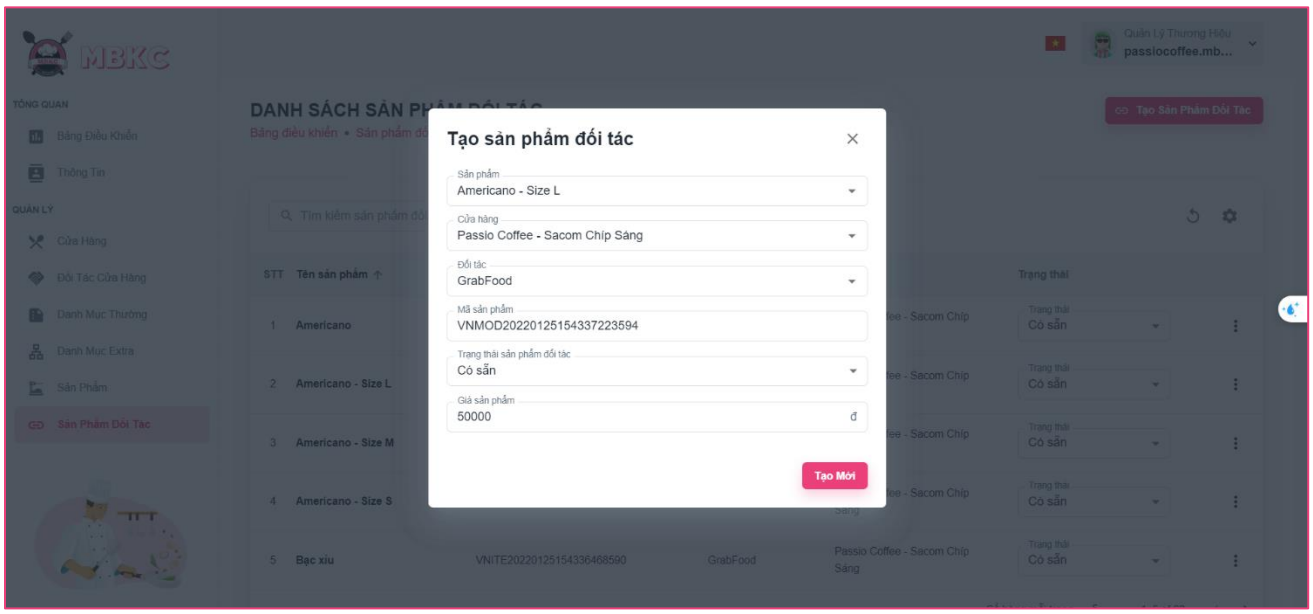


Figure 450: User Manual - Create new Partner Product – Step 5

Step 6: After filling in all required field, click “Tạo Mới” to finish creating a new partner product.

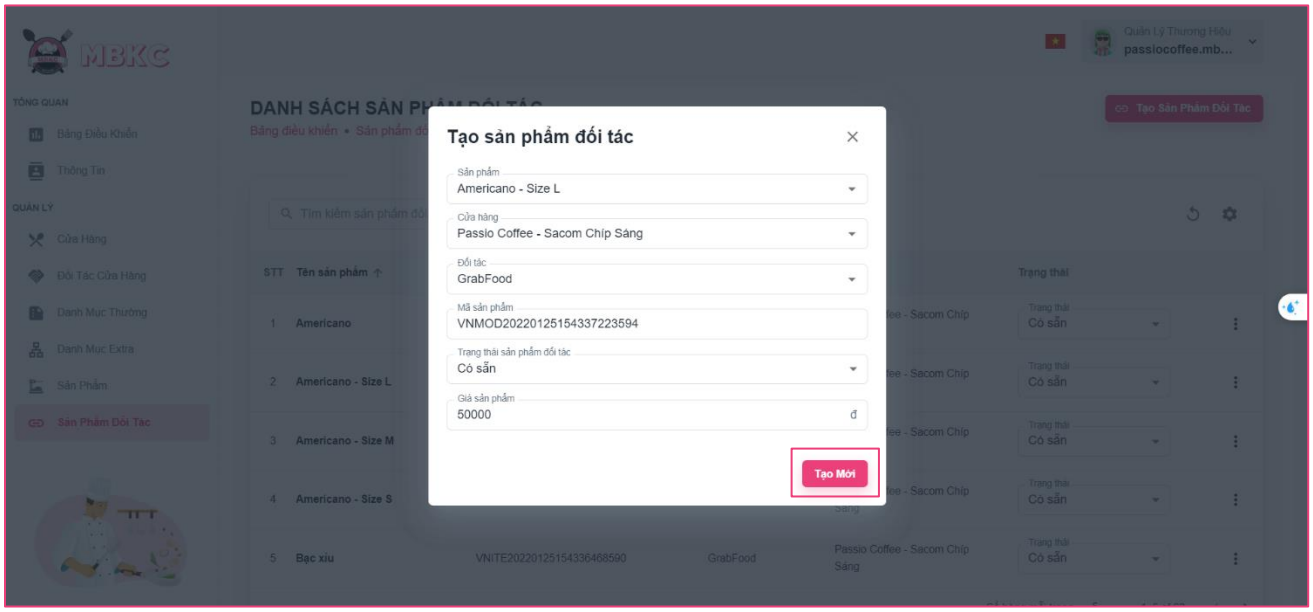


Figure 451: User Manual - Create new Partner Product – Step 6

Step 7: List of Partner Products Page will be reloaded to show the list of partner products and a toast message “Đã tạo thành công sản phẩm đối tác mới”.

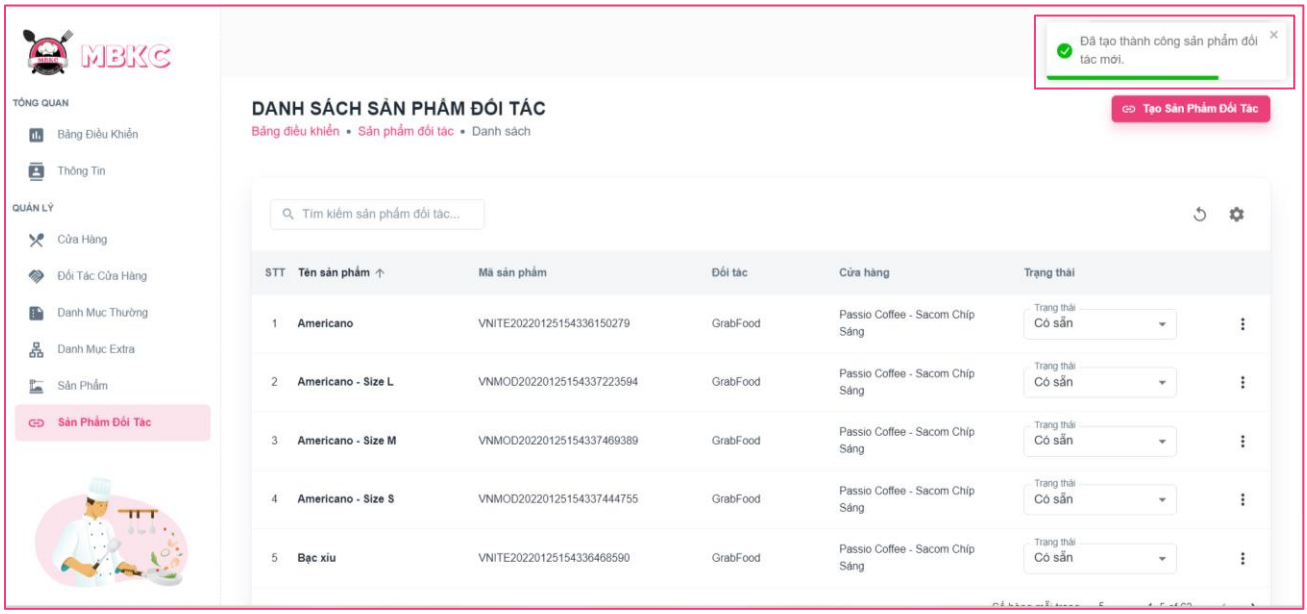


Figure 452: User Manual – Create new Partner Product – Step 7

3.2.10.2. View list of Partner Products

3.2.10.2.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of partner products.

This web application supports view list of partner products with account as a Brand Manager.

3.2.10.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

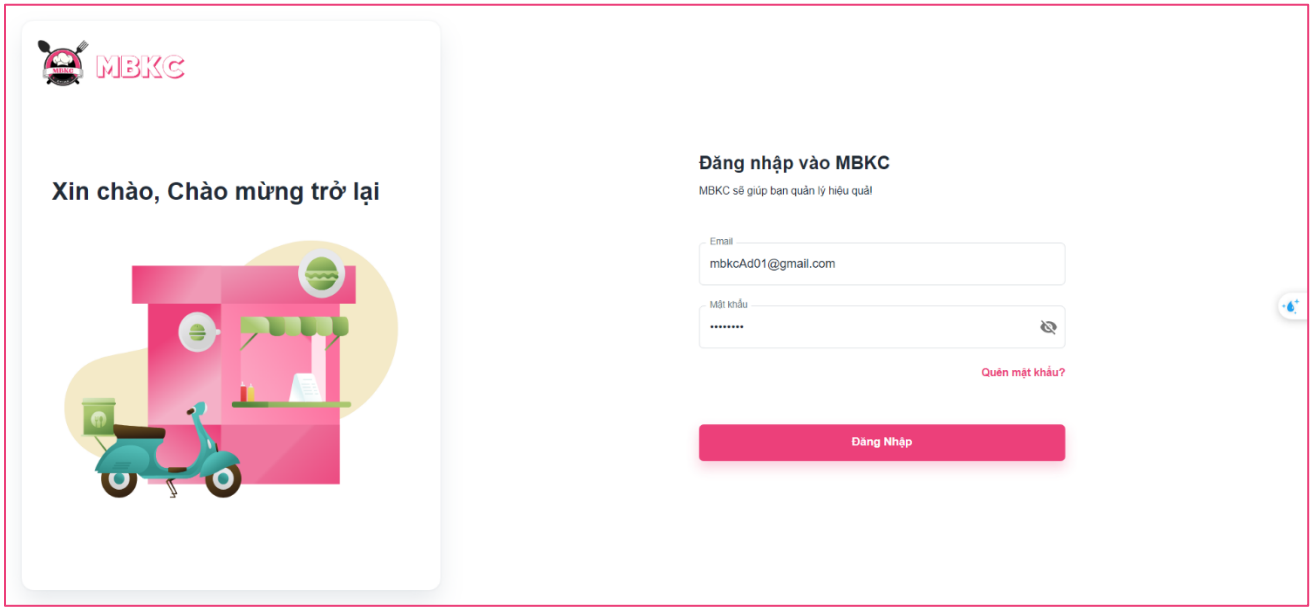


Figure 453: User Manual - View list of Partner Products – Step 1

Step 2: Click “Đăng Nhập” button.

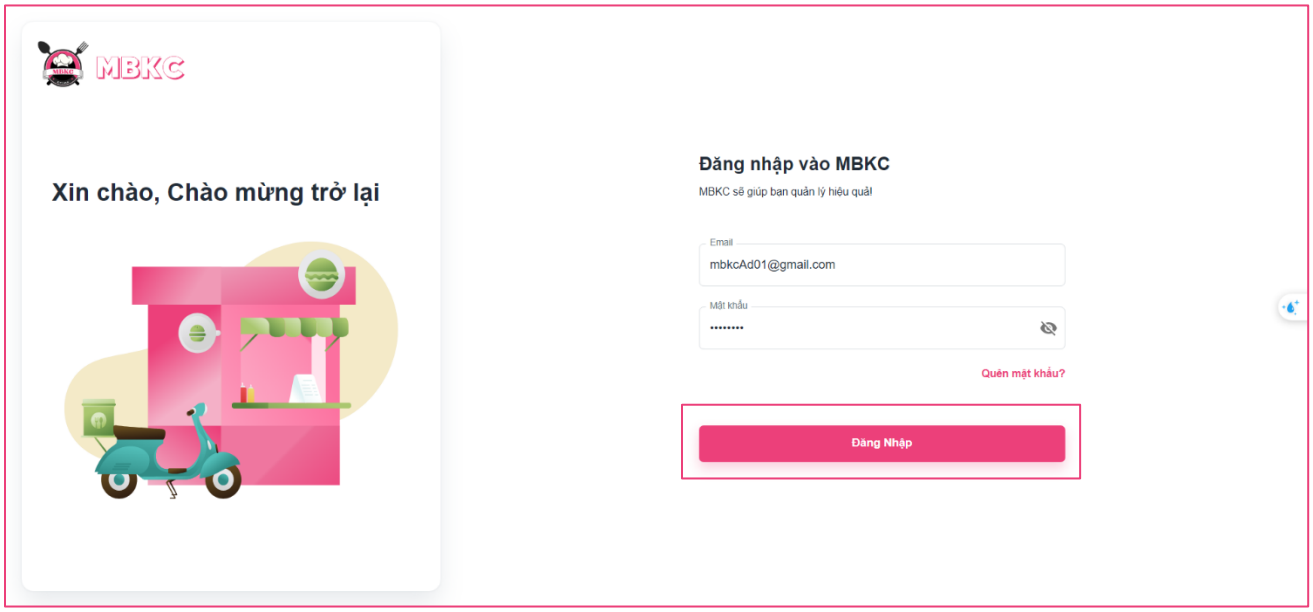


Figure 454: User Manual - View list of Partner Products – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm Đối Tác” from the left sidebar.

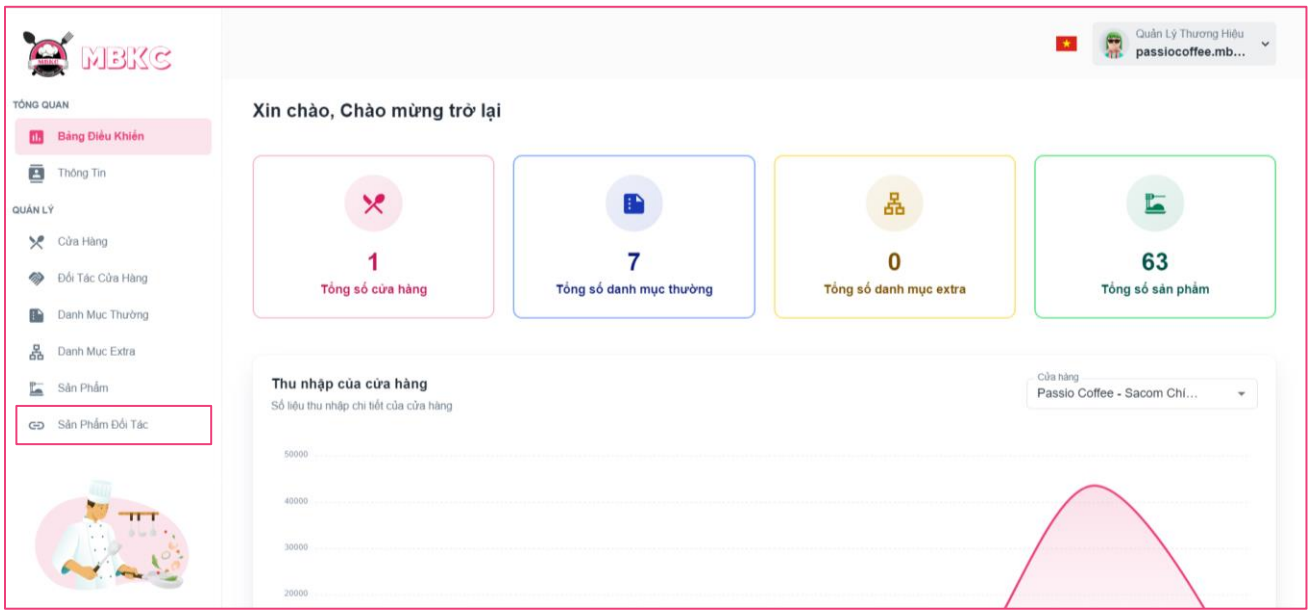


Figure 455: User Manual - View list of Partner Products – Step 3

Step 4: Search, reload data, and show or hide columns in the table list of partner products.

Case 1: Fill in the product's name in the search box to search partner products in the table list.

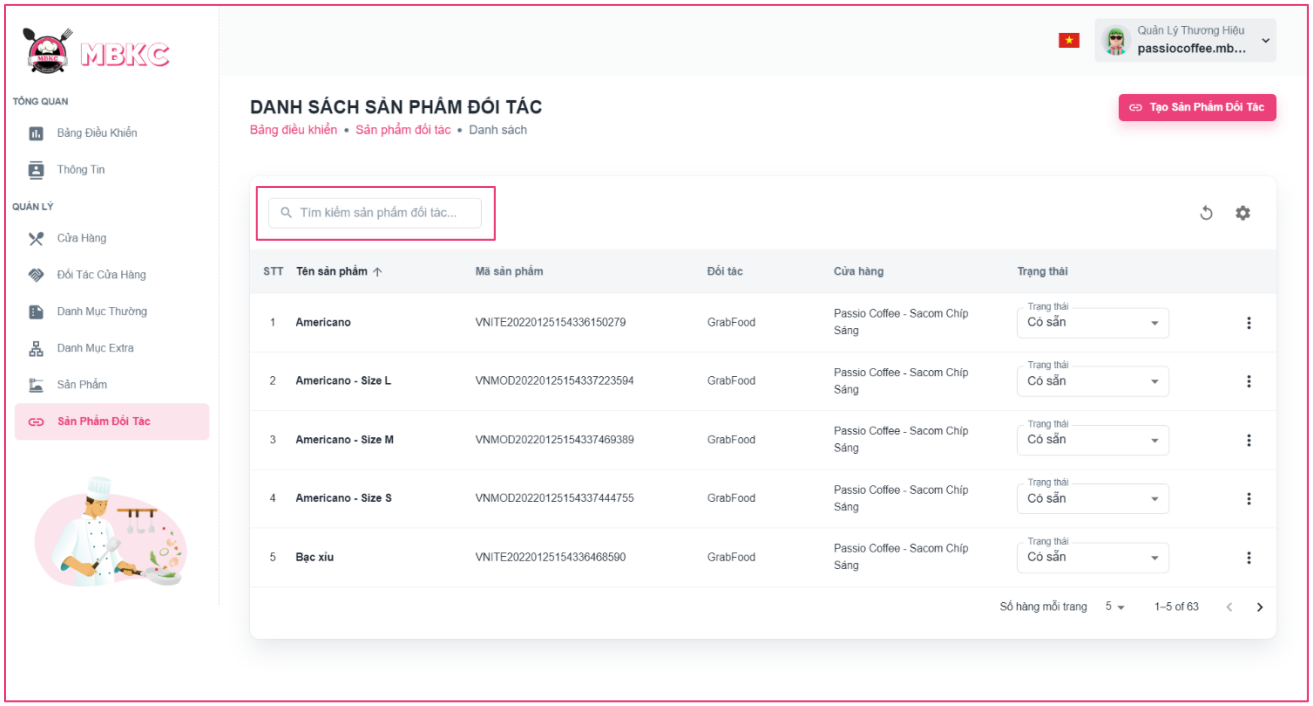


Figure 456: User Manual - View list of Partner Products – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of partner products.

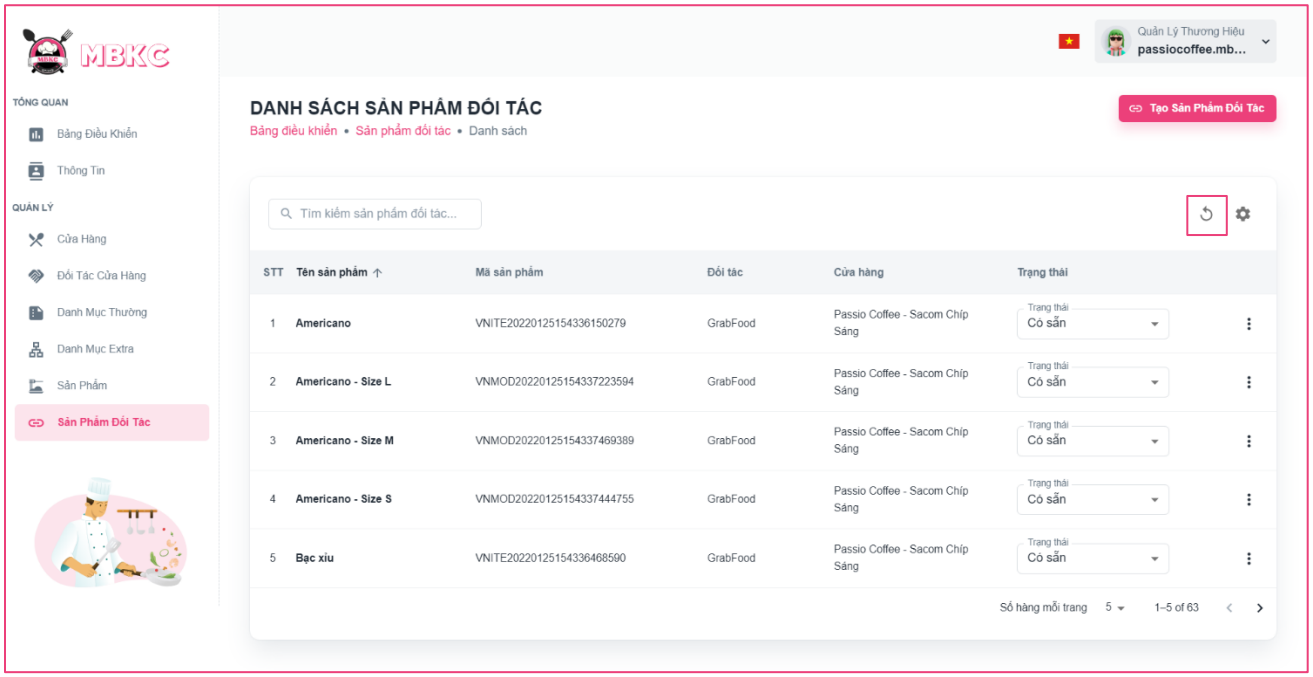


Figure 457: User Manual - View list of Partner Products – Step 4 – Case 2

Case 3: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of partner products.

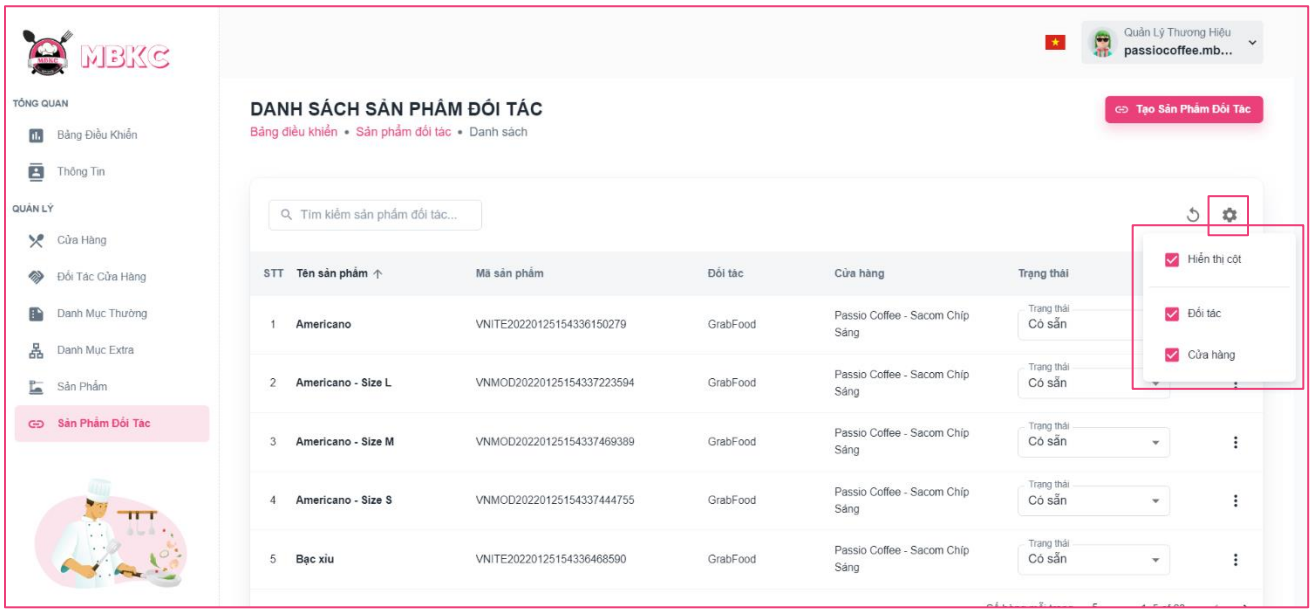


Figure 458: User Manual - View list of Partner Products – Step 4 – Case 3

3.2.10.3. View details Partner Product

3.2.10.3.1. Description

Use this guideline to view details partner product.

This web application supports view details partner product with account as a Brand Manager.

3.2.10.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

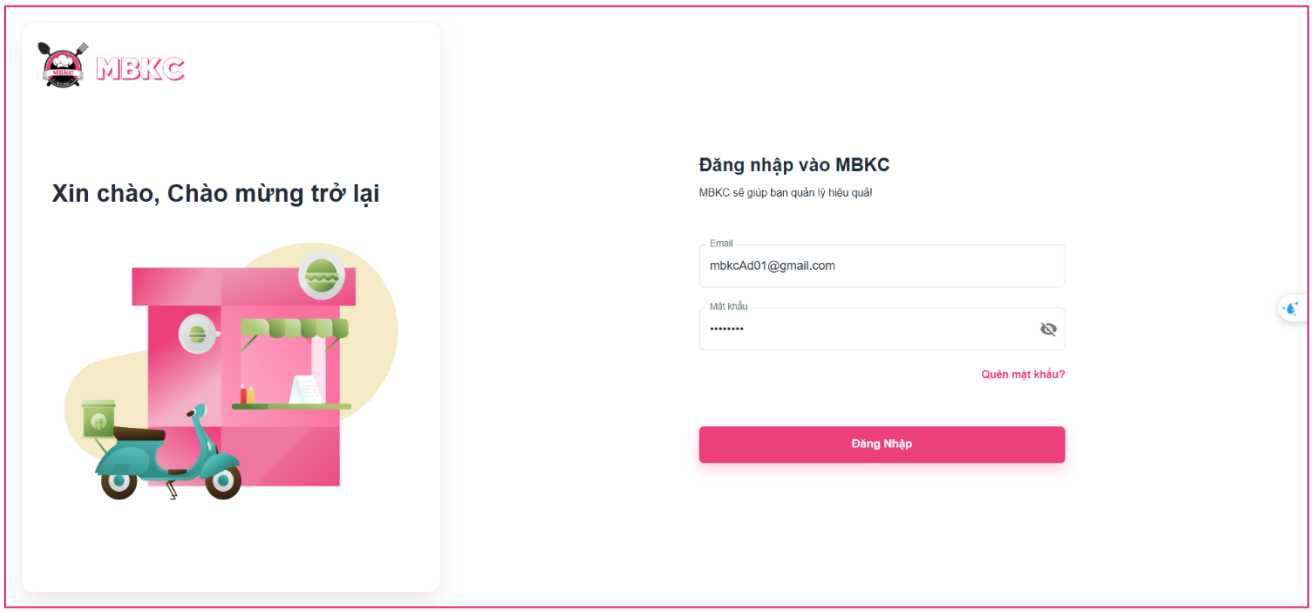


Figure ...459 User Manual - View details Partner Product – Step 1

Step 2: Click “Đăng Nhập” button.

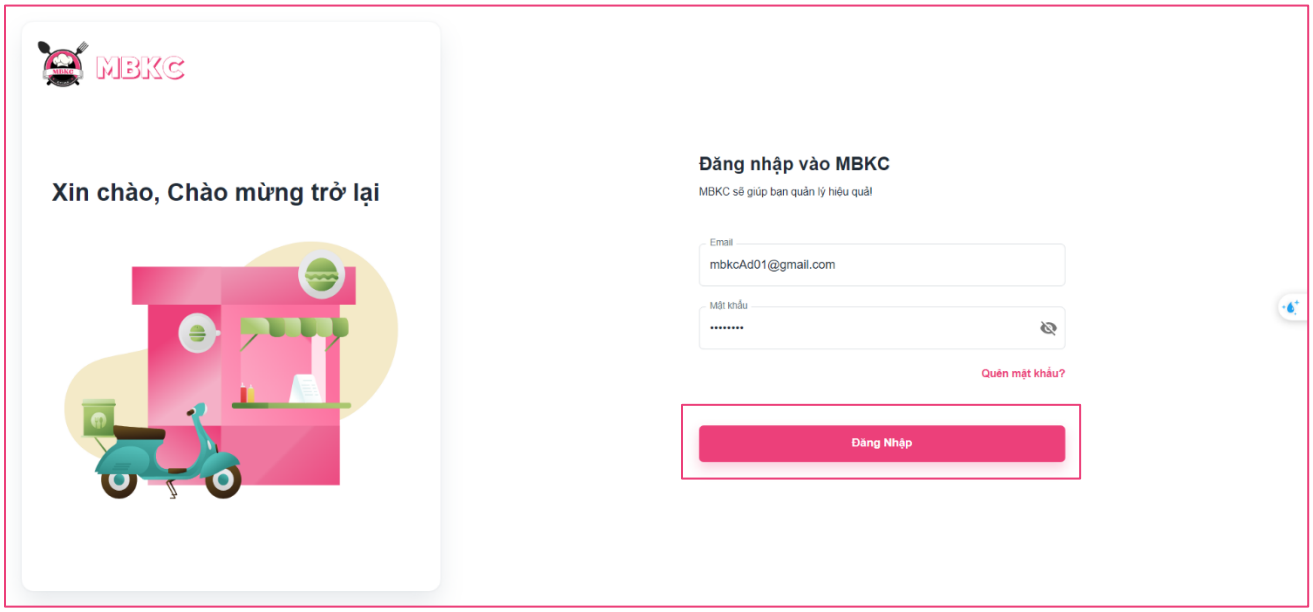


Figure 460: User Manual - View details Partner Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm Đối Tác” from the left sidebar.

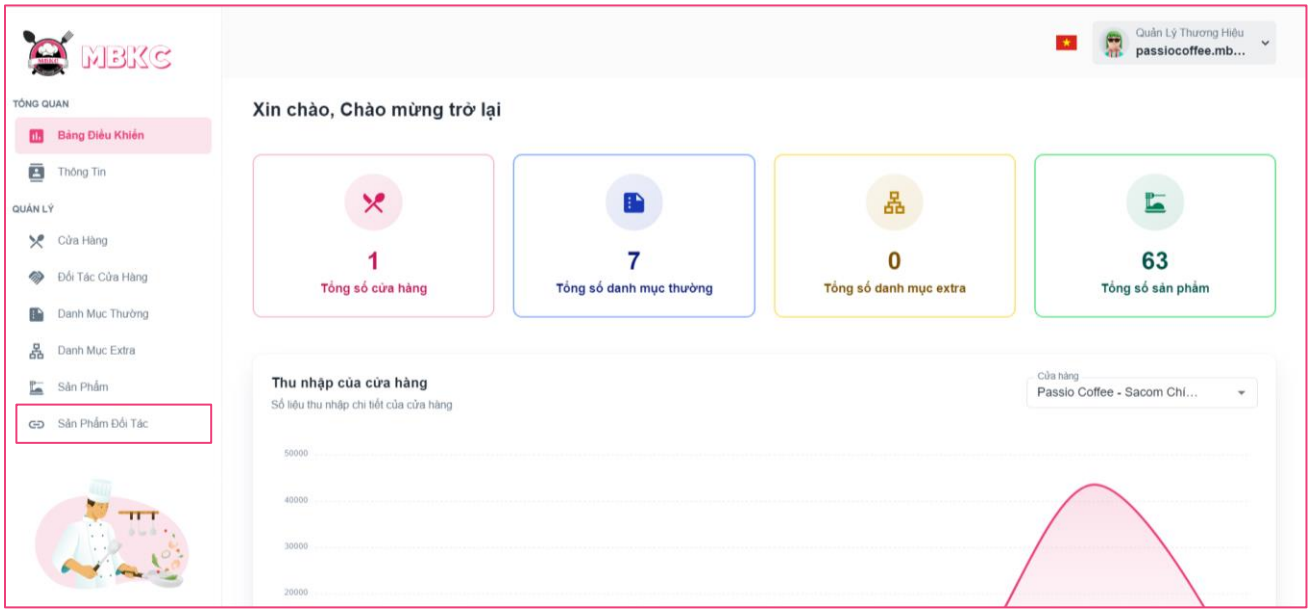


Figure 461: User Manual - View details Partner Product – Step 3

Step 4: Click the partner product row to navigate to Partner Product Details Page.

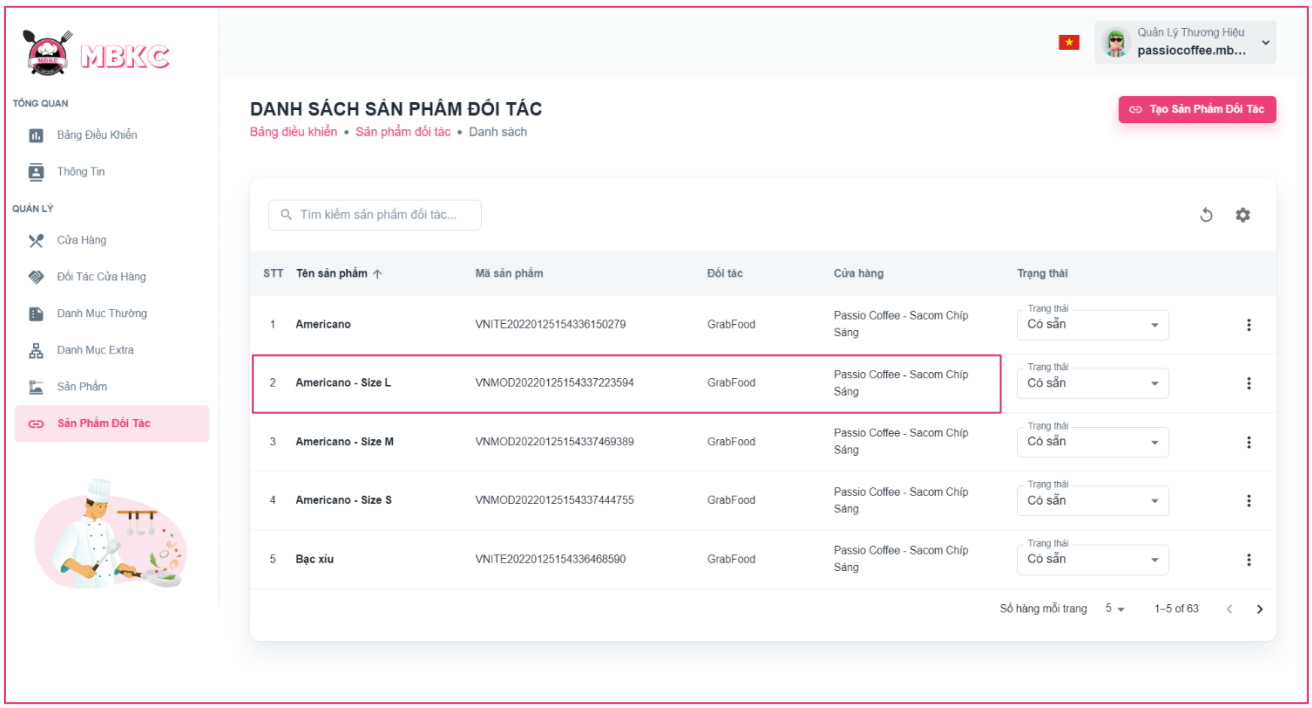


Figure 462: User Manual - View details Partner Product – Step 4

Step 5: Navigate to Partner Product Details Page to show information details of partner product.

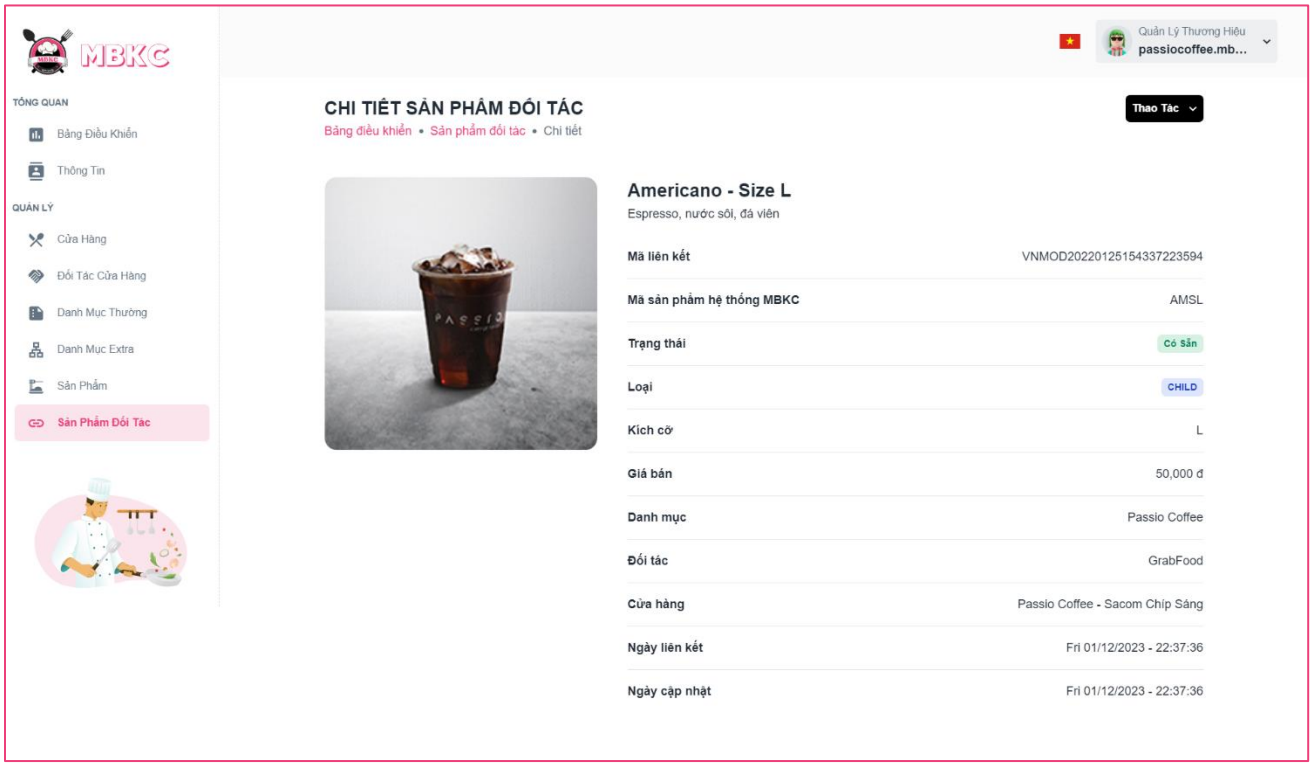


Figure 463: User Manual - View details Partner Product – Step 5

3.2.10.4. Update Status Partner Product

3.2.10.4.1. Description

Use this guideline to update the status of the partner product.

This web application supports update the status of the partner product with account as a Brand Manager.

3.2.10.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

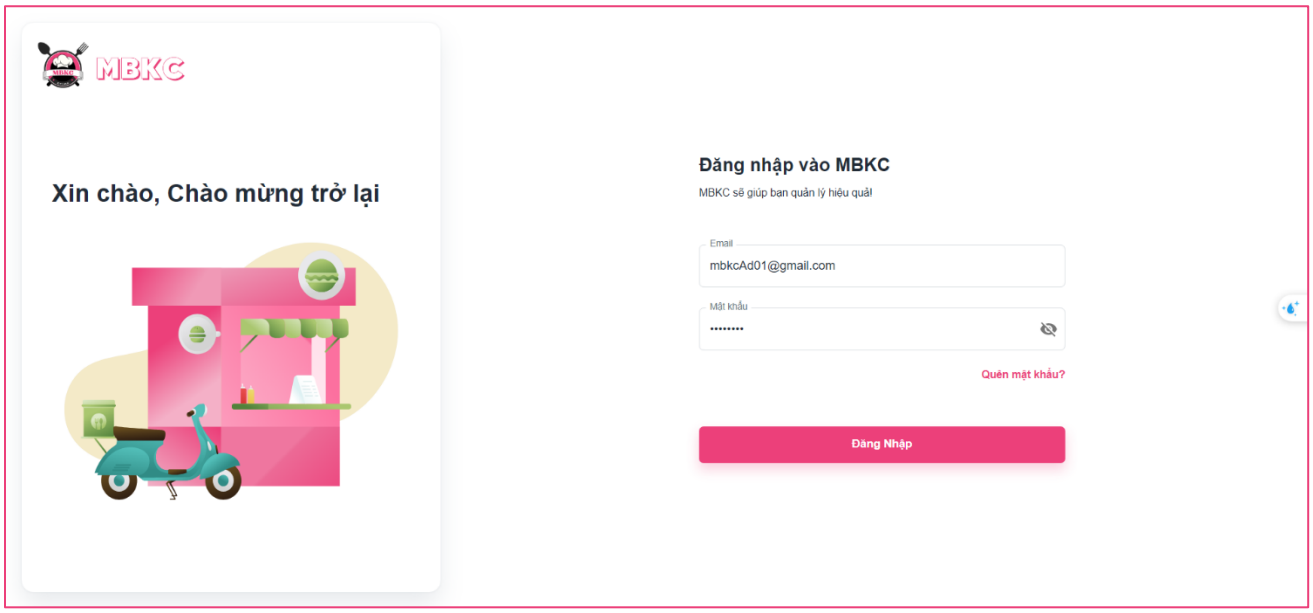


Figure 464: User Manual – Update Status Partner Product – Step 1

Step 2: Click “Đăng Nhập” button.

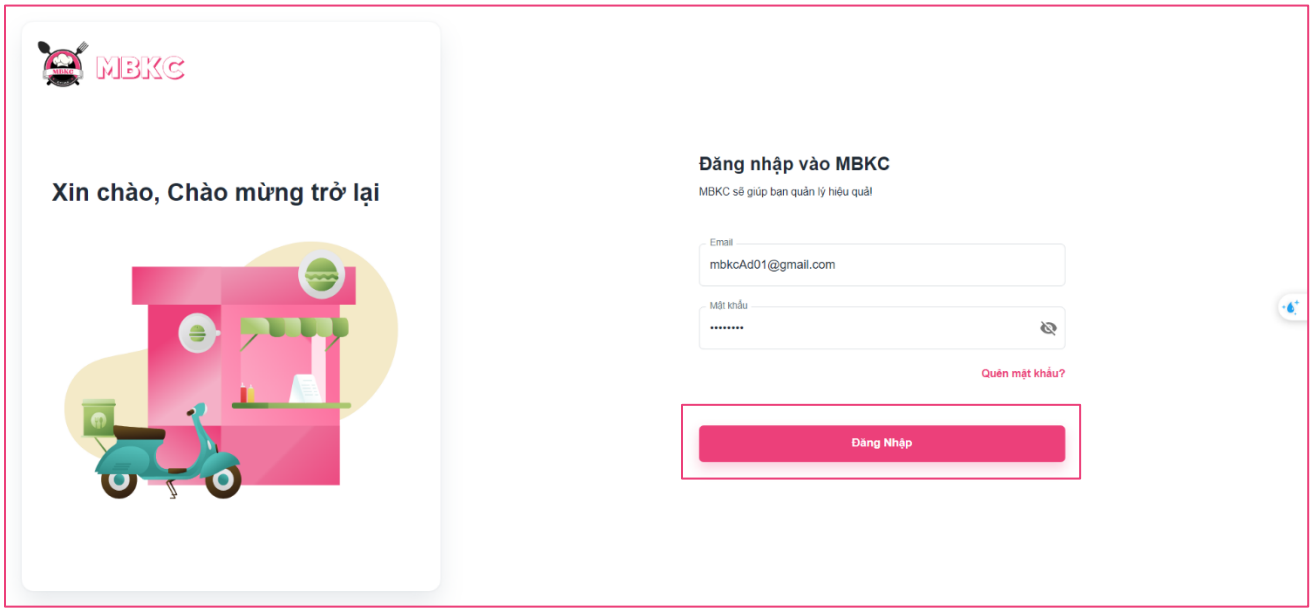


Figure 465: User Manual - Update Status Partner Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm Đối Tác” from the left sidebar.

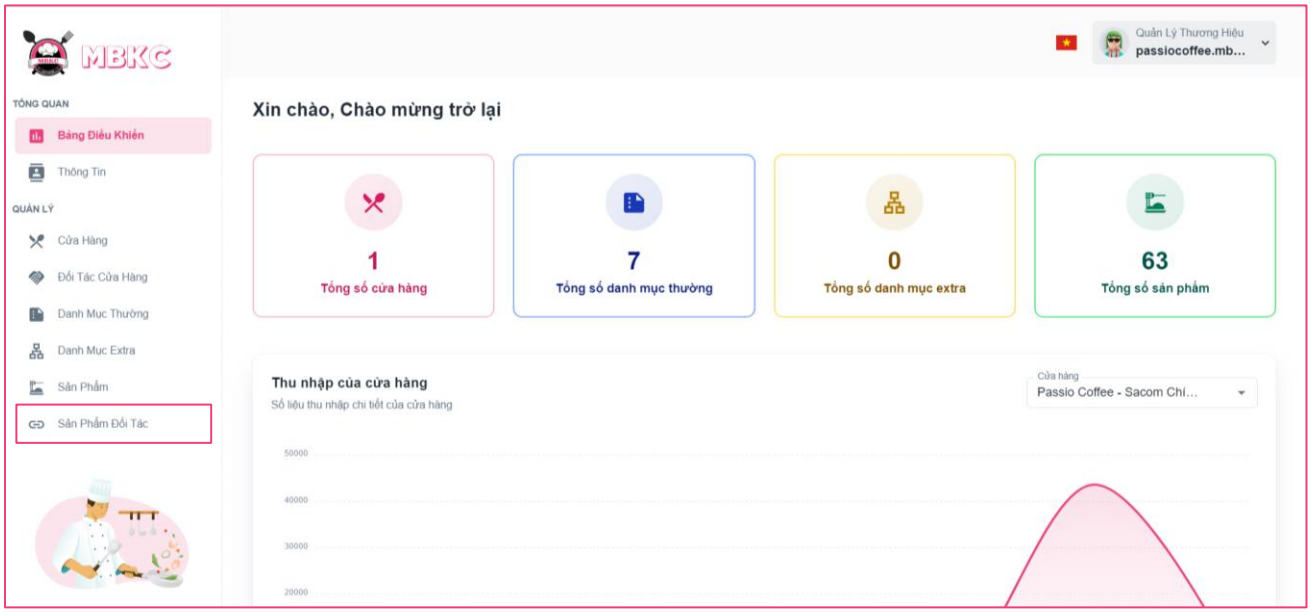


Figure 466: User Manual - Update Status Partner Product – Step 3

Step 4: Click on the box select “Trạng thái” on the partner product row and select status want to update for a partner product.

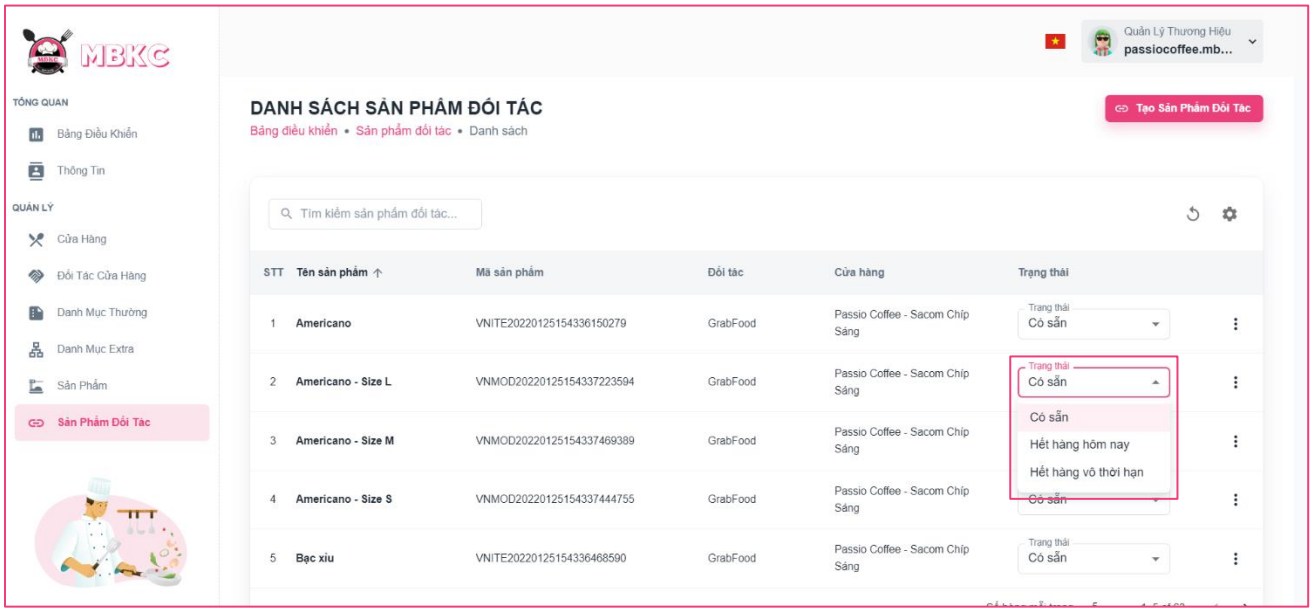


Figure 467: User Manual - Update Status Partner Product – Step 4

Step 5: List of partner products will be reloaded and a toast message “Đã cập nhật trạng thái sản phẩm của đối tác thành công”. And the status of the partner product has been updated.

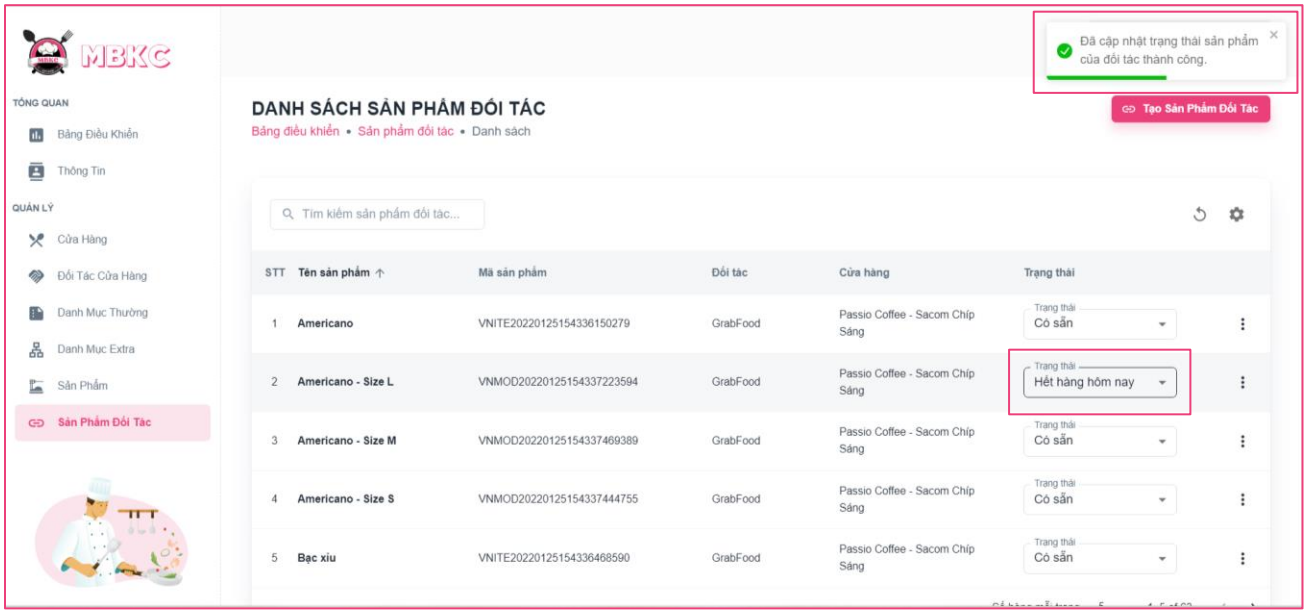


Figure 468: User Manual - Update Status Partner Product – Step 5

Case: If you update the status of a partner product whose product type is "Parent", the status of partner products whose product type is "Child" will be updated to the same as the parent product.

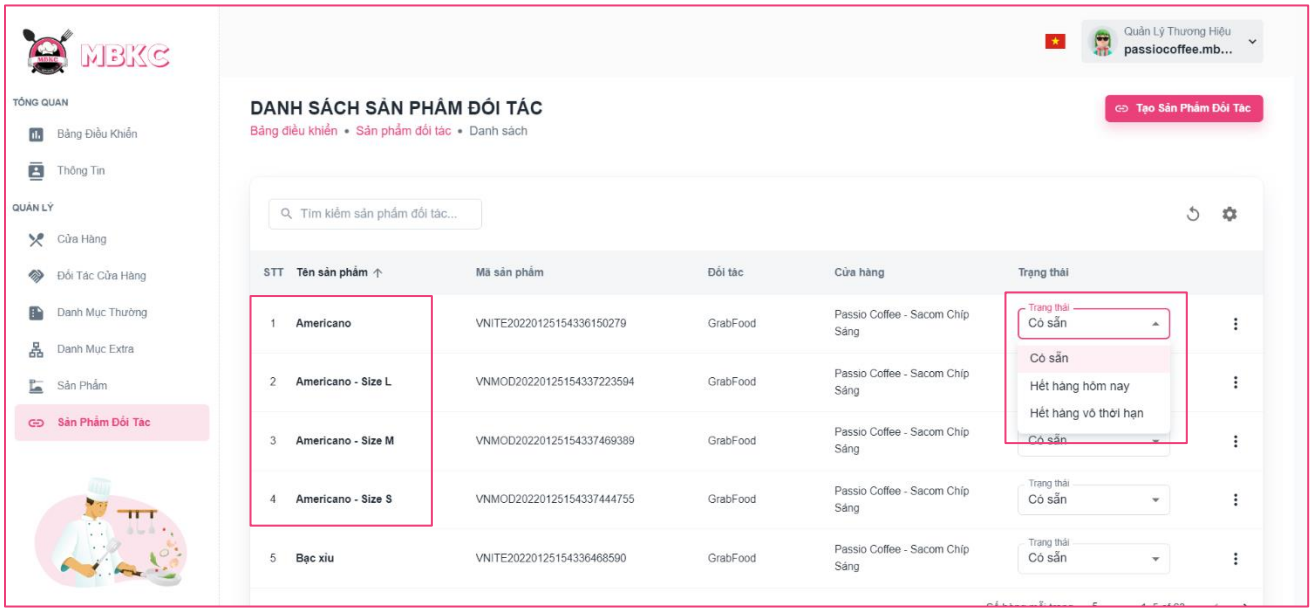


Figure 469: User Manual - Update Status Partner Product - Case - 1

The statuses have been updated accordingly.

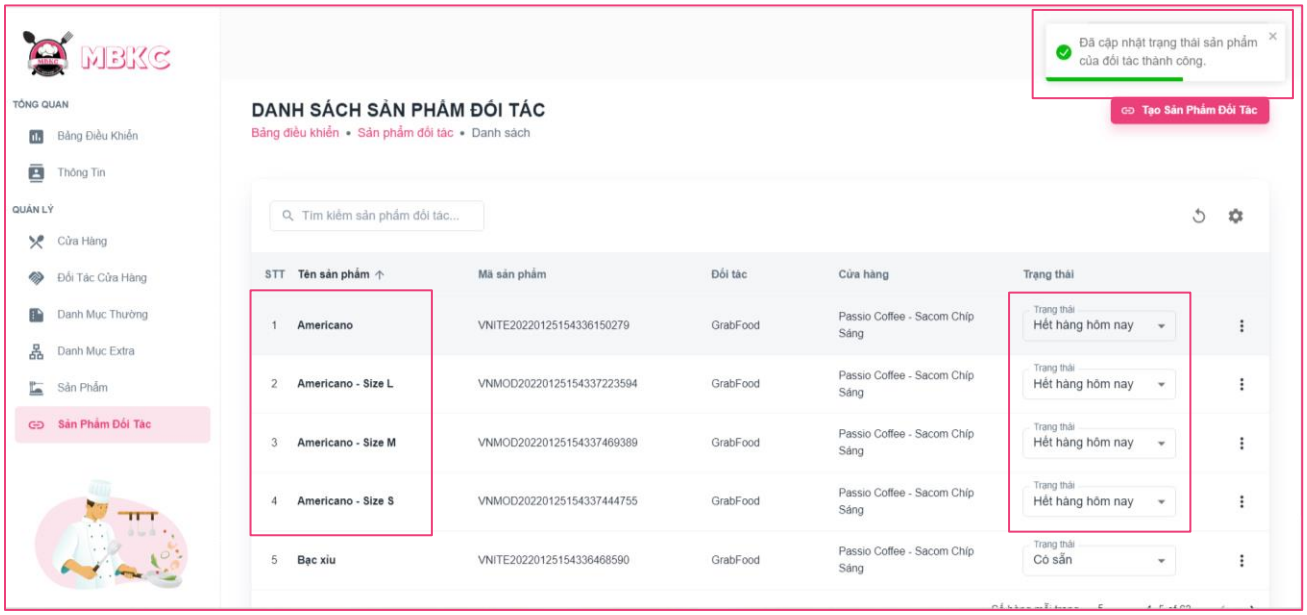


Figure 470: User Manual - Update Status Partner Product - Case - 2

3.2.10.5. Update Information Partner Product

3.2.10.5.1. Description

Use this guideline to update information of the partner product.

This web application supports update information of the partner product with account as a Brand Manager.

3.2.10.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

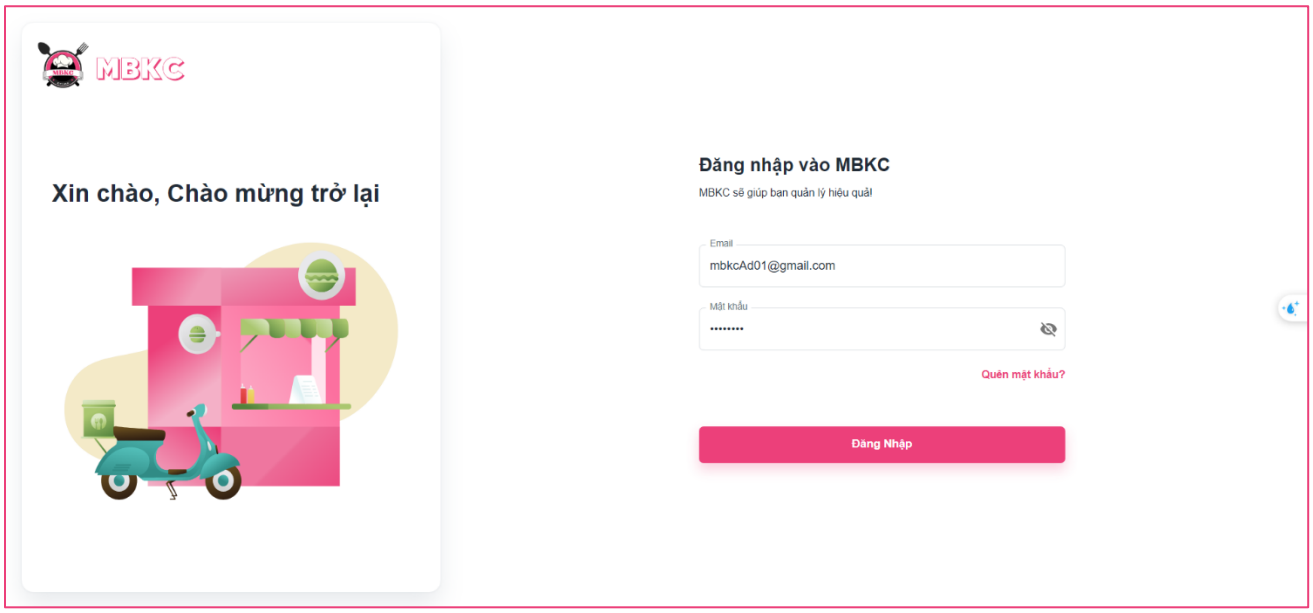


Figure 471: User Manual – Update Information Partner Product – Step 1

Step 2: Click “Đăng Nhập” button.

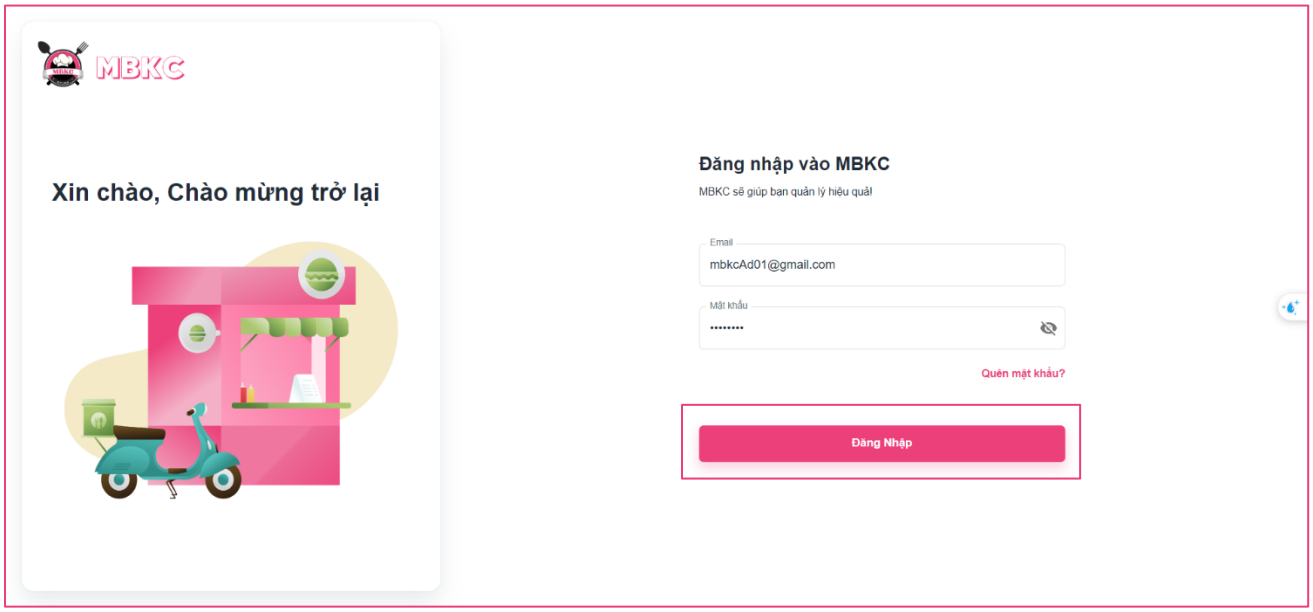


Figure 472: User Manual - Update Information Partner Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm Đối Tác” from the left sidebar.

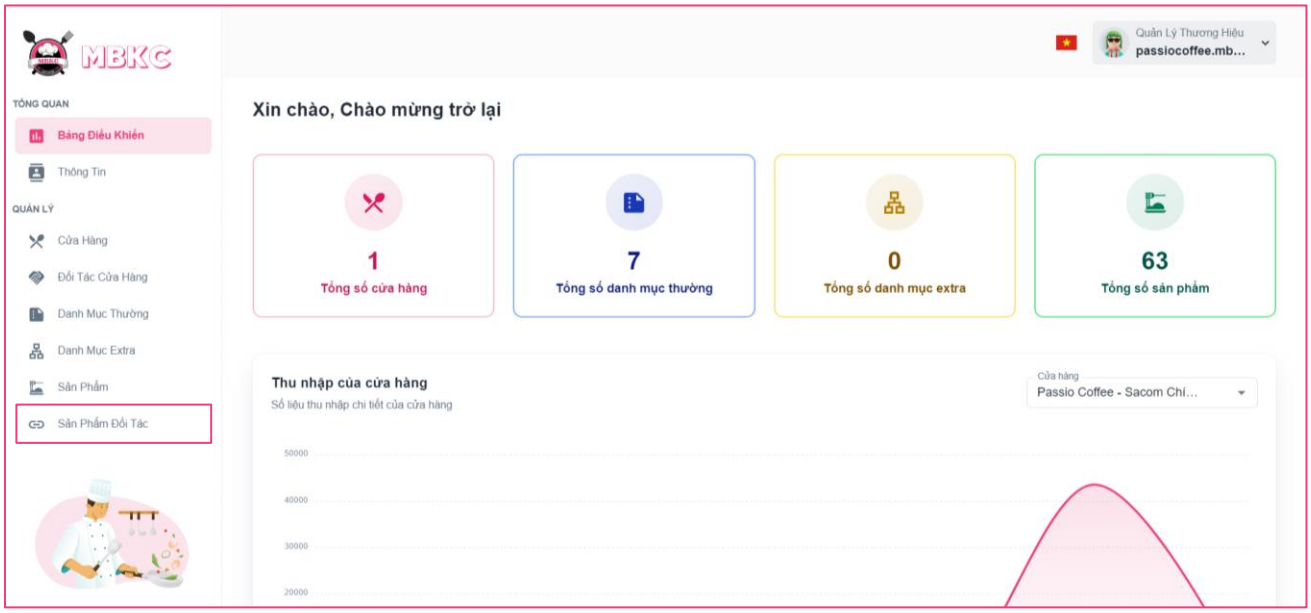


Figure 473: User Manual - Update Information Partner Product – Step 3

Step 4: Update information of the partner product.

Case 1: Update information of the partner product in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the partner product row want to pop up Update Partner Product Modal.

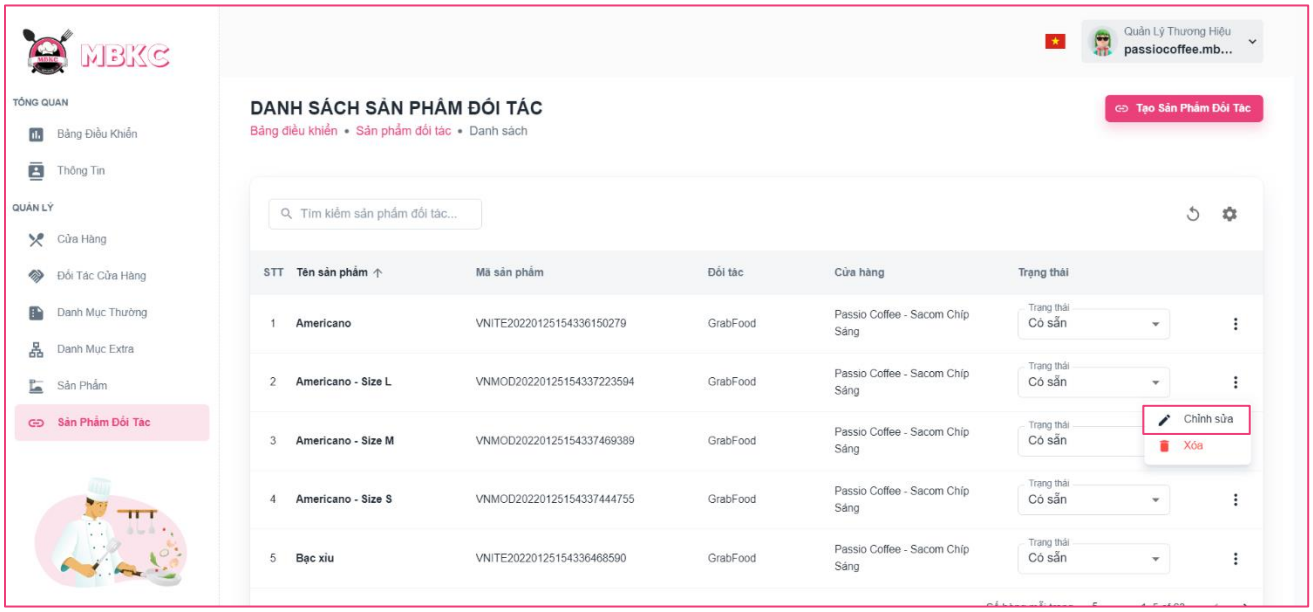


Figure 474: User Manual - Update Information Partner Product – Step 4 – Case 1

Case 2: Update information of the partner product on the details page.

1: Click the partner product row to navigate to Partner Product Details Page.

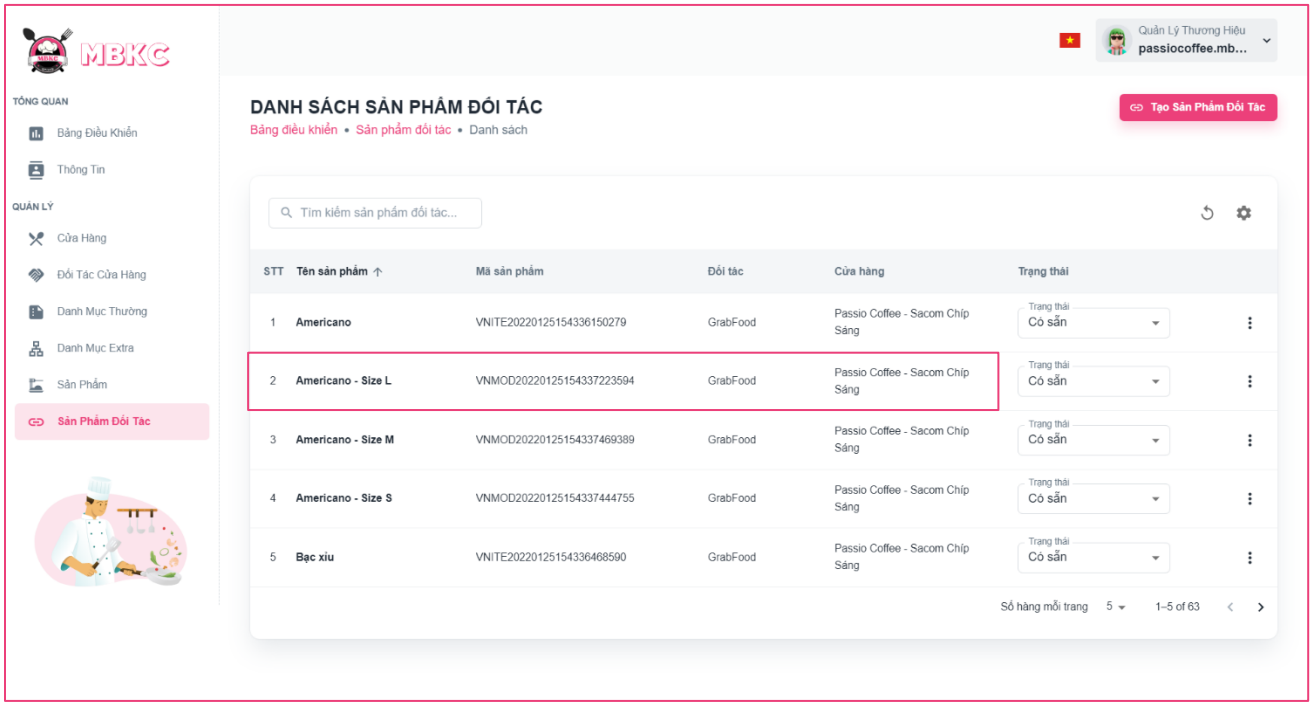


Figure 475: User Manual – Update Information Partner Product – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item to delete.

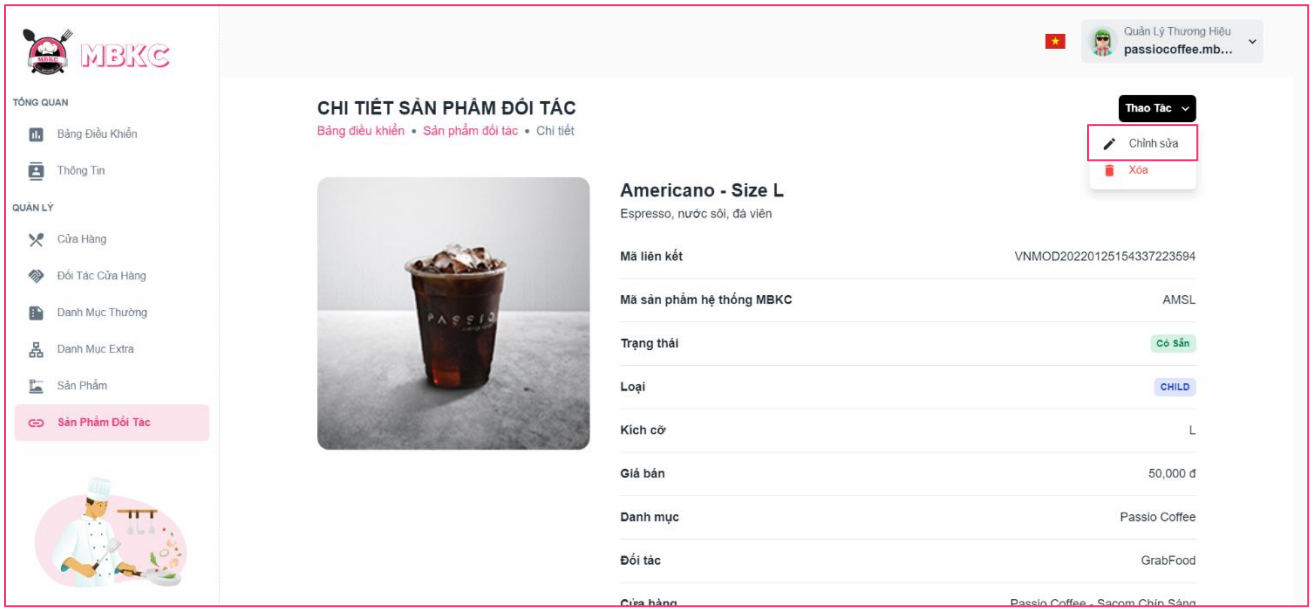


Figure 476: User Manual – Update Information Partner Product – Step 4 – Case 2 - 2

Step 5: Partner product information will be automatically filled in the form. Then edit the information as desired.

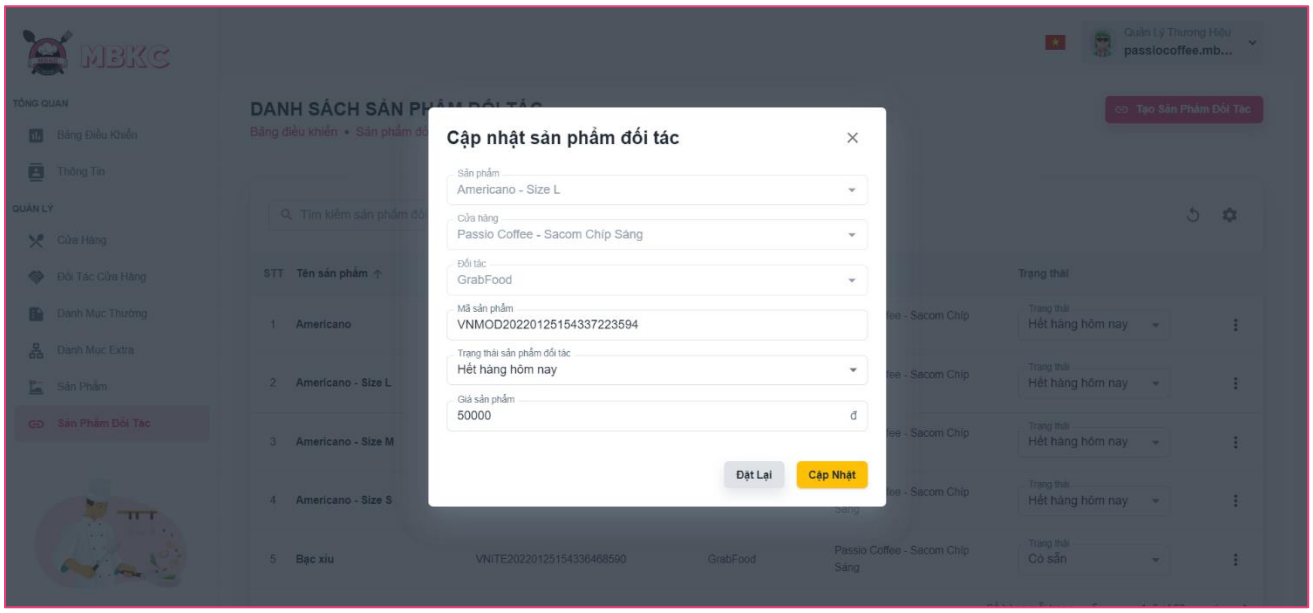


Figure 477: User Manual - Update Information Partner Product – Step 5

Step 6: Click “Cập nhật” button.

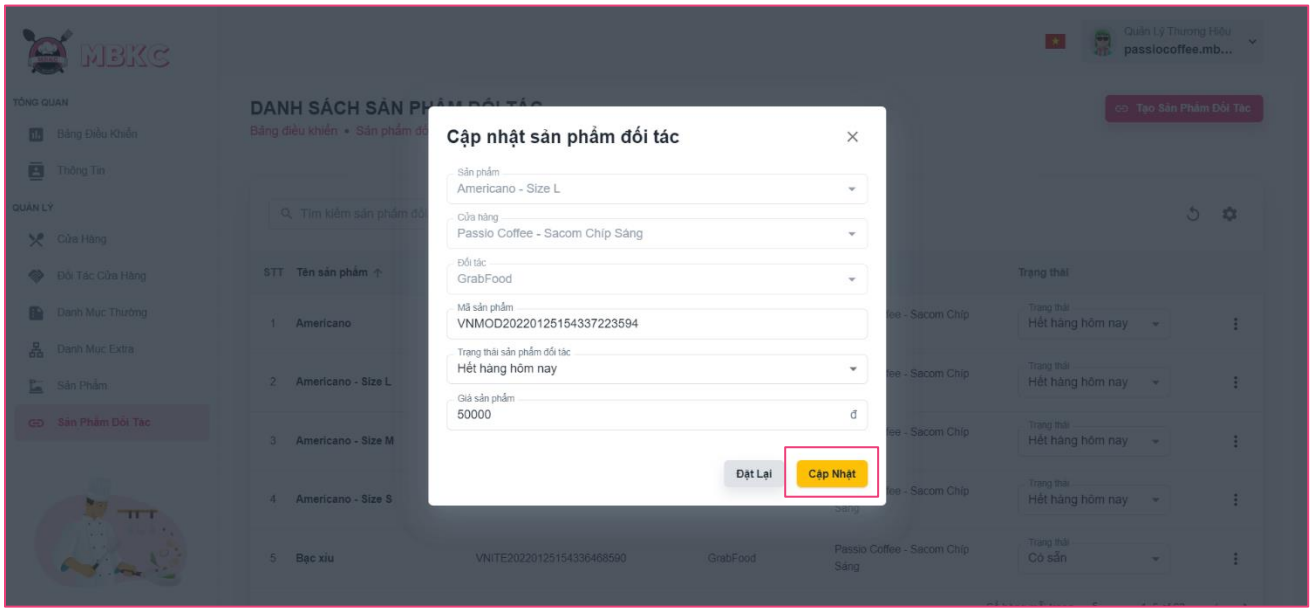


Figure 478: User Manual - Update Information Partner Product – Step 6

Step 7:

Case 1: Partner Product Details Page will be reloaded to show the partner product details when start edit information of partner product from Partner Product Details Page and a toast message “Cập nhật sản phẩm đối tác thành công”.

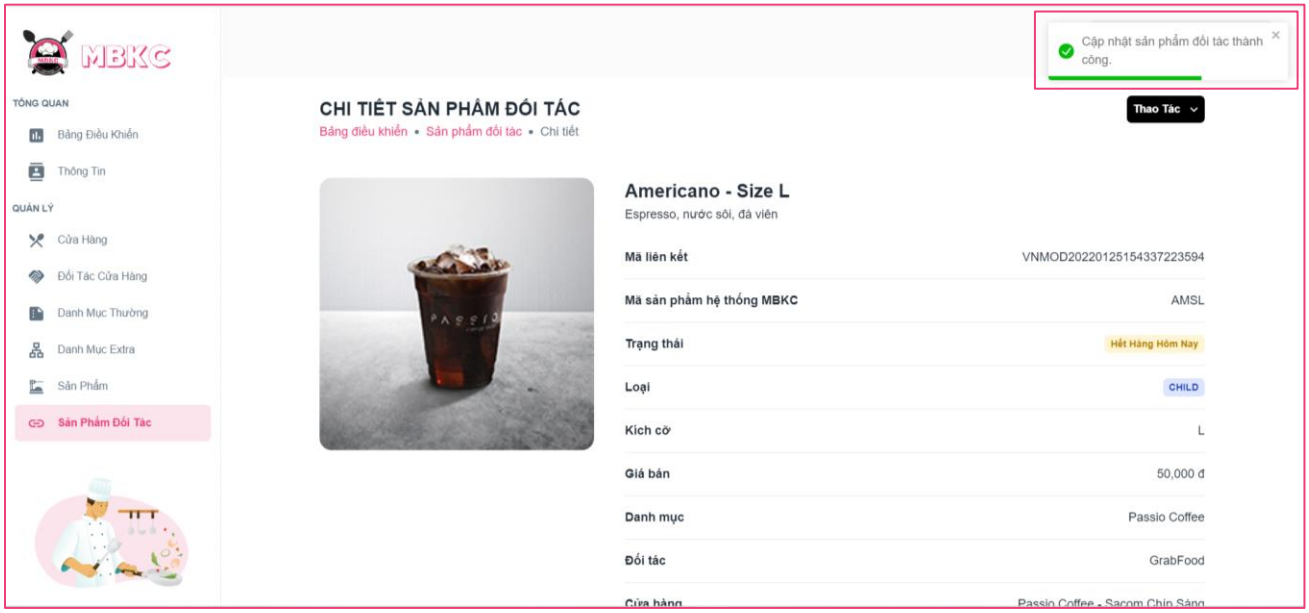


Figure 479: User Manual - Update Information Partner Product – Step 7 – Case 1

Case 2: List of Partner Products Page will be reloaded to show the list of partner products when start edit information of partner product from List of Partner Products Page and a toast message “Cập nhật sản phẩm đối tác thành công”.

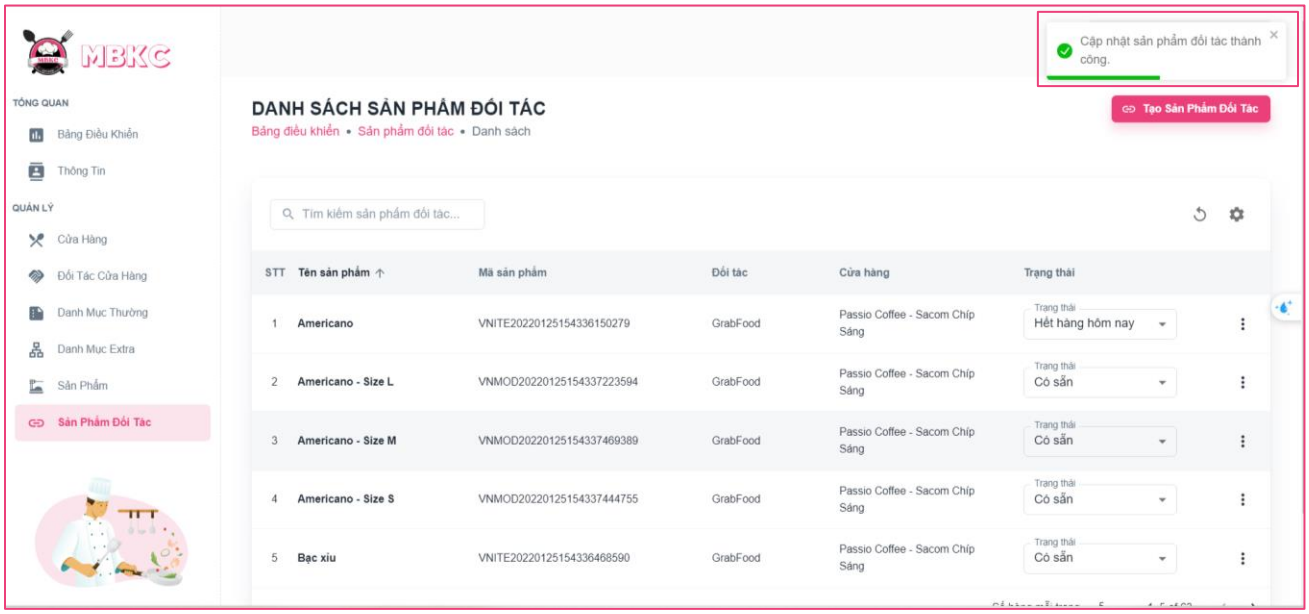


Figure 480: User Manual - Update Information Partner Product – Step 7 – Case 2

3.2.10.6. Delete Partner Product

3.2.10.6.1. Description

Use this guideline to delete partner product.

This web application supports delete partner product with account as a Brand Manager.

3.2.10.6.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Brand Manager.

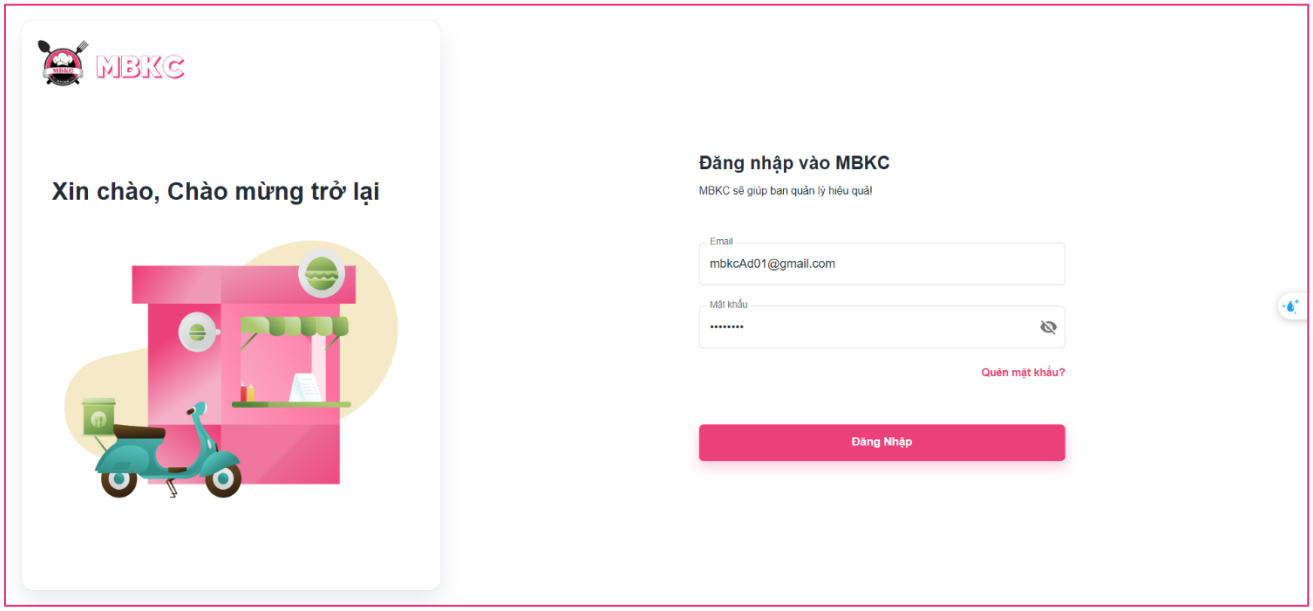


Figure 481: User Manual – Delete Partner Product – Step 1

Step 2: Click “Đăng Nhập” button.

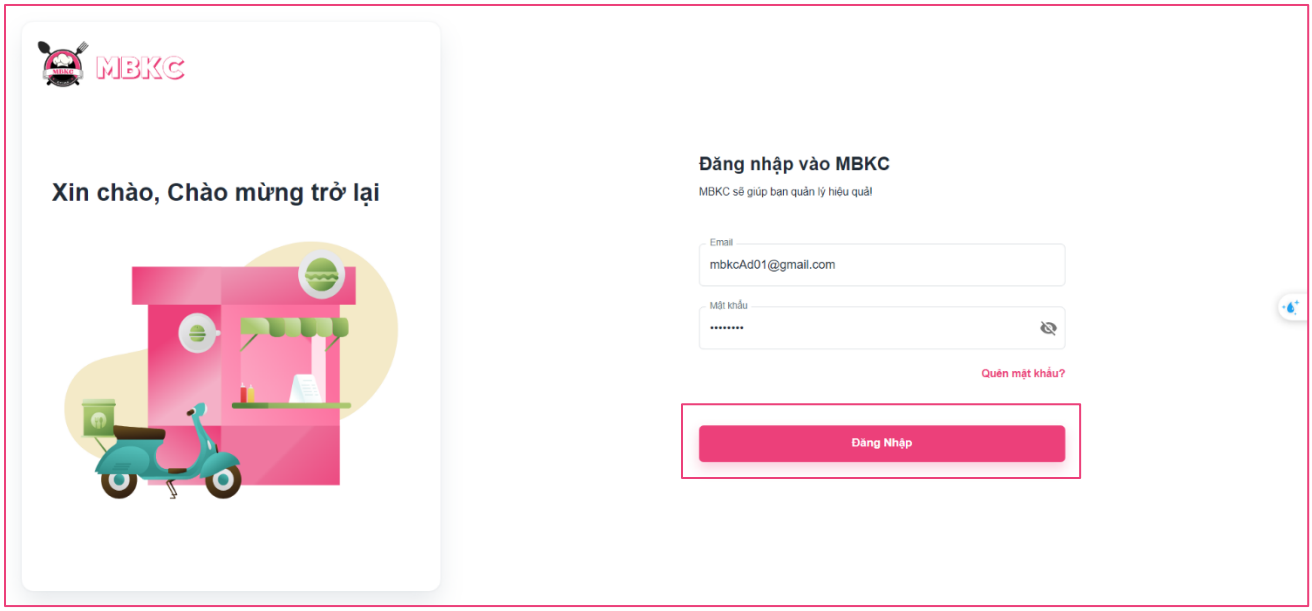


Figure 482: User Manual – Delete Partner Product – Step 2

Step 3: From Brand Dashboard Page, choose “Sản Phẩm Đối Tác” from the left sidebar.

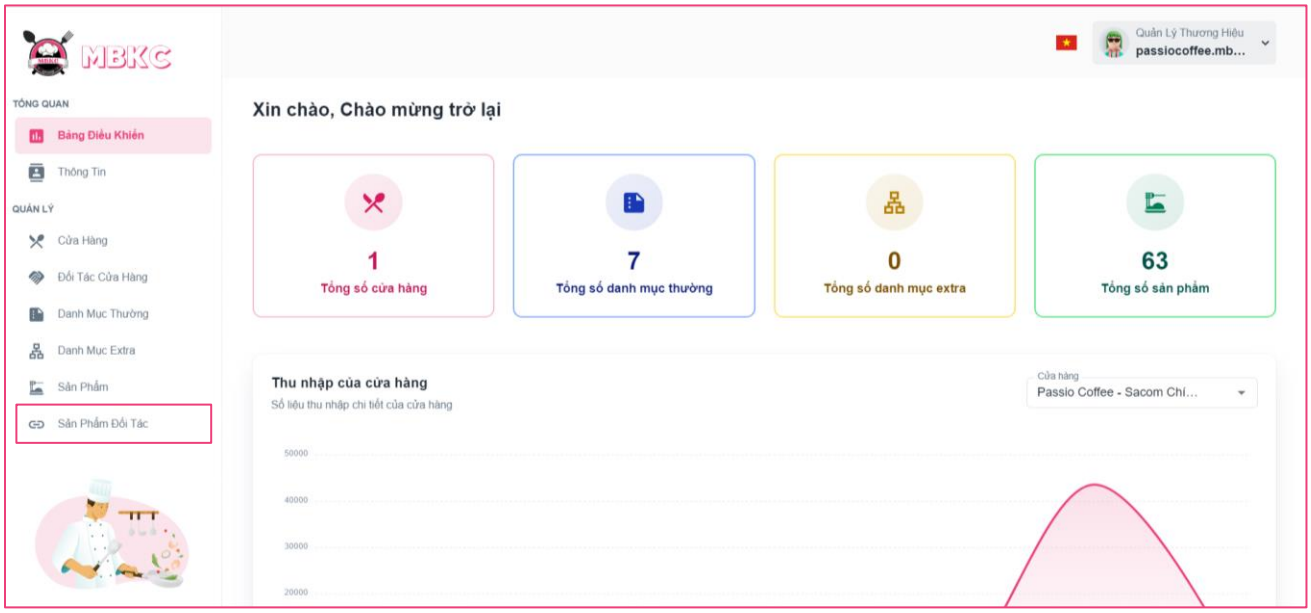


Figure 483: User Manual – Delete Partner Product – Step 3

Step 4: Delete partner product.

Case 1: Delete partner product in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the partner product row to delete.

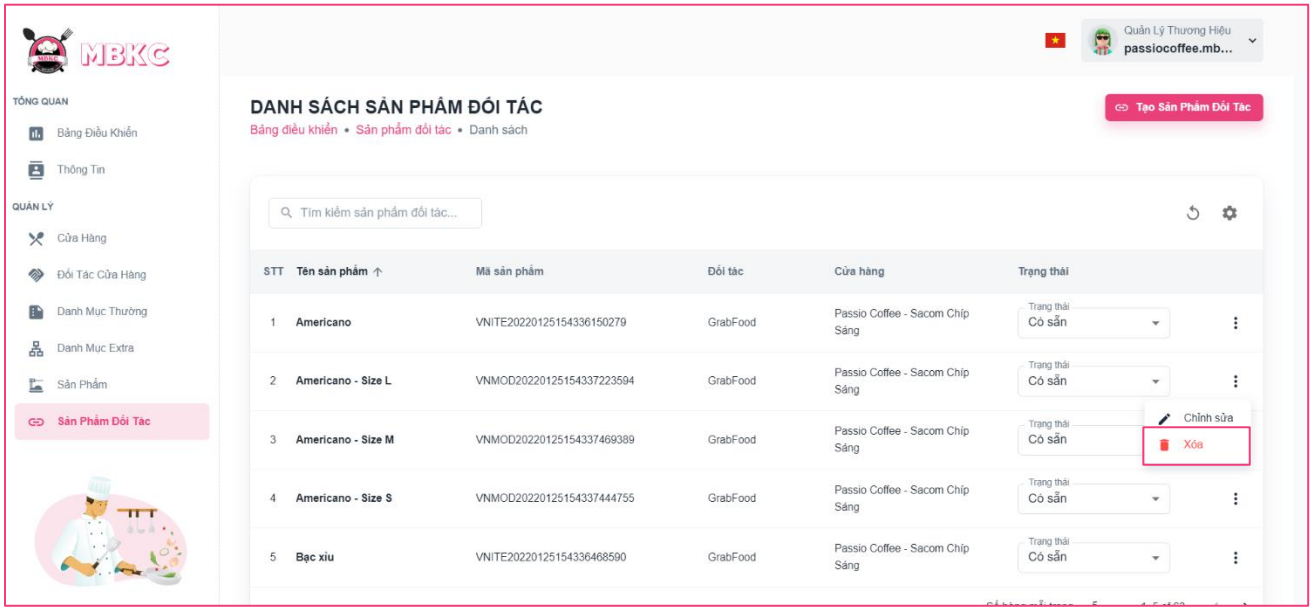


Figure 484: User Manual – Delete Partner Product – Step 4 – Case 1

Case 2: Delete partner product on the details page.

1: Click the partner product row to navigate to Partner Product Details Page.

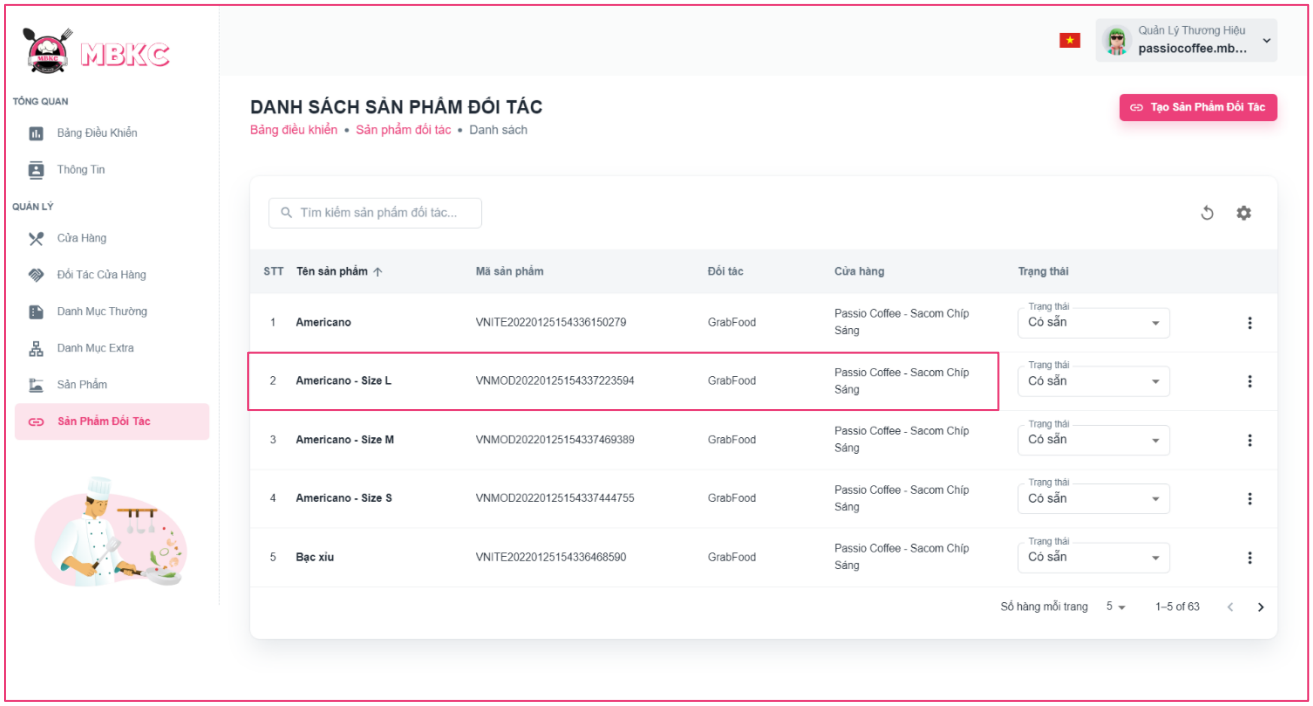


Figure 485: User Manual – Delete Partner Product – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item to delete.

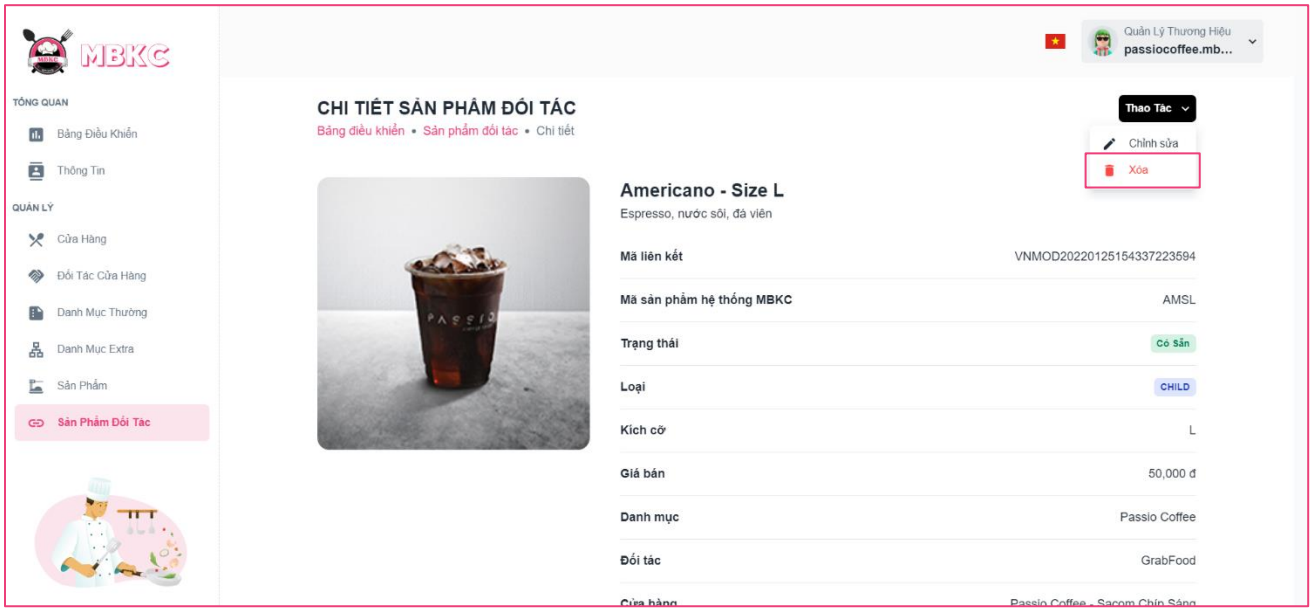


Figure 486: User Manual – Delete Partner Product – Step 4 – Case 2 - 2

Step 5: Click “Xác Nhận” button.

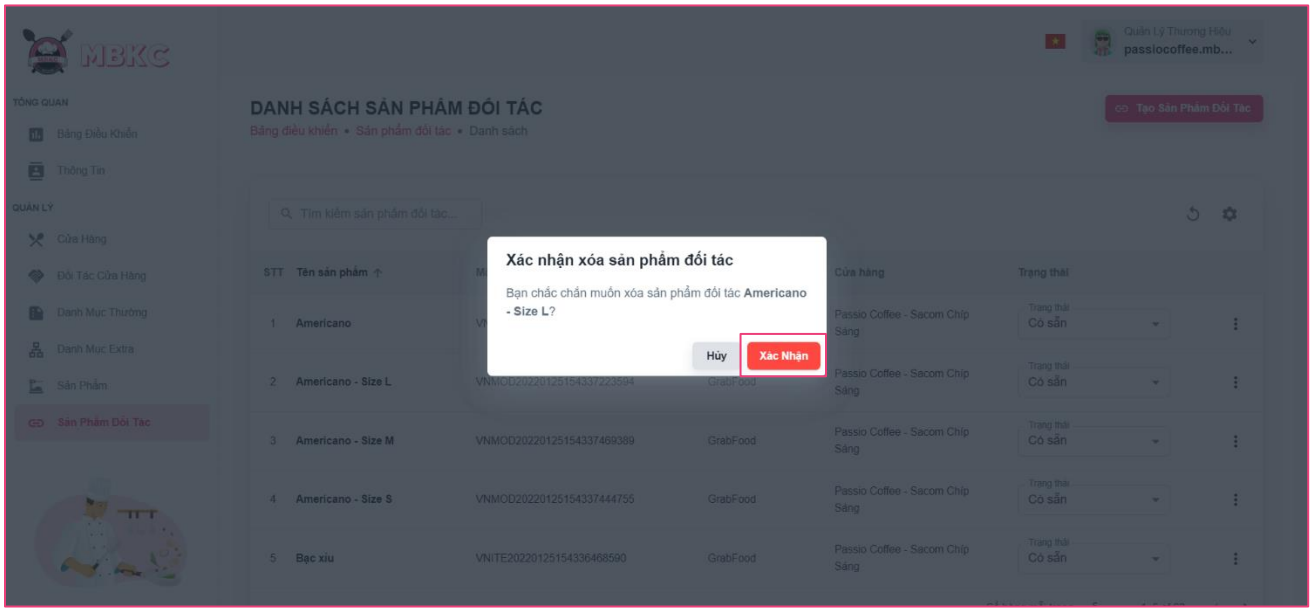


Figure 487: User Manual – Delete Partner Product – Step 5

Step 6: If you in Partner Product Details Page, you will navigate to List of Partner Products Page. List of partner products will be reloaded and a toast message “Đã xóa sản phẩm đối tác thành công”. And partner product has been deleted.

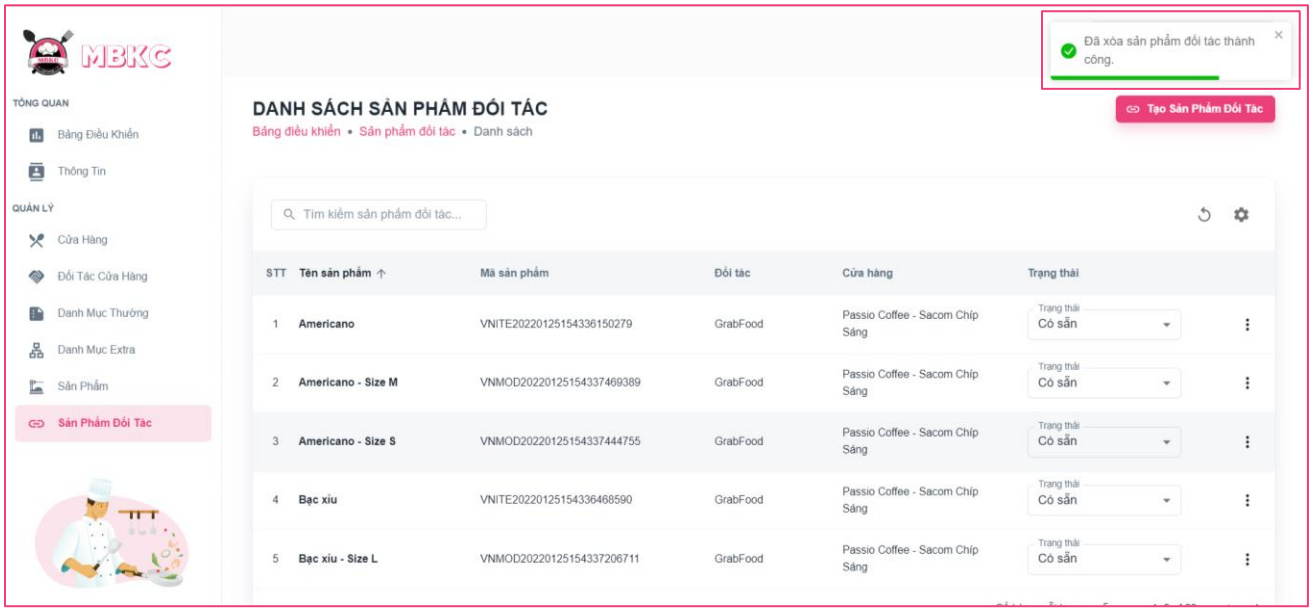


Figure 488: User Manual – Delete Partner Product – Step 6

3.2.11. Cashier management

3.2.11.1. Create new Cashier

3.2.11.1.1. Description
Use this guideline to login into MBKC account to create a new cashier.

This web application supports creating a new cashier with an account as a Kitchen Center Manager.

3.2.11.1.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

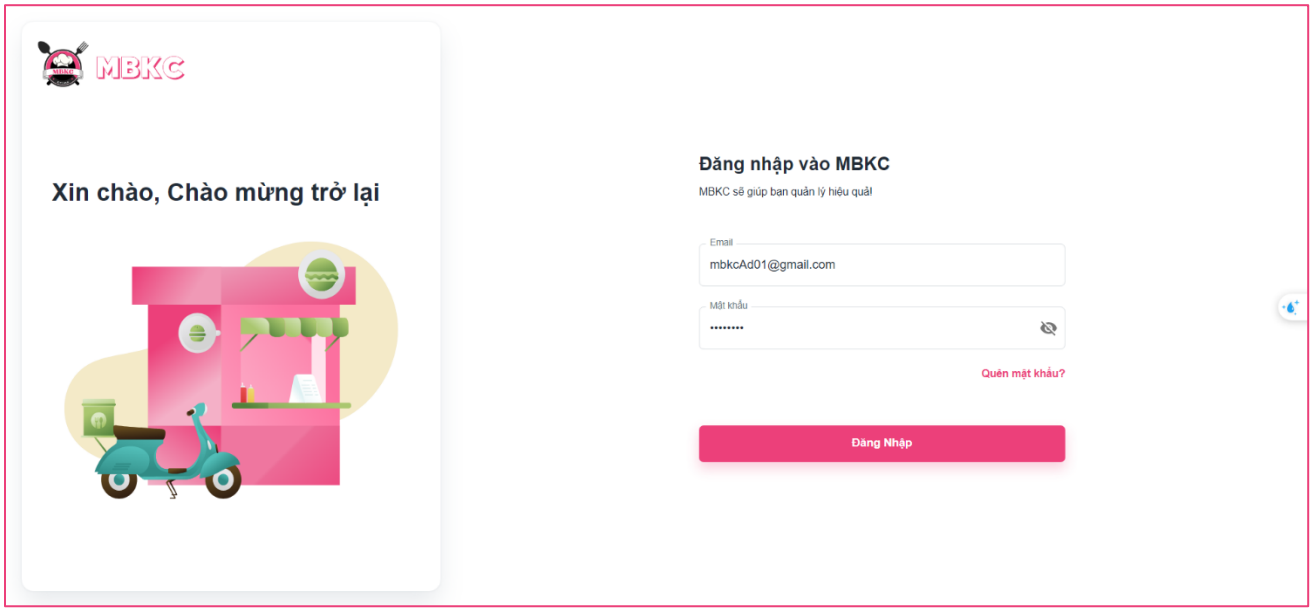


Figure 489: User Manual - Create new Cashier – Step 1

Step 2: Click “Đăng Nhập” button.

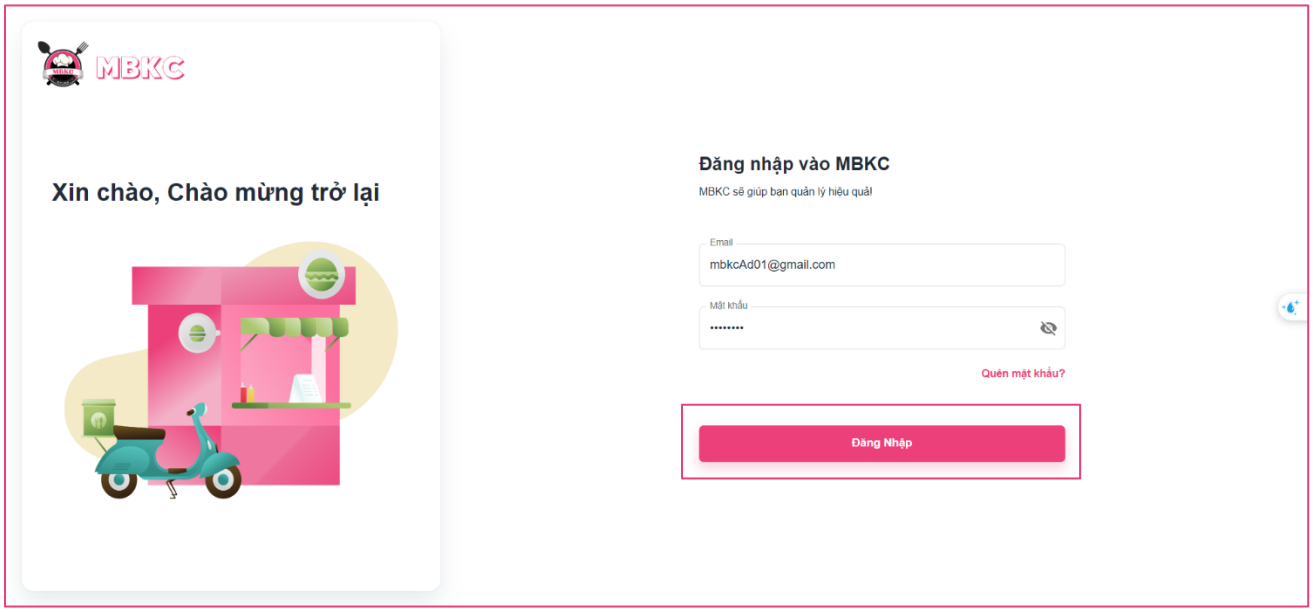


Figure 490: User Manual - Create new Cashier – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Nhân Viên Thu Ngân” from the left sidebar.

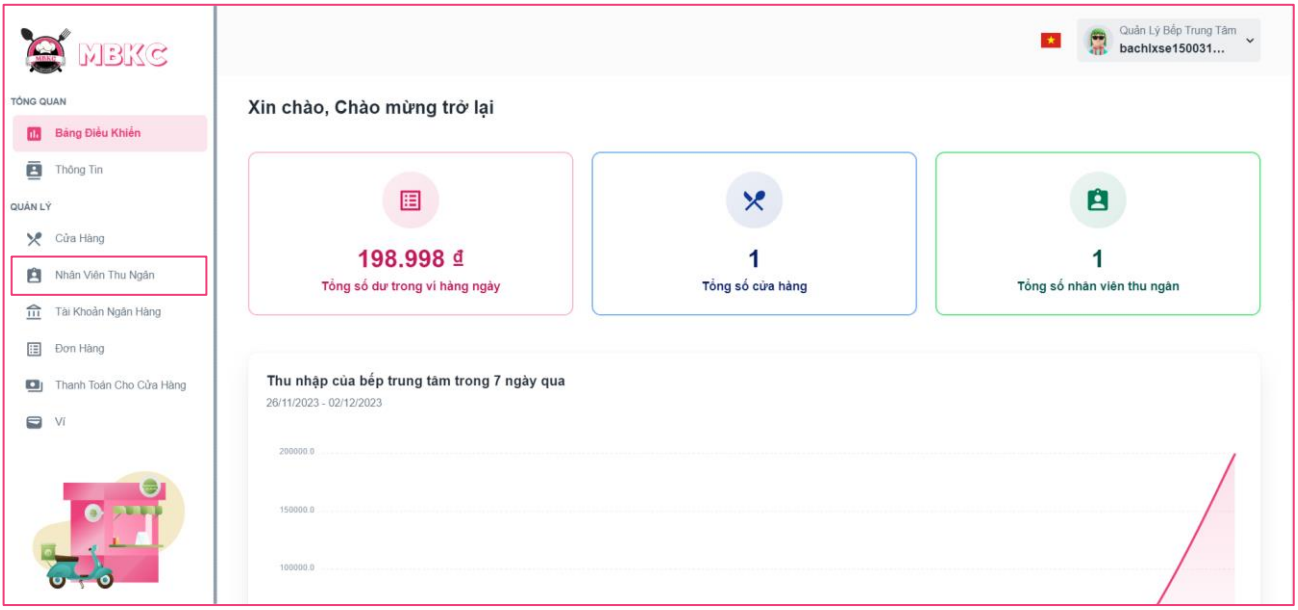


Figure 491: User Manual - Create new Cashier – Step 3

Step 4: Click “Thêm Nhân Viên Thu Ngân” button on the screen and navigate to Create New Cashier Page to add a new cashier.

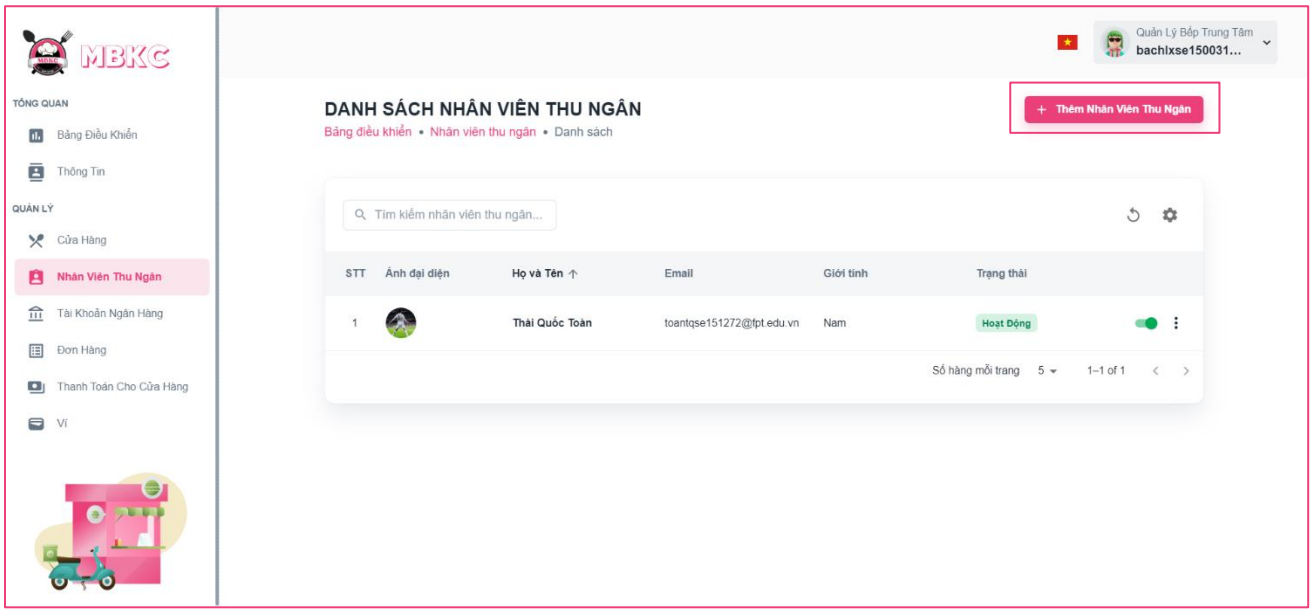


Figure 492: User Manual - Create new Cashier – Step 4

Step 5: Filling all the required information to add a cashier including: “Hình ảnh”, “Email", “Họ và Tên", “Giới tính”, “Ngày tháng năm sinh" and “Số chứng minh nhân dân”.

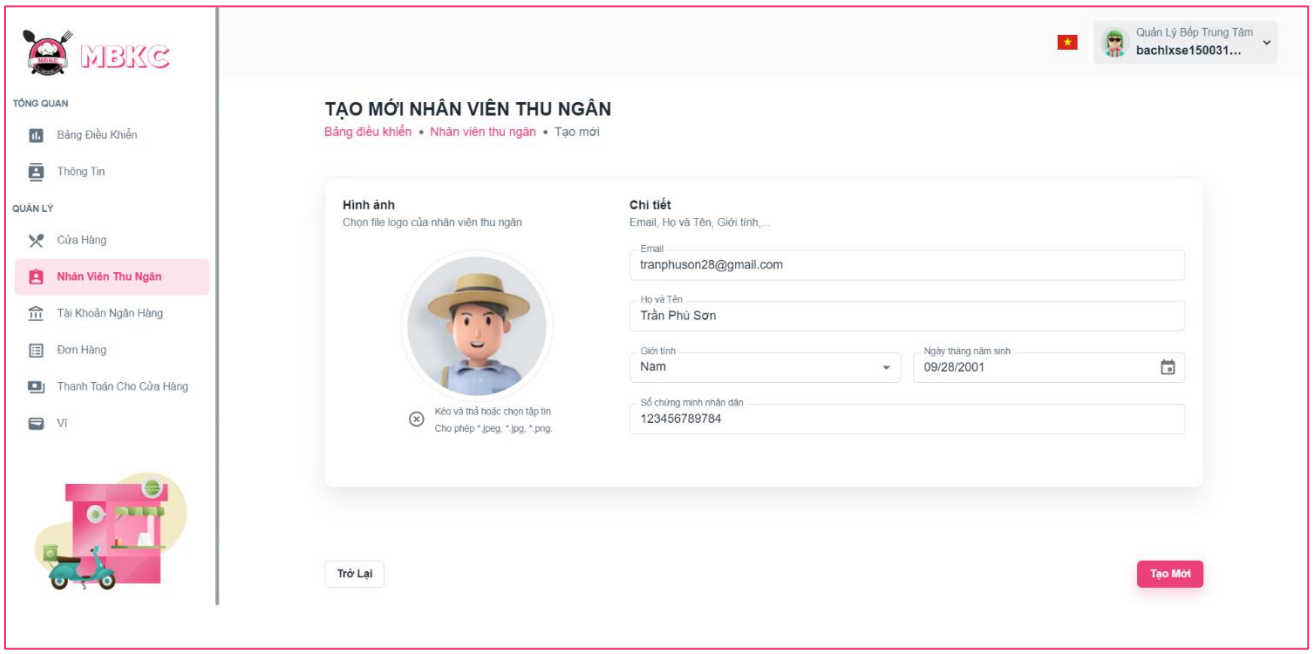


Figure 493: User Manual - Create new Cashier – Step 5

Step 6: After filling in all required field, click “Tạo Mới” to finish creating a new cashier.

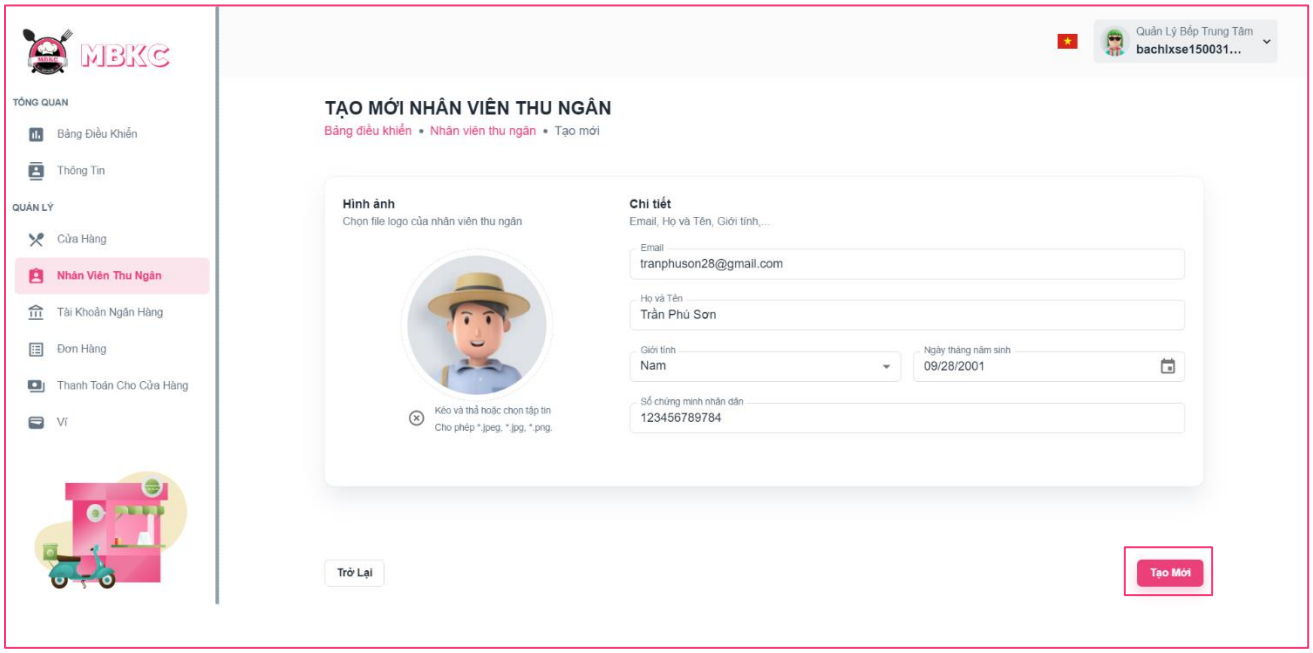


Figure 494: User Manual - Create new Cashier – Step 6

Step 7: You will be navigated to List of Cashiers Page showing the list of cashiers and a toast message “Tạo mới nhân viên thu ngân thành công”.

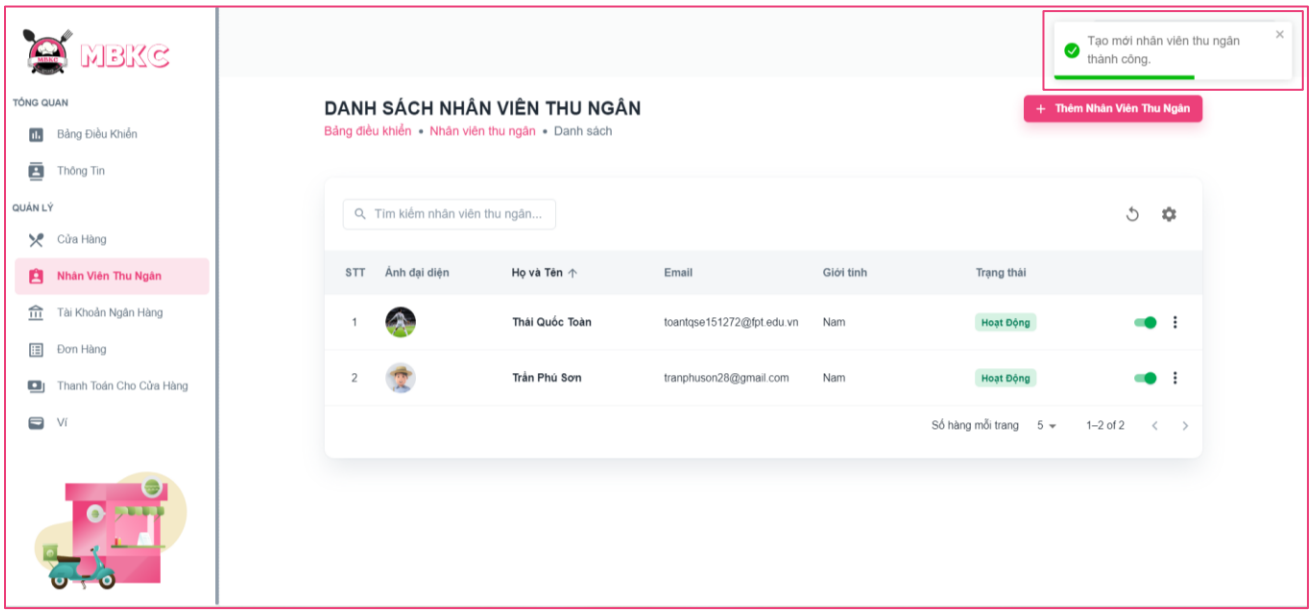


Figure 495: User Manual – Create new Cashier – Step 7

3.2.11.2. View list of Cashiers

3.2.11.2.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of cashiers.

This web application supports view list of cashiers with an account as a Kitchen Center Manager.

3.2.11.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

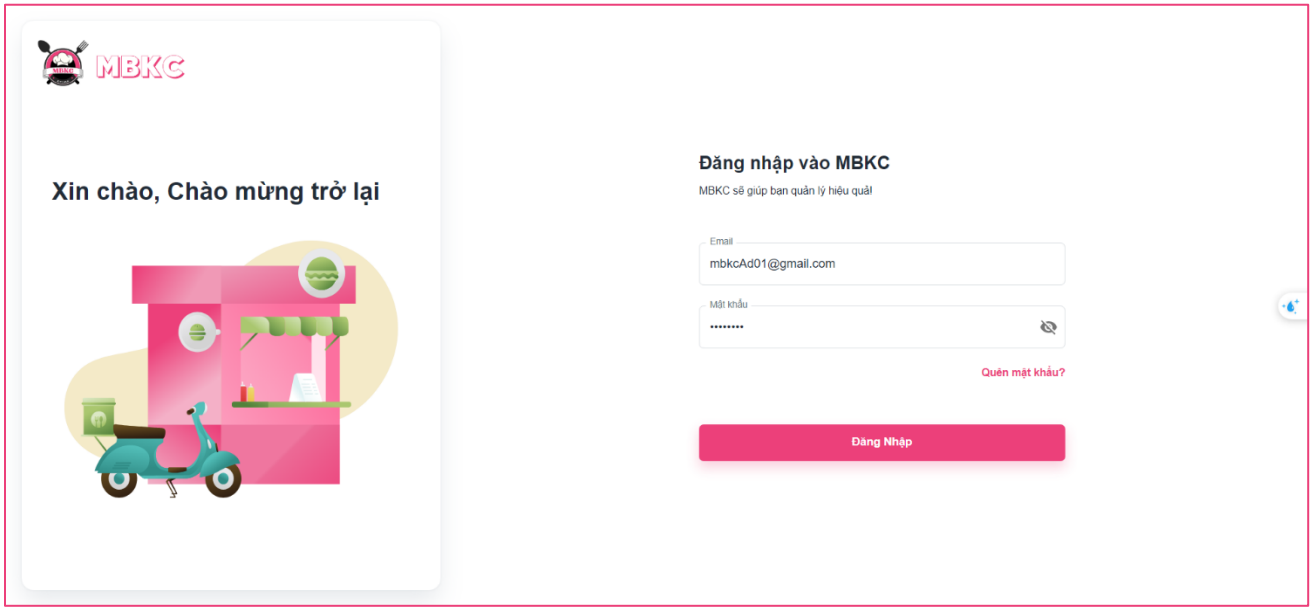


Figure 496: User Manual - View list of Cashiers – Step 1

Step 2: Click “Đăng Nhập” button.

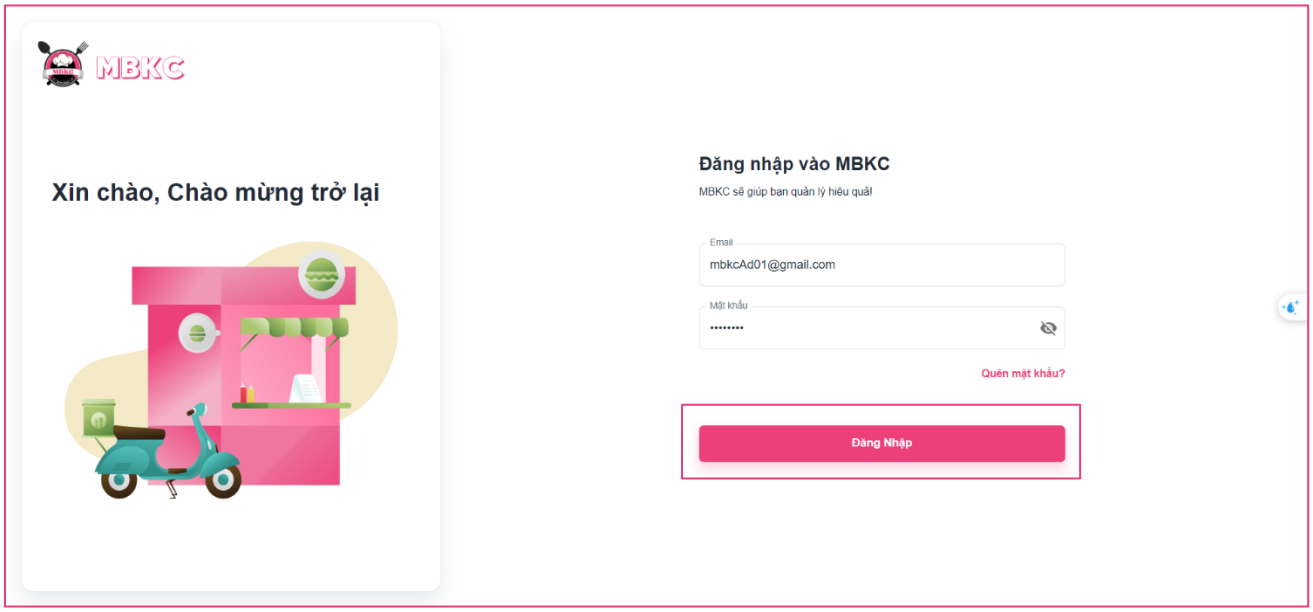


Figure 497: User Manual - View list of Cashiers – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Nhân Viên Thu Ngân” from the left sidebar.

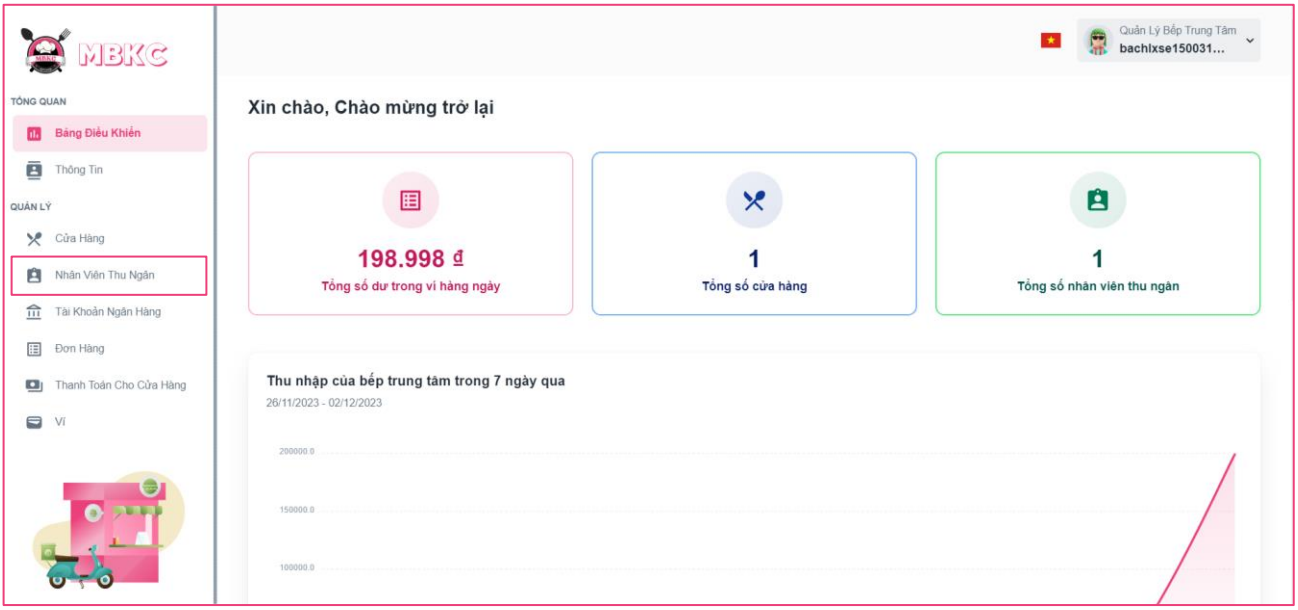


Figure 498: User Manual - View list of Cashiers – Step 3

Step 4: Search, reload data, and show or hide columns in the table list of cashiers.

Case 1: Fill in the cashier's name in the search box to search cashiers in the table list.

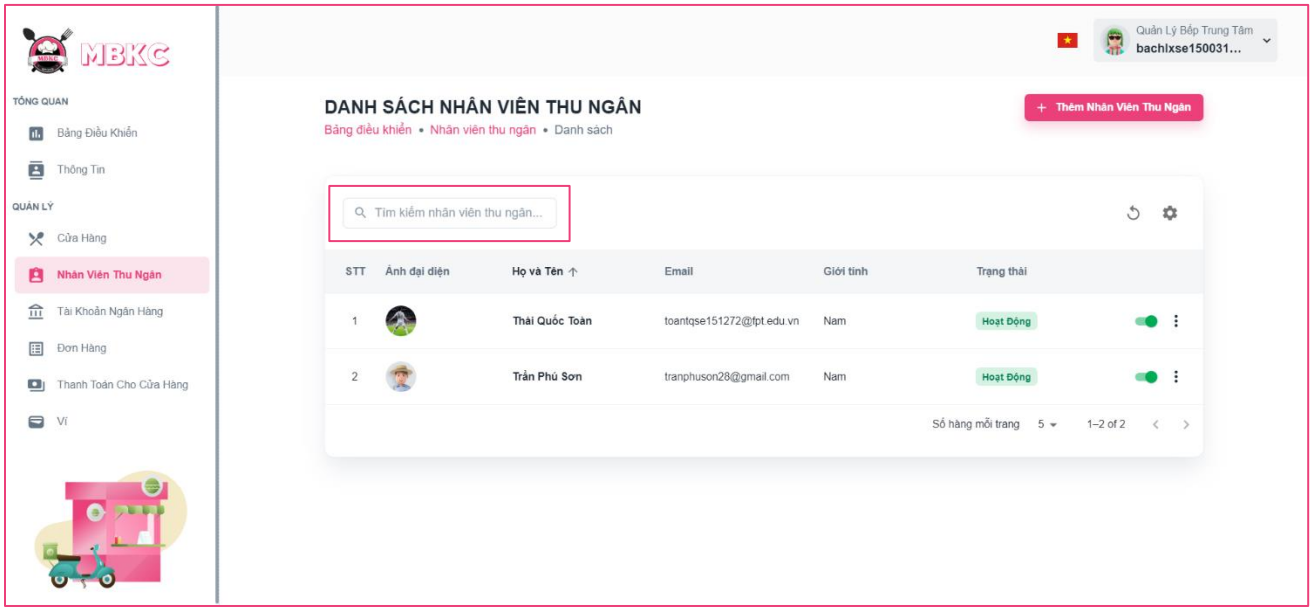


Figure 499: User Manual - View list of Cashiers – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of cashiers.

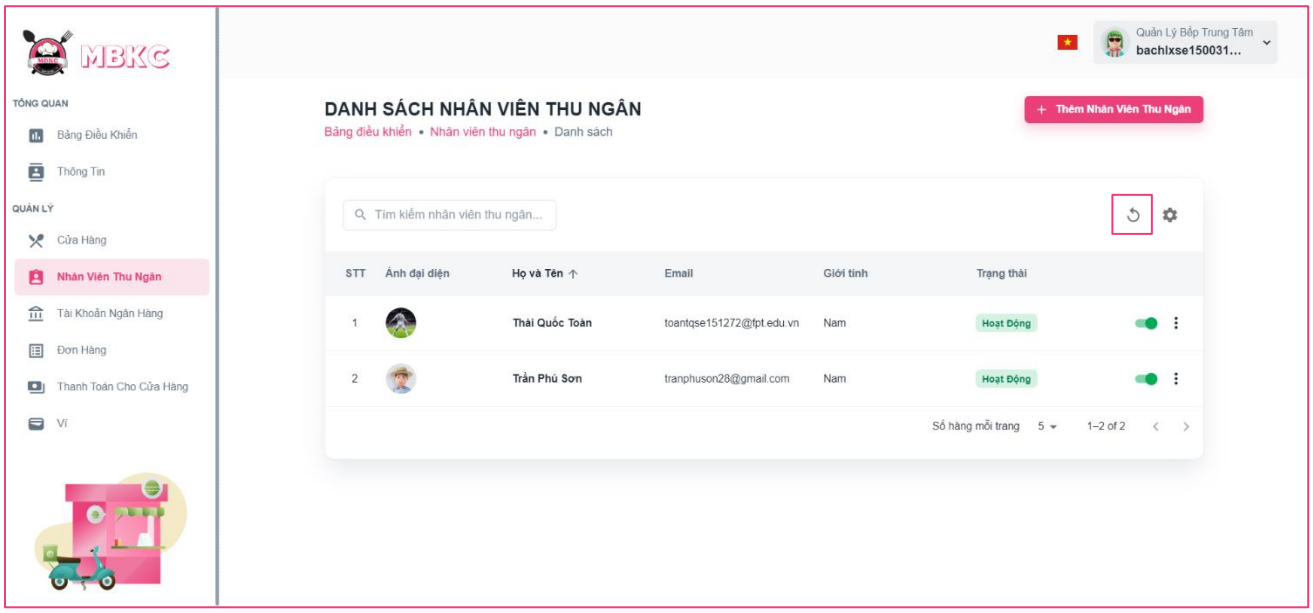


Figure 500: User Manual - View list of Cashiers – Step 4 – Case 2

Case 3: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of cashiers.

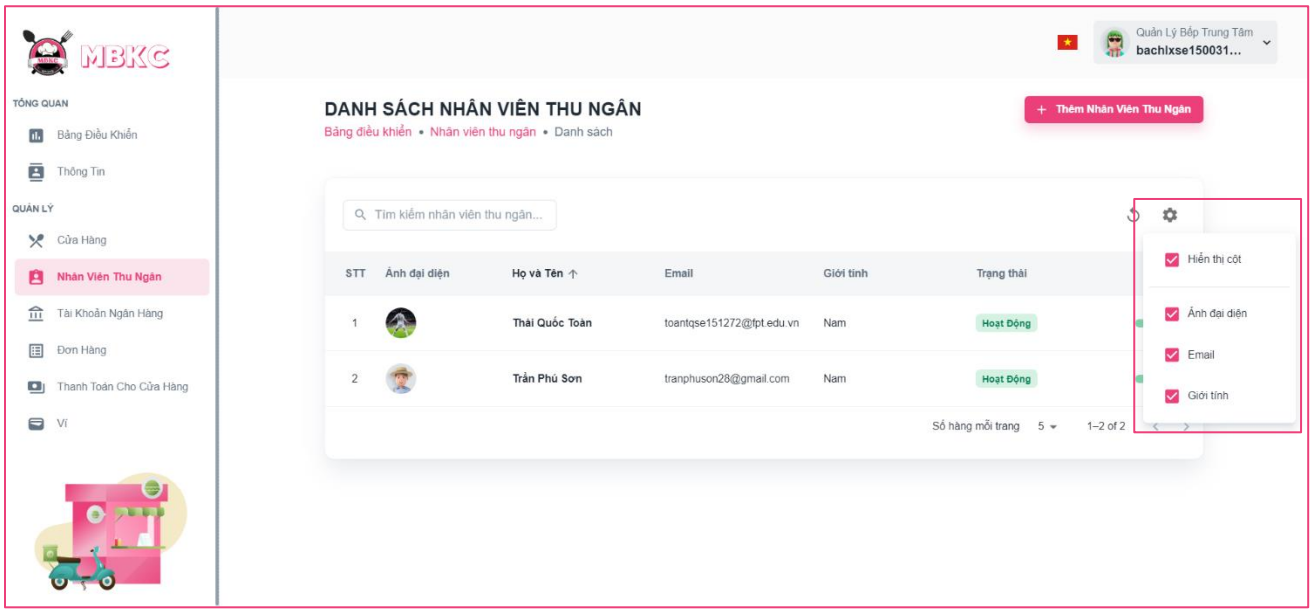


Figure 501: User Manual - View list of Cashiers – Step 4 – Case 3

3.2.11.3. View details Cashier

3.2.11.3.1. Description

Use this guideline to view details cashier.

This web application supports view details cashier with an account as a Kitchen Center Manager.

3.2.11.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

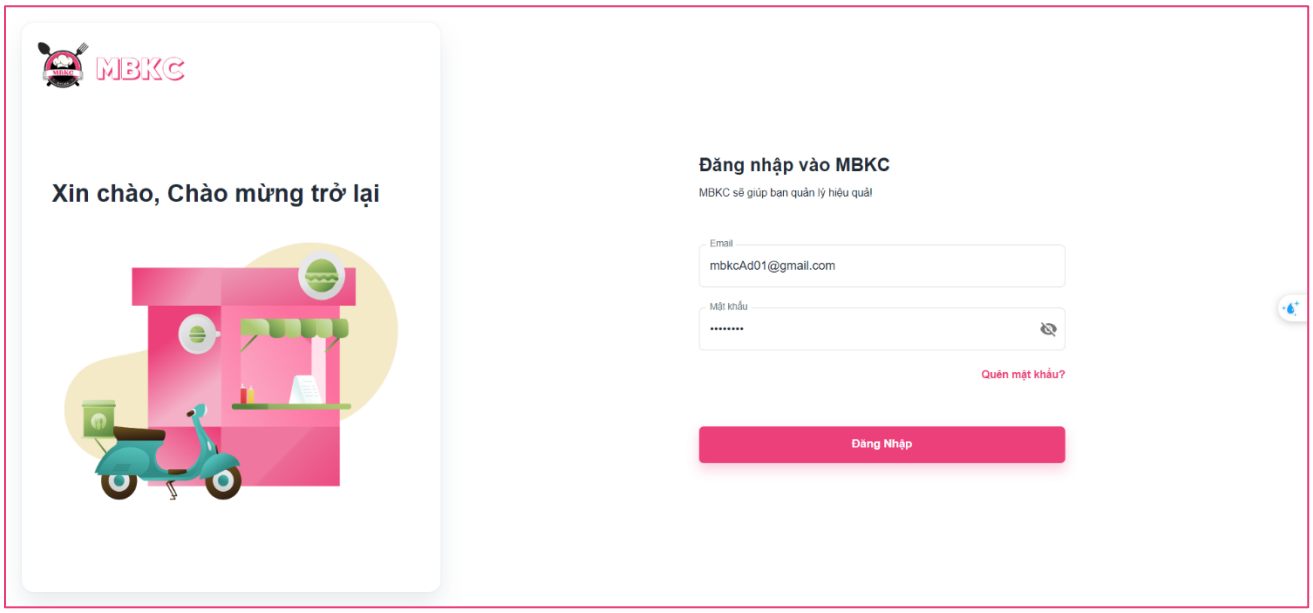


Figure 502: User Manual - View details Cashier – Step 1

Step 2: Click “Đăng Nhập” button.

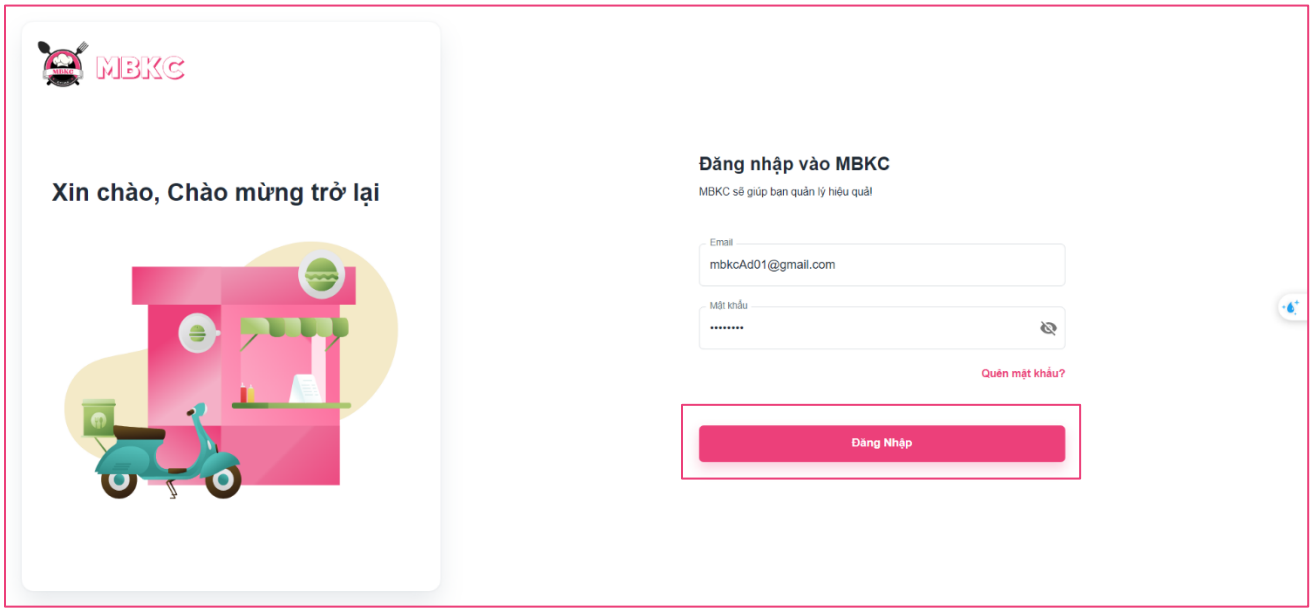


Figure 503: User Manual - View details Cashier – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Nhân Viên Thu Ngân” from the left sidebar.

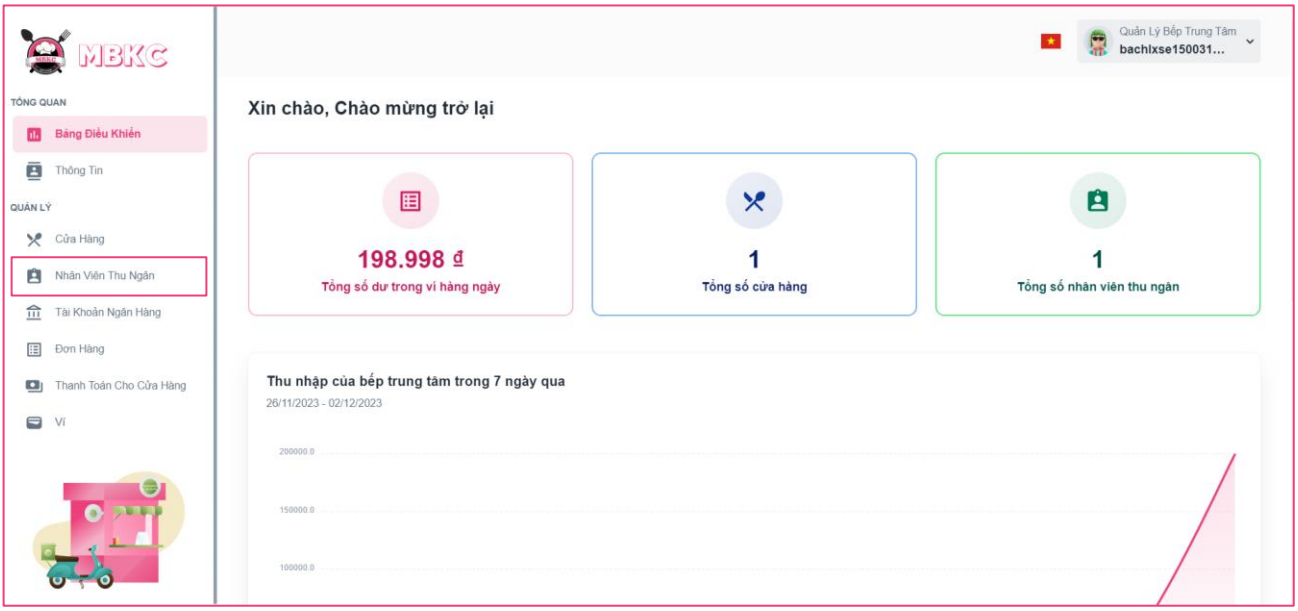


Figure 504: User Manual - View details Cashier – Step 3

Step 4: Click the cashier row to navigate to Cashier Details Page.

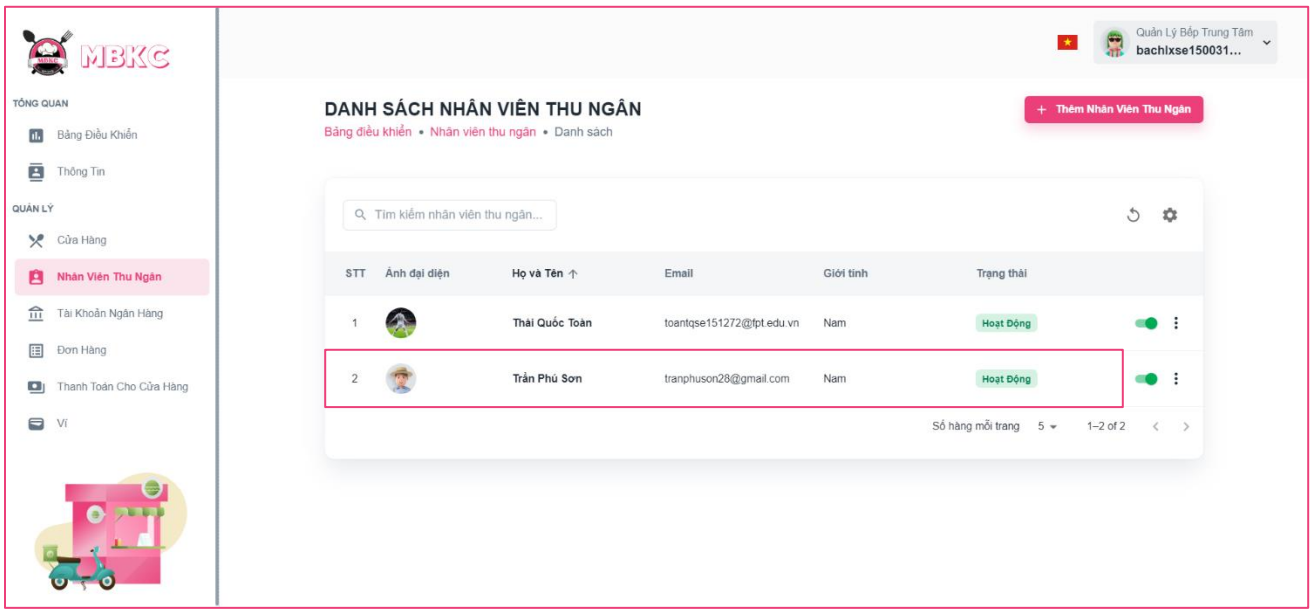


Figure 505: User Manual - View details Cashier – Step 4

Step 5: Navigate to Cashier Details Page to show information details of cashier.

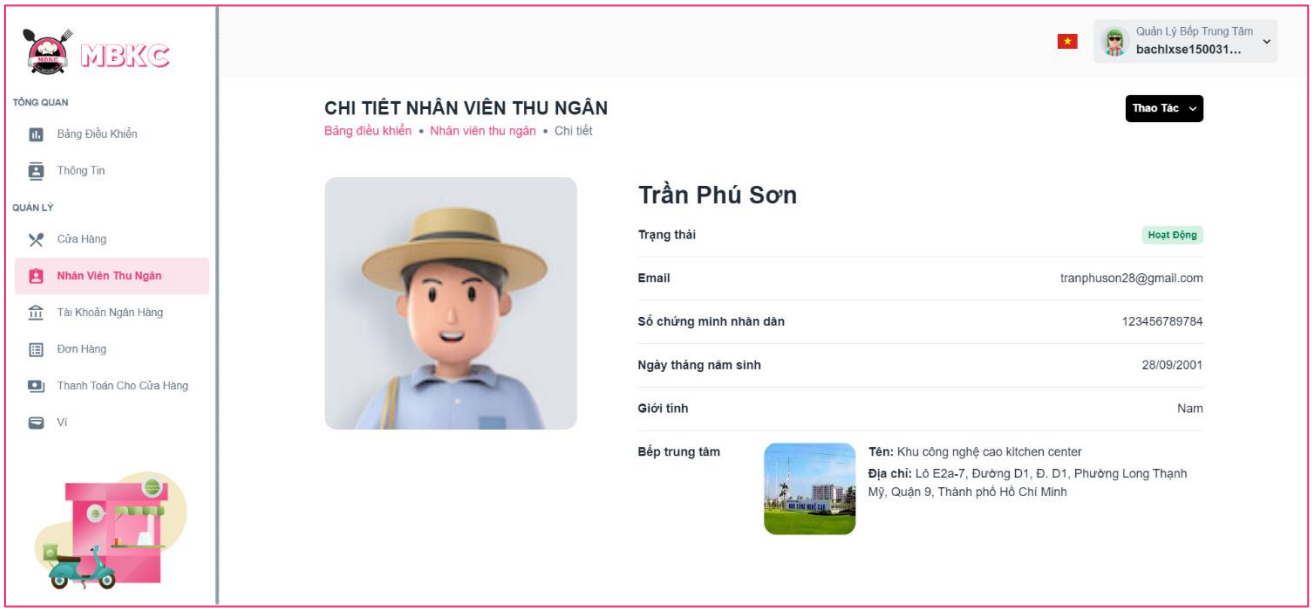


Figure 506: User Manual - View details Cashier – Step 5

3.2.11.4. Update Status Cashier

3.2.11.4.1. Description

Use this guideline to update the status of the cashier.

This web application supports update the status of the cashier with an account as a Kitchen Center Manager.

3.2.11.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

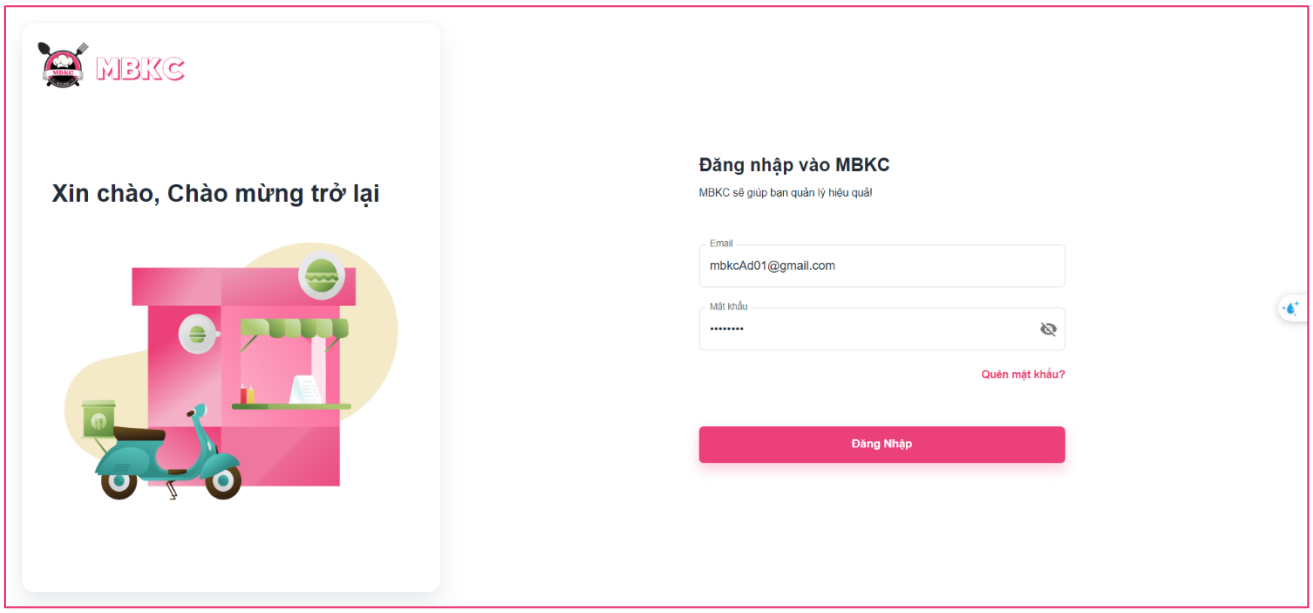


Figure 507: User Manual – Update Status Cashier – Step 1

Step 2: Click “Đăng Nhập” button.

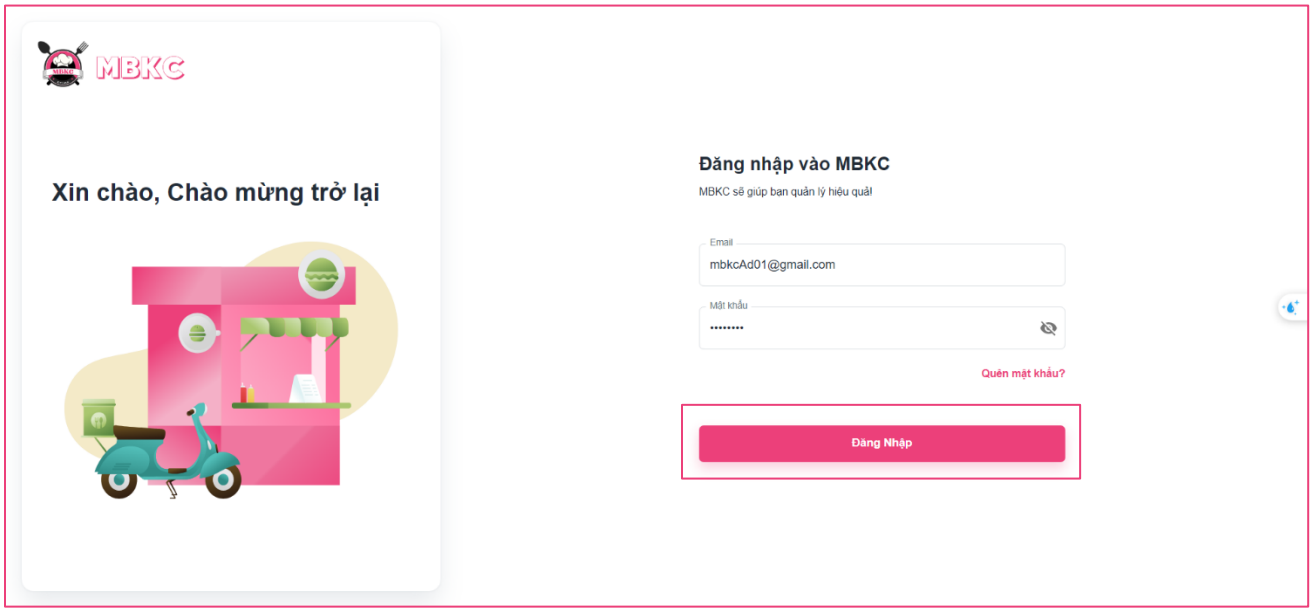


Figure 508: User Manual - Update Status Cashier – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Nhân Viên Thu Ngân” from the left sidebar.

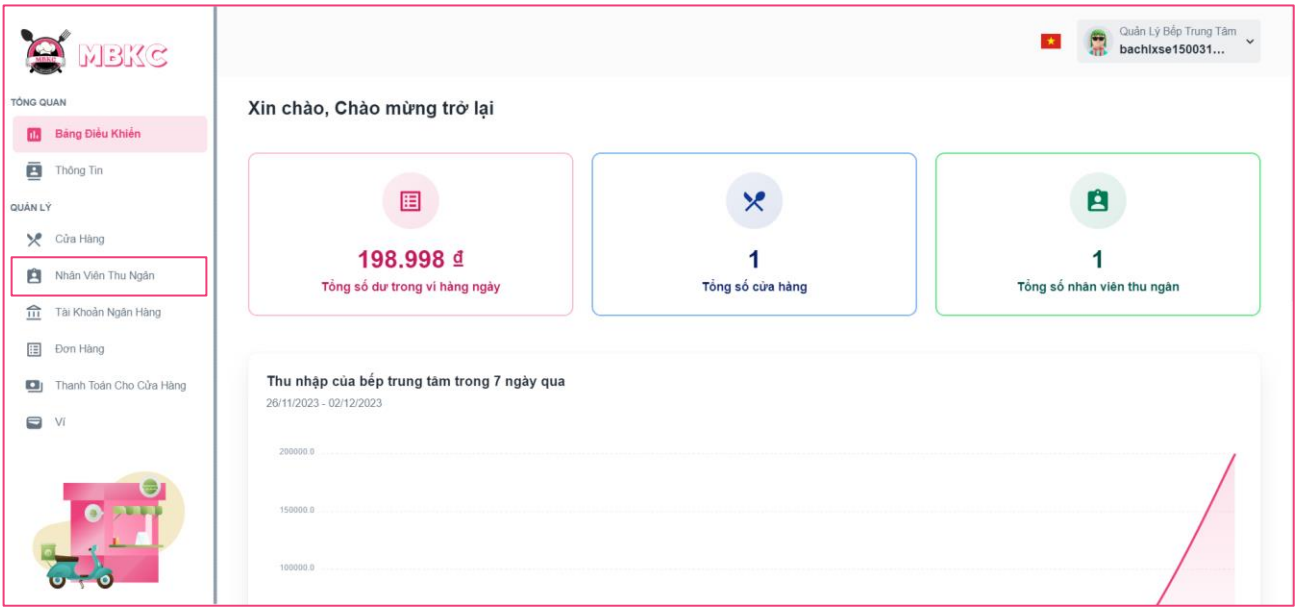


Figure 509: User Manual - Update Status Cashier – Step 3

Step 4: Click on the switch button on the cashier row to update the status.

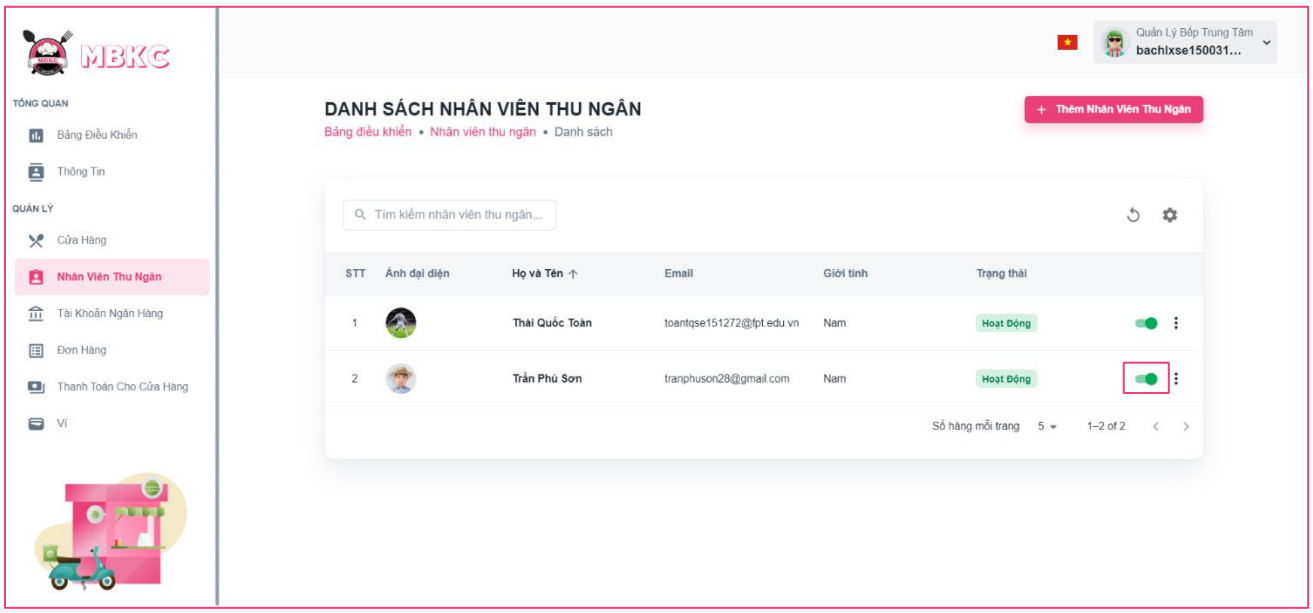


Figure 510: User Manual - Update Status Cashier – Step 4

Step 5: List of cashiers will be reloaded and a toast message “Cập nhật trạng thái nhân viên thu ngân thành công”. And the status of the cashier has been updated.

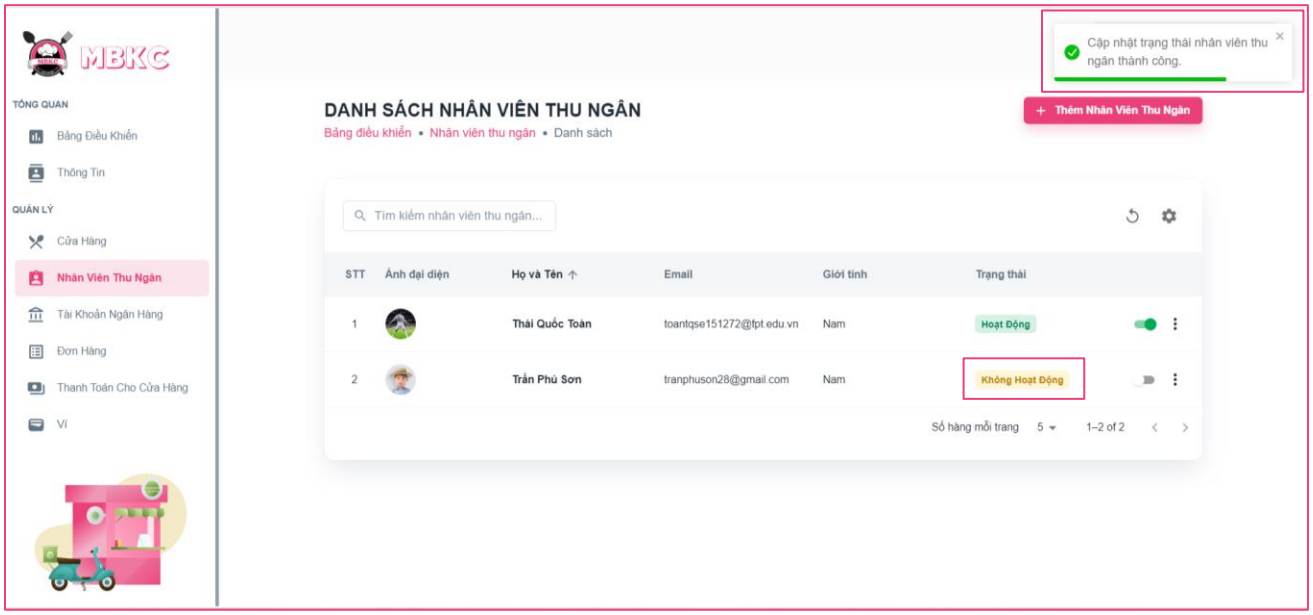


Figure 511: User Manual - Update Status Cashier – Step 5

3.2.11.5. Update Information Cashier

3.2.11.5.1. Description

Use this guideline to update information of the cashier.

This web application supports update information of the cashier with an account as a Kitchen Center Manager.

3.2.11.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

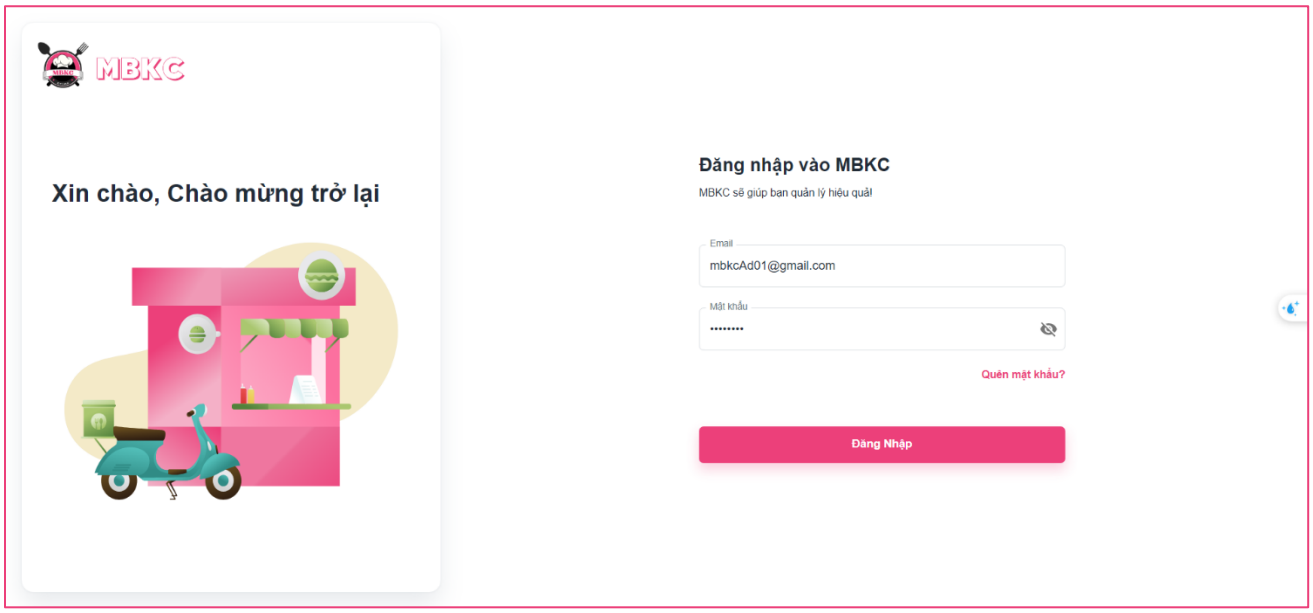


Figure 512: User Manual – Update Information Cashier – Step 1

Step 2: Click “Đăng Nhập” button.

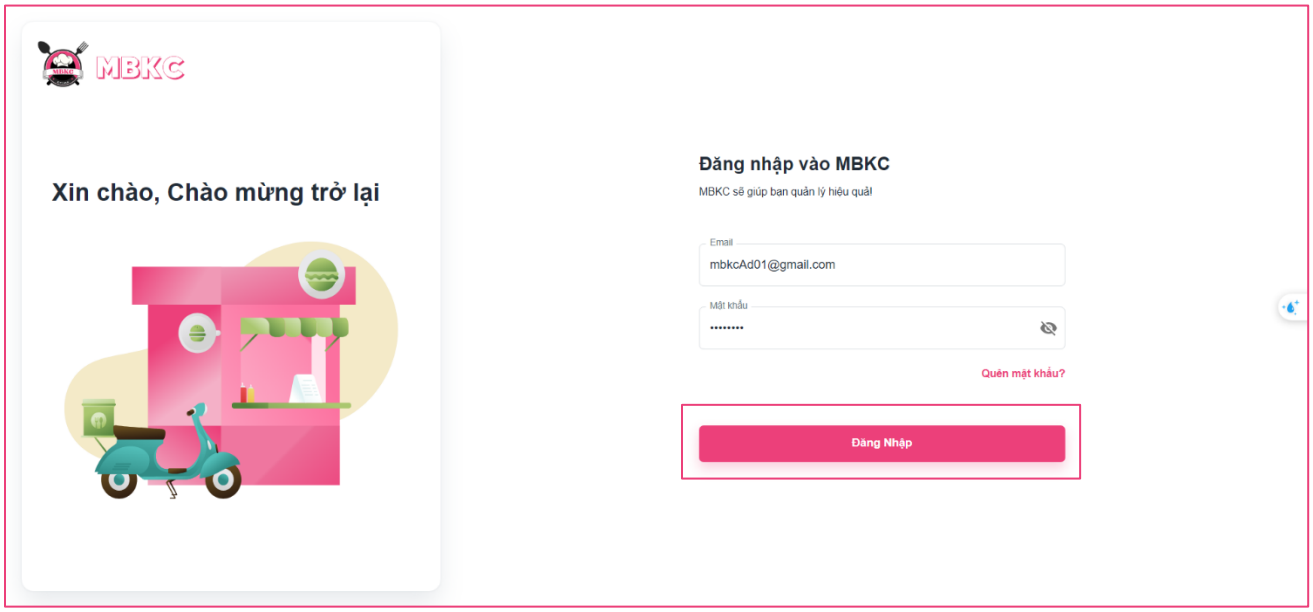


Figure 513: User Manual - Update Information Cashier – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Nhân Viên Thu Ngân” from the left sidebar.

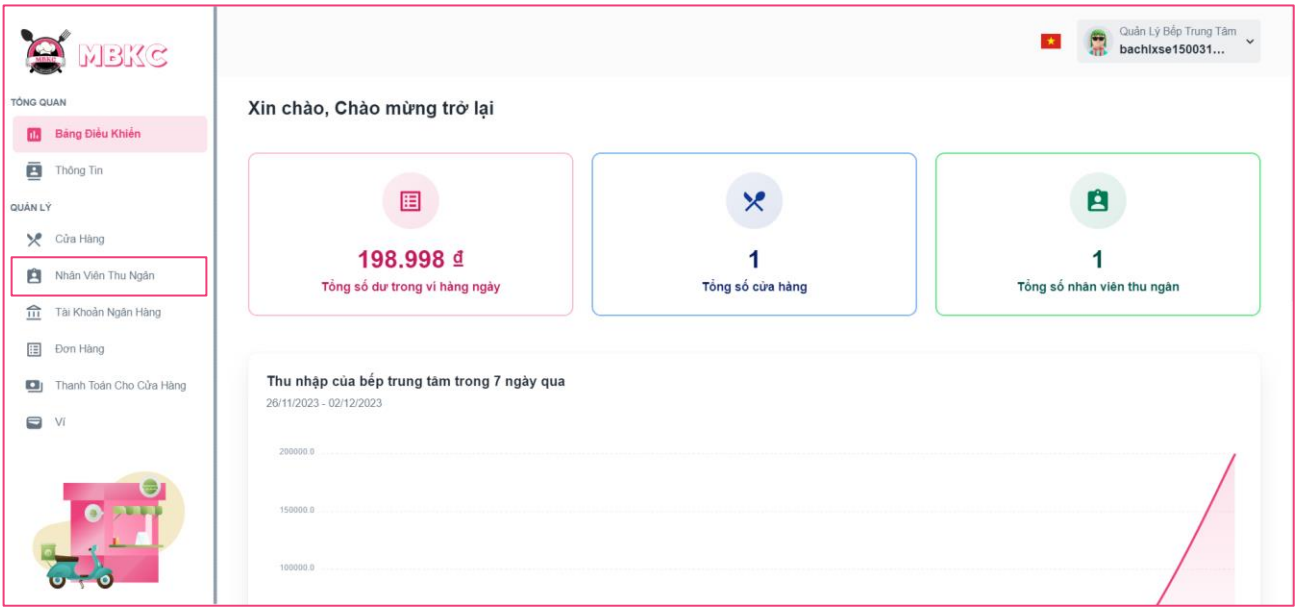


Figure 514: User Manual - Update Information Cashier – Step 3

Step 4: Update information of the cashier.

Case 1: Update information of the cashier in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the cashier row want to navigate Update Cashier Page.

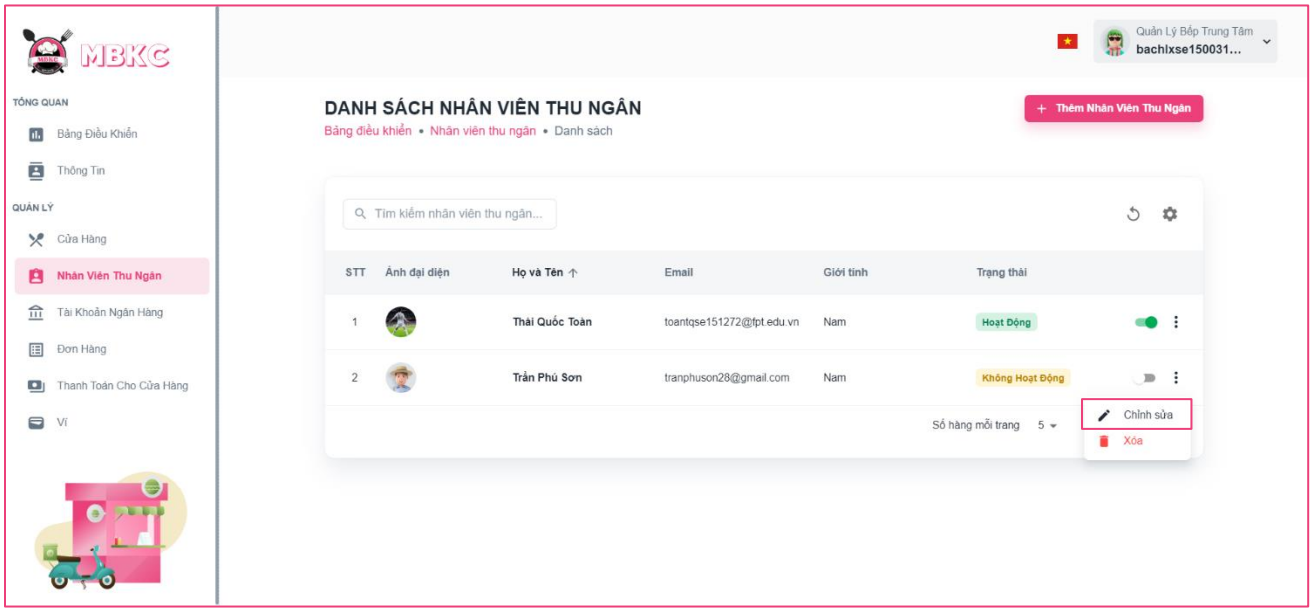


Figure 515: User Manual - Update Information Cashier – Step 4 – Case 1

Case 2: Update information of the cashier on the details page.

1: Click the cashier row to navigate to Cashier Details Page.

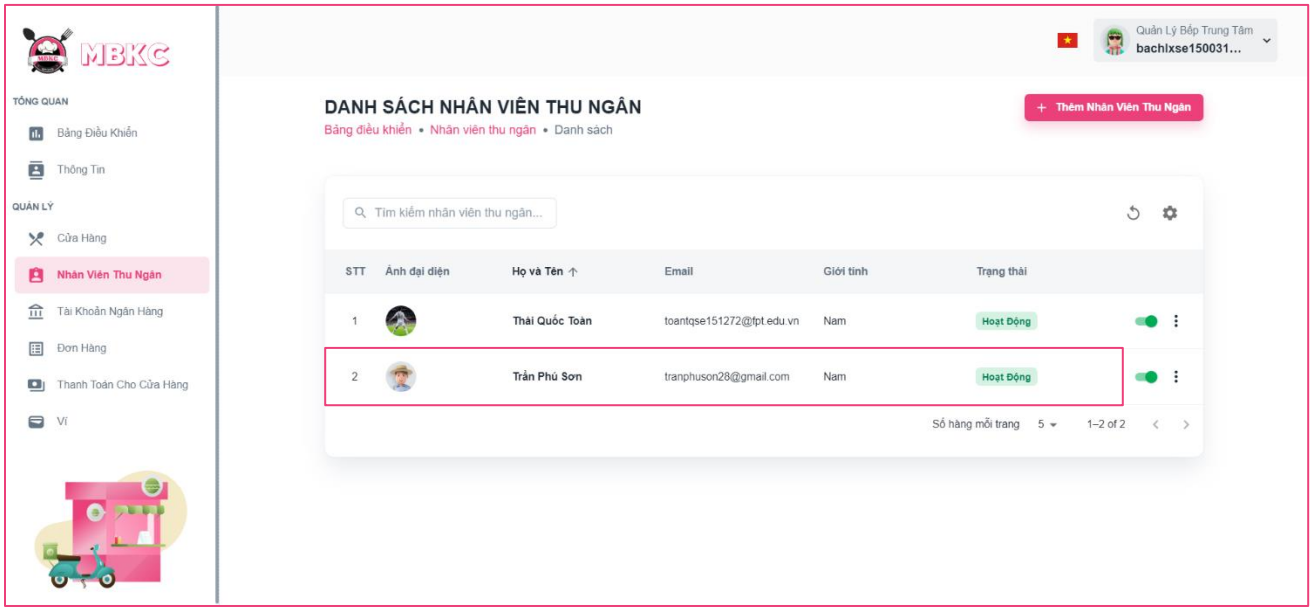


Figure 516: User Manual - Update Information Cashier – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Chỉnh sửa” menu item to navigate Update Cashier Page.

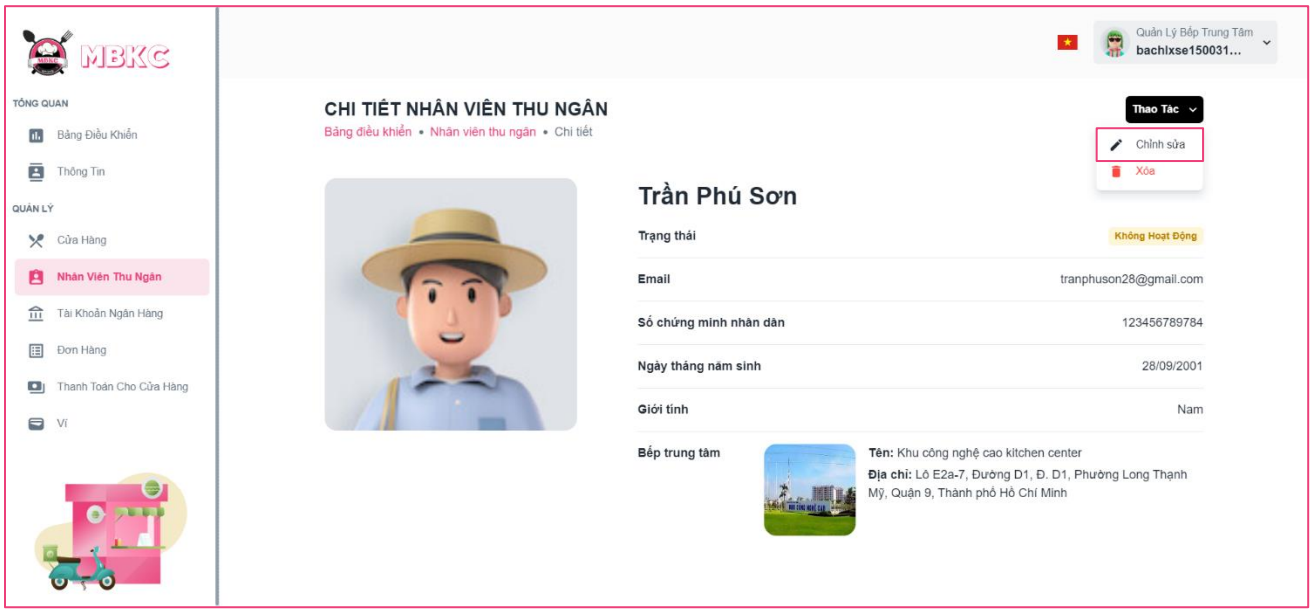


Figure 517: User Manual - Update Information Cashier – Step 4 – Case 2 - 2

Step 5: Cashier information will be automatically filled in the form. Then edit the information as desired.

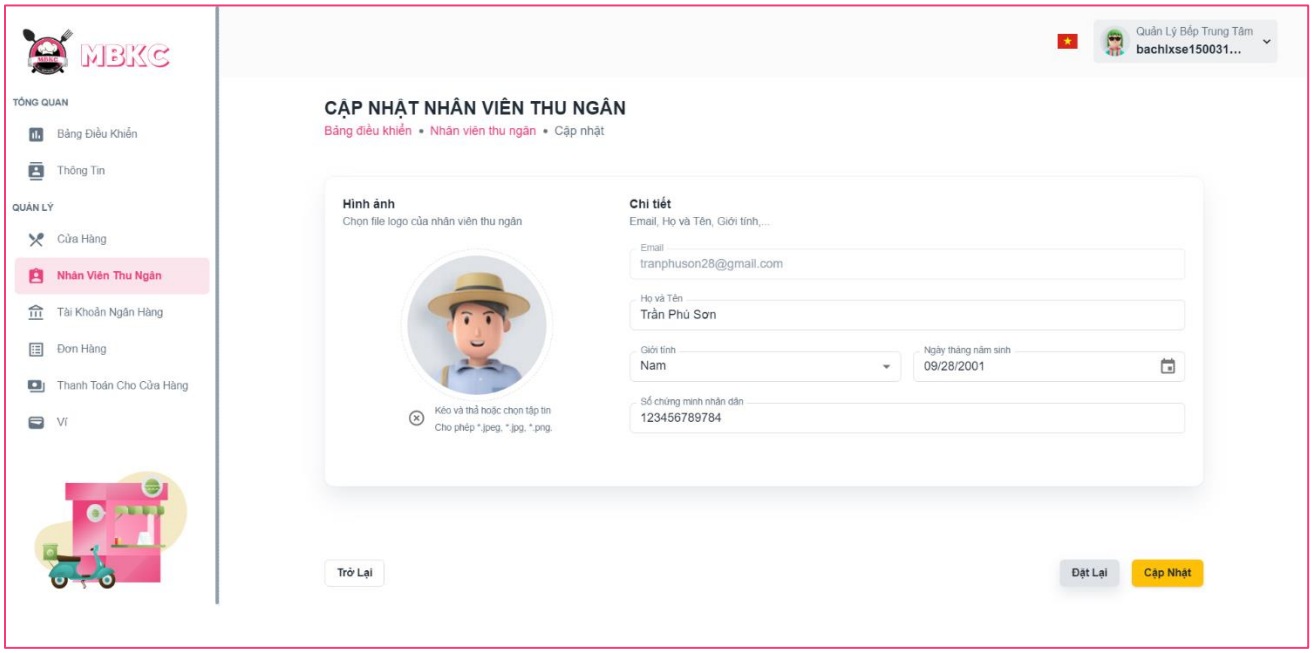


Figure 518: User Manual - Update Information Cashier – Step 5

Step 6: Click “Cập nhật” button.

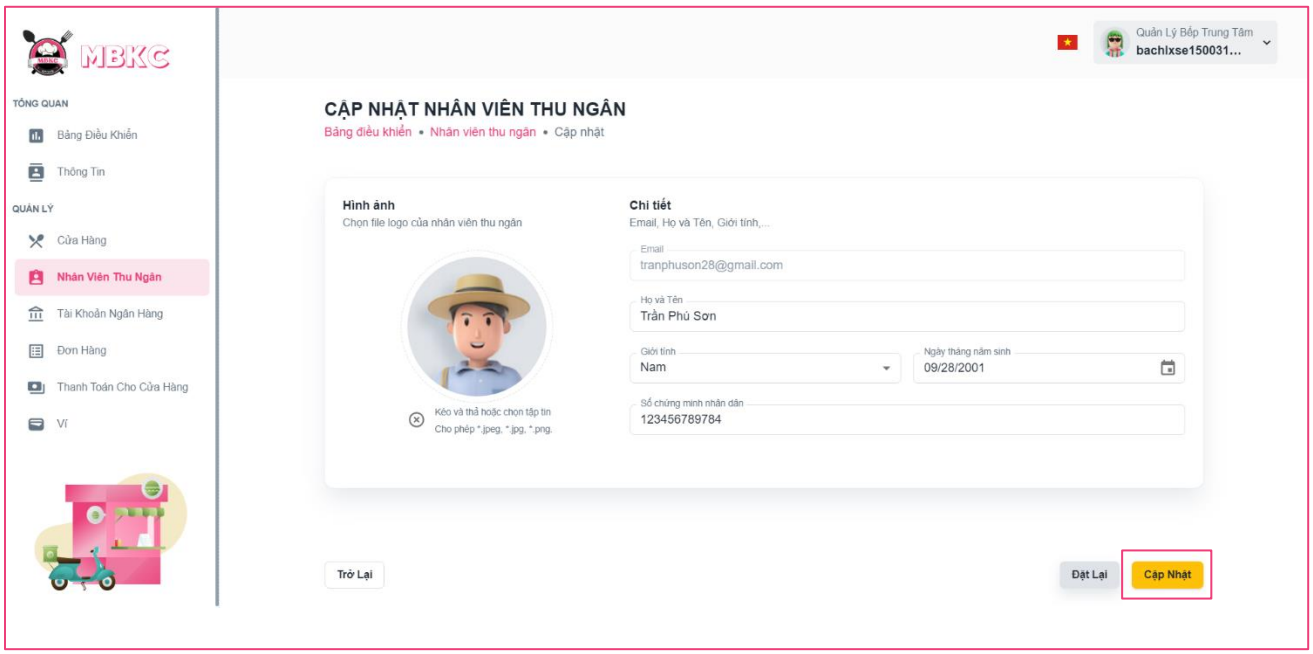


Figure 519: User Manual - Update Information Cashier – Step 6

Step 7:

Case 1: You will be navigated to Cashier Detail Page showing the cashier details when start edit information of cashier from Cashier Detail Page and a toast message “Cập nhật nhân viên thu ngân thành công”.

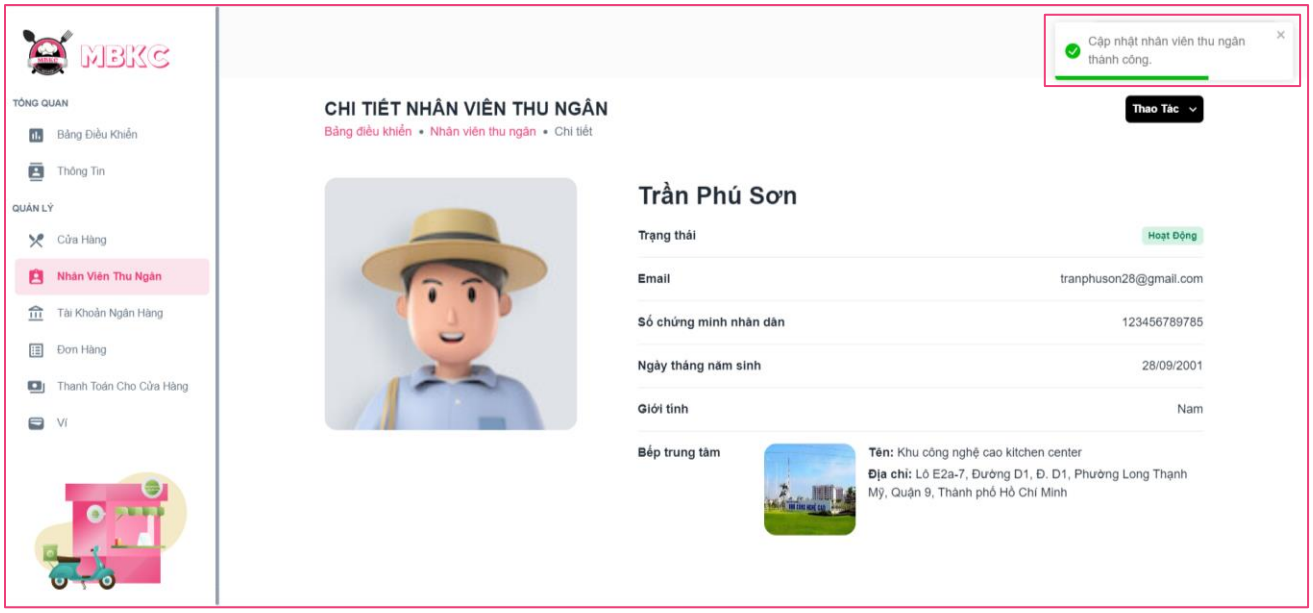


Figure 520: User Manual - Update Information Cashier – Step 7 – Case 1

Case 2: You will be navigated to List of Cashiers Page showing the list of cashiers when start edit information of cashier from List of Cashiers Page and a toast message “Cập nhật nhân viên thu ngân thành công”.

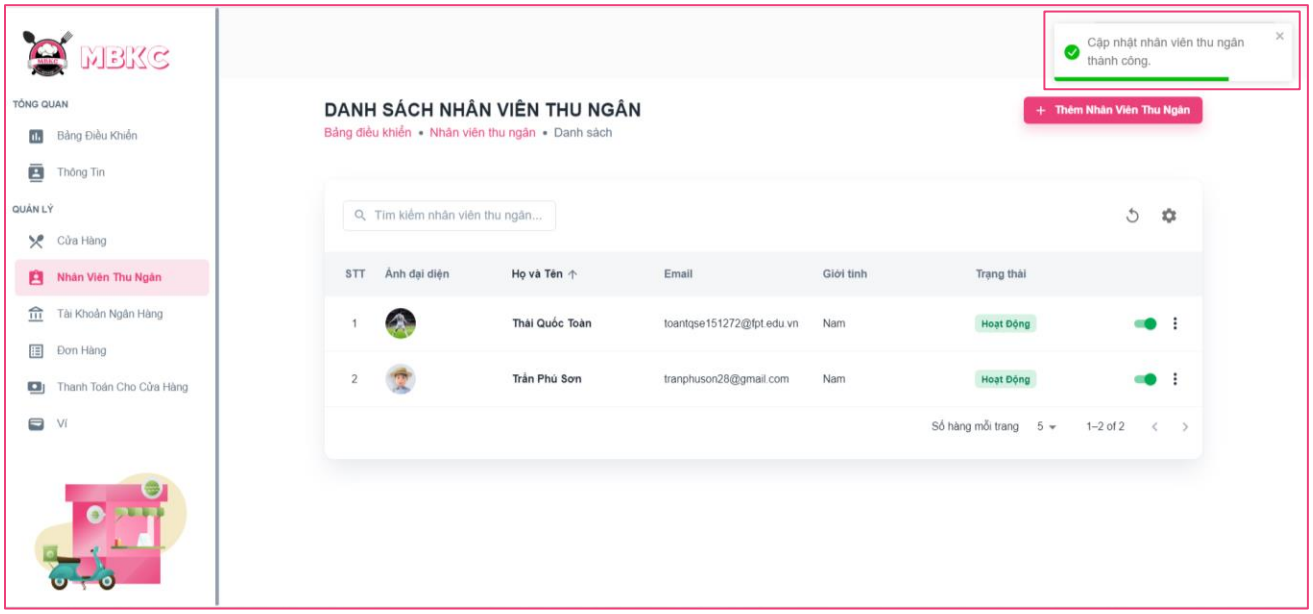


Figure 521: User Manual - Update Information Cashier – Step 7 – Case 2

3.2.11.6. Delete Cashier

3.2.11.6.1. Description
Use this guideline to delete cashier.

This web application supports delete cashier with an account as a Kitchen Center Manager.

3.2.11.6.2. Guide Steps
Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

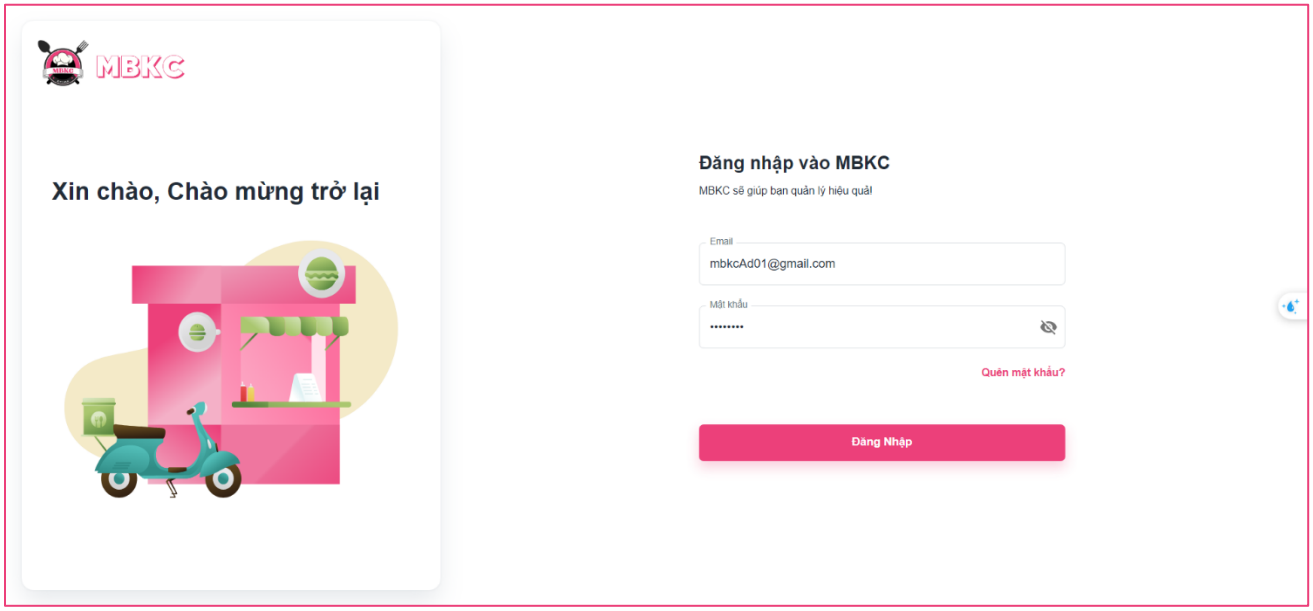


Figure 522: User Manual – Delete Cashier – Step 1

Step 2: Click “Đăng Nhập” button.

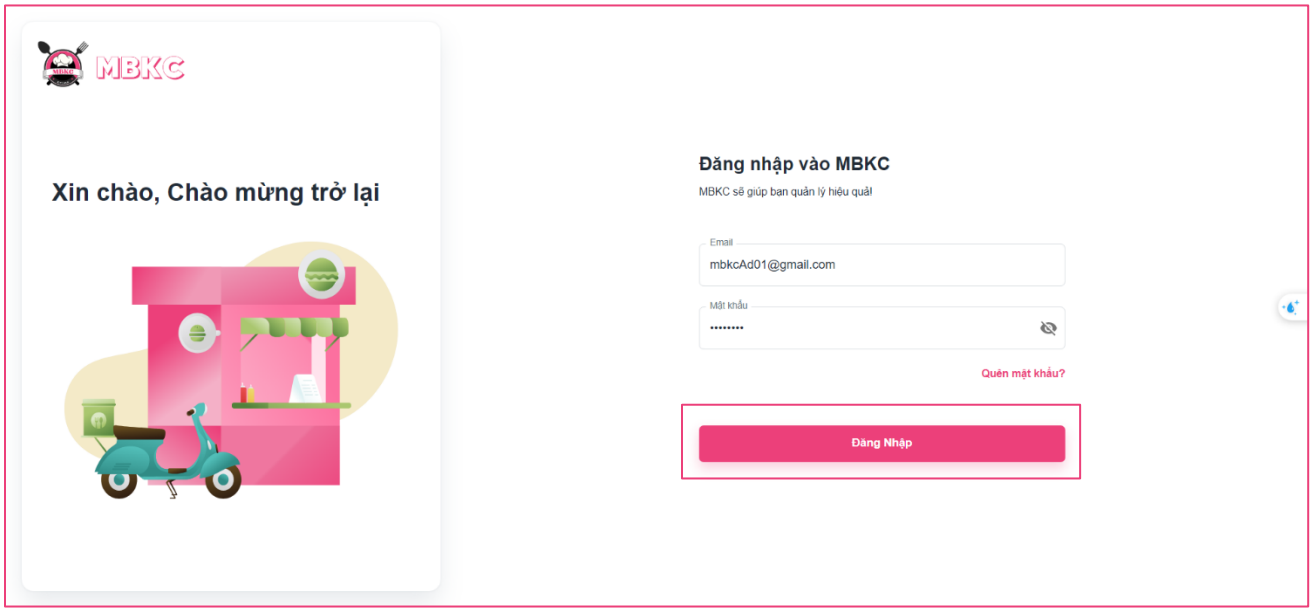


Figure 523: User Manual - Delete Cashier – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Nhân Viên Thu Ngân” from the left sidebar.

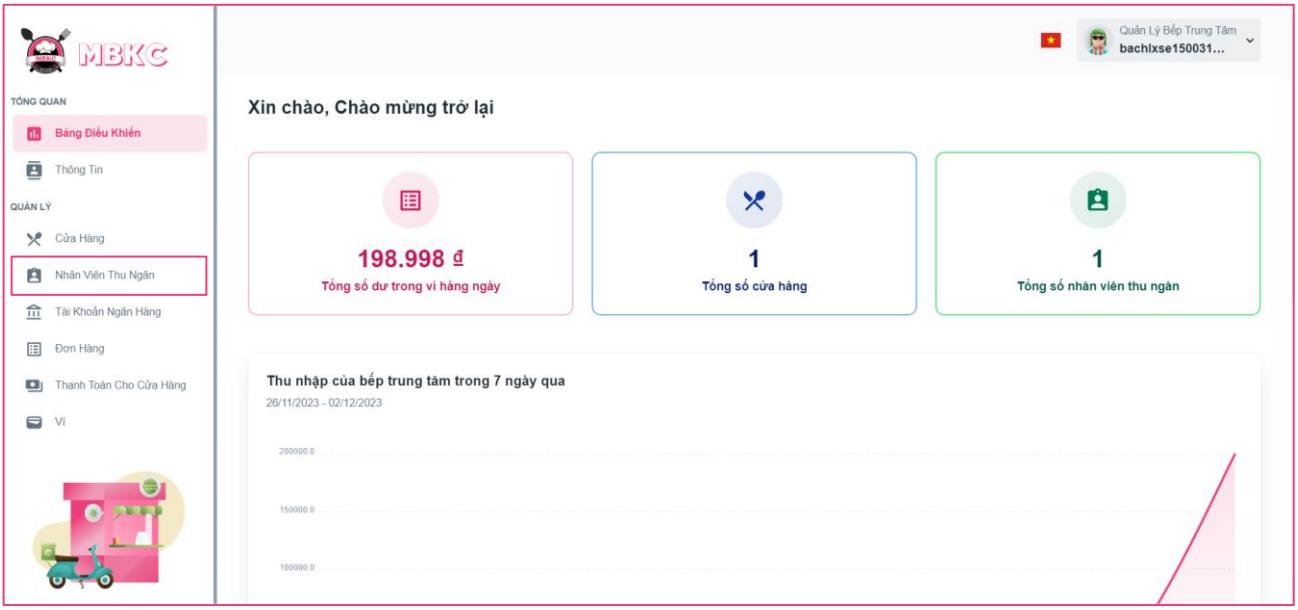


Figure 524: User Manual - Delete Cashier – Step 3

Step 4: Delete cashier.

Case 1: Delete cashier in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the cashier row to delete.

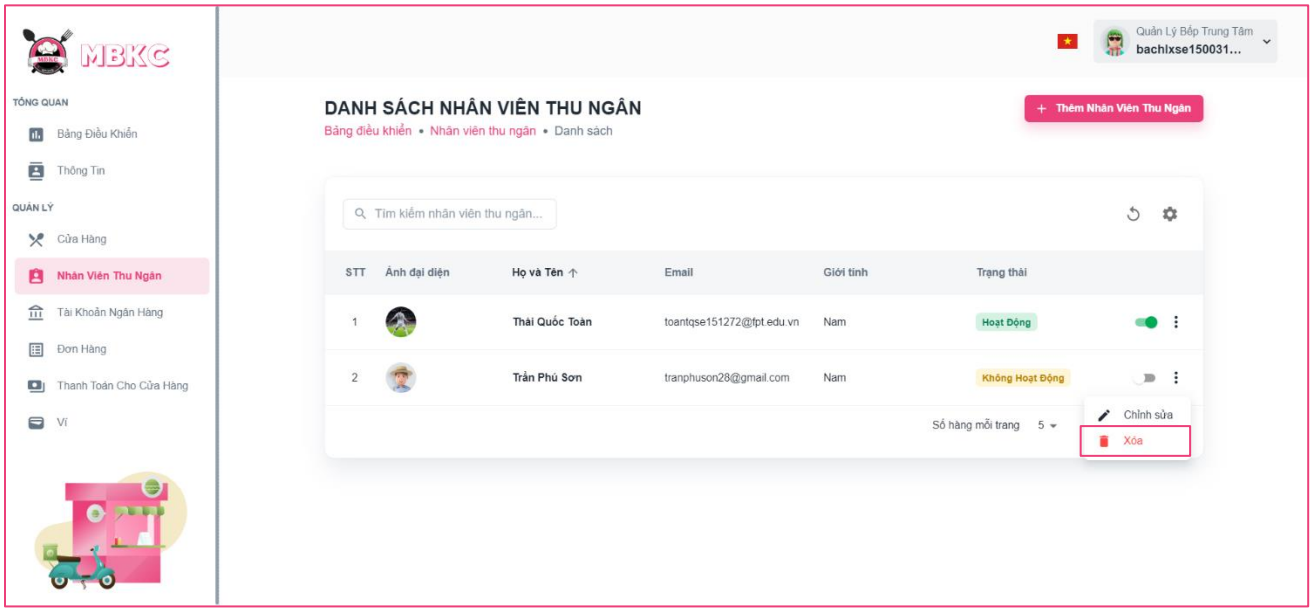


Figure 525: User Manual - Delete Cashier – Step 4 – Case 1

Case 2: Delete cashier on the details page.

1: Click the cashier row to navigate to Cashier Details Page.

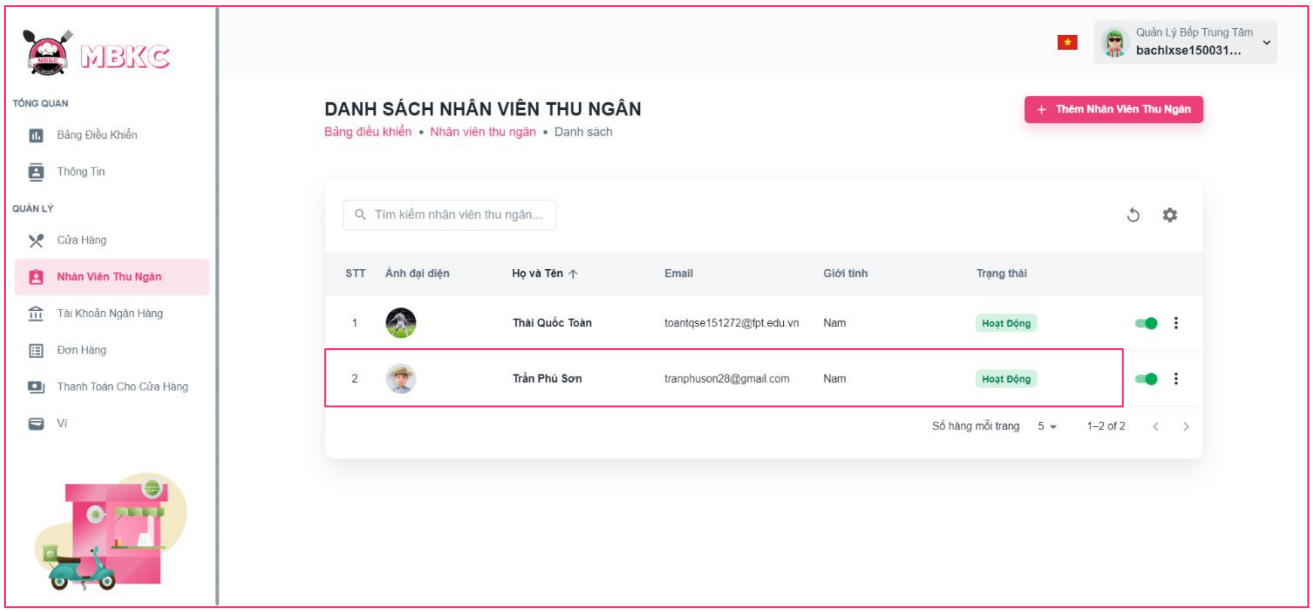


Figure 526: User Manual - Delete Cashier – Step 4 – Case 2 – 1

2: Click on “Thao Tác” button. Then click “Xóa” menu item to delete.

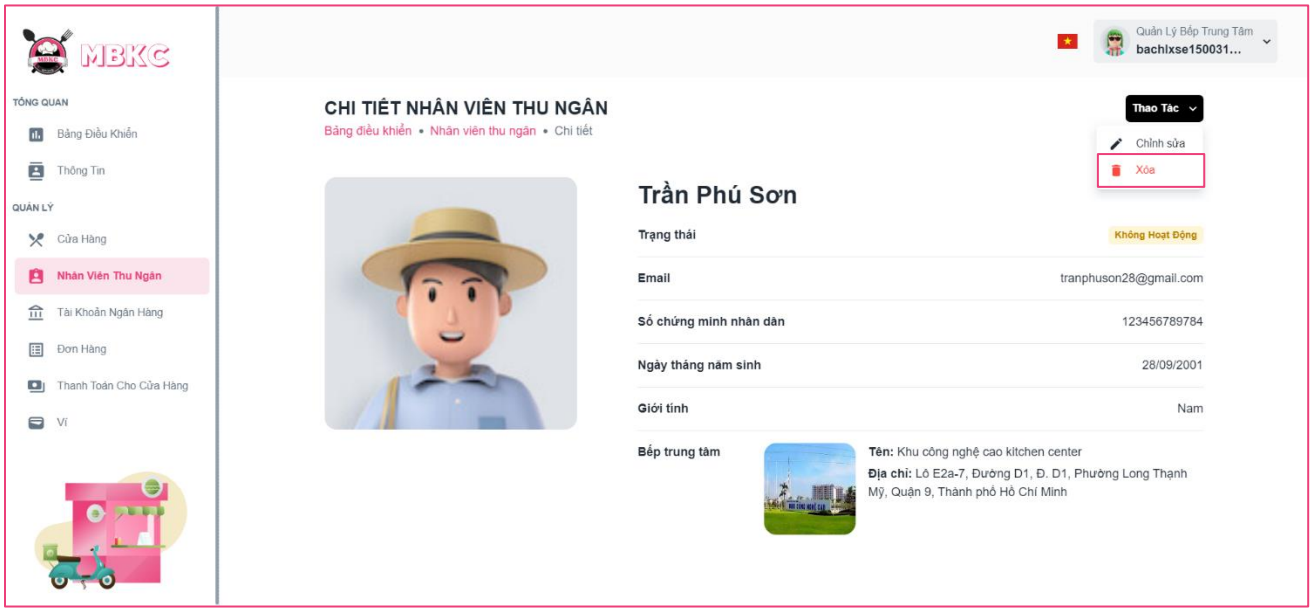


Figure 527: User Manual - Delete Cashier – Step 4 – Case 2 - 2

Step 5: Click “Xác Nhận” button.

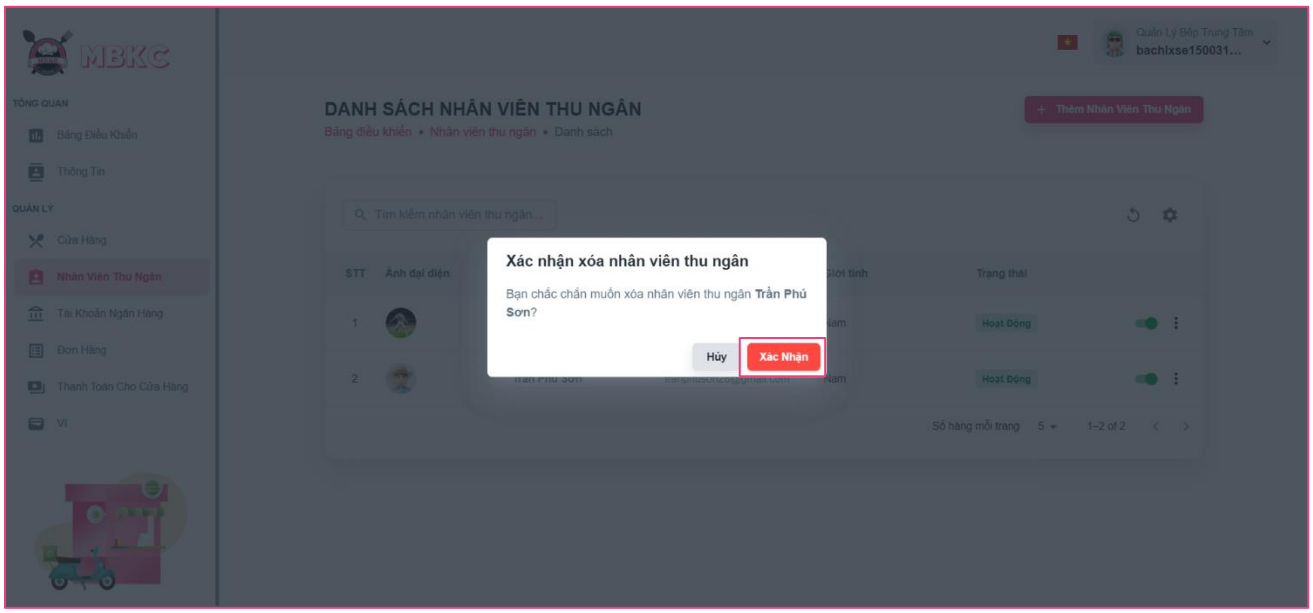


Figure 528: User Manual - Delete Cashier – Step 5

Step 6: If you in Cashier Details Page, you will navigate to List of Cashiers Page. List of cashiers will be reloaded and a toast message “Đã xóa nhân viên thu ngân thành công”. And cashier has been deleted.

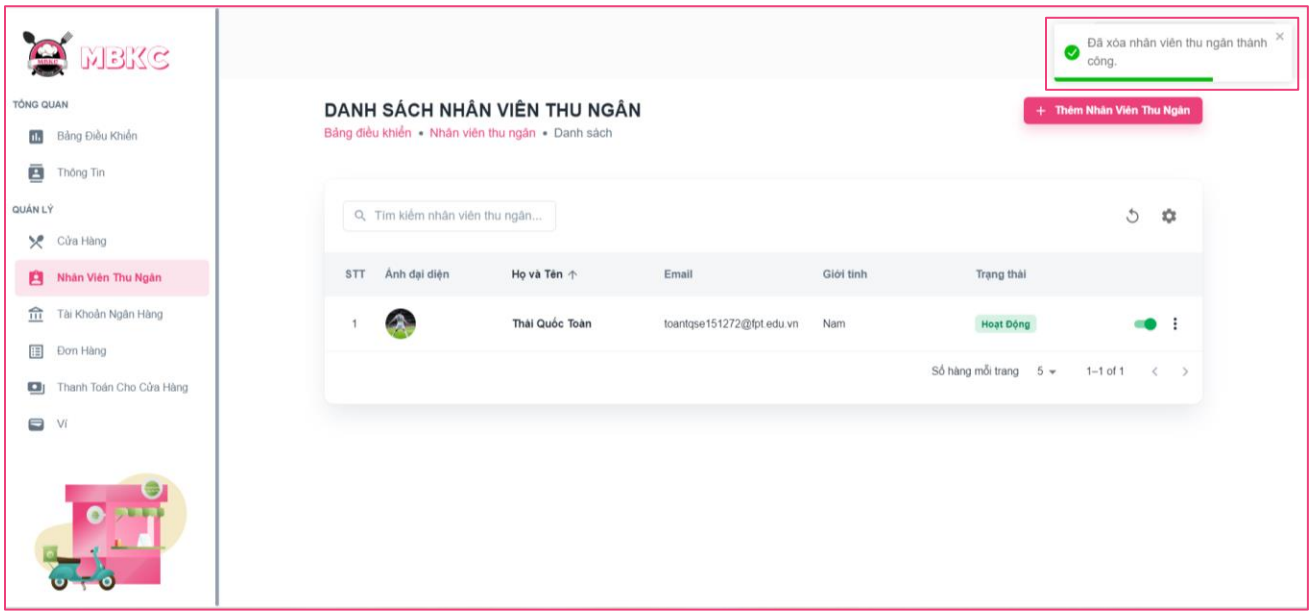


Figure 529: User Manual – Delete Cashier – Step 6

3.2.12. Banking Account management

3.2.12.1. Create new Banking Account

3.2.12.1.1. Description

Use this guideline to login into MBKC account to create a new banking account.

This web application supports creating a new banking account with an account as a Kitchen Center Manager.

3.2.12.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

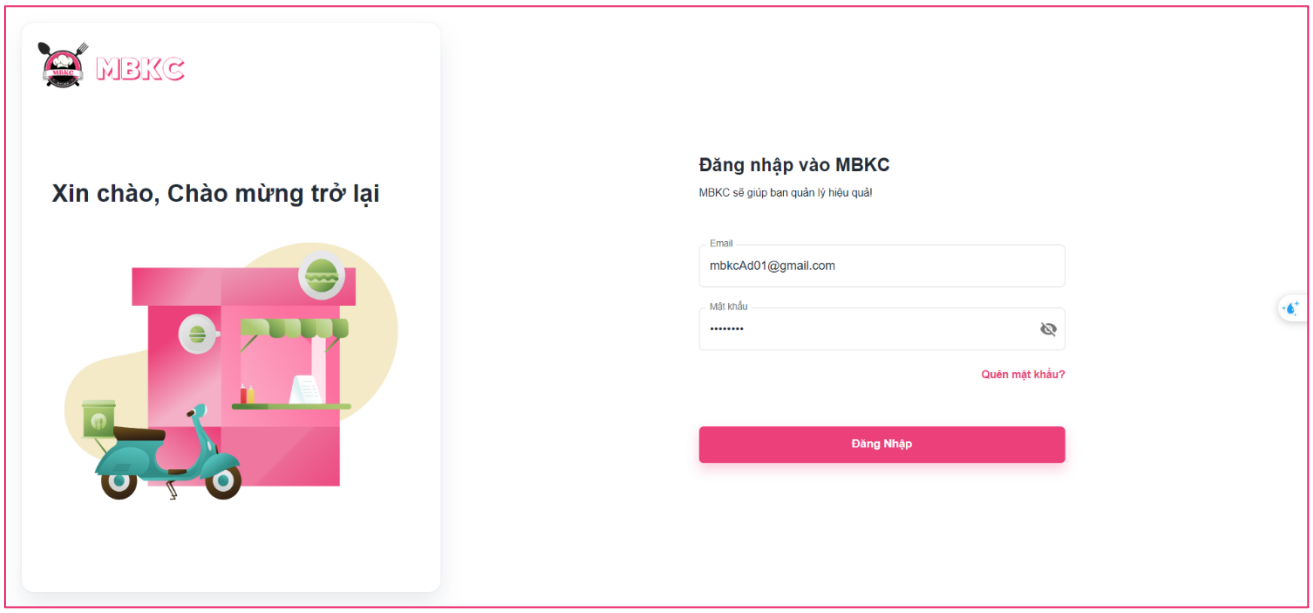


Figure 530: User Manual - Create new Brand – Step 1

Step 2: Click “Đăng Nhập” button.

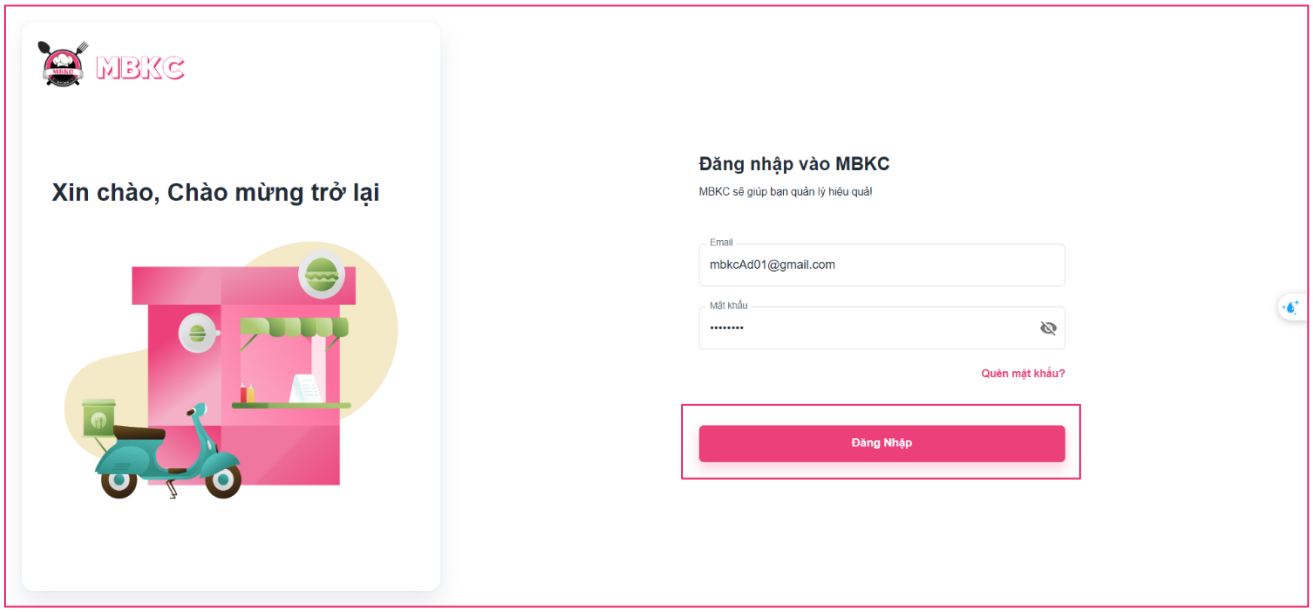


Figure 531: User Manual - Create new Banking Account – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Tài Khoản Ngân Hàng” from the left sidebar.

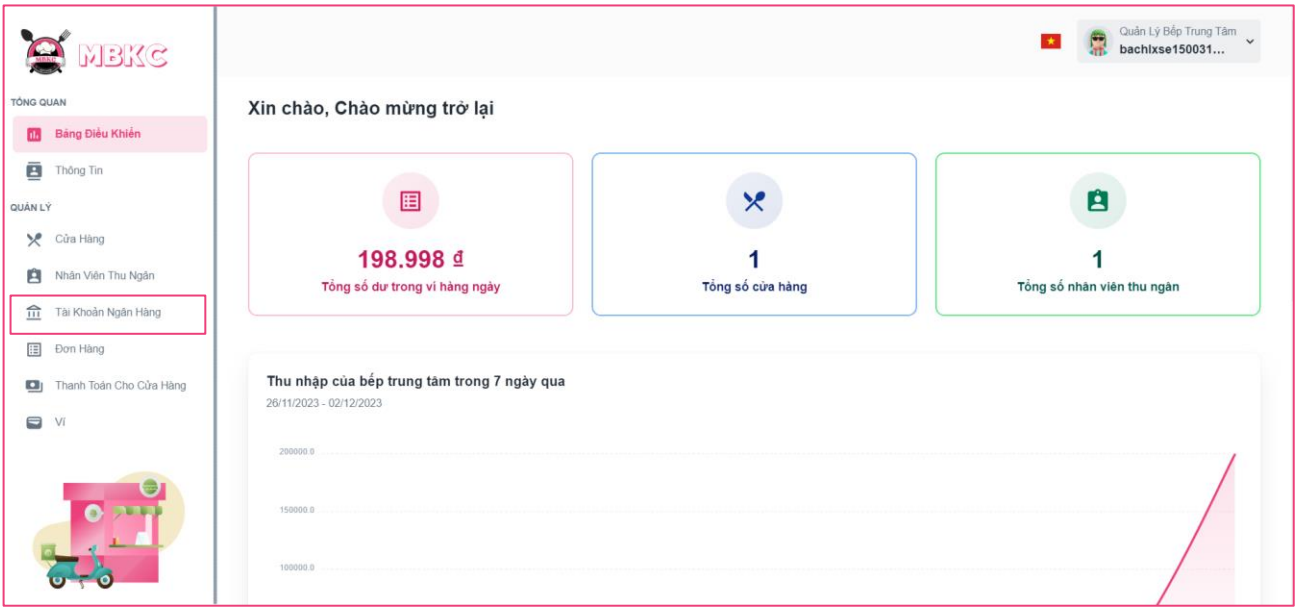


Figure 532: User Manual - Create new Banking Account – Step 3

Step 4: Click “Thêm Nhân Viên Thu Ngân” button on the screen and pop up Create New Banking Account Modal to add a new bacnking account.

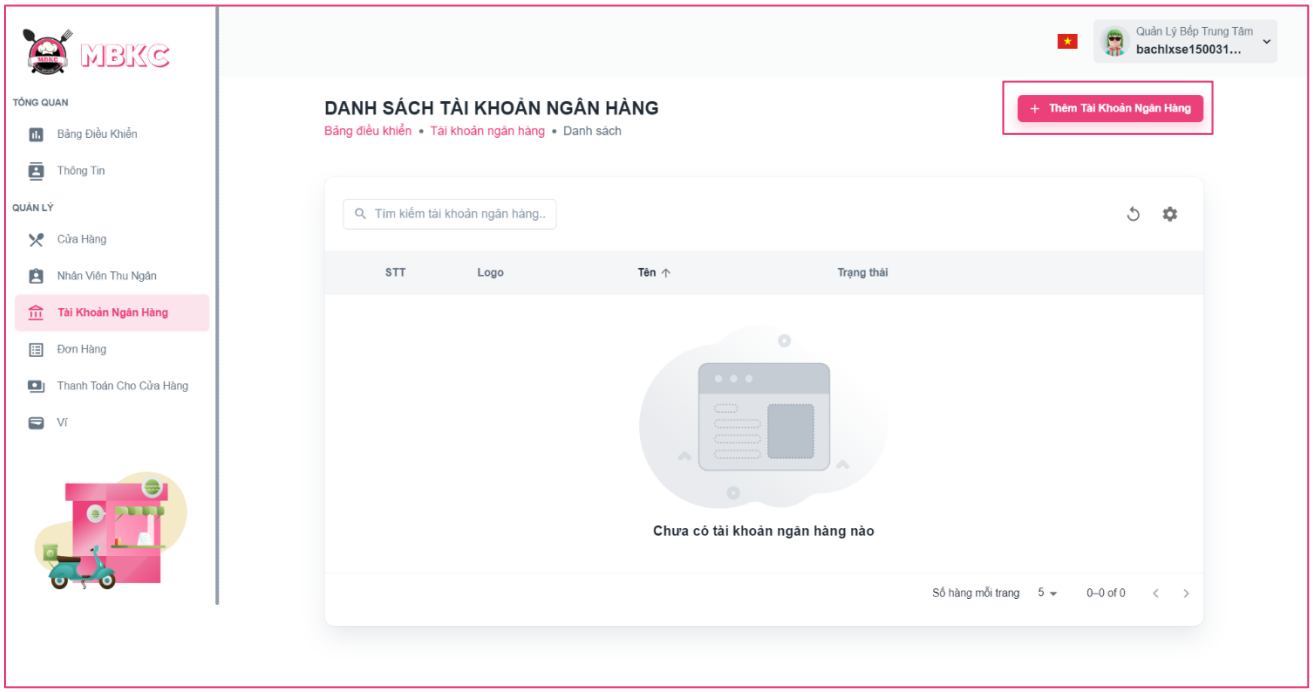


Figure 533: User Manual - Create new Banking Account – Step 4

Step 5: Filling all the required information to add a banking account including: “Hình ảnh”, “Tên tài khoản ngân hàng", and “Số tài khoản".

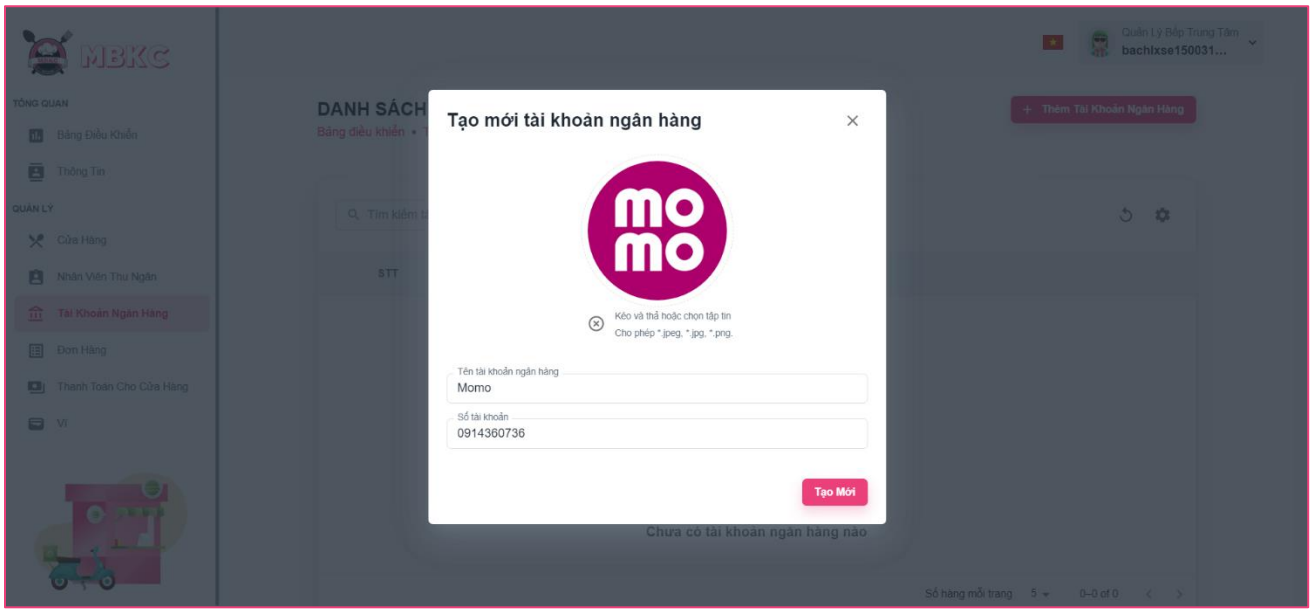


Figure 534: User Manual - Create new Banking Account – Step 5

Step 6: After filling in all required field, click “Tạo Mới” to finish creating a new banking account.

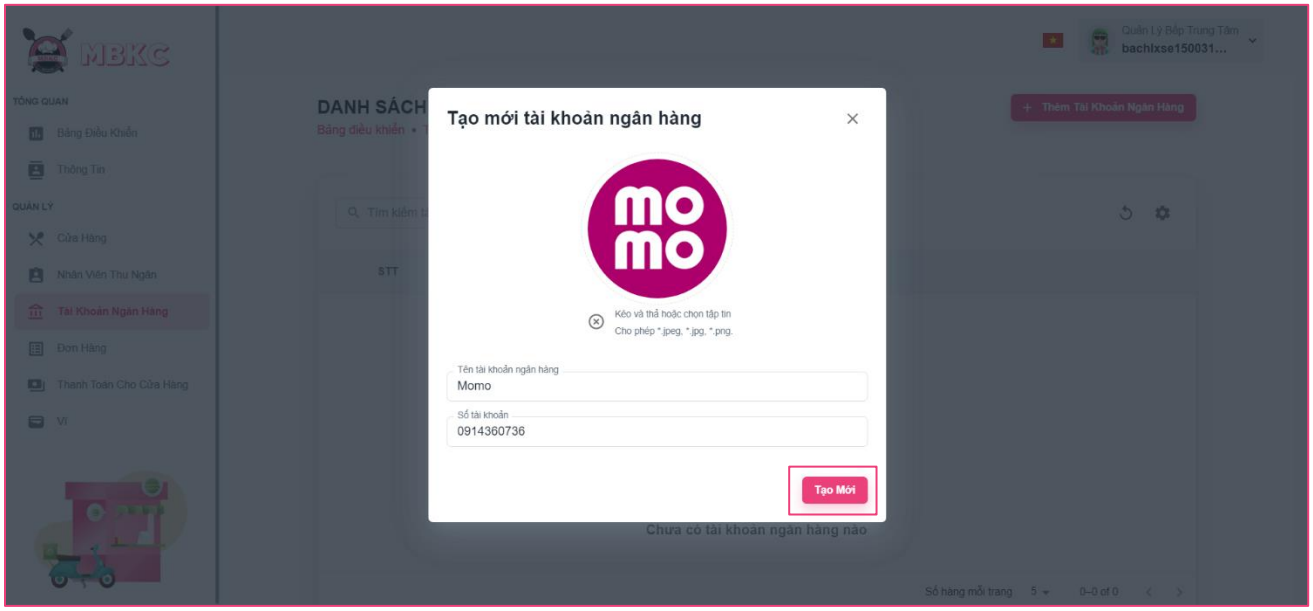


Figure 535: User Manual - Create new Banking Account – Step 6

Step 7: List of Banking Accounts Page will be reloaded to show the list of banking accounts and a toast message “Tạo mới tài khoản ngân hàng thành công”.

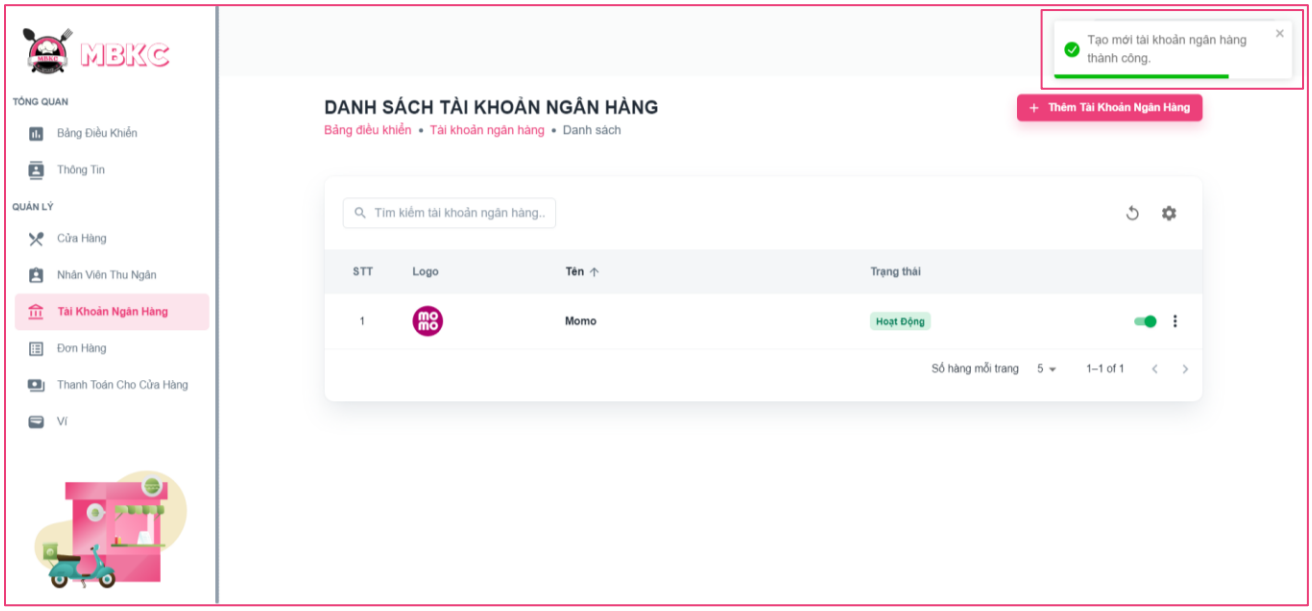


Figure 536: User Manual – Create new Banking Account – Step 7

3.2.12.2. View list of Banking Accounts

3.2.12.2.1. Description

Use this guideline to search, reload data, and show or hide columns in the table list of banking accounts.

This web application supports view list of banking accounts with an account as a Kitchen Center Manager.

3.2.12.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

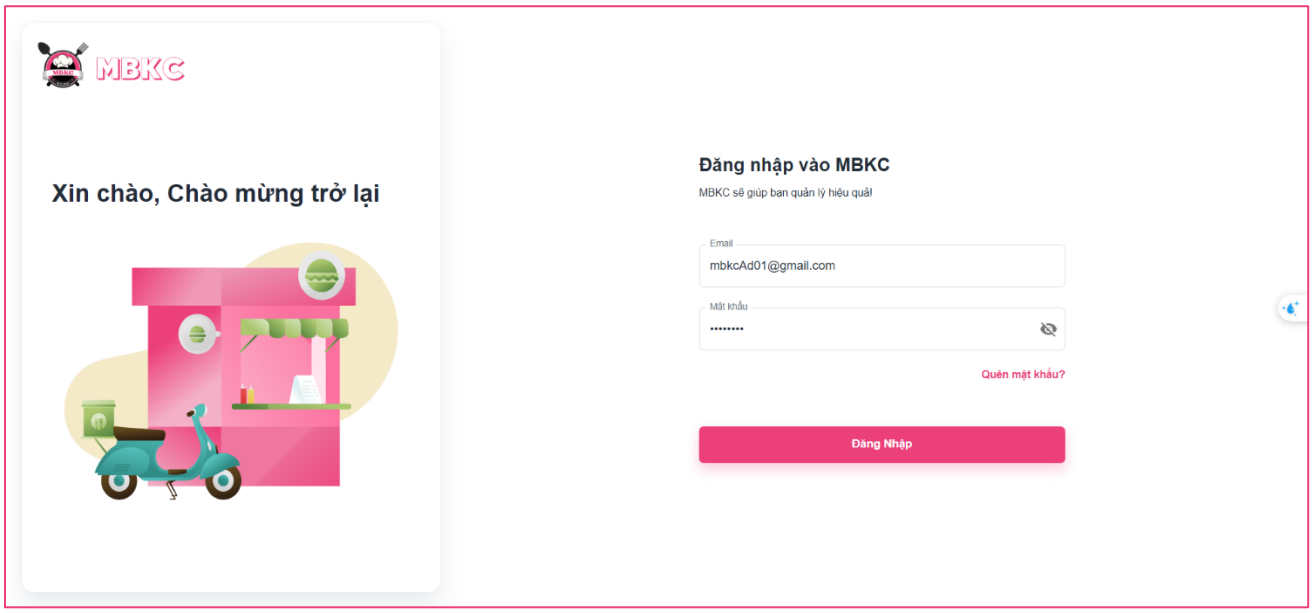


Figure 537: User Manual - View list of Banking Accounts – Step 1

Step 2: Click “Đăng Nhập” button.

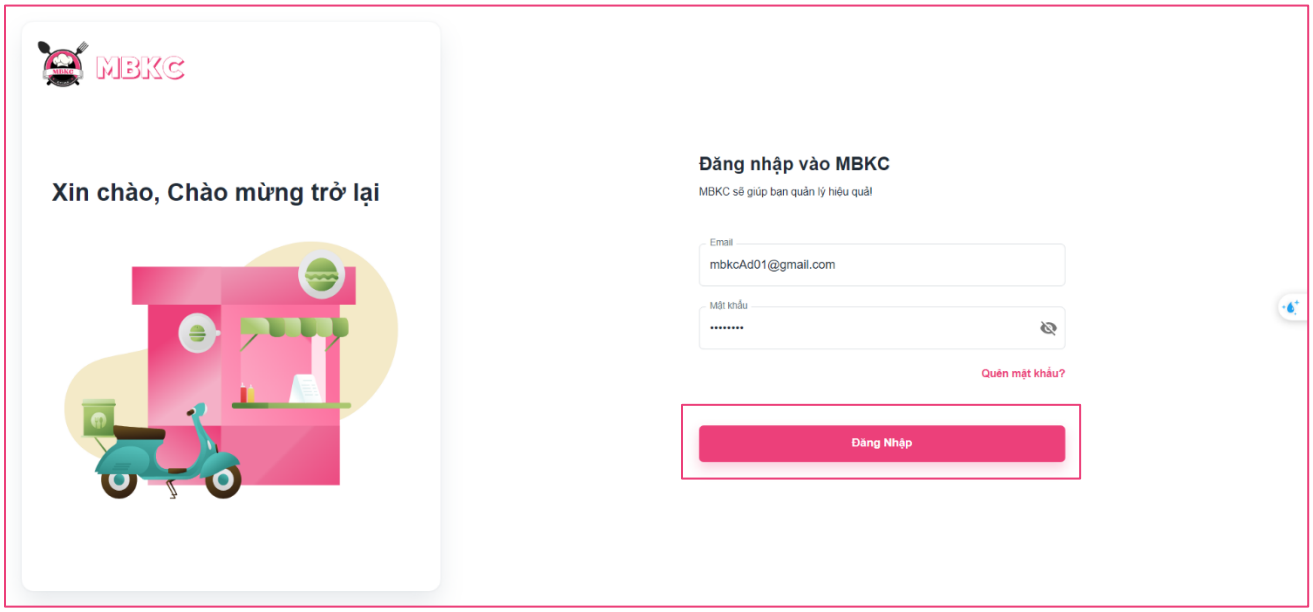


Figure 538: User Manual - View list of Banking Accounts – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Tài Khoản Ngân Hàng” from the left sidebar.

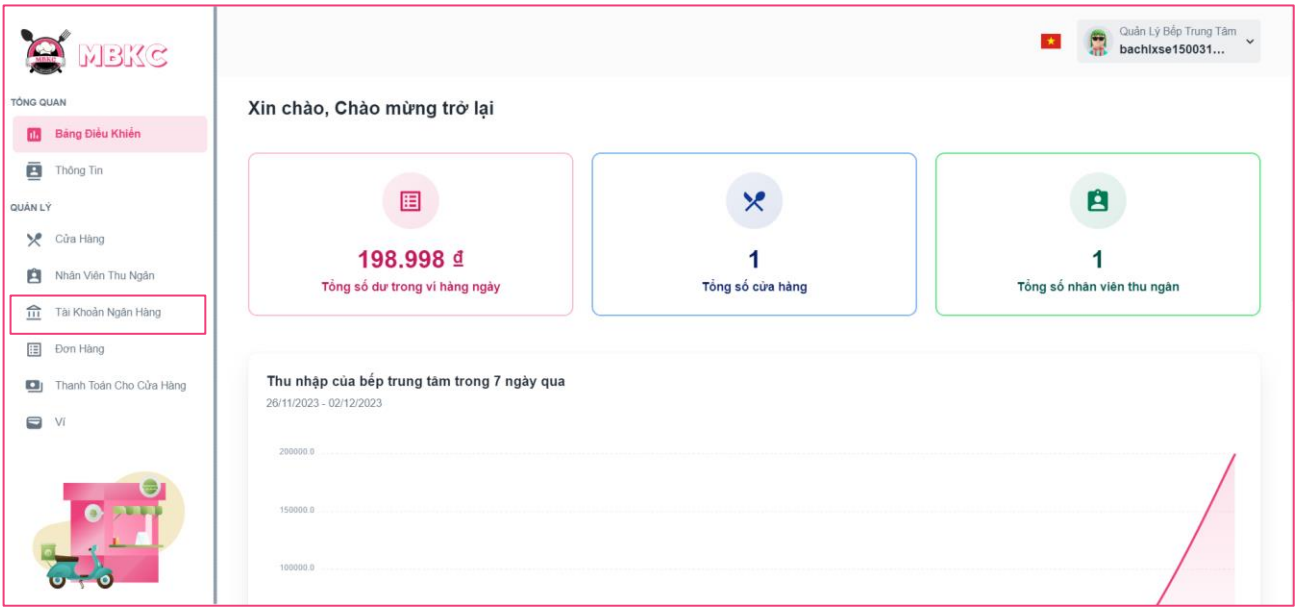


Figure 539: User Manual - View list of Banking Accounts – Step 3

Step 4: Search, reload data, and show or hide columns in the table list of banking accounts.

Case 1: Fill in the banking account’s name in the search box to search banking accounts in the table list.

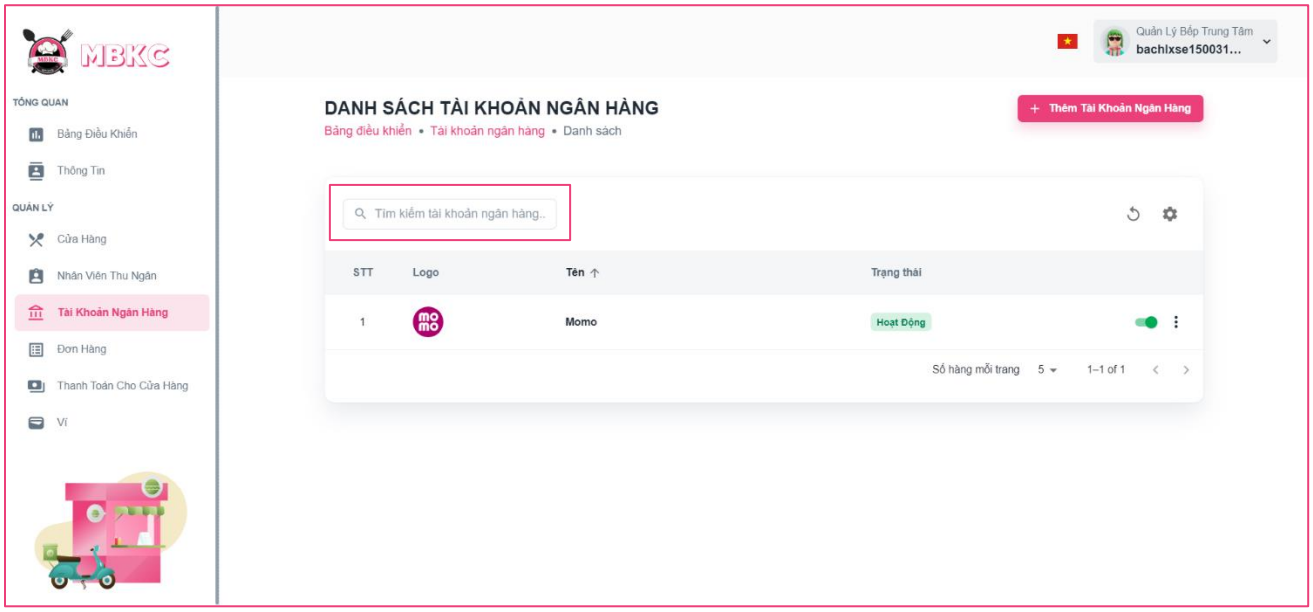


Figure 540: User Manual - View list of Banking Accounts – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of banking accounts.

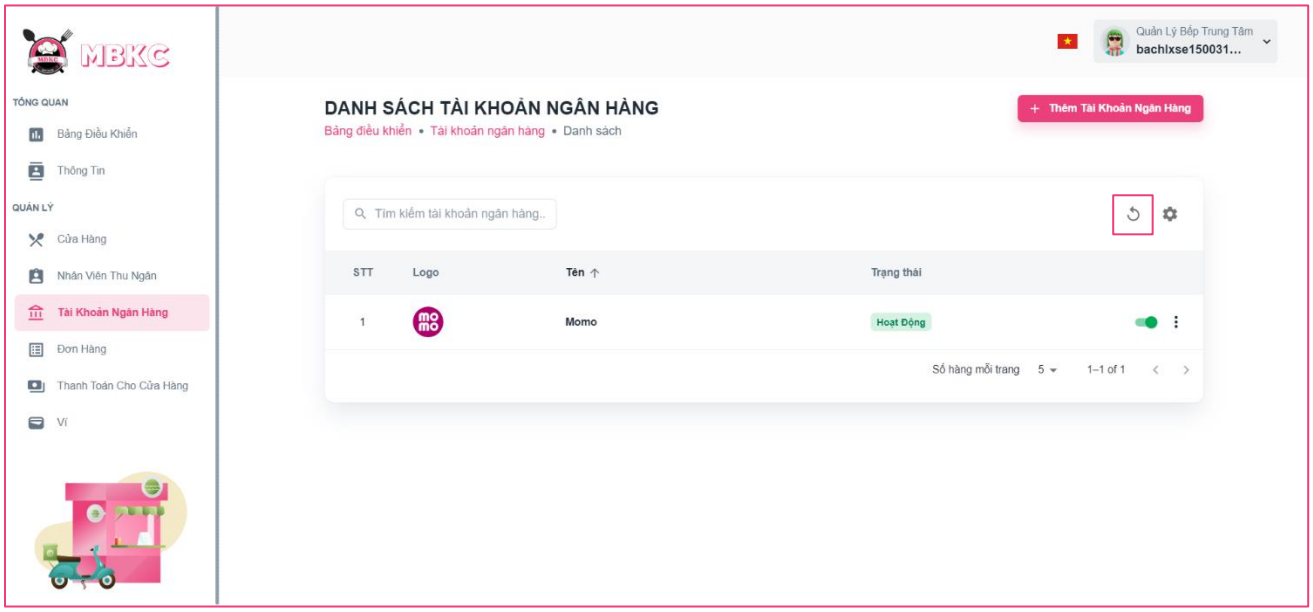


Figure 541: User Manual - View list of Banking Accounts – Step 4 – Case 2

Case 3: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of banking accounts.

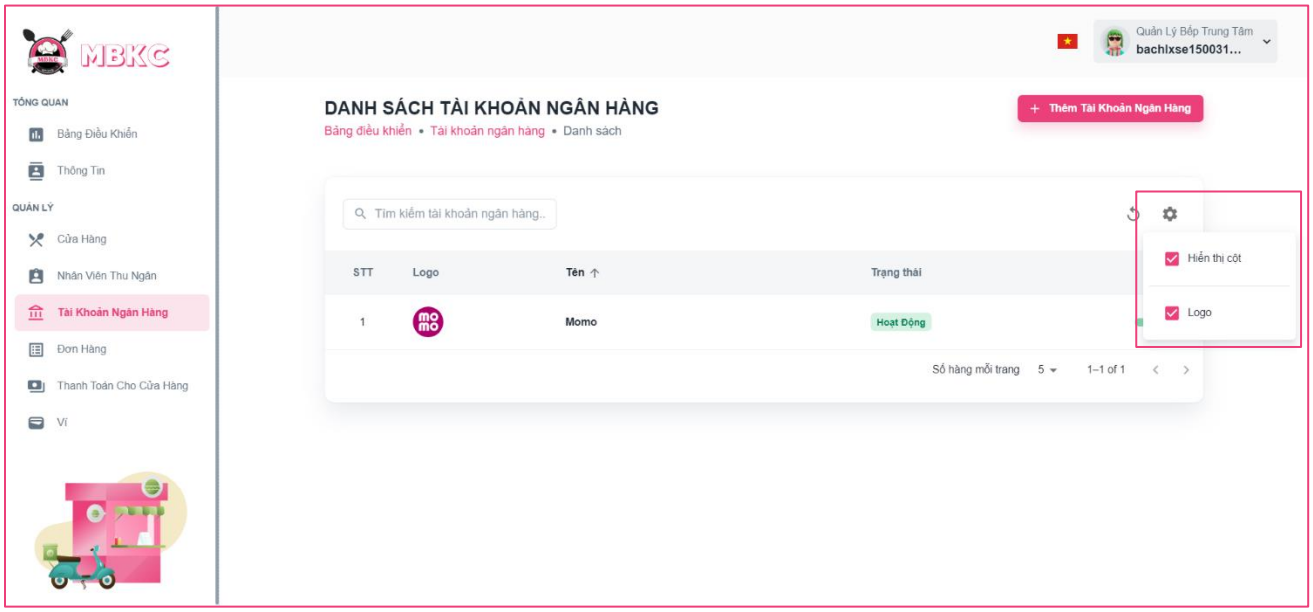


Figure 542: User Manual - View list of Banking Accounts – Step 4 – Case 3

3.2.12.3. View details Banking Account

3.2.12.3.1. Description

Use this guideline to view details banking account.

This web application supports view details banking account with an account as a Kitchen Center Manager.

3.2.12.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

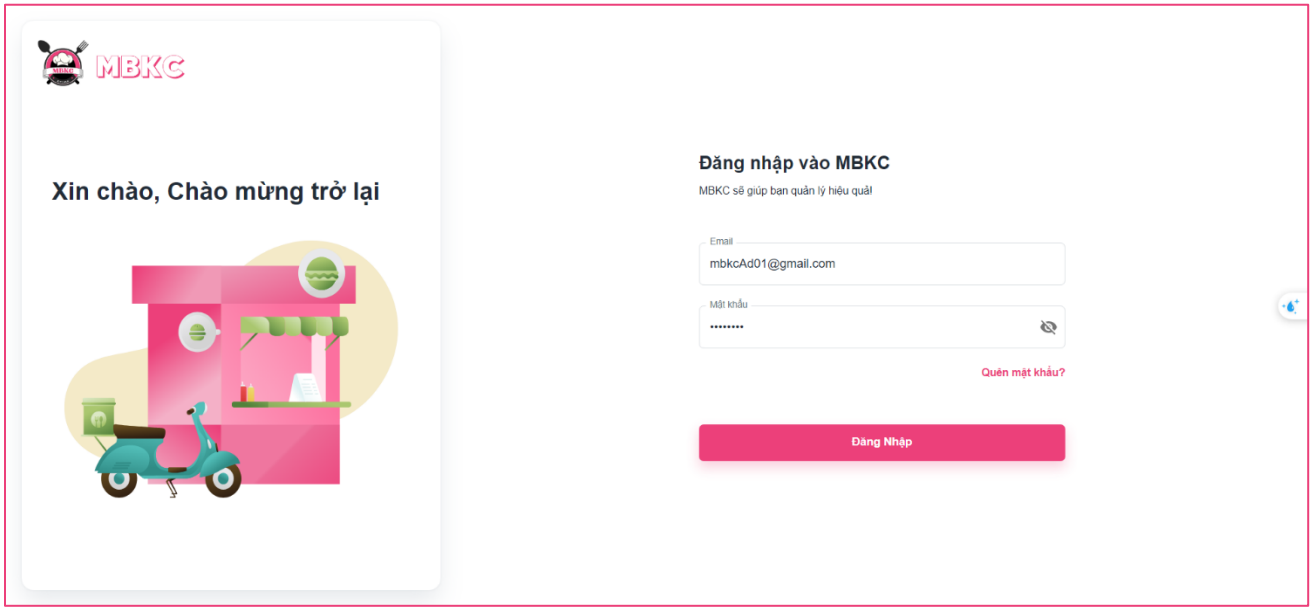


Figure 543: User Manual - View details Banking Account – Step 1

Step 2: Click “Đăng Nhập” button.

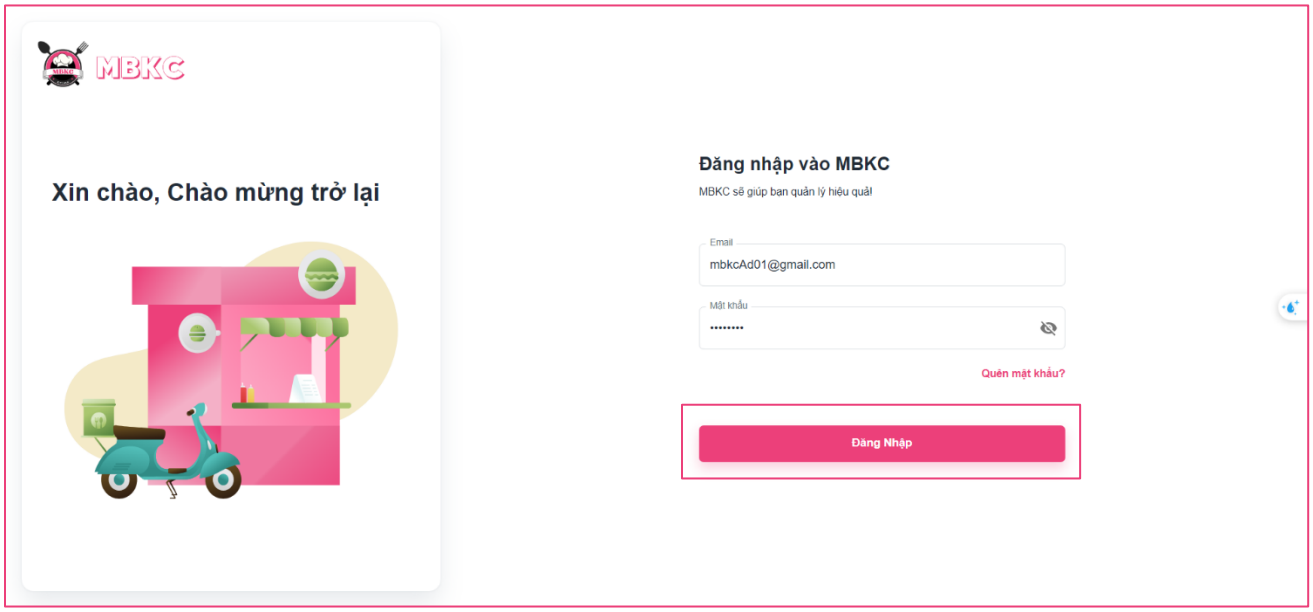


Figure 544: User Manual - View details Banking Account – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Tài Khoản Ngân Hàng” from the left sidebar.

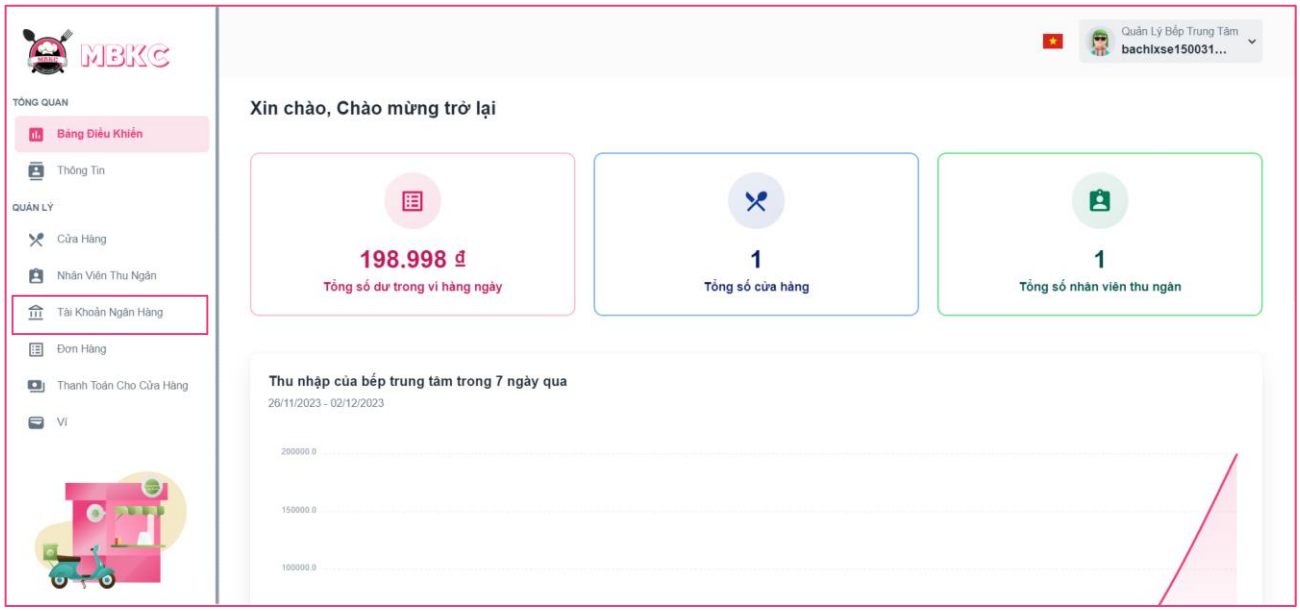


Figure 545: User Manual - View details Banking Account – Step 3

Step 4: Click the banking account row to pop up Banking Account Details Modal.

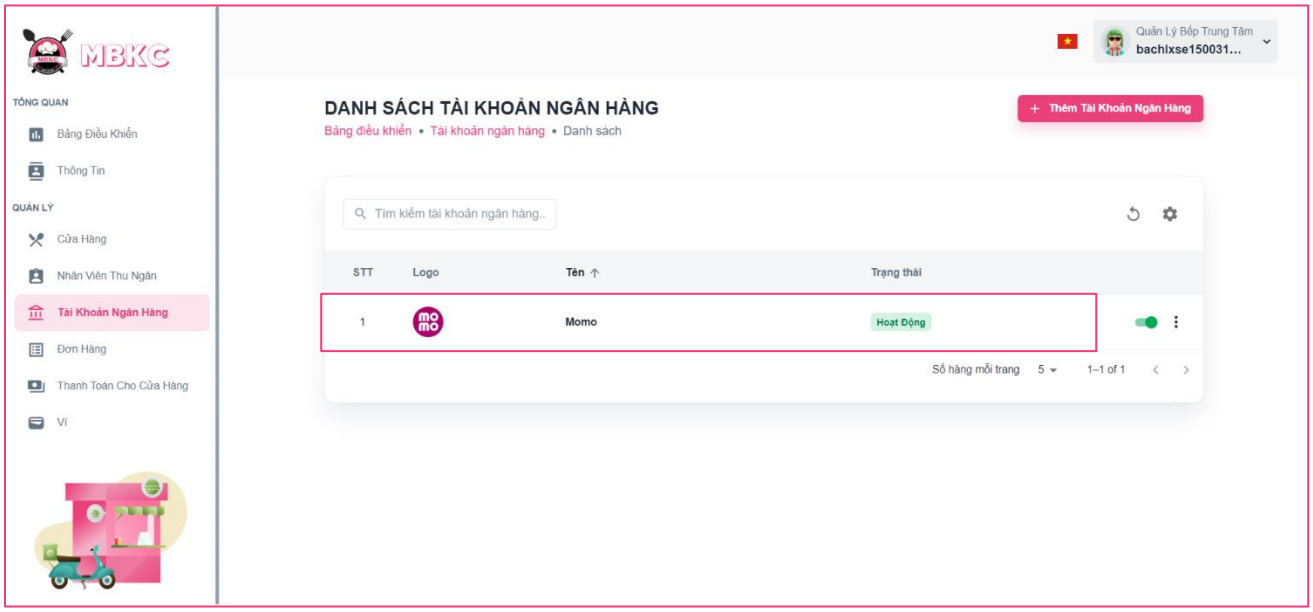


Figure 546: User Manual - View details Banking Account – Step 4

Step 5: Banking Account Details Modal will be showed to show information details of banking account.

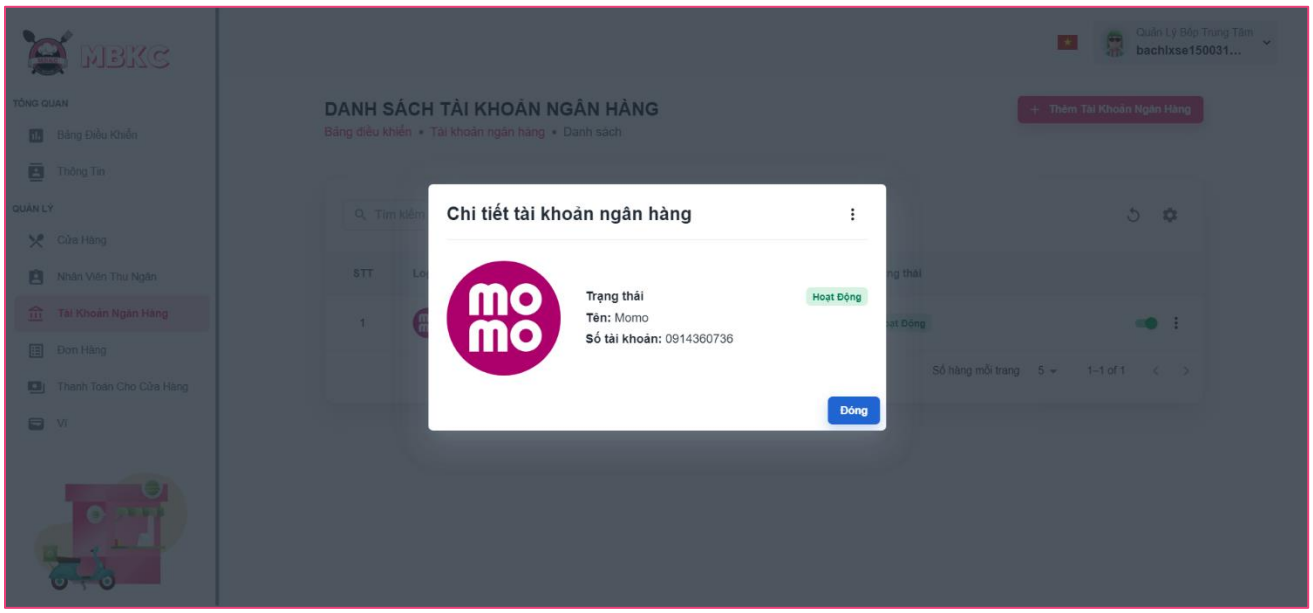


Figure 547: User Manual - View details Banking Account – Step 5

3.2.12.4. Update Status Banking Account

3.2.12.4.1. Description

Use this guideline to update the status of the banking account.

This web application supports update the status of the banking account with an account as a Kitchen Center Manager.

3.2.12.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

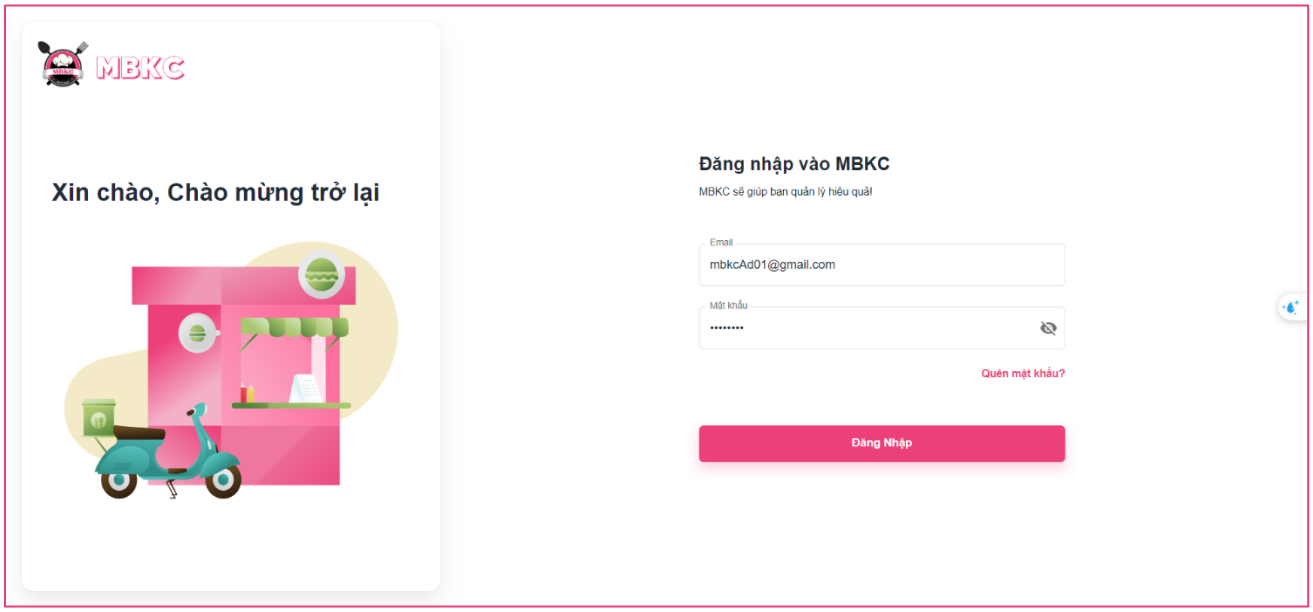


Figure 548: User Manual – Update Status Banking Account – Step 1

Step 2: Click “Đăng Nhập” button.

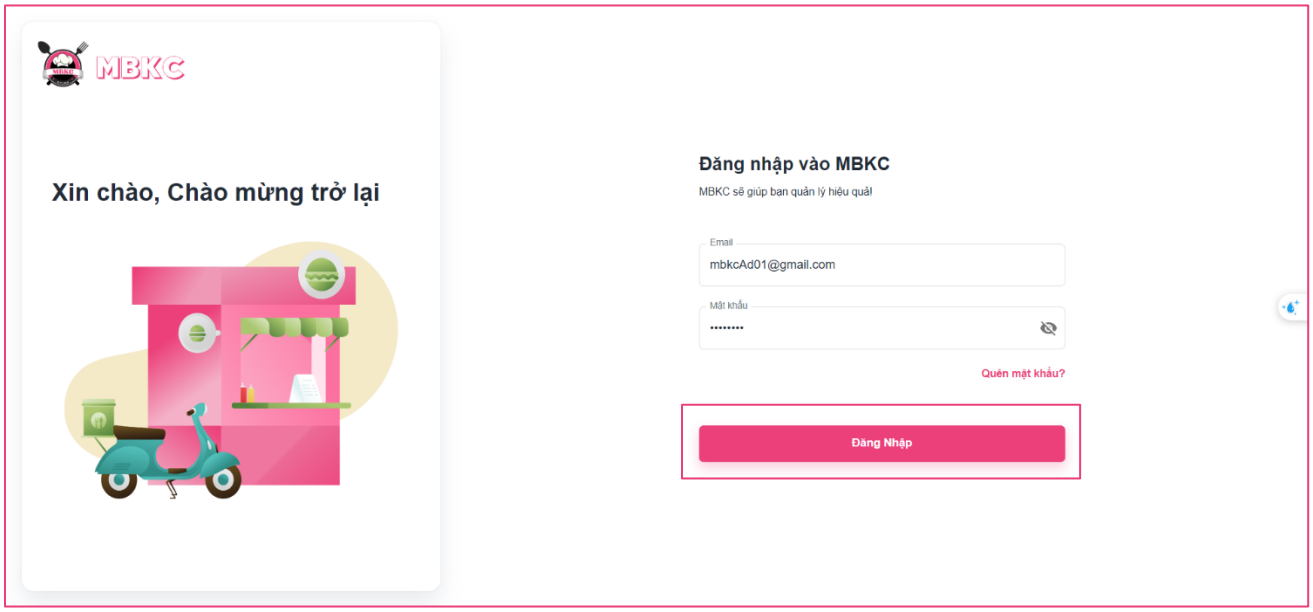


Figure 549: User Manual - Update Status Banking Account – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Tài Khoản Ngân Hàng” from the left sidebar.

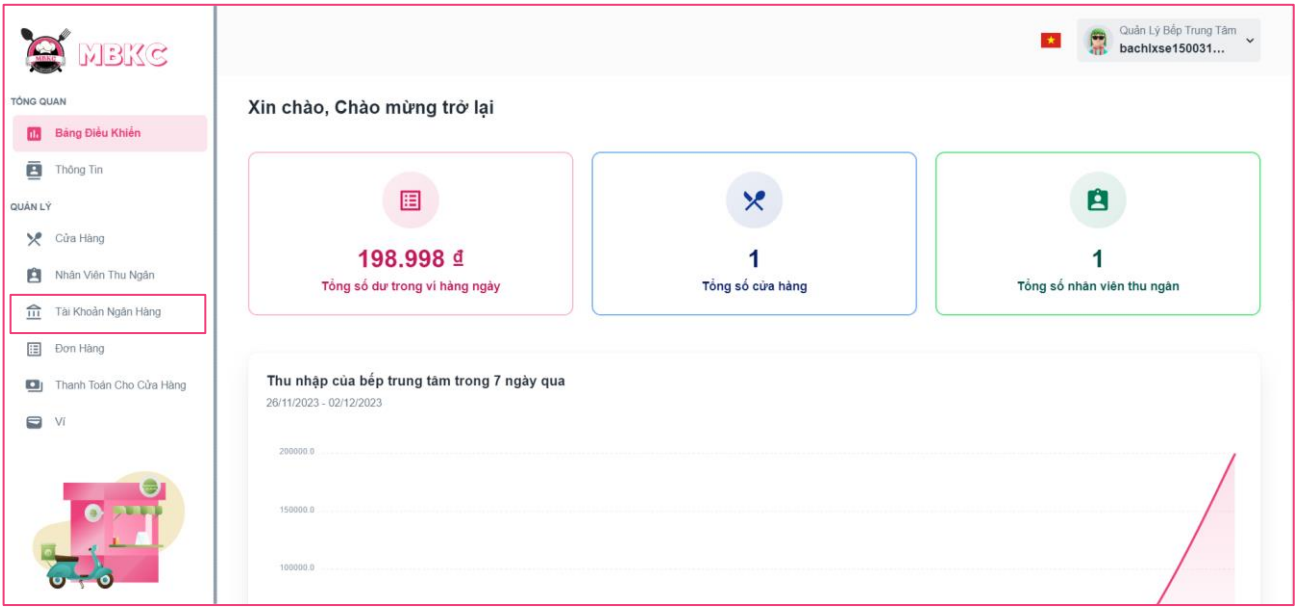


Figure …: User Manual - Update Status Banking Account – Step 3

Step 4: Click on the switch button on the banking account row to update the status.

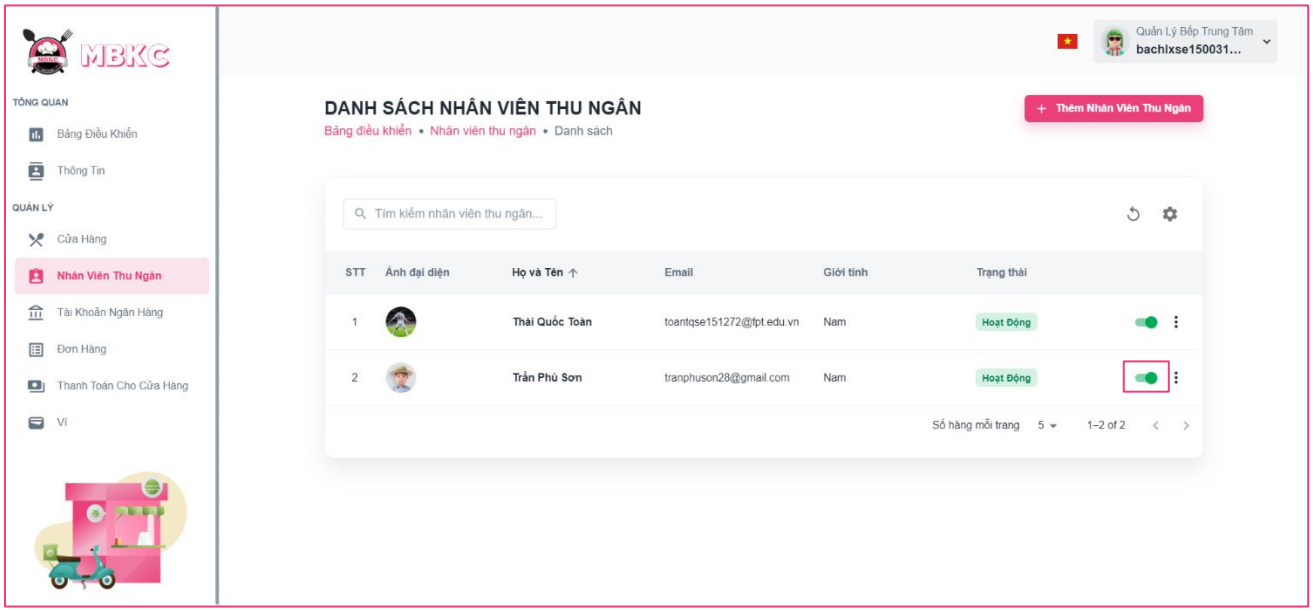


Figure 550: User Manual - Update Status Banking Account – Step 4

Step 5: List of banking accounts will be reloaded and a toast message "Cập nhật trạng thái tài khoản ngân hàng thành công". And the status of the banking account has been updated.

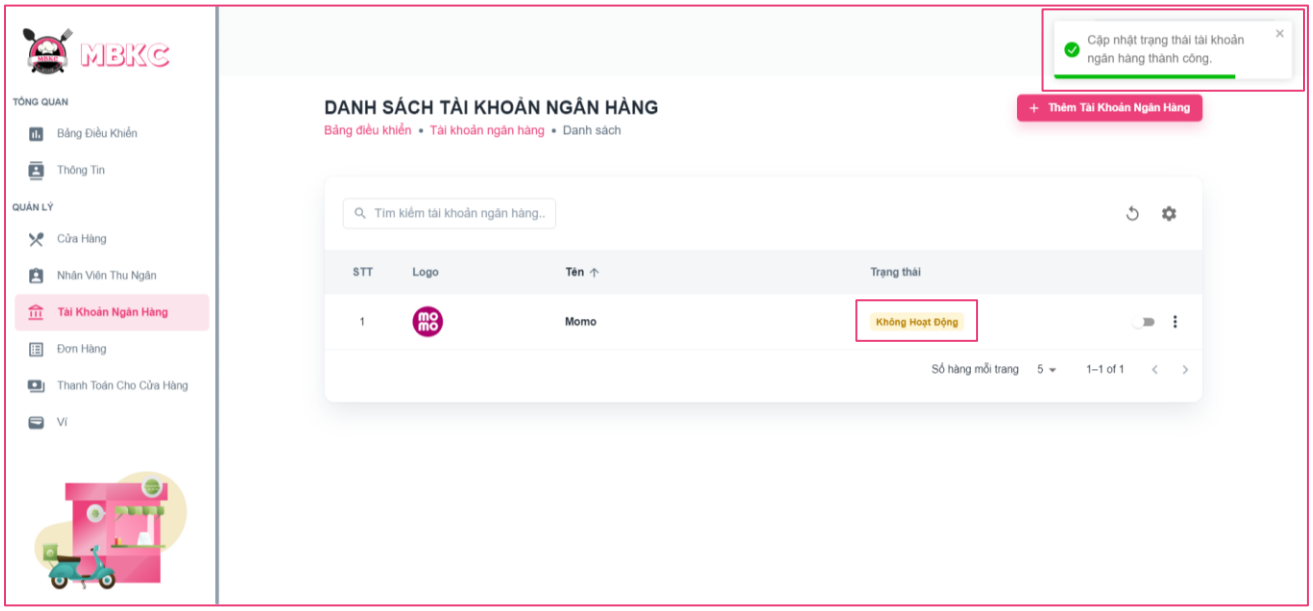


Figure 551: User Manual - Update Status Banking Account – Step 5

3.2.12.5. Update Information Banking Account

3.2.12.5.1. Description

Use this guideline to update information of the banking account.

This web application supports update information of the banking account with an account as a Kitchen Center Manager.

3.2.12.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

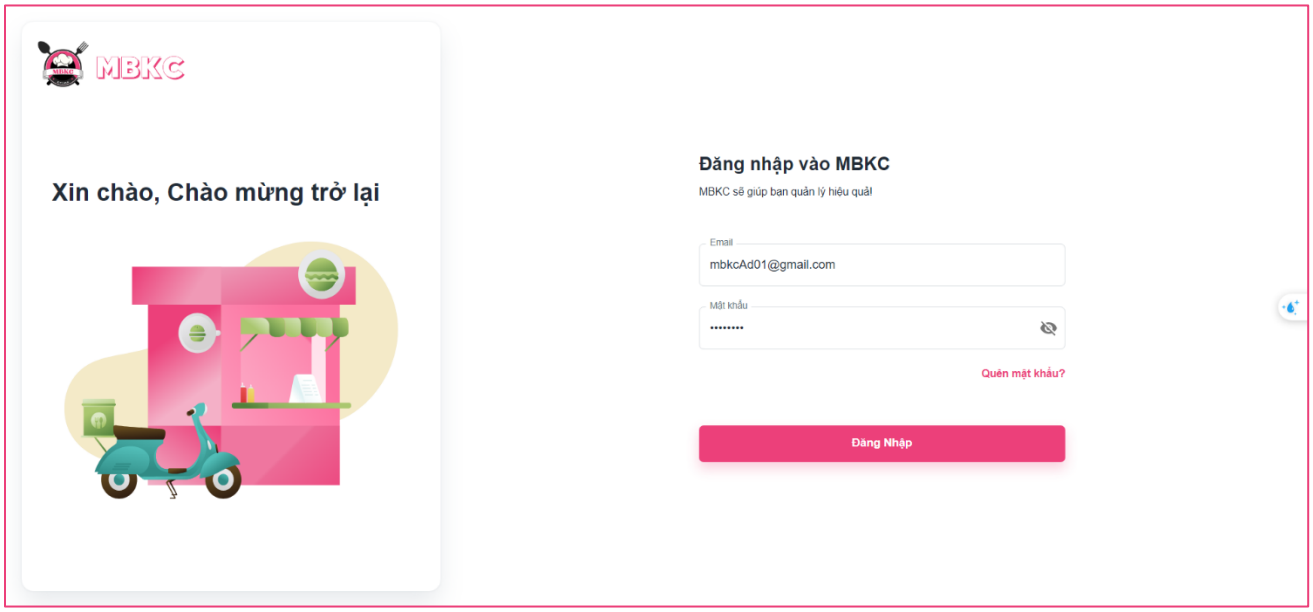


Figure 552: User Manual – Update Information Banking Account – Step 1

Step 2: Click “Đăng Nhập” button.

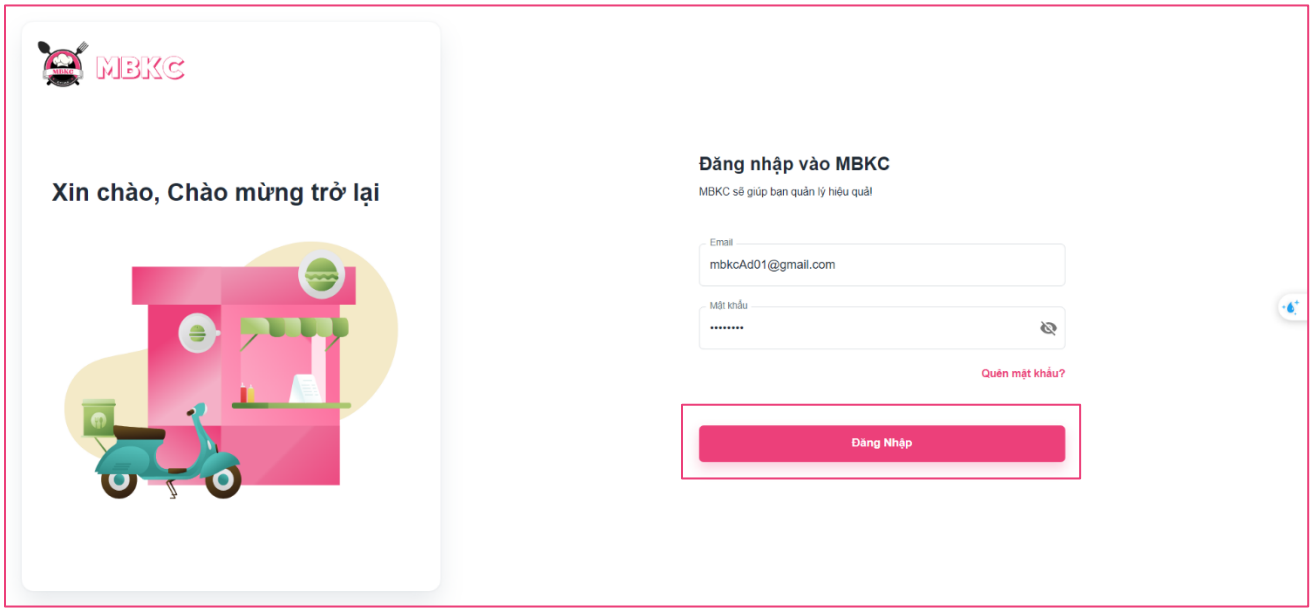


Figure 553: User Manual - Update Information Banking Account – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Tài Khoản Ngân Hàng” from the left sidebar.

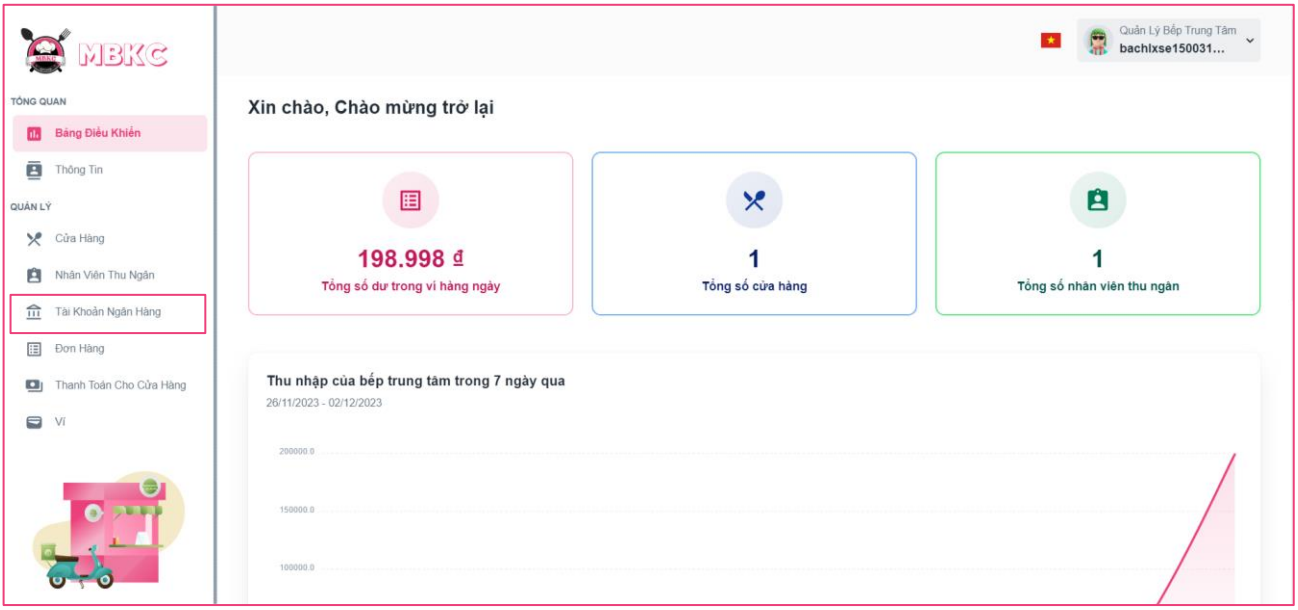


Figure 554: User Manual - Update Information Banking Account – Step 3

Step 4: Update information of the banking account.

Case 1: Update information of the banking account in table list.

Click on the Menu icon button. Then click “Chỉnh sửa” menu item on the banking account row want to pop up Update Banking Account Modal.

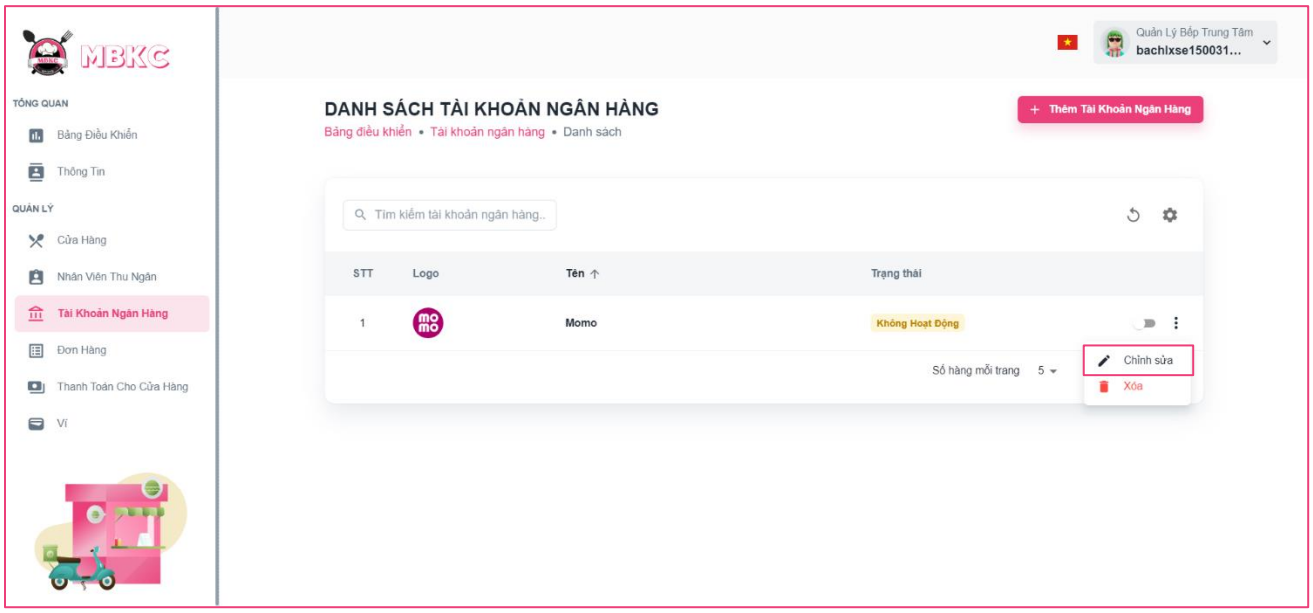


Figure 555: User Manual - Update Information Banking Account – Step 4 – Case 1

Case 2: Update information of the banking account on the detail modal.

1: Click the banking account row to pop up Banking Account Details Modal.

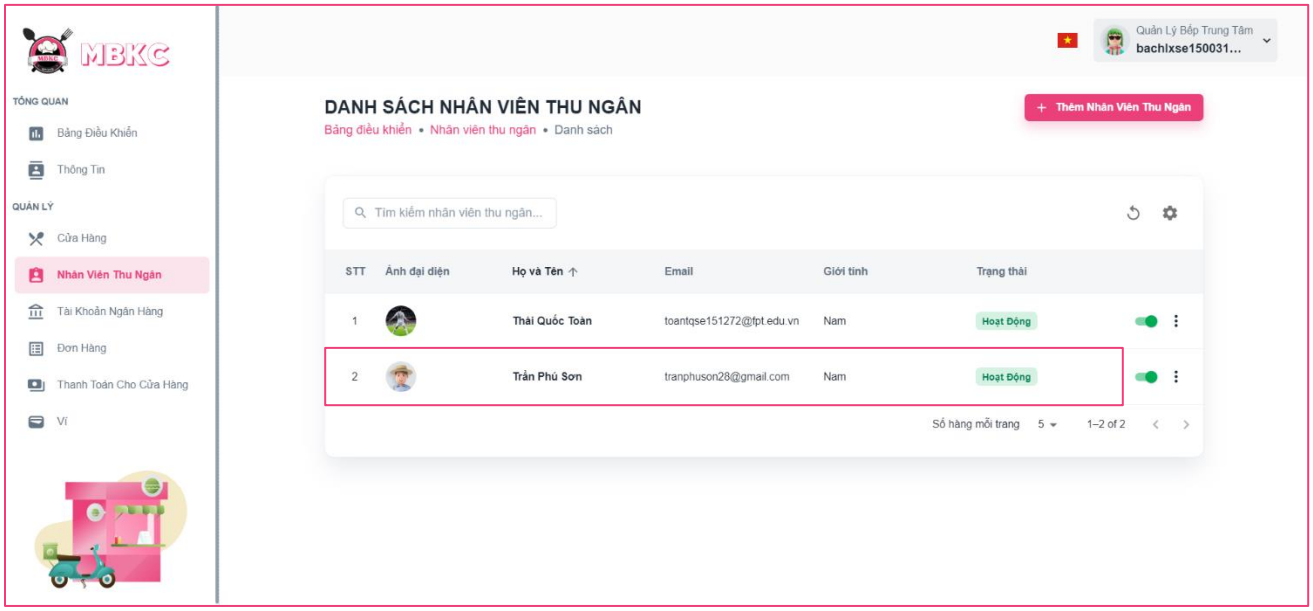


Figure 556: User Manual - Update Information Banking Account – Step 4 – Case 2 – 1

2: Click on Menu icon button. Then click “Chỉnh sửa” menu item to pop up Update Banking Account Modal.

Figure 557: User Manual - Update Information Banking Account – Step 4 – Case 2 – 2

Step 5: Banking account information will be automatically filled in the form. Then edit the information as desired.

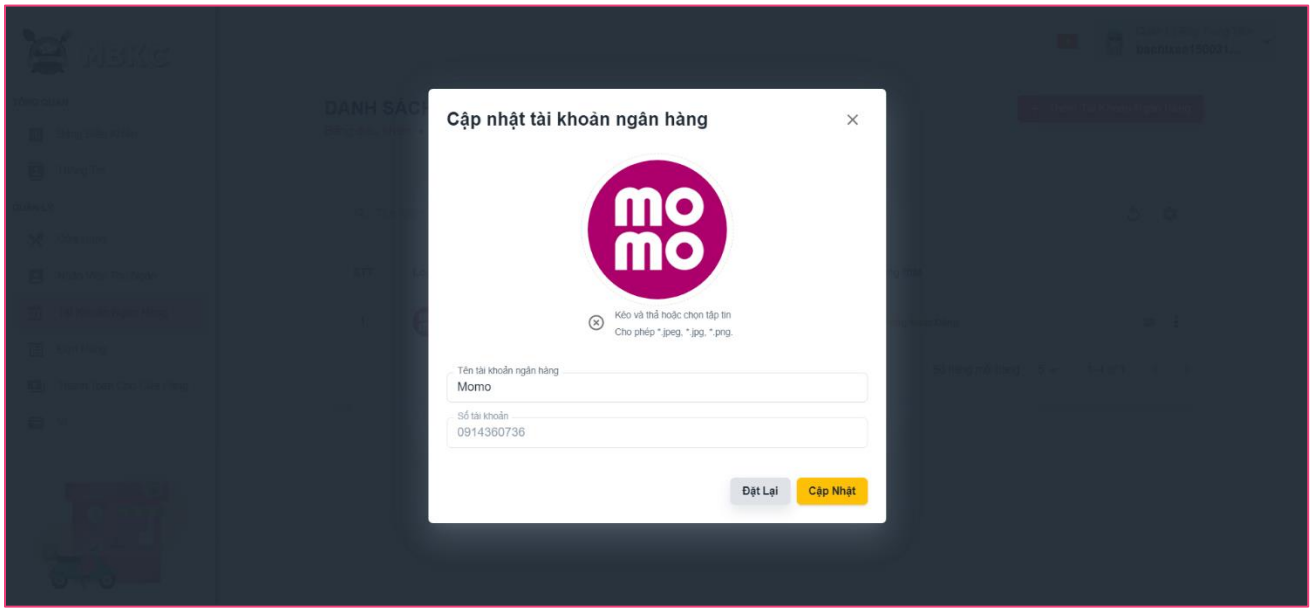


Figure 558: User Manual - Update Information Banking Account – Step 5

Step 6: Click “Cập nhật” button.

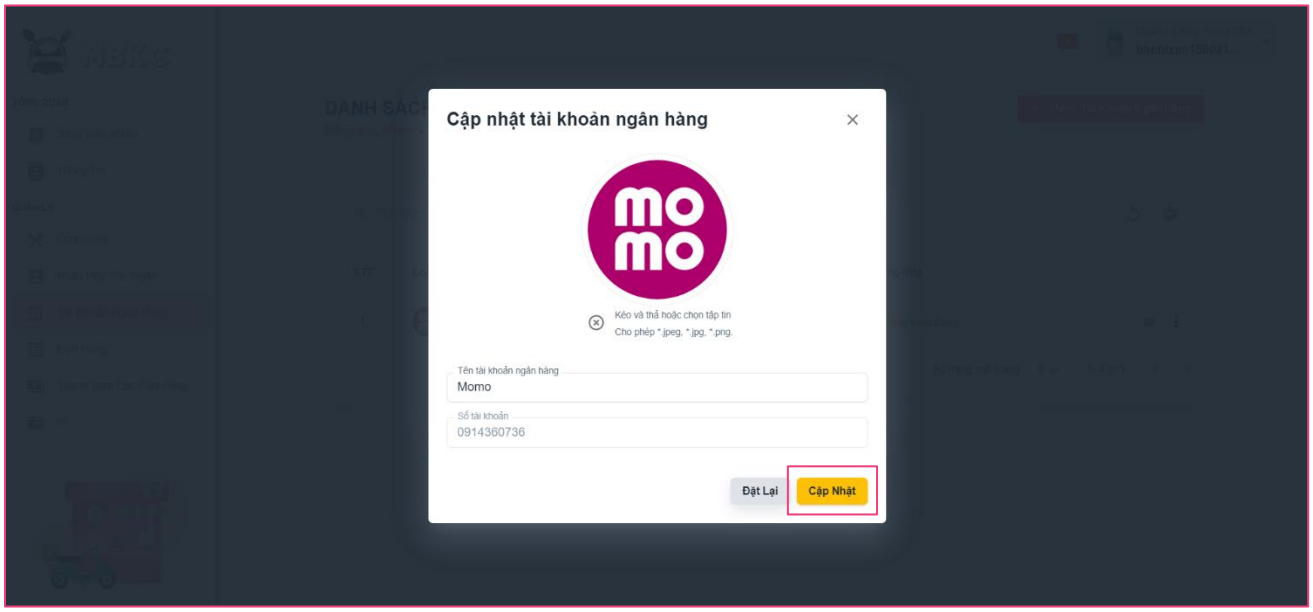


Figure 559: User Manual - Update Information Banking Account – Step 6

Step 7: List of Banking Accounts Page showing the list of banking accounts when start edit information of banking account from List of Banking Accounts Page and a toast message “Cập nhật nhân viên thu ngân thành công”.

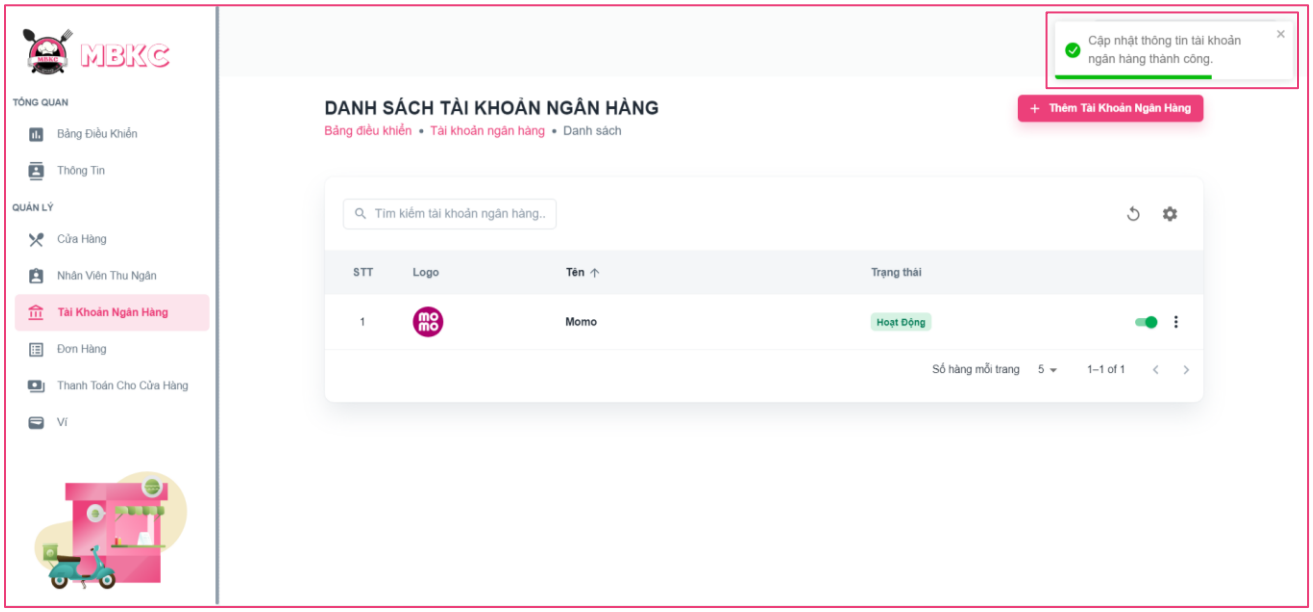


Figure 560: User Manual - Update Information Banking Account – Step 7

3.2.12.6. Delete Banking Account

3.2.12.6.1. Description

Use this guideline to delete banking account.

This web application supports delete banking account with an account as a Kitchen Center Manager.

3.2.12.6.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

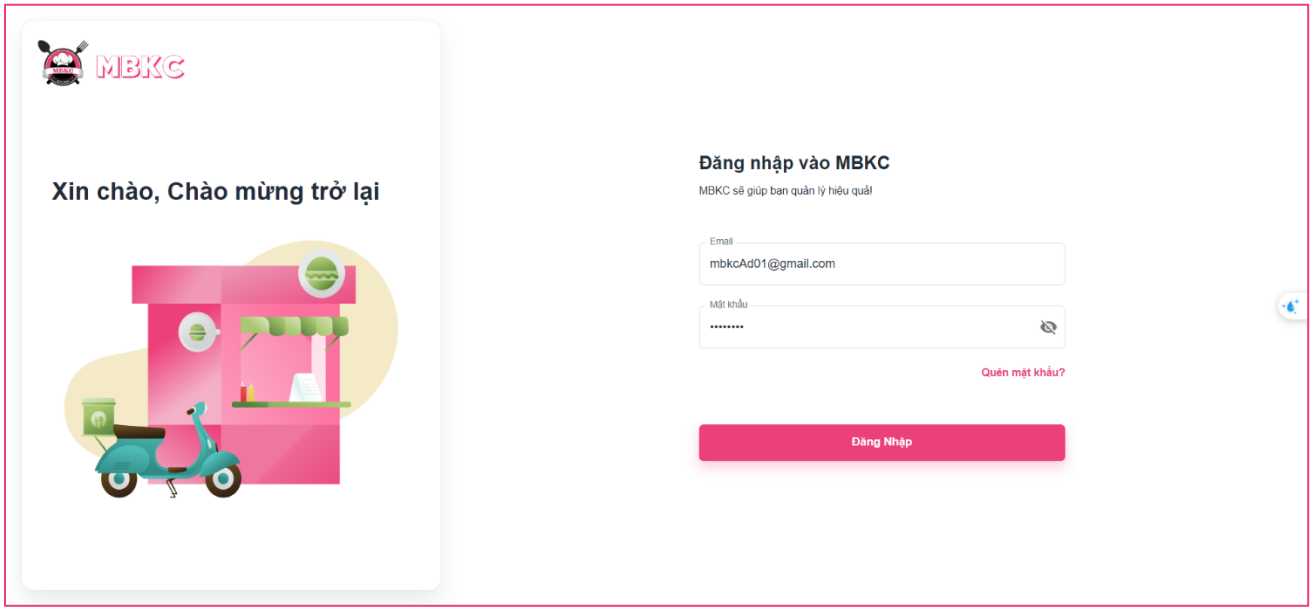


Figure 561: User Manual – Delete Banking Account – Step 1

Step 2: Click “Đăng Nhập” button.

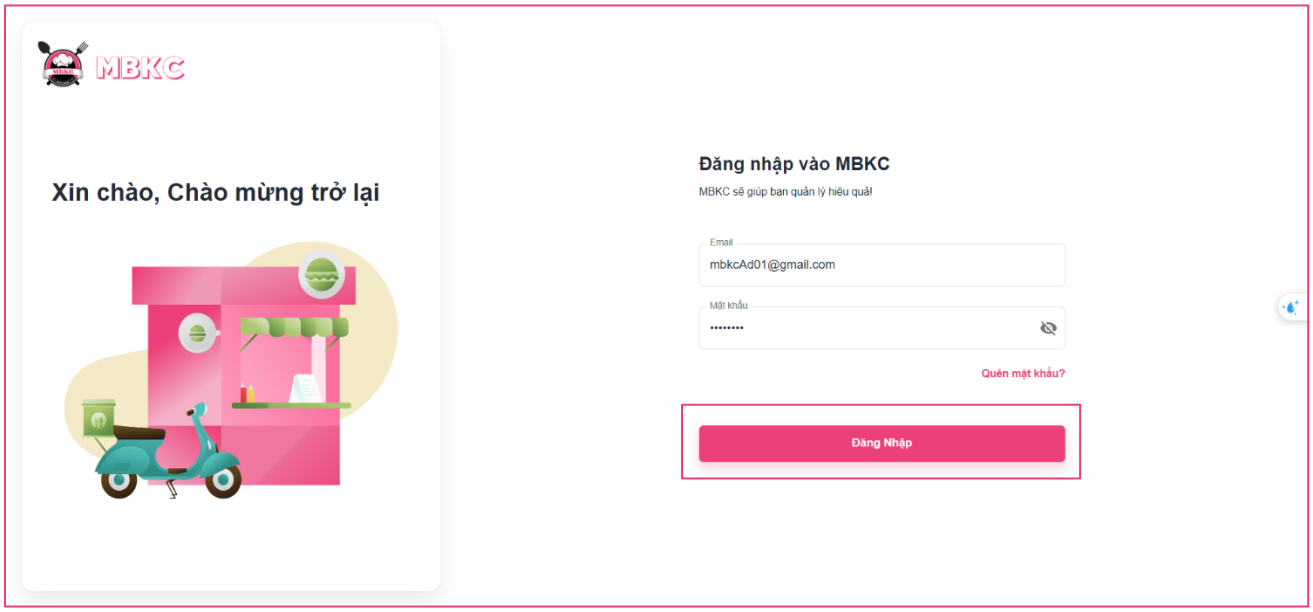


Figure 562: User Manual - Delete Banking Account – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Tài Khoản Ngân Hàng” from the left sidebar.

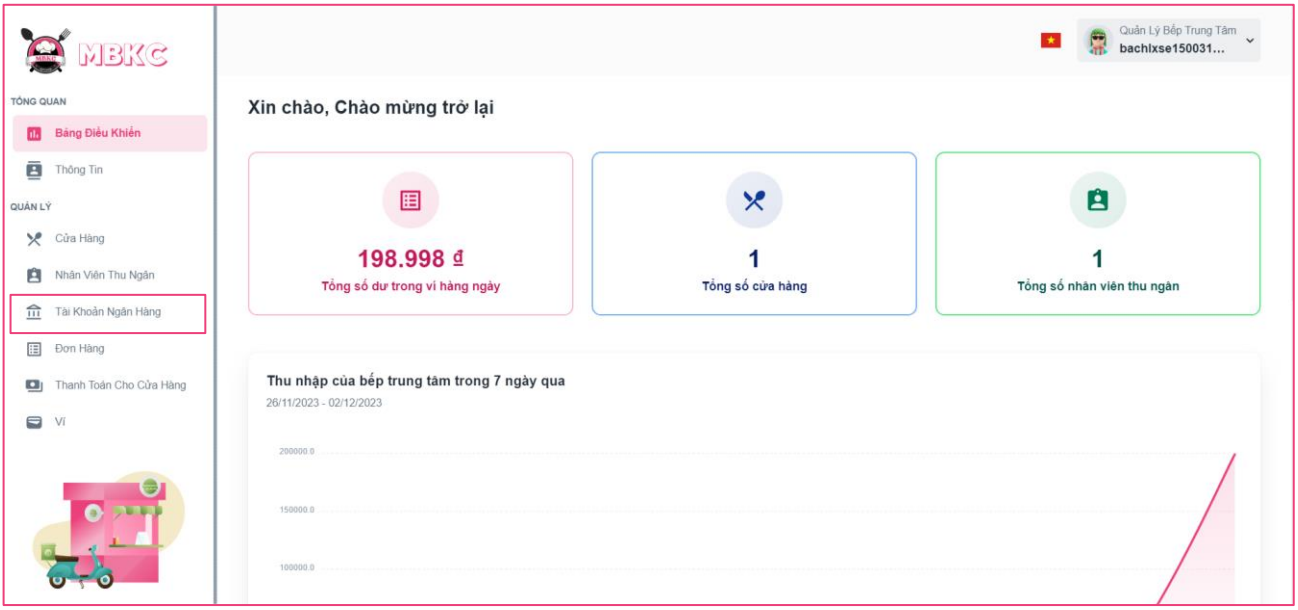


Figure 563: User Manual - Delete Banking Account – Step 3

Step 4: Delete banking account.

Case 1: Delete banking account in table list.

Click on the Menu icon button. Then click “Xóa” menu item on the banking account row to delete.

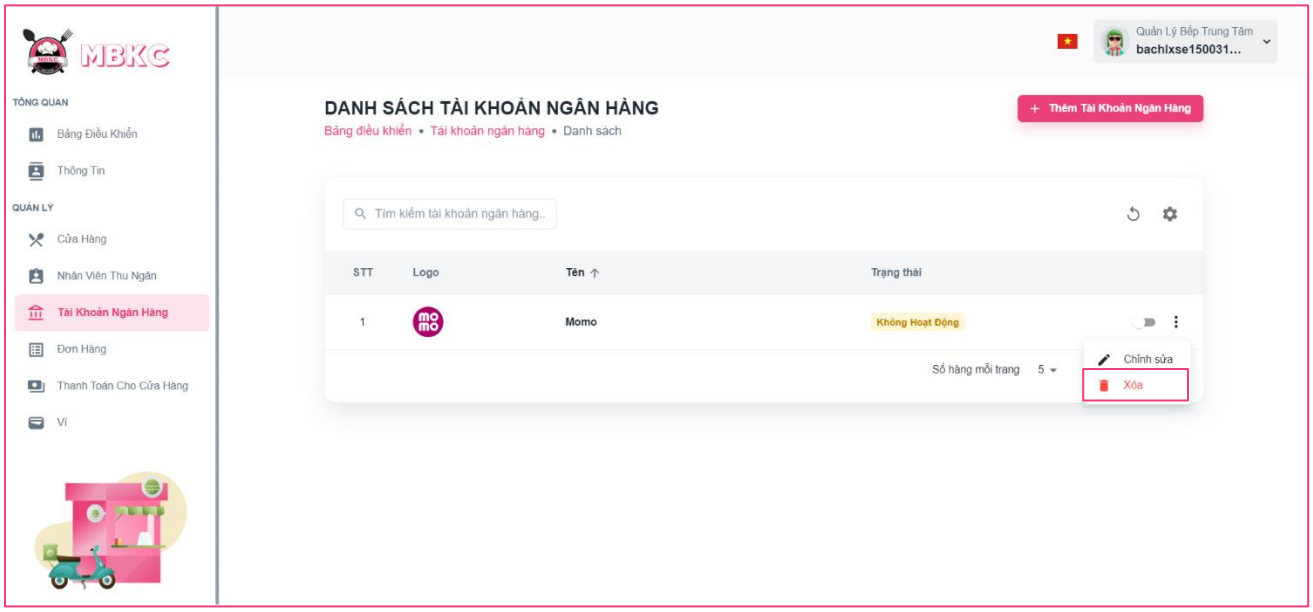


Figure 564: User Manual - Delete Banking Account – Step 4 – Case 1

Case 2: Delete banking account on the detail modal.

1: Click the banking account row to pop up Banking Account Details Modal.

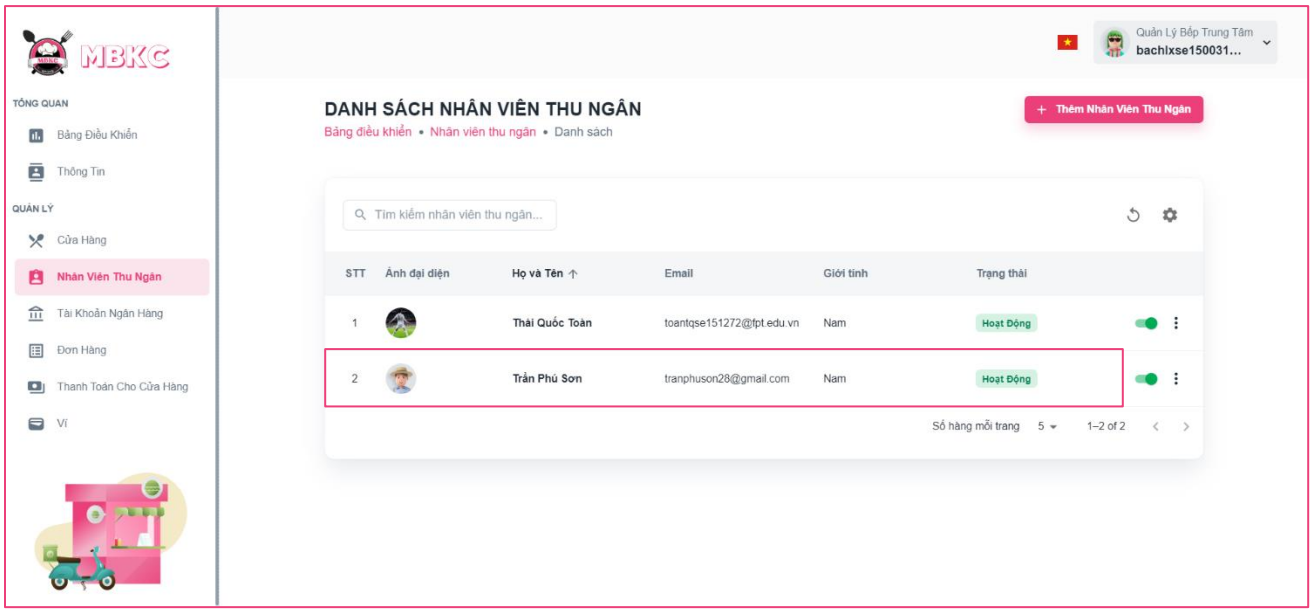


Figure 565: User Manual - Delete Banking Account – Step 4 – Case 2 – 1

2: Click on Menu icon button. Then click “Xóa” menu item to delete.

Figure 566: User Manual - Delete Banking Account – Step 4 – Case 2 - 2

Step 5: Click “Xác Nhận” button.

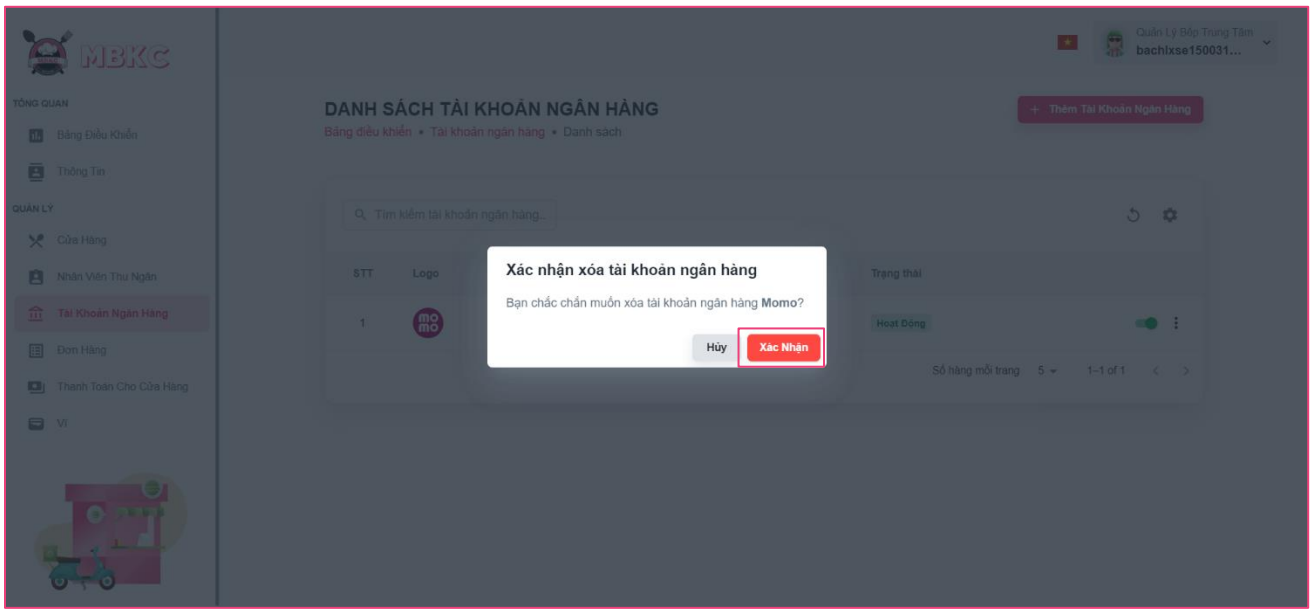


Figure 567: User Manual - Delete Banking Account – Step 5

Step 6: List of banking accounts will be reloaded and a toast message “Đã xóa nhân viên thu ngân thành công”. And banking account has been deleted.

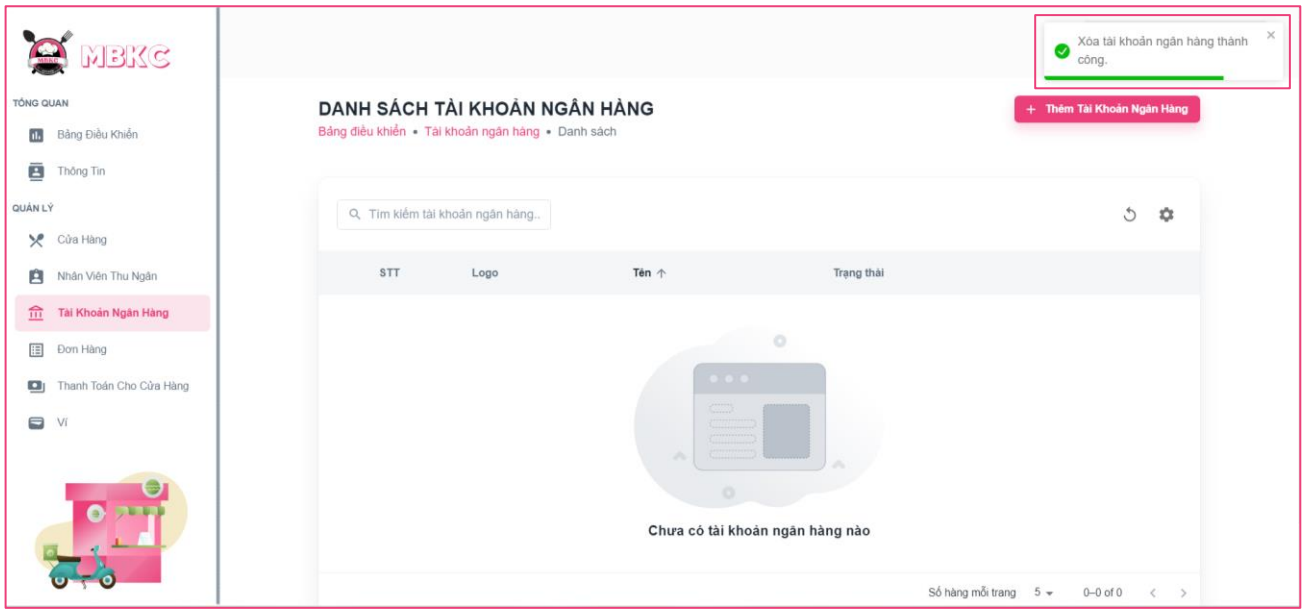


Figure 568: User Manual – Delete Banking Account – Step 6

3.2.13. Order management

3.2.13.1. View list of Orders

3.2.13.1.1. Description

Use this guideline to search, reload data, filter by date from and date to, filter by system status, filter by partner order status and show or hide columns in the table list of orders.

This web application supports view list of orders with an account as a Kitchen Center Manager or Cashier.

3.2.13.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager or Cashier.

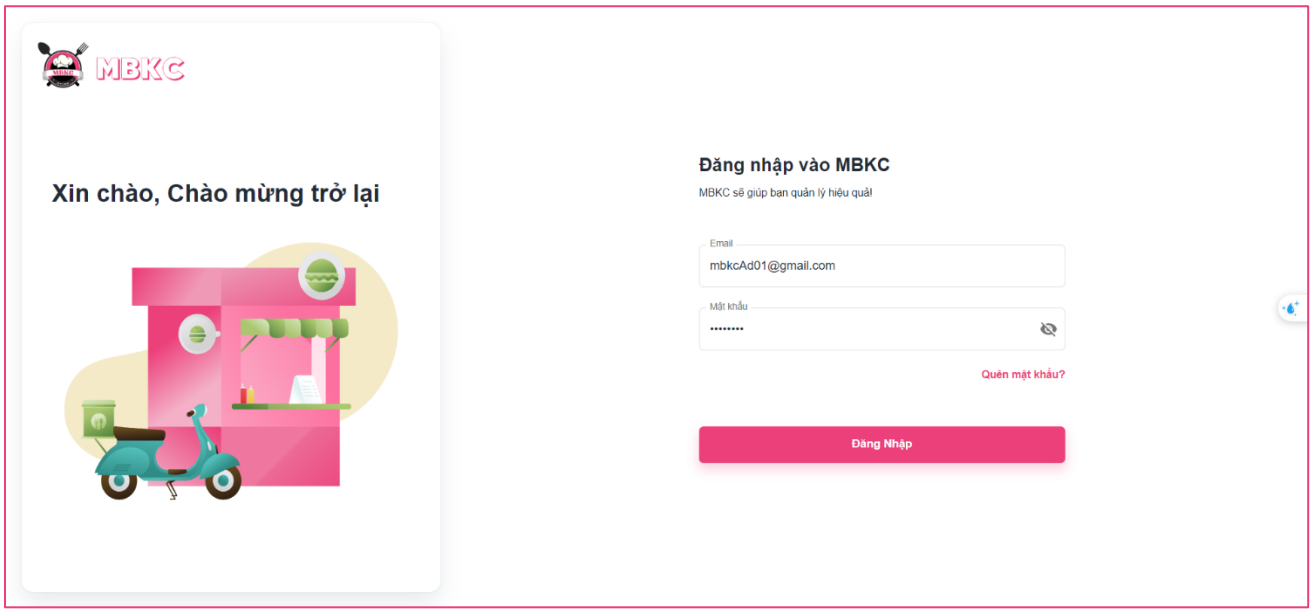


Figure 569: User Manual - View list of Orders – Step 1

Step 2: Click “Đăng Nhập” button.

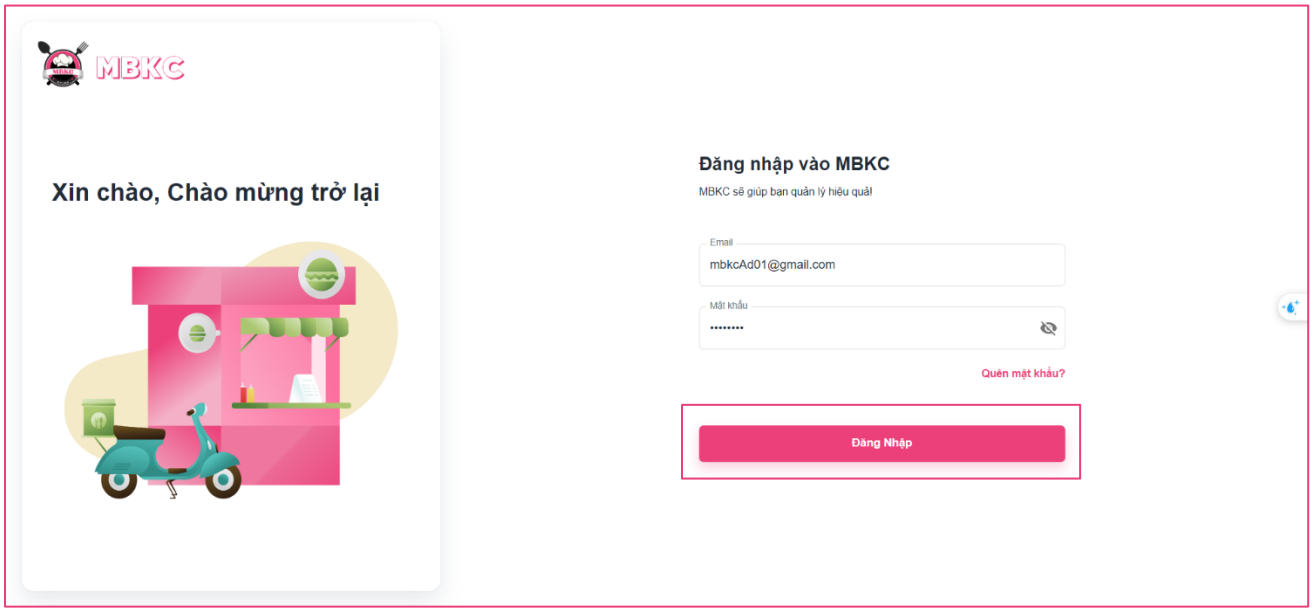


Figure 570: User Manual - View list of Orders – Step 2

Step 3: From Kitchen Center Dashboard Page or Cashier Dashboard Page, choose "Đơn Hàng" from the left sidebar.

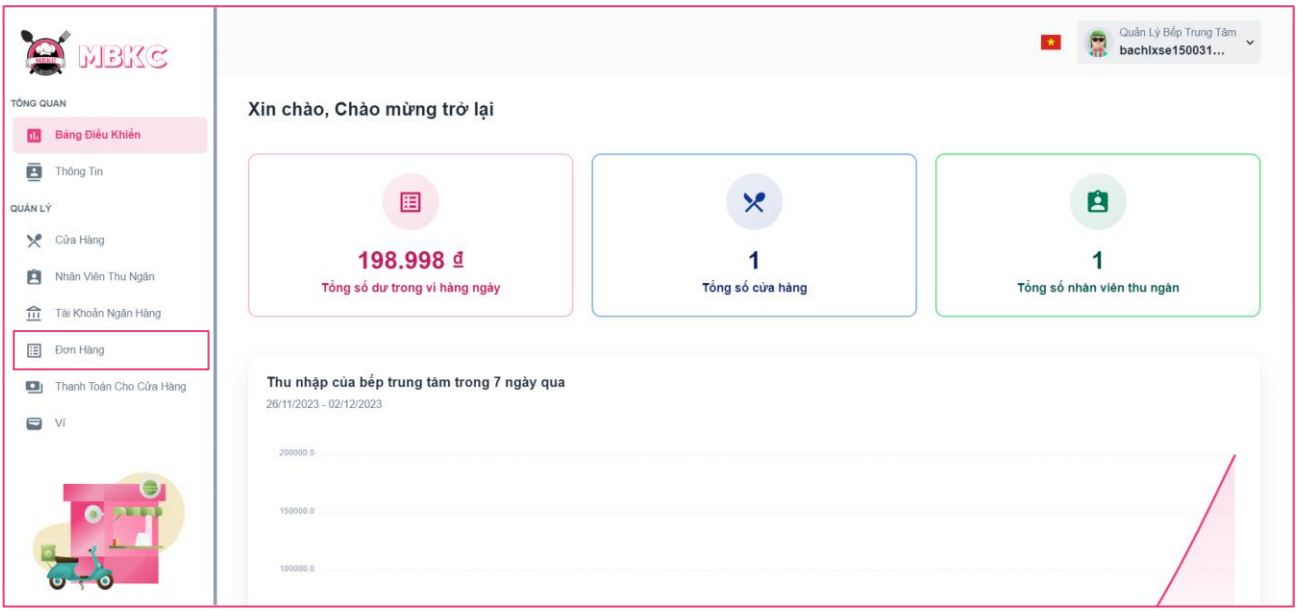


Figure 571: User Manual - View list of Orders – Step 3

Step 4: Search, reload data, filter by date from and date to, filter by system status, filter by partner order status, and show or hide columns in the table list of orders.

Case 1: Fill in ID đối tác in the search box to search orders in the table list.

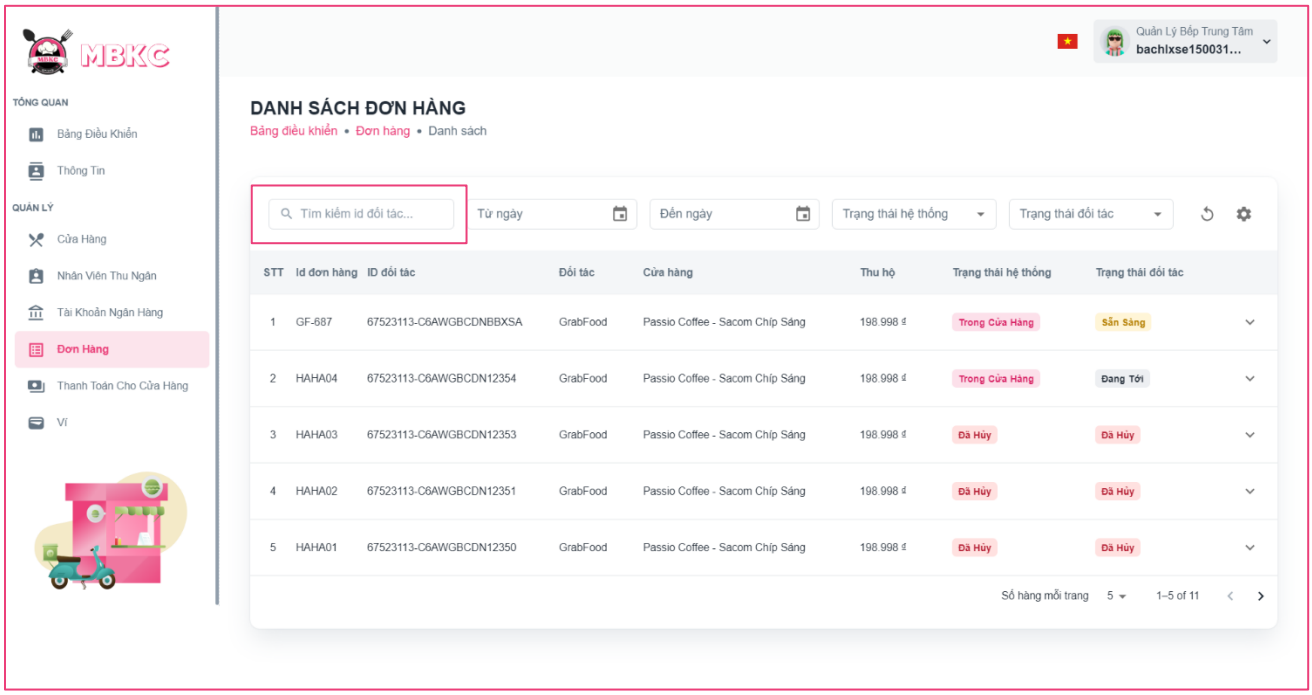


Figure 572: User Manual - View list of Orders – Step 4 – Case 1

Case 2: Filter by "Từ ngày" and "Đến ngày" to view a list of orders by selected date.

+ Filter “Từ ngày” to view list of order from the selected date to the present.

+ Filter “Đến ngày” to view a list of orders from the selected date to earlier.

+ Filter “Từ ngày” and "Đến ngày" to view a list of orders from “Từ ngày” selected to “Đến ngày” selected.

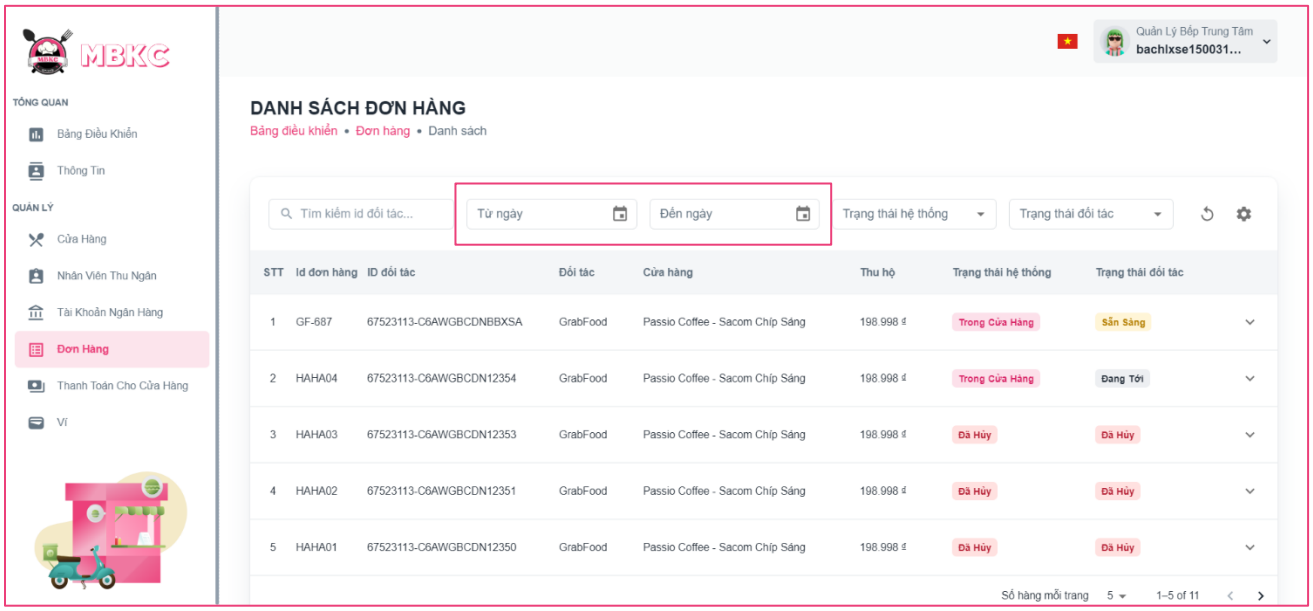


Figure 573: User Manual - View list of Orders – Step 4 – Case 2

Case 3: Filter by “Trạng thái hệ thống” to view the list of orders that have system status corresponding.

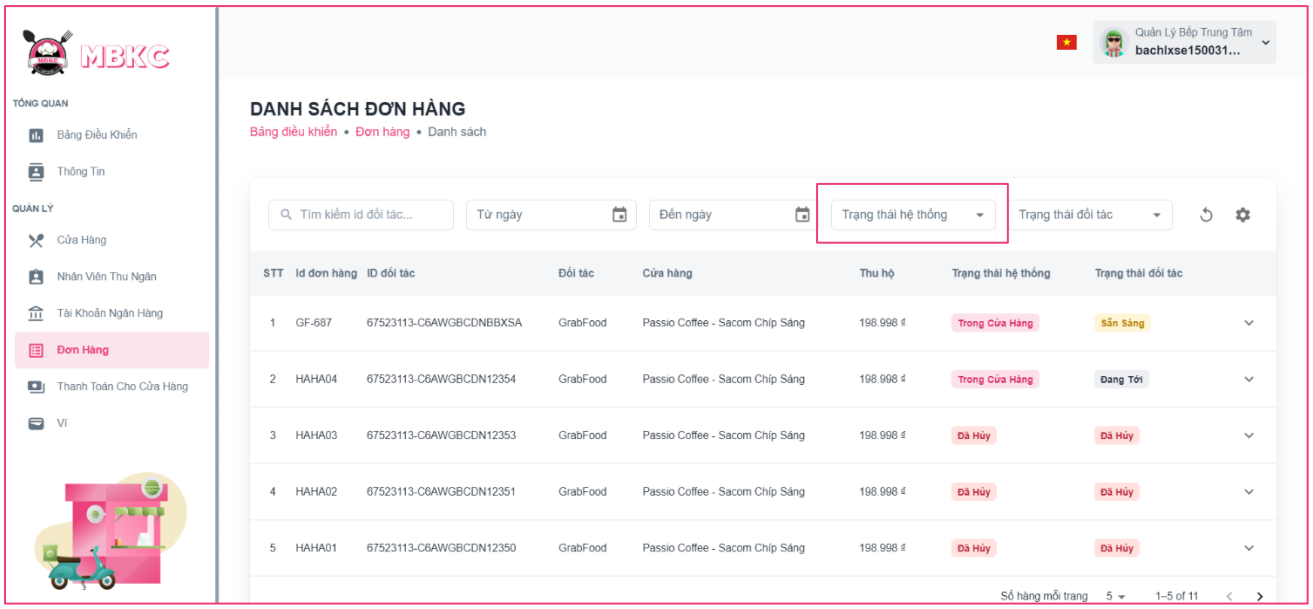


Figure 574: User Manual - View list of Orders – Step 4 – Case 3

Case 4: Filter by “Trạng thái đối tác” to view the list of orders that have system status corresponding.

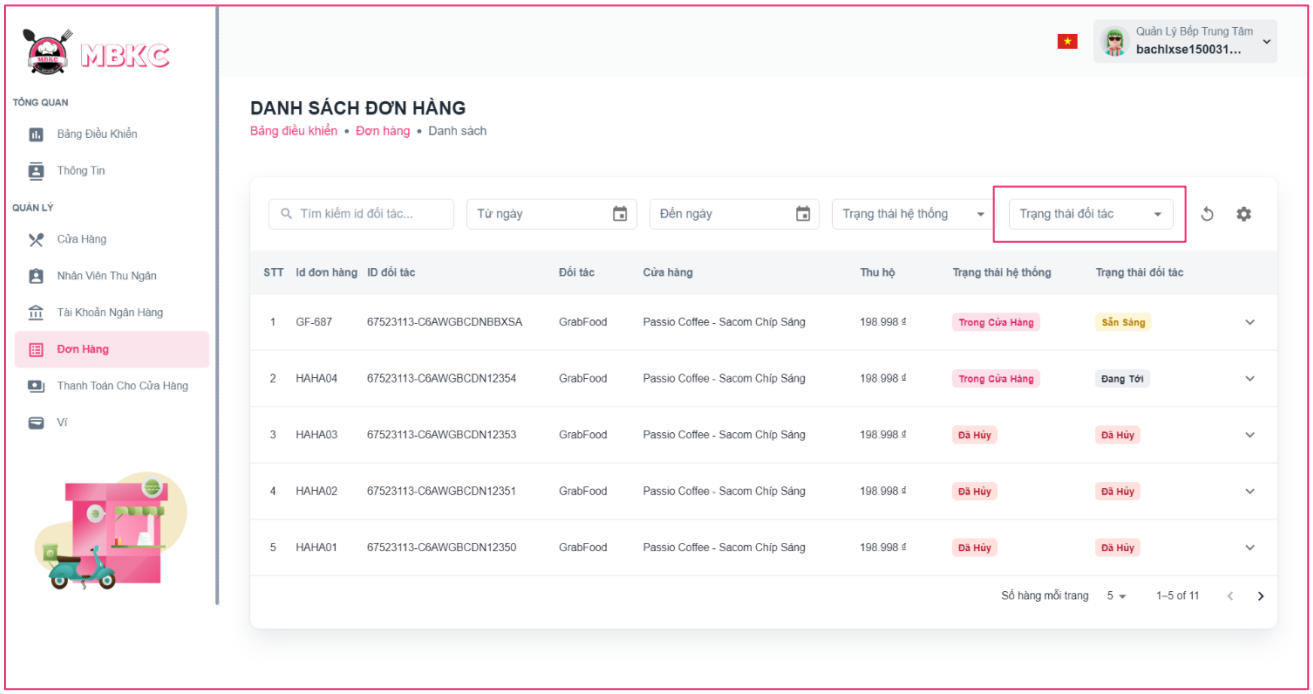


Figure 575: User Manual - View list of Orders – Step 4 – Case 4

Case 5: Click the Reload icon button to reload the list of orders.

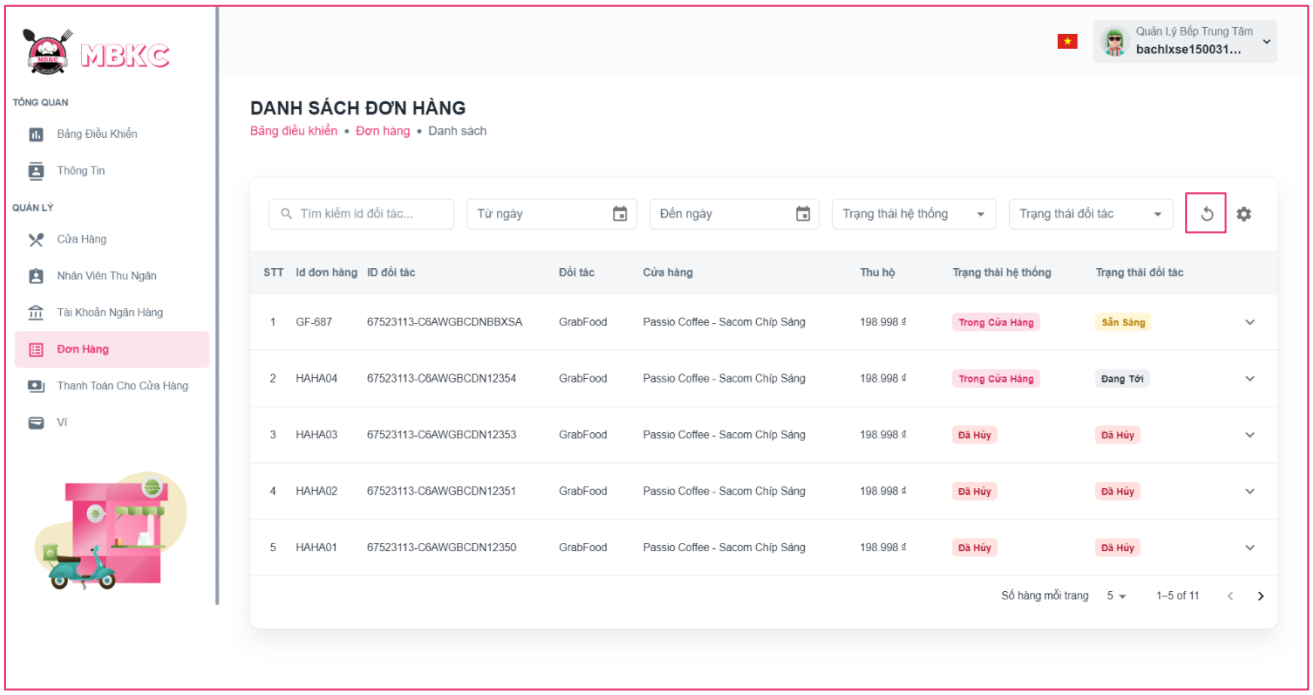


Figure 576: User Manual - View list of Orders – Step 4 – Case 5

Case 6: Click the Setting icon button and select menu items corresponding you want to show or hide columns in the table list of cashiers.

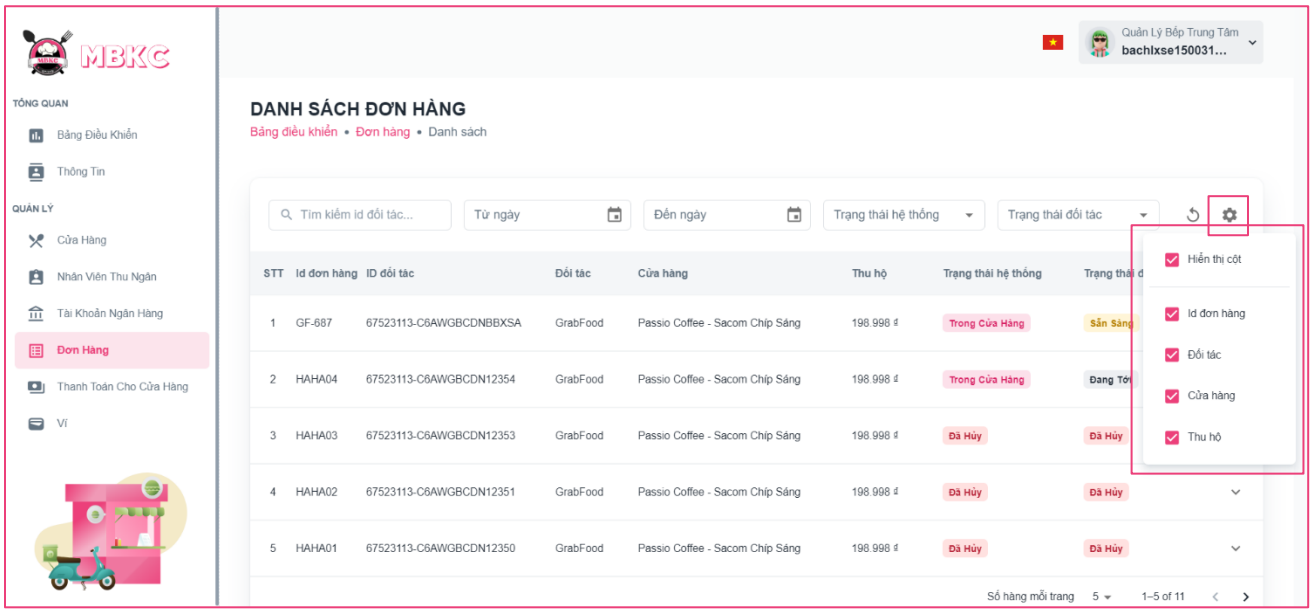


Figure 577: User Manual - View list of Orders – Step 4 – Case 6

3.2.13.2. View details Order

3.2.13.2.1. Description

Use this guideline to view details order.

This web application supports view details order with an account as a Kitchen Center Manager order Cashier.

3.2.13.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager or Cashier.

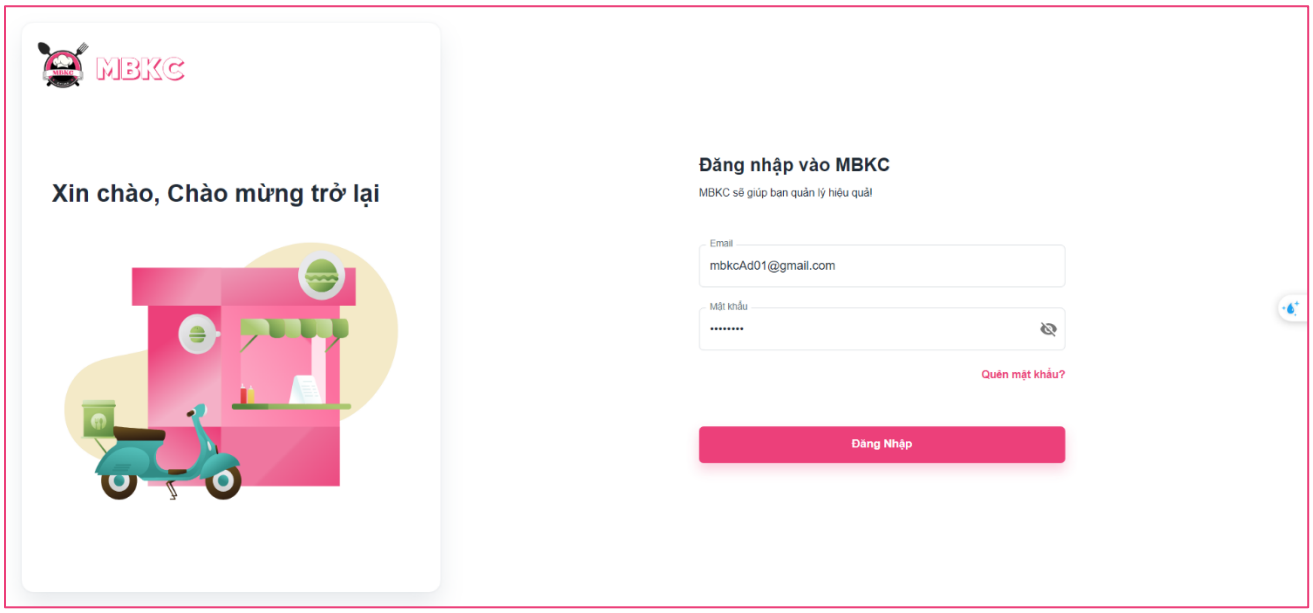


Figure 578: User Manual - View details Order – Step 1

Step 2: Click “Đăng Nhập” button.

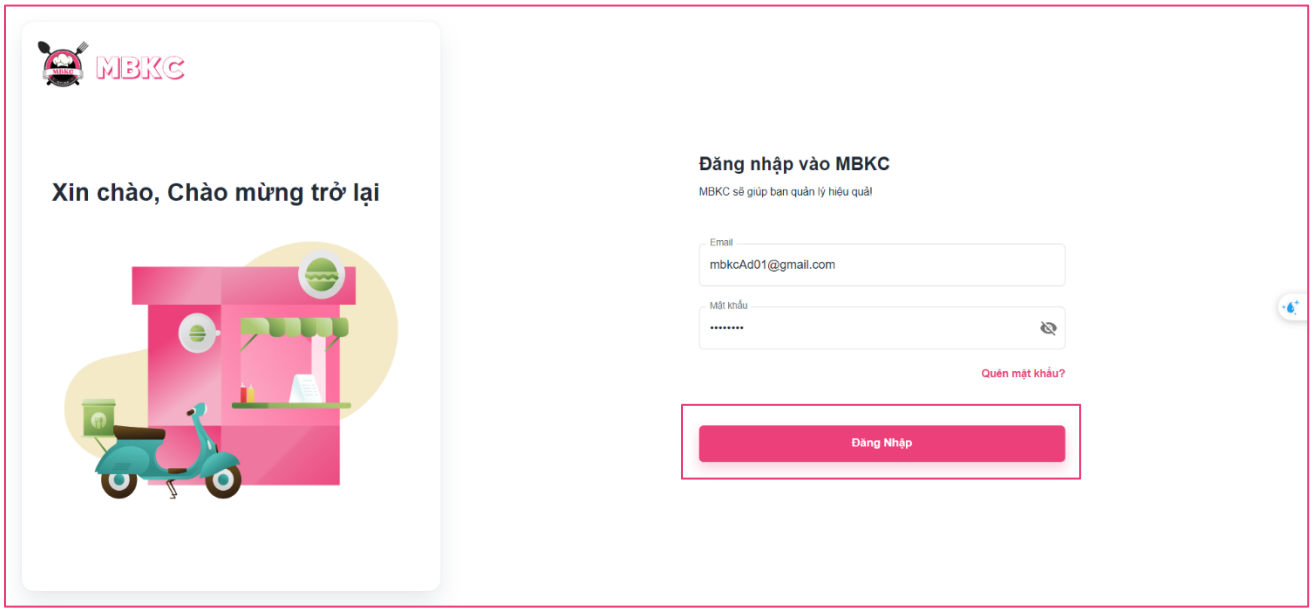


Figure 579: User Manual - View details Order – Step 2

Step 3: From Kitchen Center Dashboard Page or Cashier Dashboard Page, choose “Đơn Hàng” from the left sidebar.

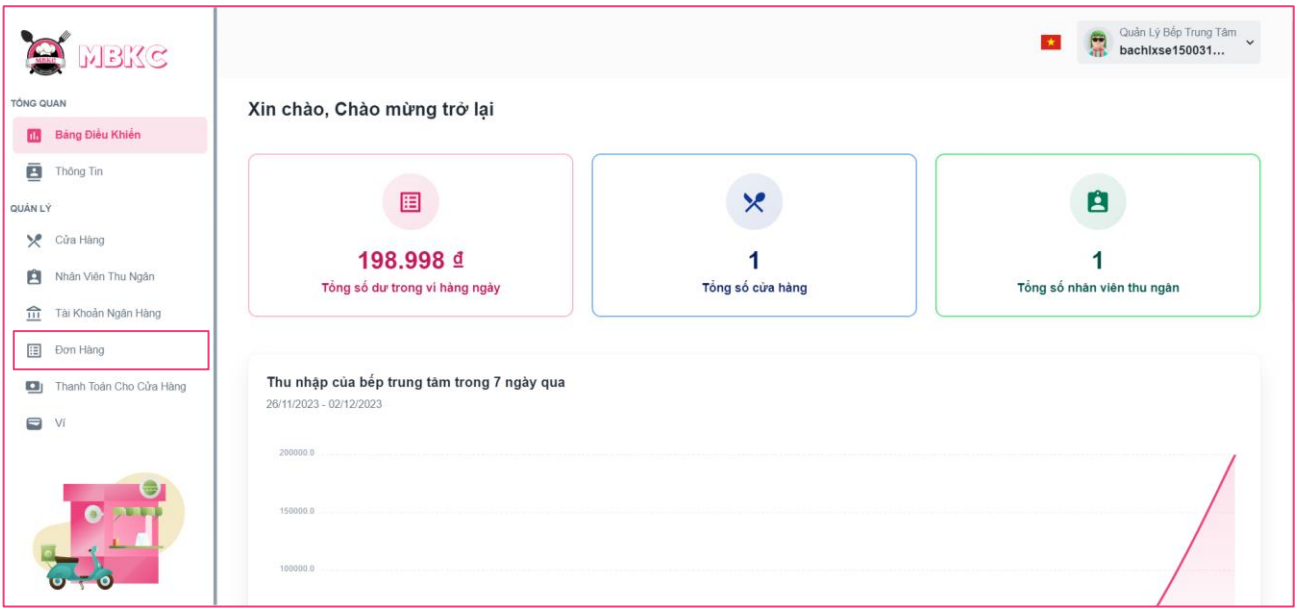


Figure 580: User Manual - View details Order – Step 3

Step 4: Click the order row to navigate to Order Details Page.

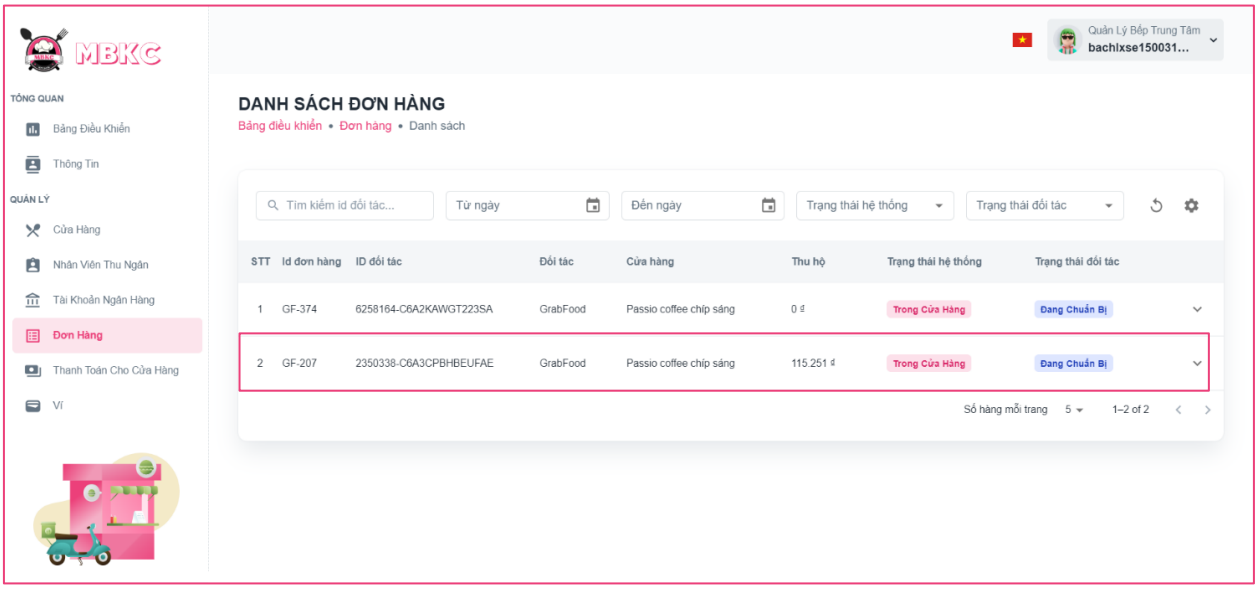


Figure 581: User Manual - View details Order – Step 4

Step 5: Navigate to Order Details Page to show information details of order.

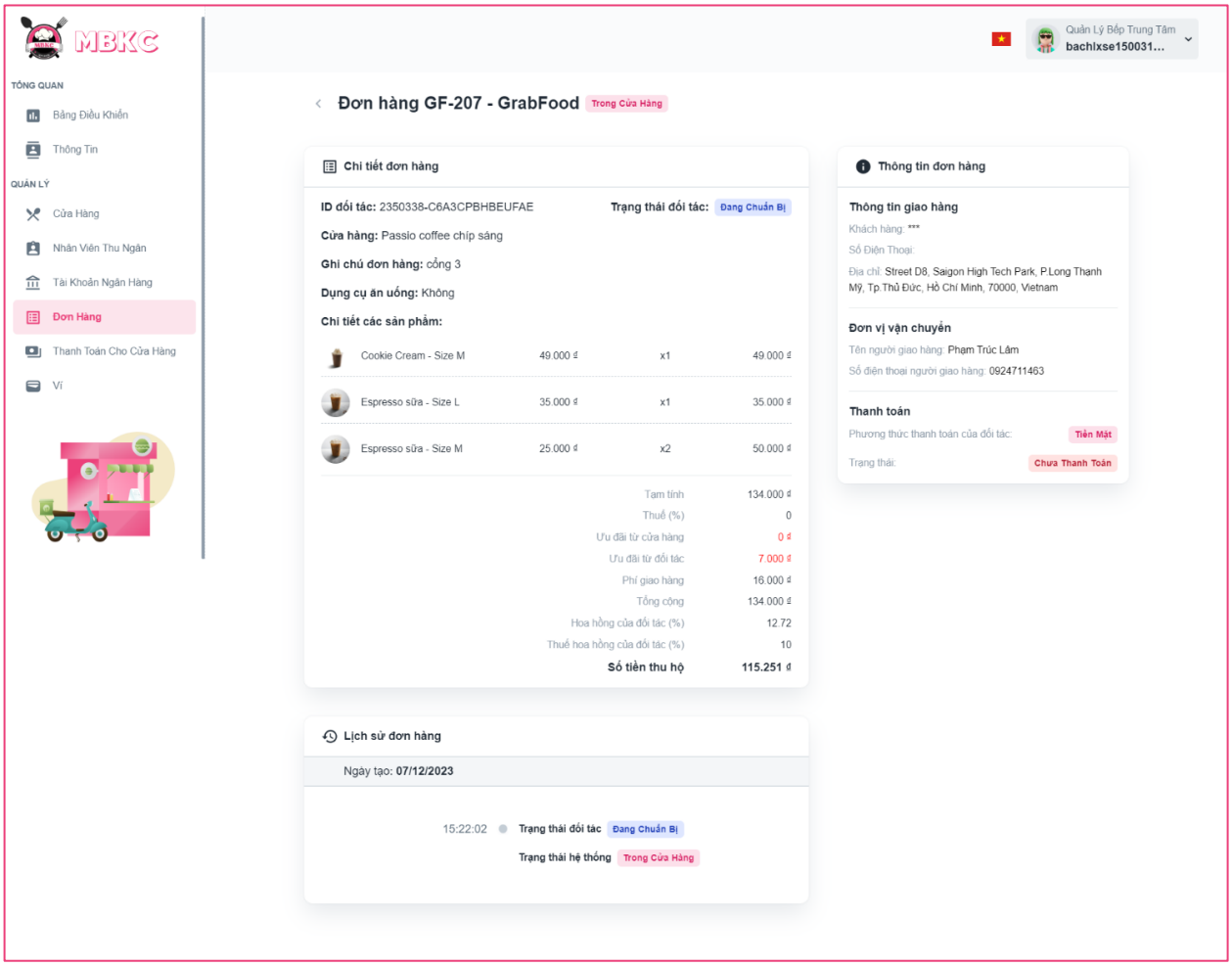


Figure 582: User Manual - View details Order – Step 5

3.2.13.3. Confirm Order Ready to Delivery

3.2.13.3.1. Description

Use this guideline to confirm order ready to delivery.

This web application supports confirm order ready to delivery with an account as a Cashier.

3.2.13.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Cashier.

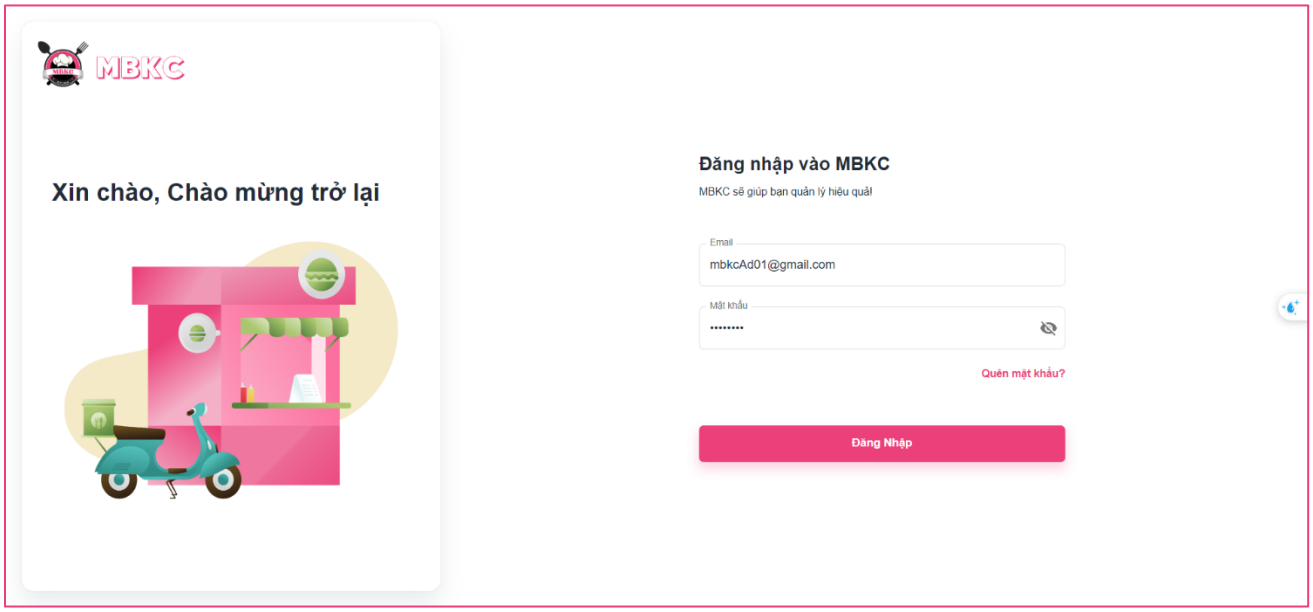


Figure 583: User Manual – Confirm Order Ready to Delivery – Step 1

Step 2: Click “Đăng Nhập” button.

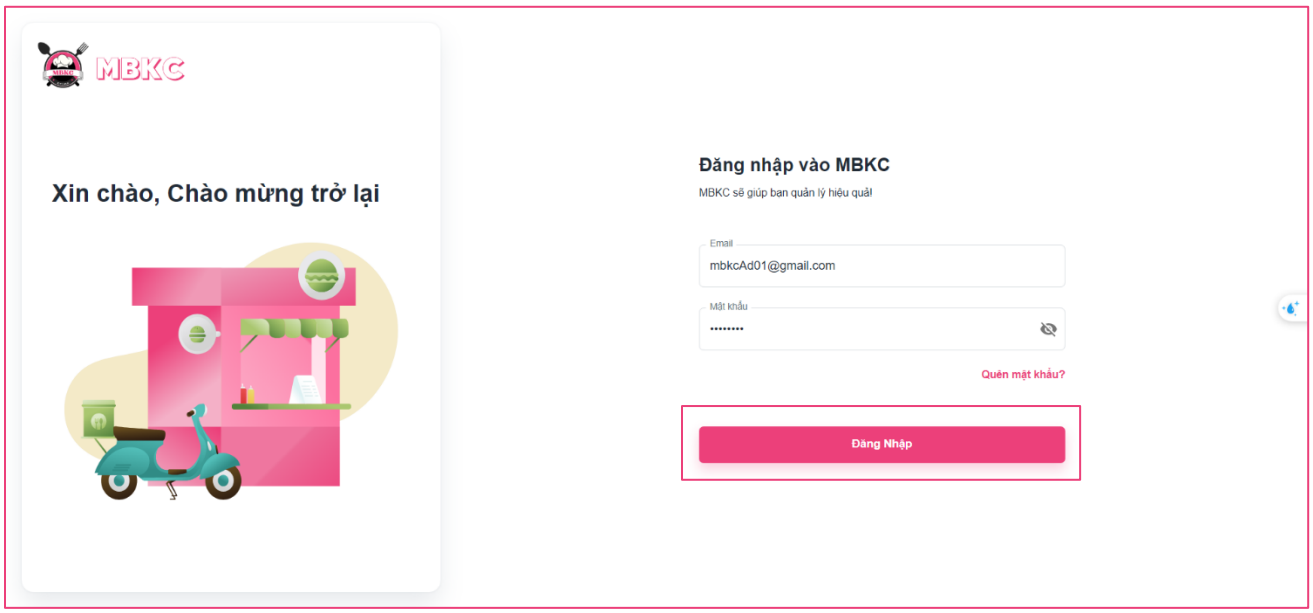


Figure 584: User Manual - Confirm Order Ready to Delivery – Step 2

Step 3: From Cashier Dashboard Page, choose “Đơn Hàng” from the left sidebar.

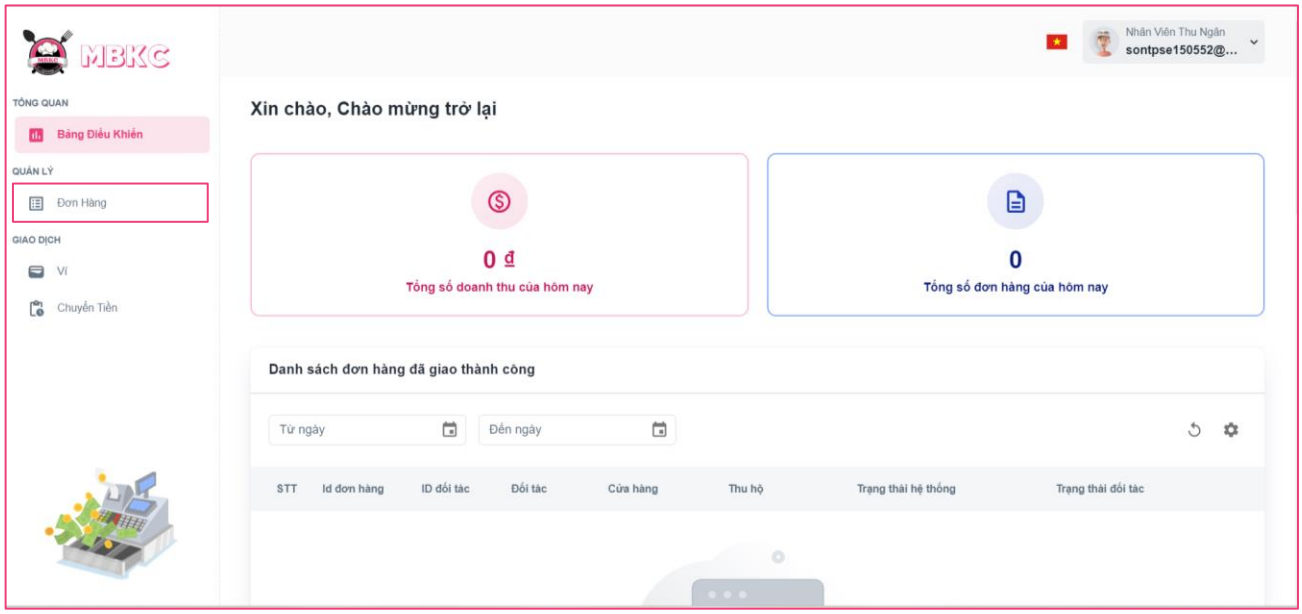


Figure 585: User Manual - Confirm Order Ready to Delivery – Step 3

Step 4: Click the order row have system status “Trong cửa hàng” – “In store” and order part status “Sẵn Sàng” – “Ready” to navigate to Order Details Page.

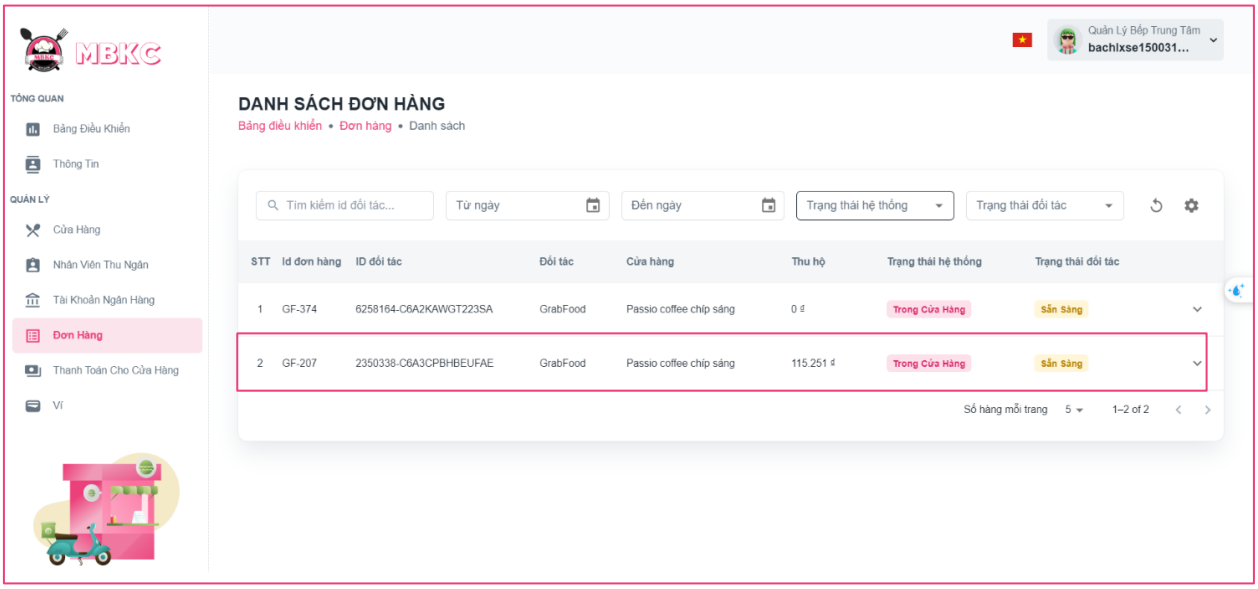


Figure 586: User Manual - Confirm Order Ready to Delivery – Step 4

Step 4: Click “Sẵn Sàng Để Giao” to confirm order ready to delivery.

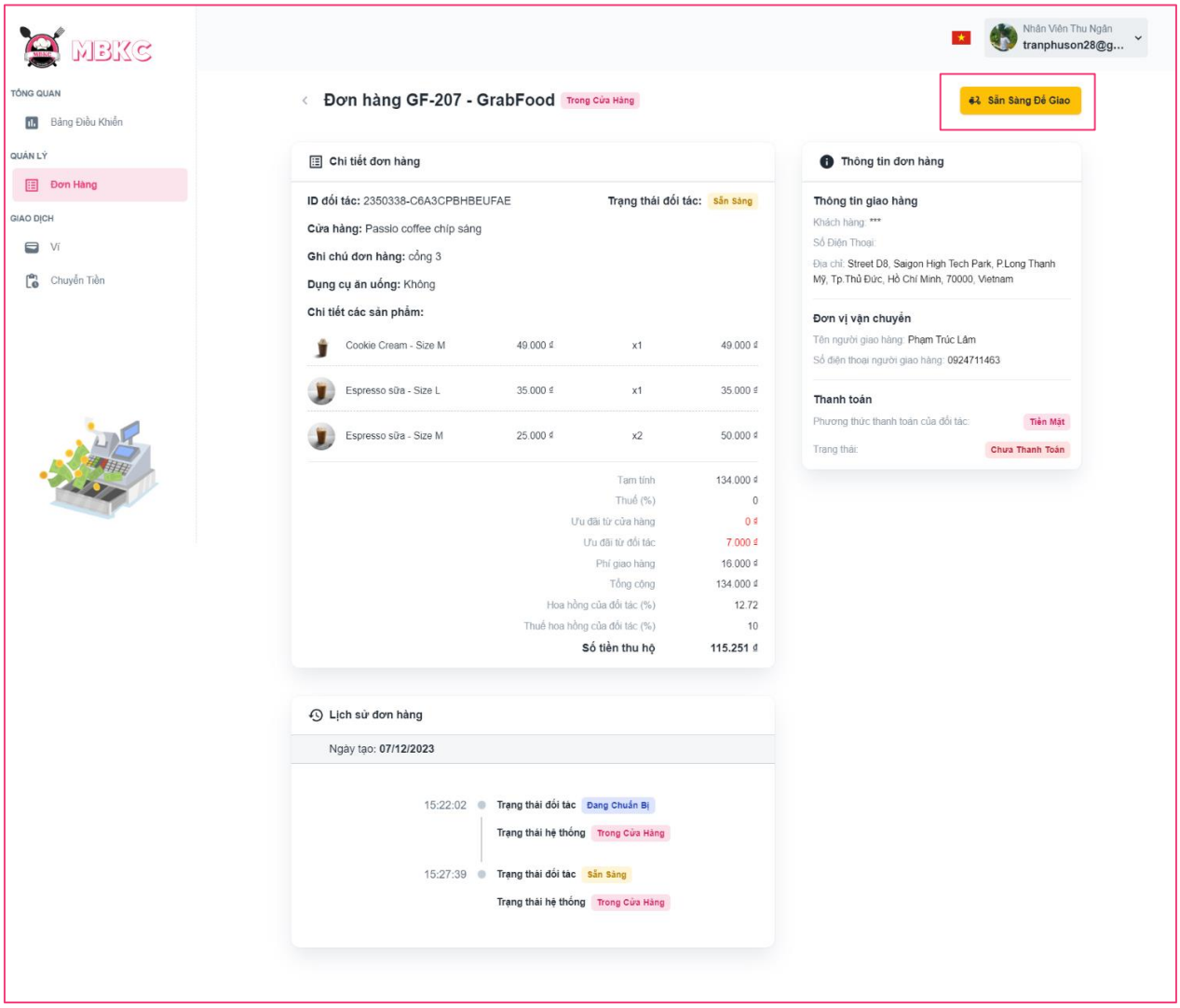


Figure 587: User Manual - Confirm Order Ready to Delivery – Step 4

Step 5: Click “Xác nhận” button.

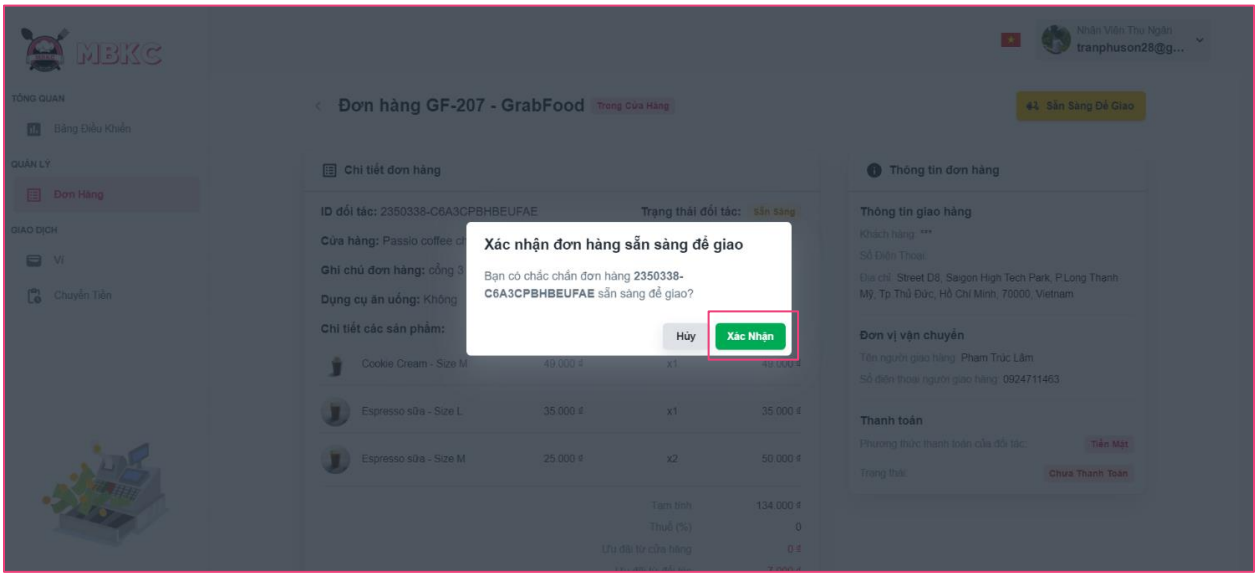


Figure 588: User Manual - Confirm Order Ready to Delivery – Step 5

Step 6: Order Details Page will be reloaded to show the order details updated and a toast message “Cập nhật đơn hang thành công”.

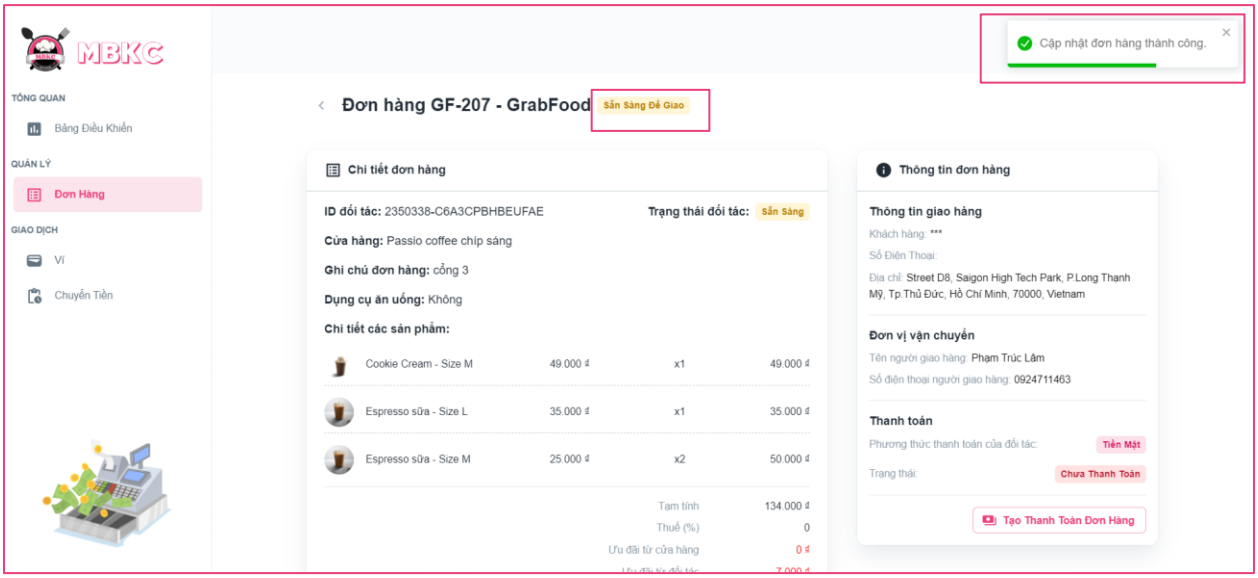


Figure 589: User Manual - Confirm Order Ready to Delivery – Step 6

3.2.13.4. Create Shipper Payment

3.2.13.4.1. Description

Use this guideline to create shipper payment.

This web application supports create shipper payment with an account as a Cashier.

3.2.13.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Cashier.

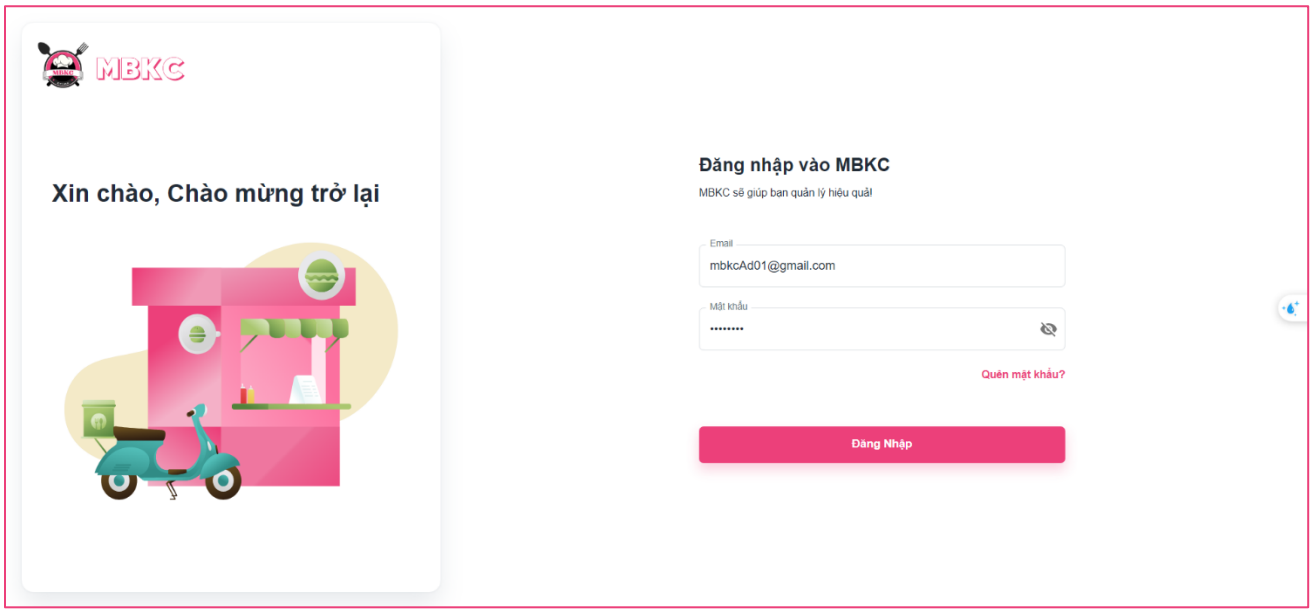


Figure 590: User Manual – Create Shipper Payment – Step 1

Step 2: Click “Đăng Nhập” button.

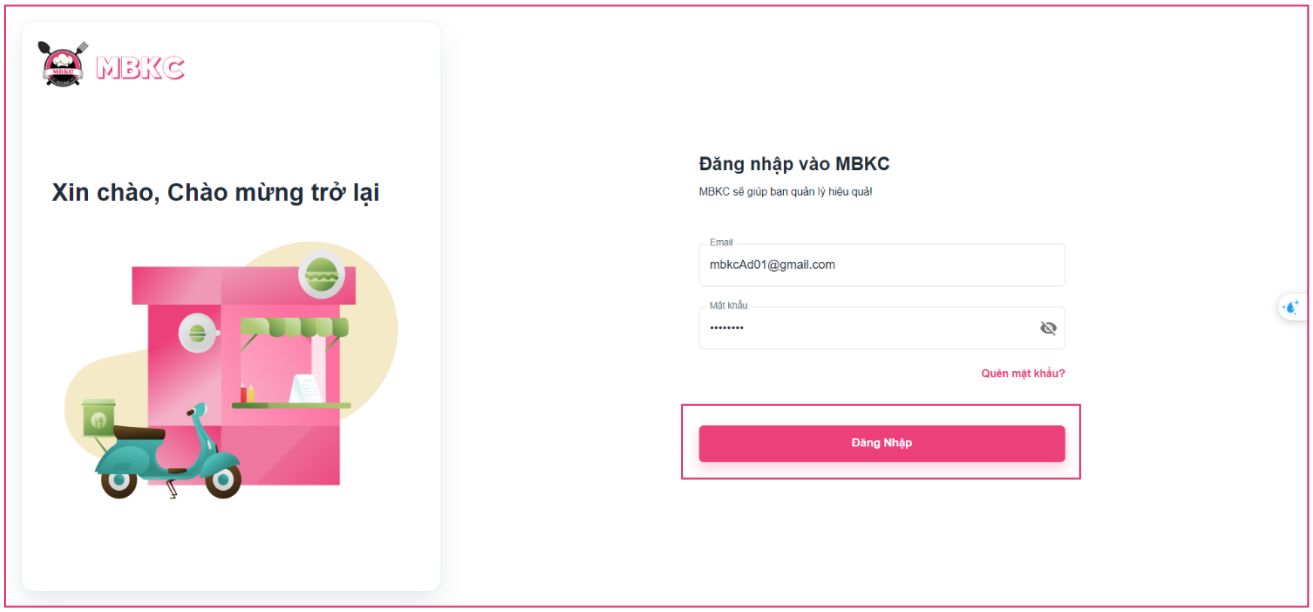


Figure 591: User Manual - Create Shipper Payment – Step 2

Step 3: From Cashier Dashboard Page, choose “Đơn Hàng” from the left sidebar.

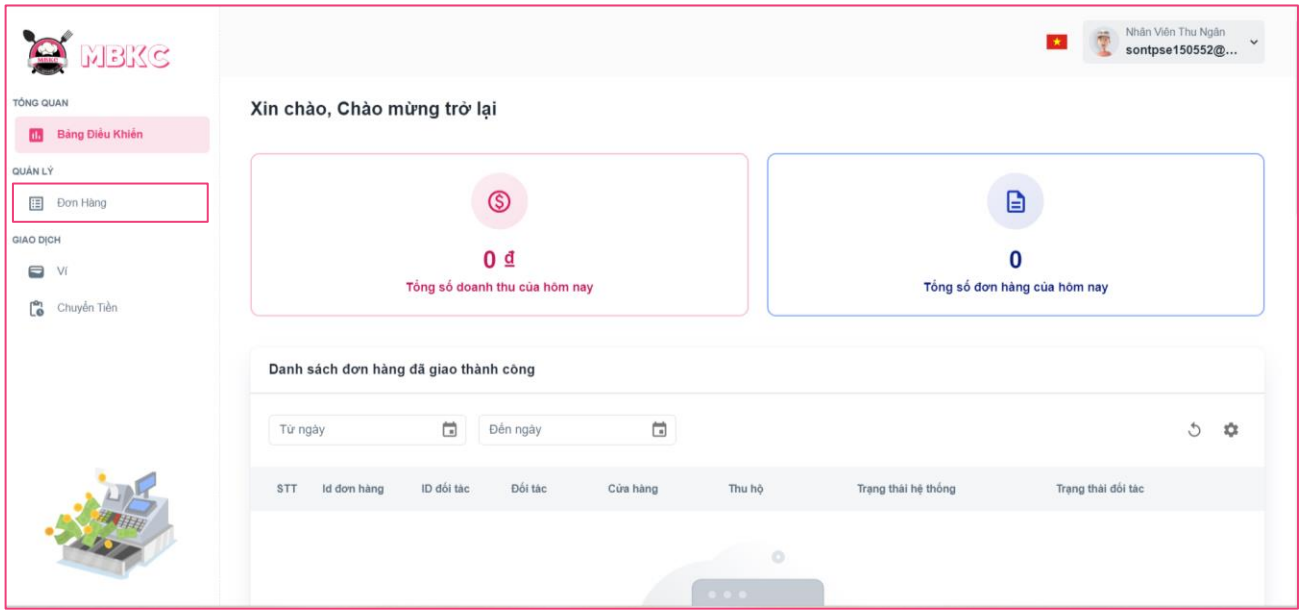


Figure 592: User Manual - Create Shipper Payment – Step 3

Step 4: Click the order row have system status “Sẵn Sàng Để Giao” – “Ready Delivery” and order part status “Sẵn Sàng” – “Ready” to navigate to Order Details Page.

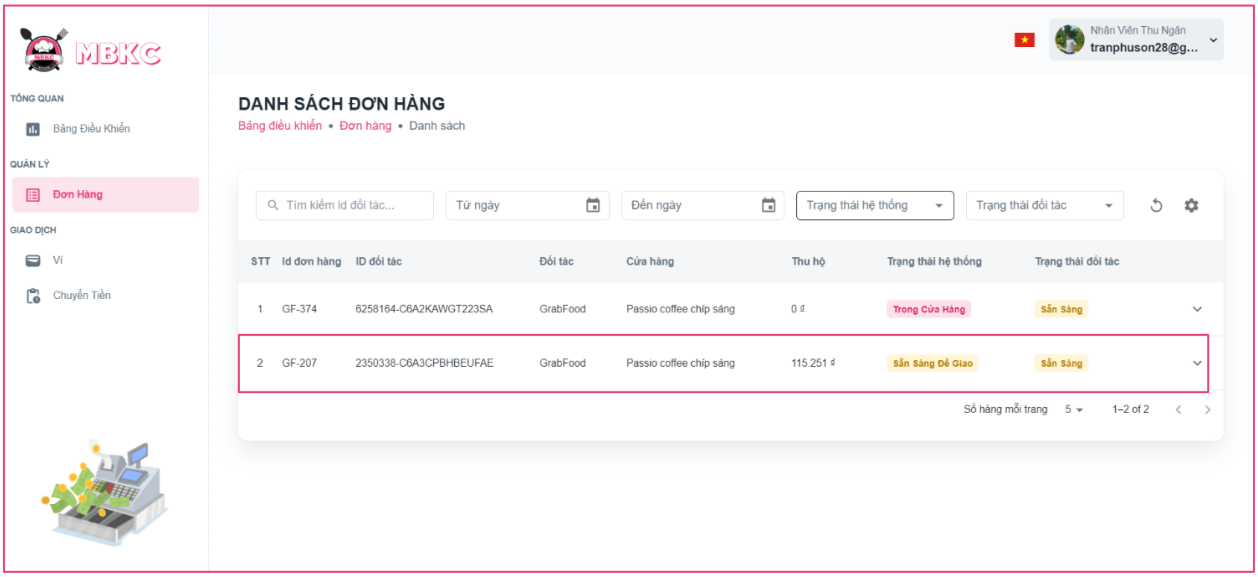


Figure 593: User Manual - Create Shipper Payment – Step 4

Step 4: Click “Tạo Thanh Toán Đơn Hàng” to confirm order delivery.

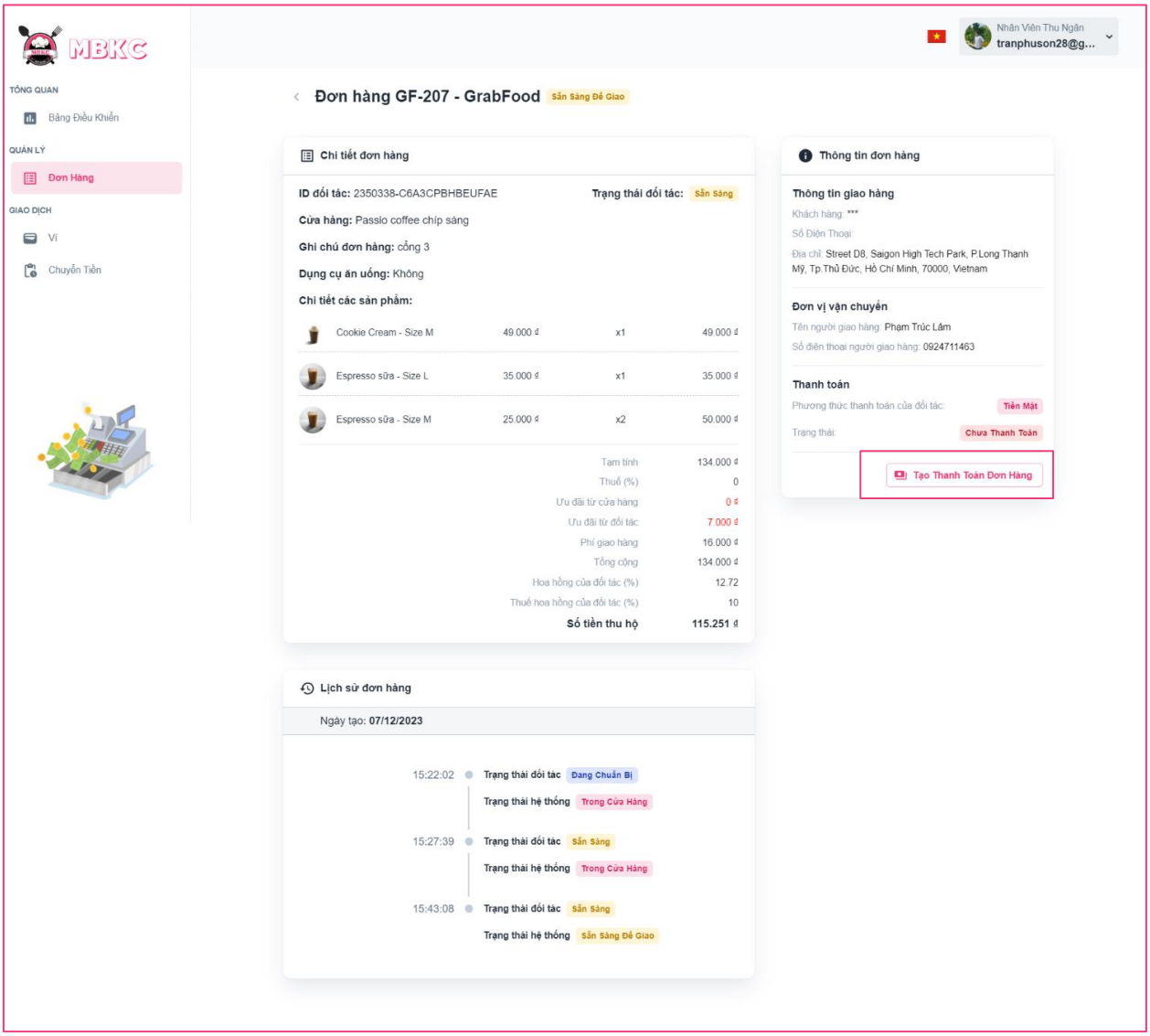


Figure 594: User Manual - Create Shipper Payment – Step 4

Step 5: Fill in “Hình ảnh” and “Phương thức thanh toán”.

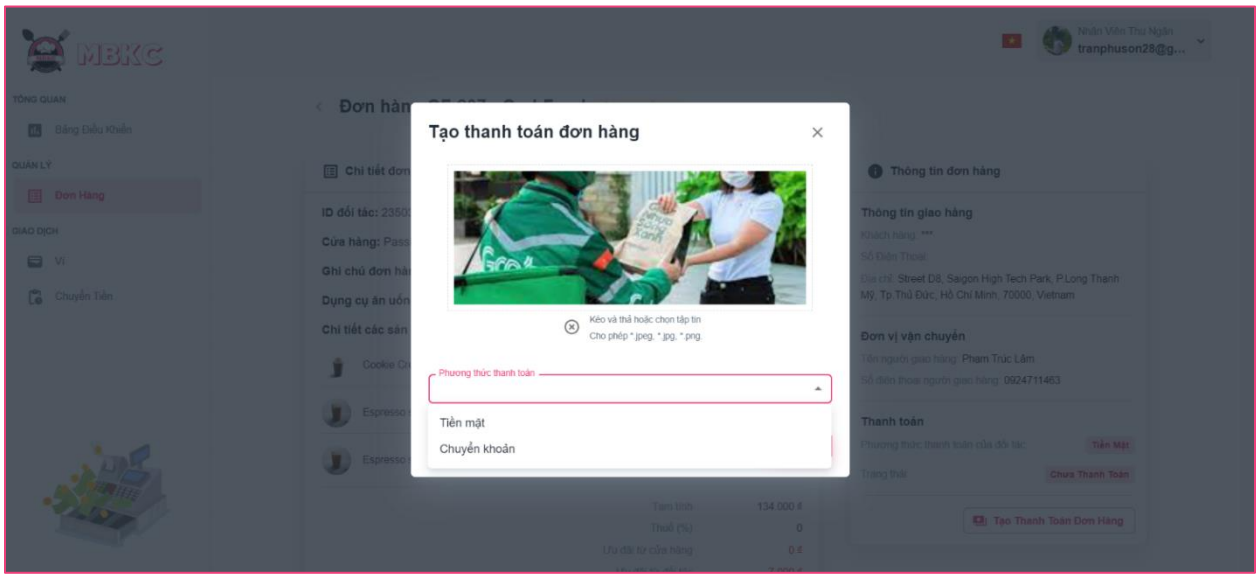


Figure 595: User Manual - Create Shipper Payment – Step 5 - 1

Fill in “Hình ảnh” and “Phương thức thanh toán”. If “Phương thức thanh toán” is “Chuyển khoản” the n you must fill “Tài khoản ngân hàng” to create shipper payment.

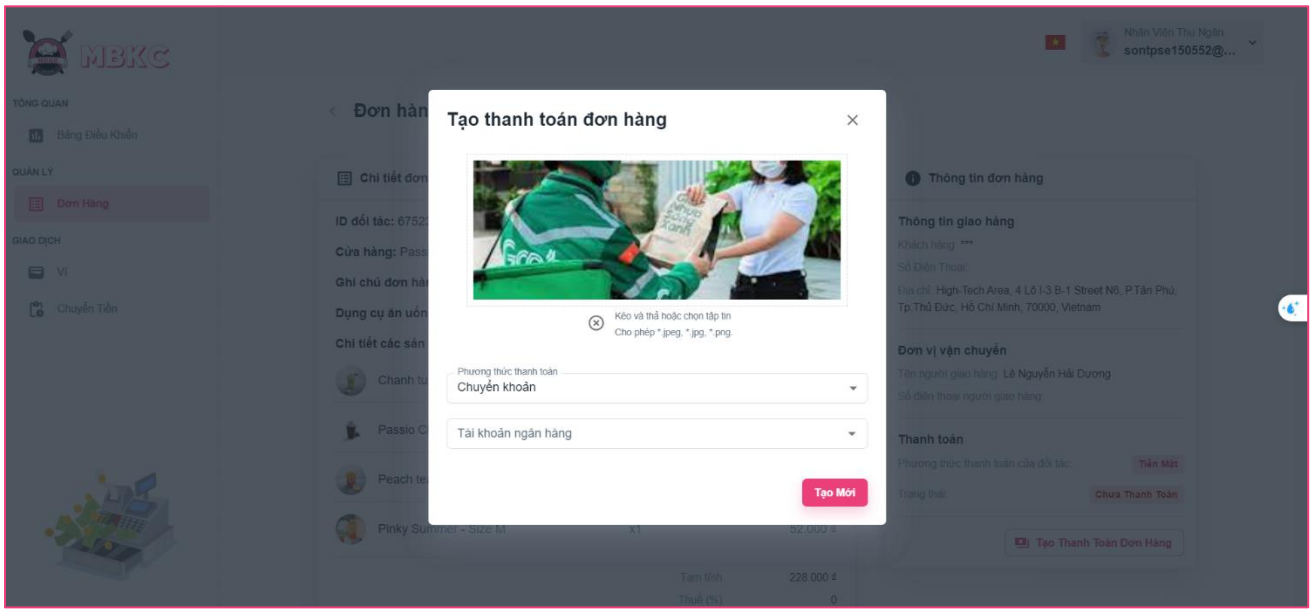


Figure 596: User Manual - Create Shipper Payment – Step 5 – 2

Step 6: Click “Tạo Mới” to create shipper payment.

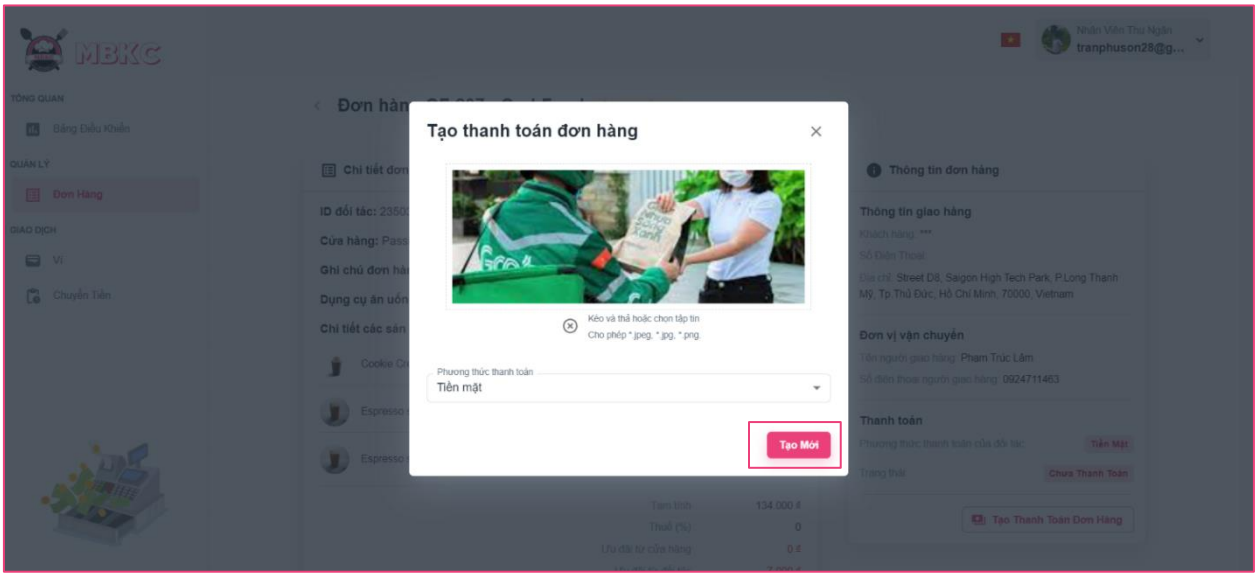


Figure 597: User Manual - Create Shipper Payment – Step 6

Step 7: Order Details Page will be reloaded to show the order details updated and a toast message “Cập nhật đơn hàng thành công”.

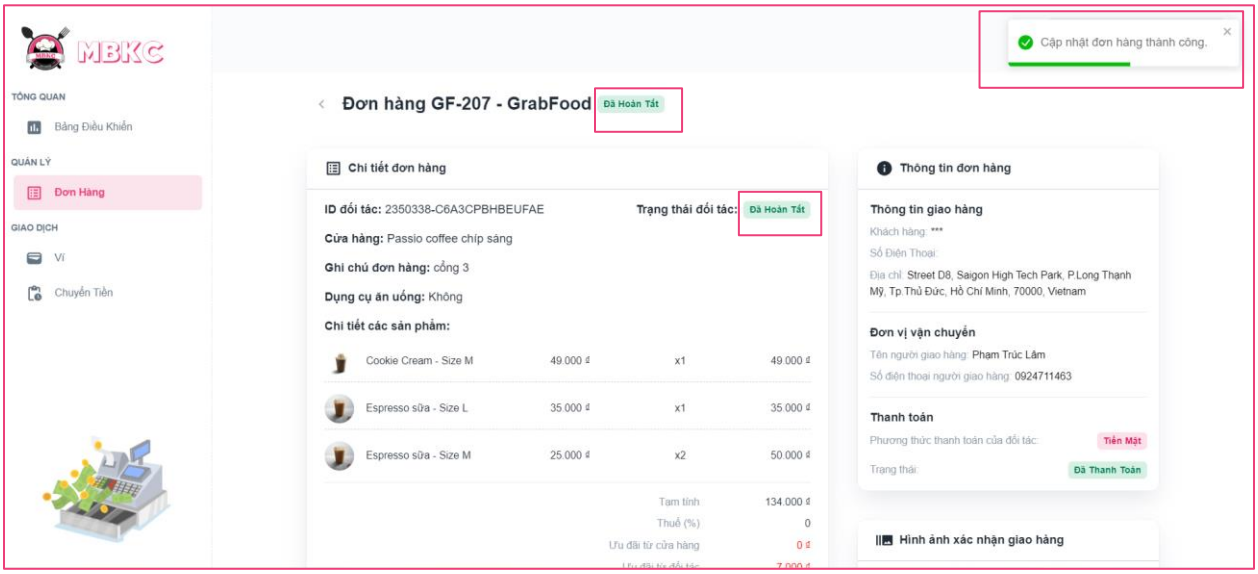


Figure 598: User Manual - Create Shipper Payment – Step 7 - 1

The order history has been updated and a delivery confirmation image has appeared.

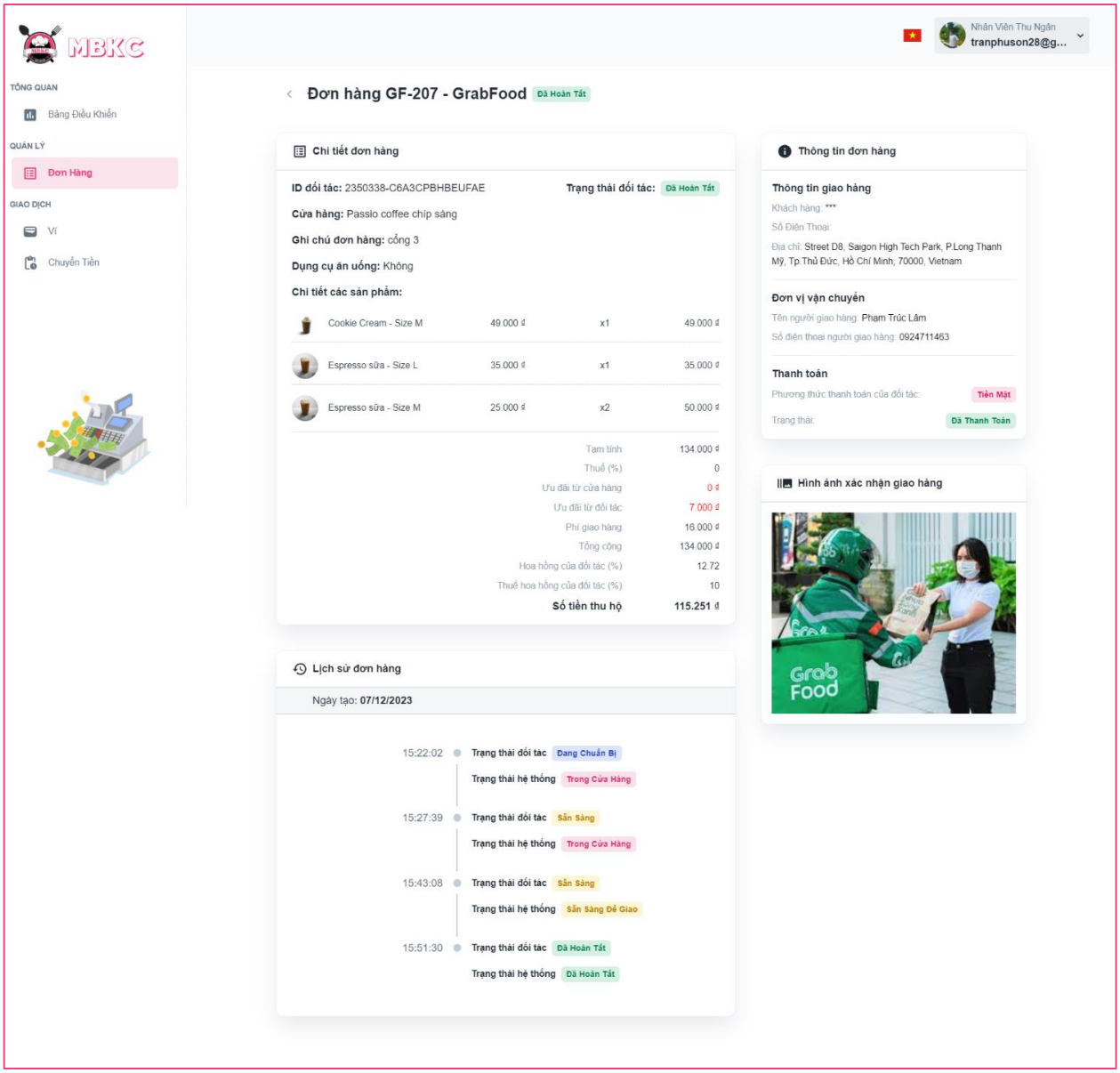


Figure 599: User Manual - Create Shipper Payment – Step 7 - 2

3.2.14. Wallet management

3.2.14.1. View wallet of cashier

3.2.14.1.1. Description

Use this guideline to view wallet of cashier.

This web application supports view wallet of cashier with an account as a Cashier.

3.2.14.1.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Cashier.

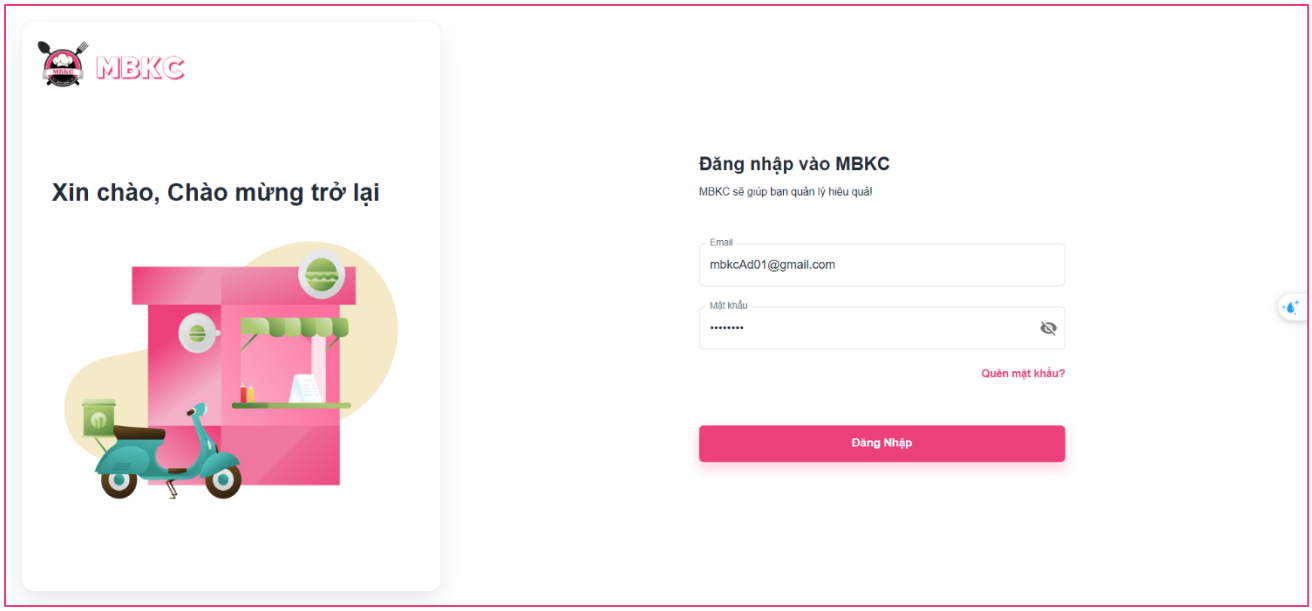


Figure 600: User Manual - View wallet of cashier – Step 1

Step 2: Click “Đăng Nhập” button.

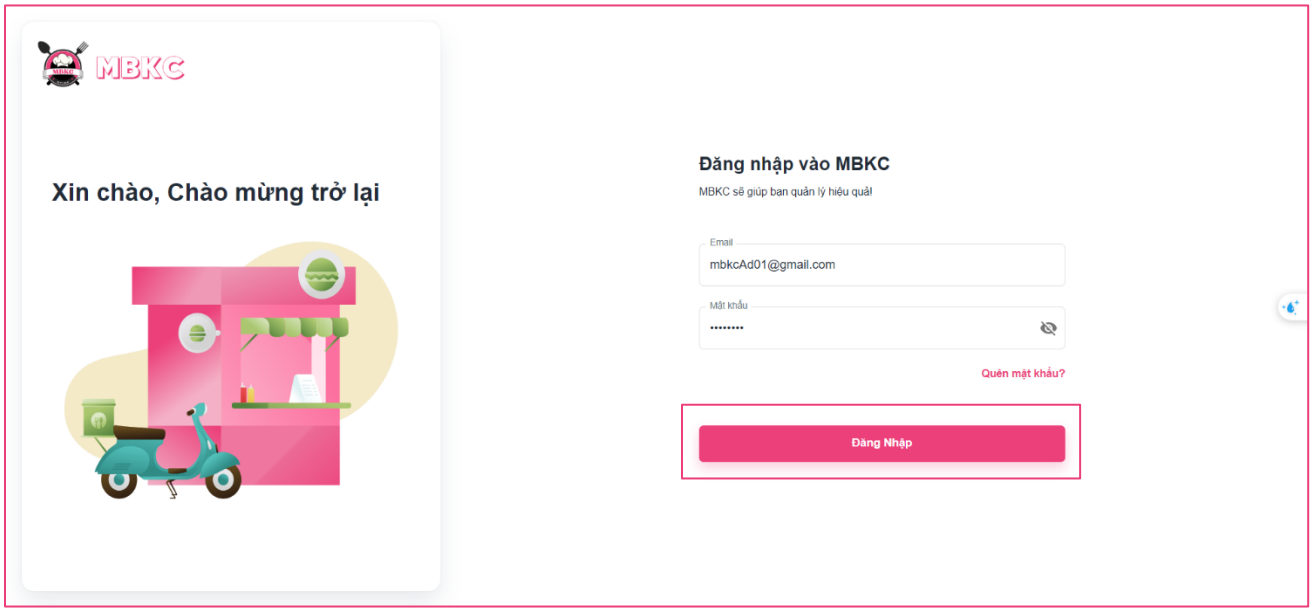


Figure 601: User Manual - View wallet of cashier – Step 2

Step 3: From Cashier Dashboard Page, choose “Ví” from the left sidebar.

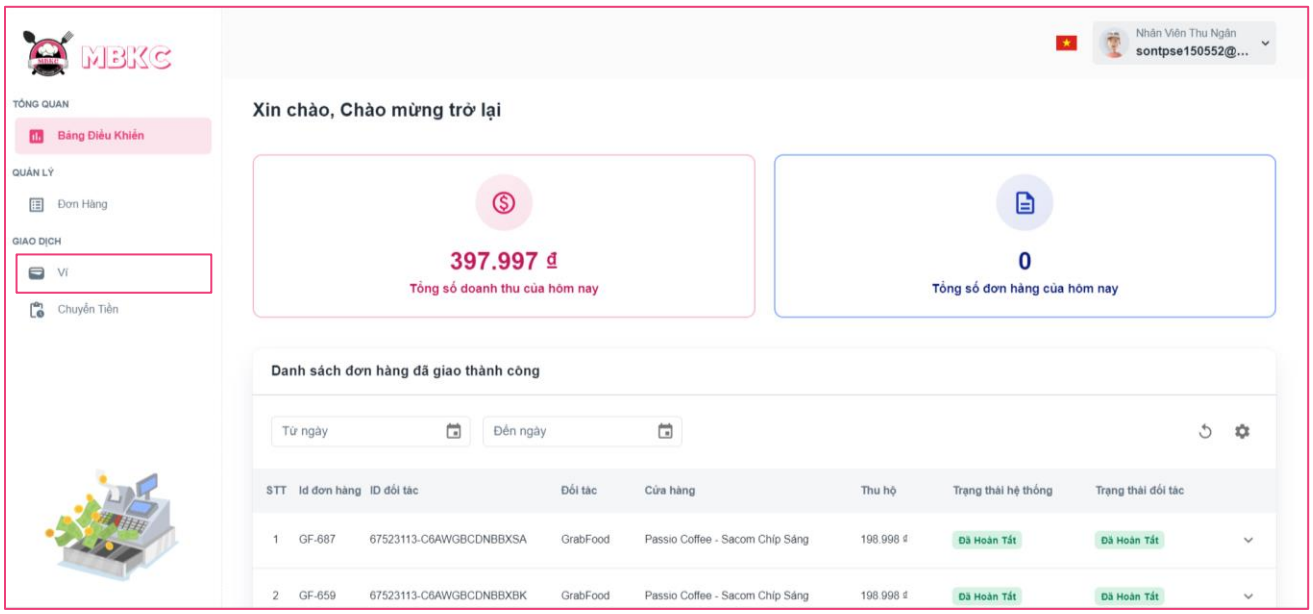


Figure 602: User Manual - View wallet of cashier – Step 3

Step 4: Information wallet of Cashier include: balances in wallet, total daily transactions, total daily shipper payments, list of money exchanges, list of shipper payments.

Case: You can filter by date to view the Cashier's wallet information for the selected date.

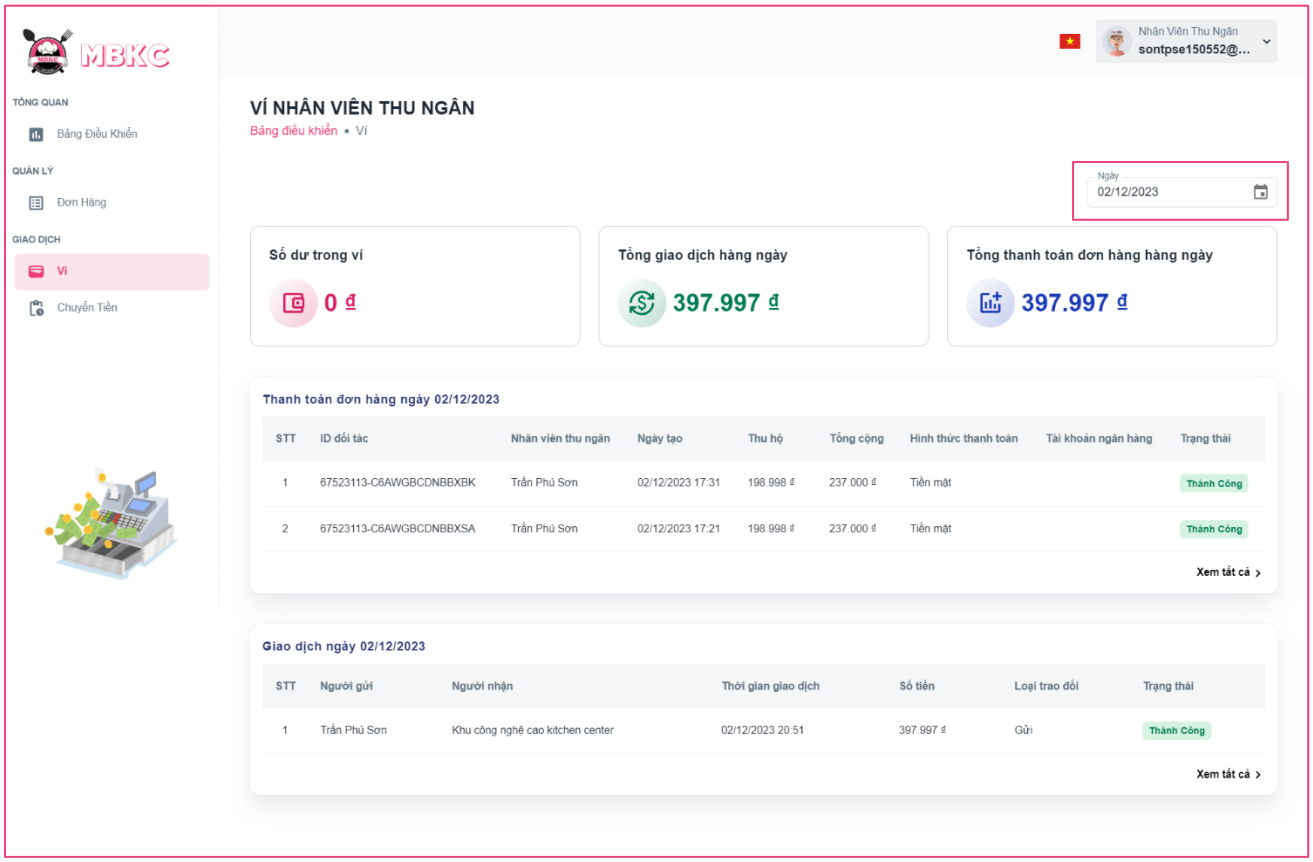


Figure 603: User Manual - View wallet of cashier – Step 4

3.2.14.2. View wallet of kitchen center

3.2.14.2.1. Description

Use this guideline to view wallet of kitchen center.

This web application supports view wallet of kitchen center with an account as a Kitchen Center.

3.2.14.2.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center.

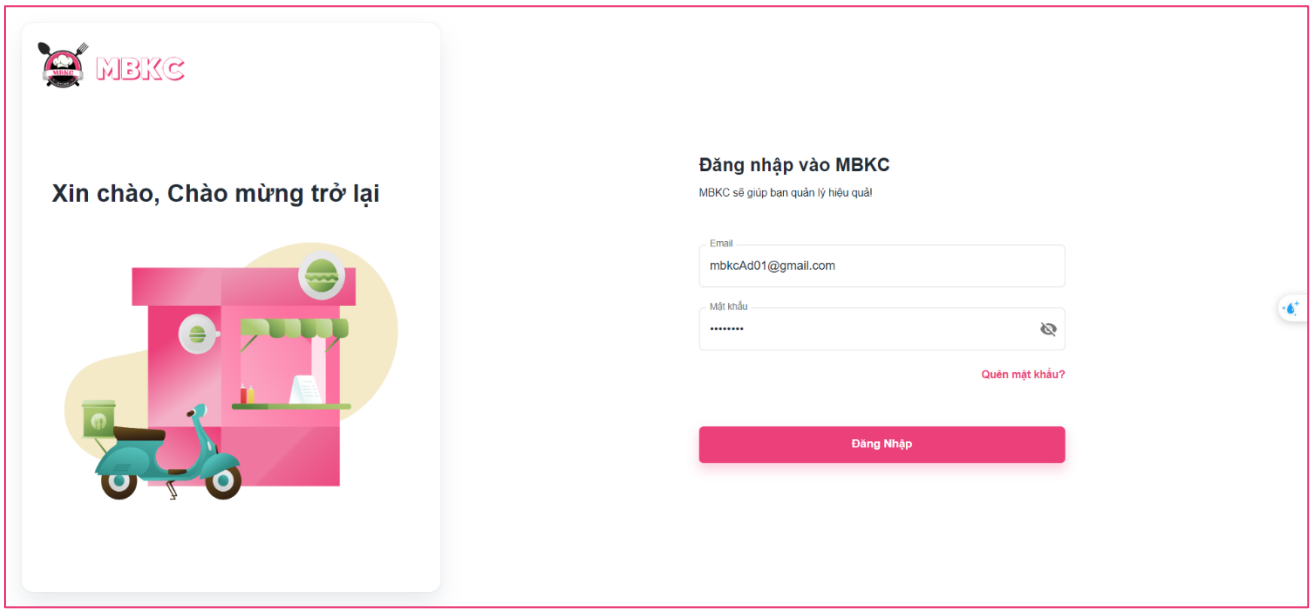


Figure 604: User Manual - View wallet of kitchen center – Step 1

Step 2: Click “Đăng Nhập” button.

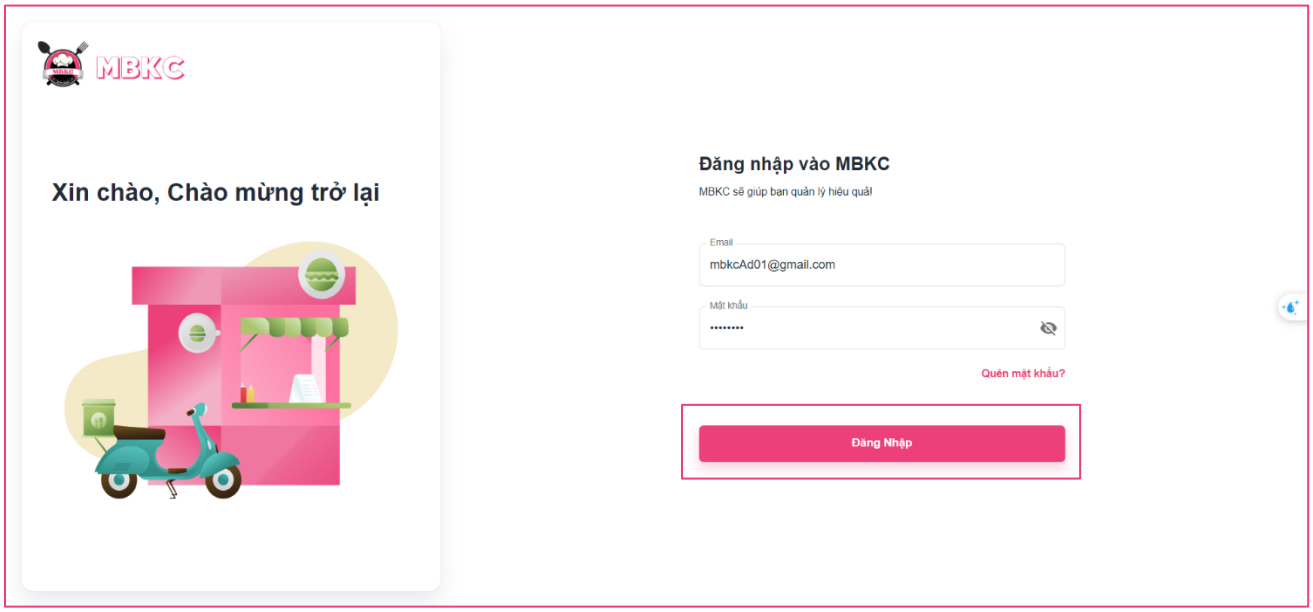


Figure 605: User Manual - View wallet of kitchen center – Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Ví” from the left sidebar.

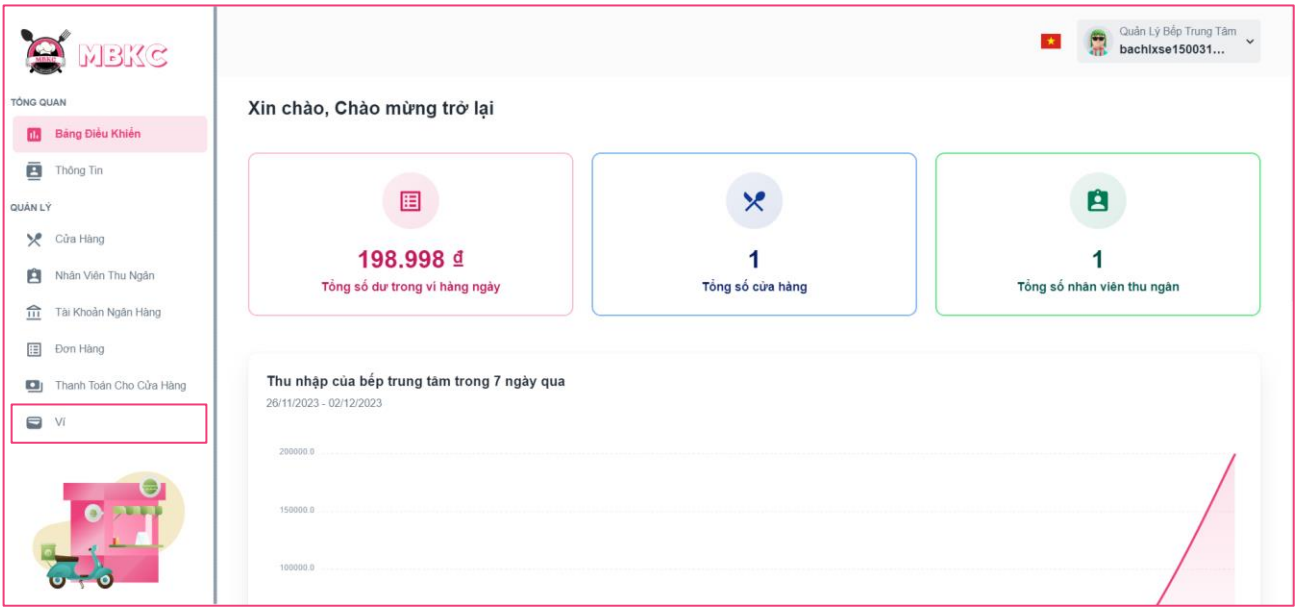


Figure 606: User Manual - View wallet of kitchen center – Step 3

Step 4: Information wallet of Kitchen Center include: balances in wallet, total daily receive, total daily send, list of money exchanges.

Case: You can filter by date to view the Kitchen Center's wallet information for the selected date.

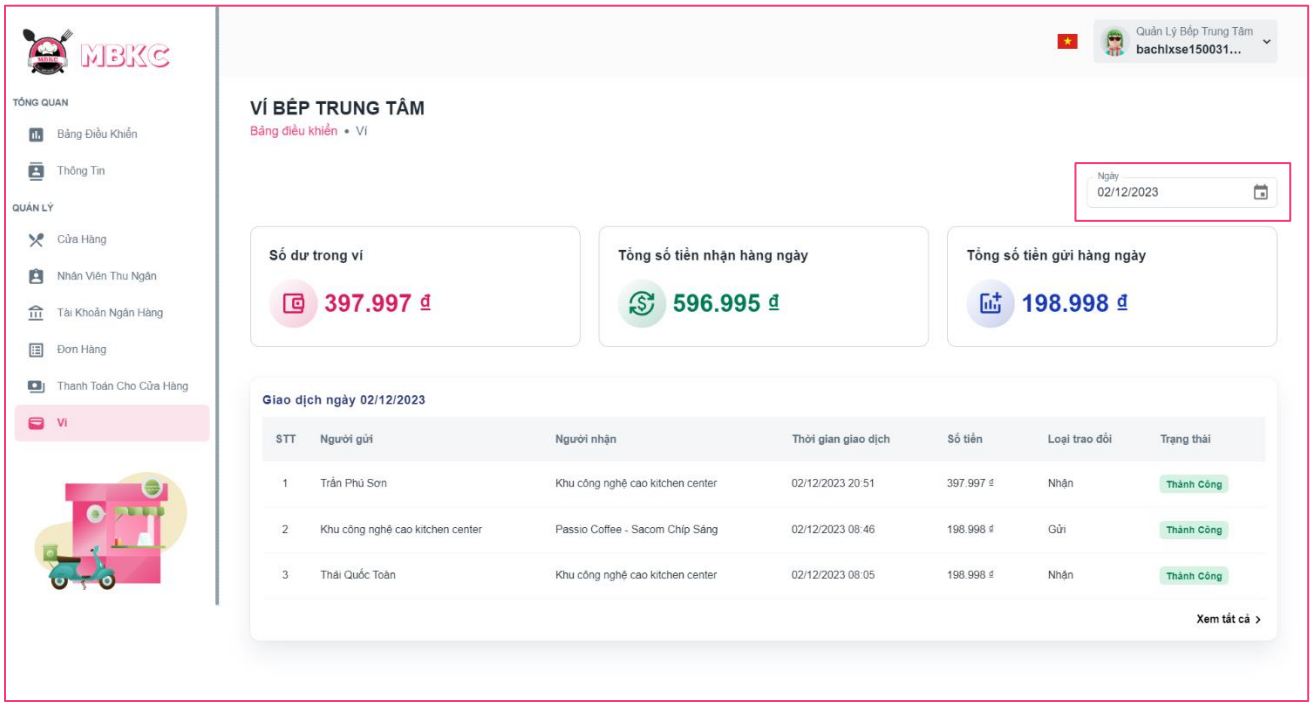


Figure 607: User Manual - View wallet of kitchen center – Step 4

3.2.14.3. View list of transactions

3.2.14.3.1. Description

Use this guideline to view list of transactions.

This web application supports view list of transactions with an account as a Kitchen Center Manager or Cashier.

3.2.14.3.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager or Cashier.

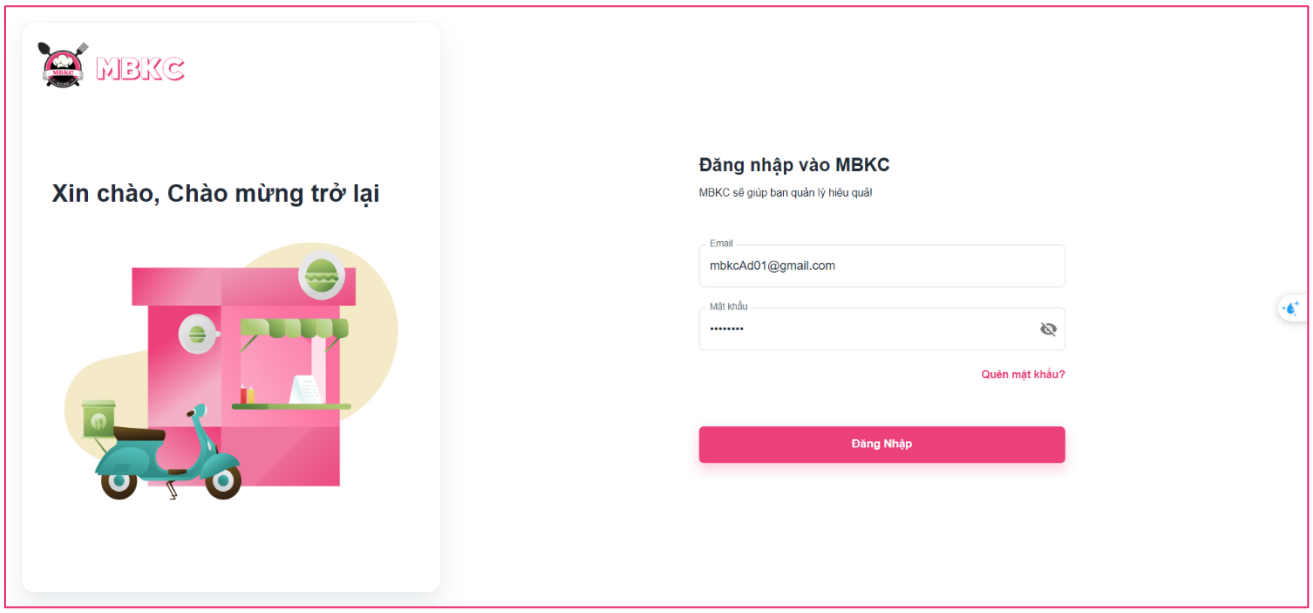


Figure 608: User Manual - View list of transactions – Step 1

Step 2: Click “Đăng Nhập” button.

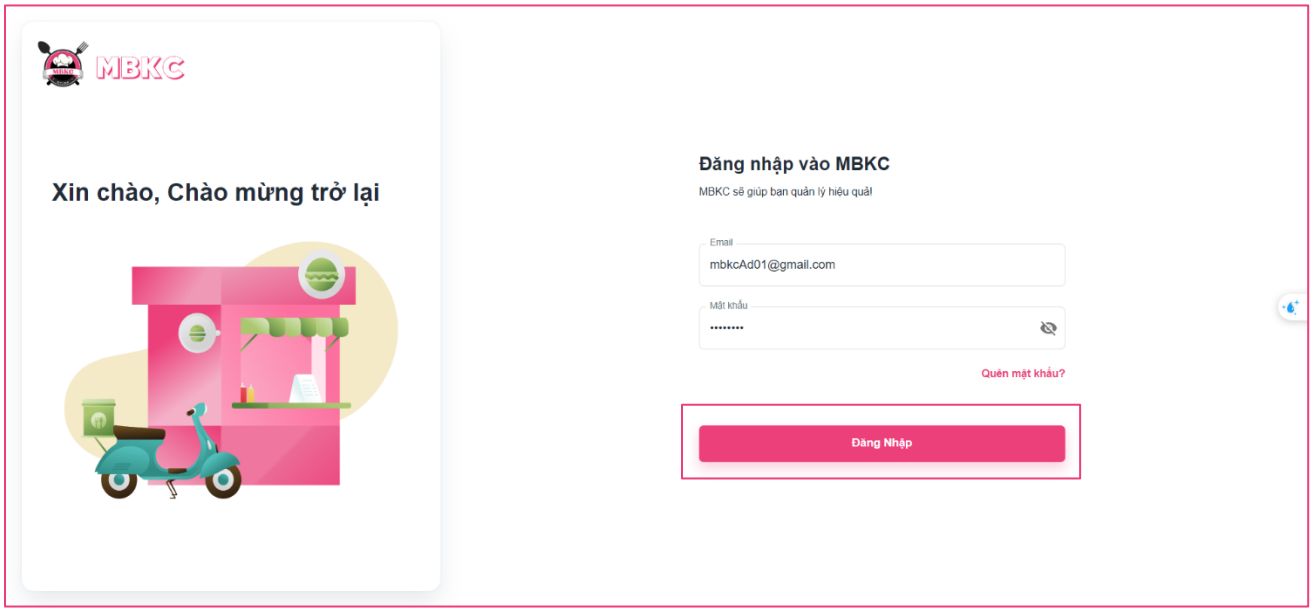


Figure 609: User Manual - View list of transactions – Step 2

Step 3: From Kitchen Center Dashboard Page or Cashier Dashboard Page, choose “Ví” from the left sidebar.

+ Kitchen Center Dashboard Page

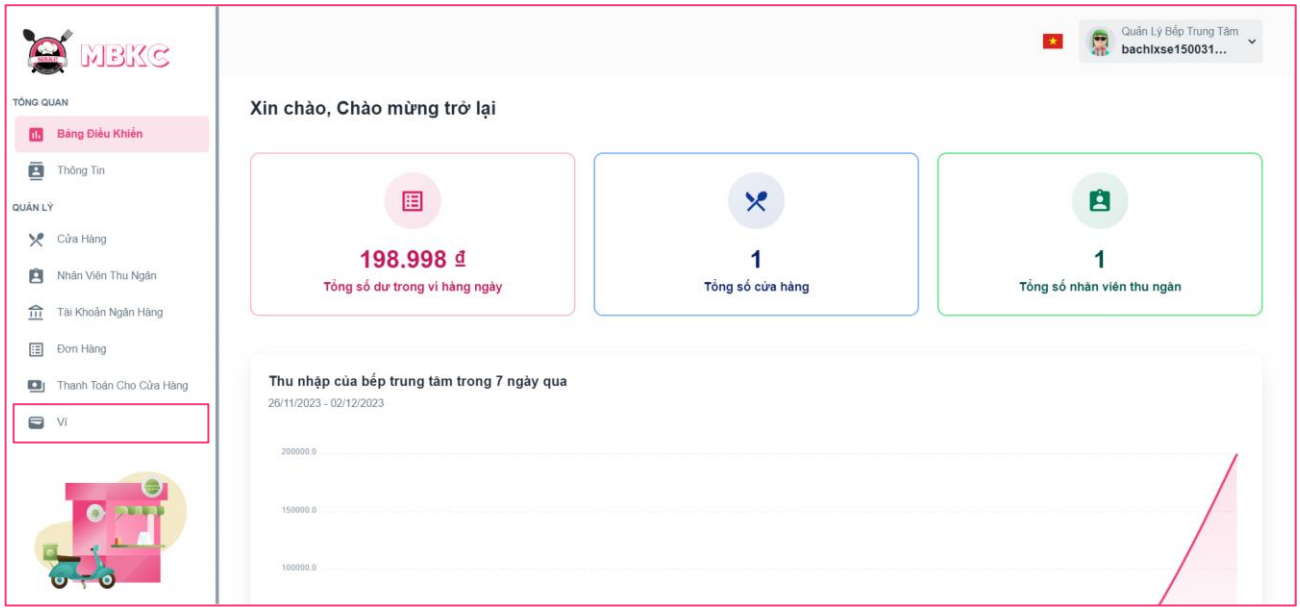


Figure 610: User Manual - View list of transactions – Step 3

+ Cashier Dashboard Page

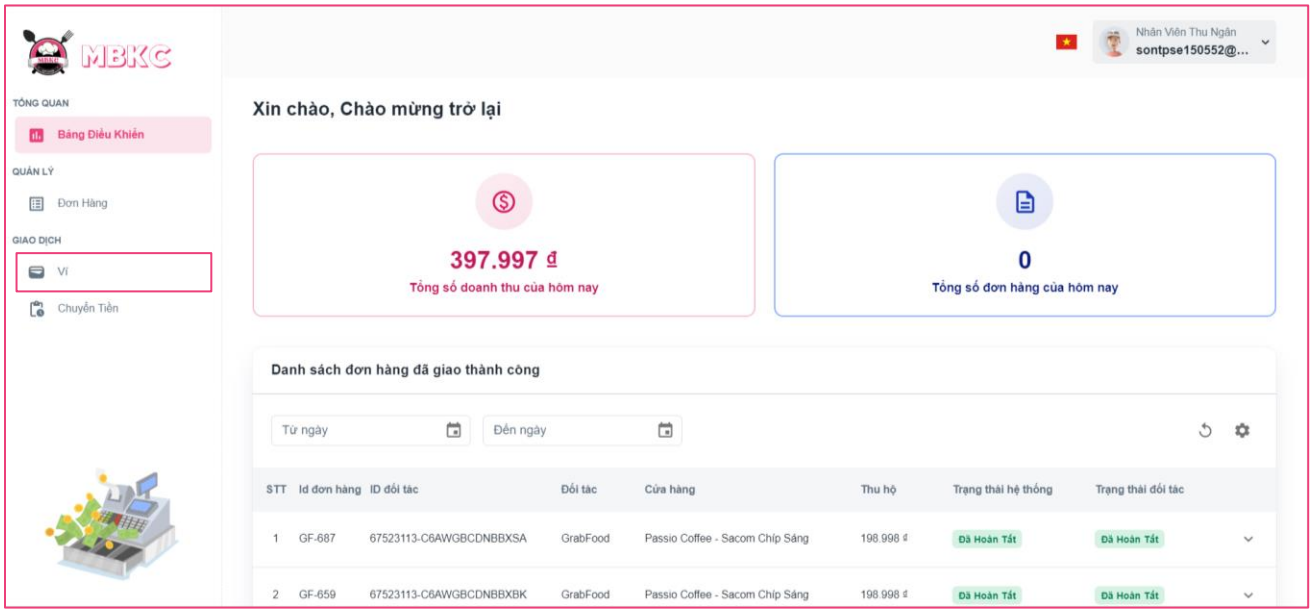


Figure 611: User Manual - View list of transactions – Step 3

Step 4: Press “Xem tất cả” to view list of transactions.

+ Kitchen Center Wallet Page

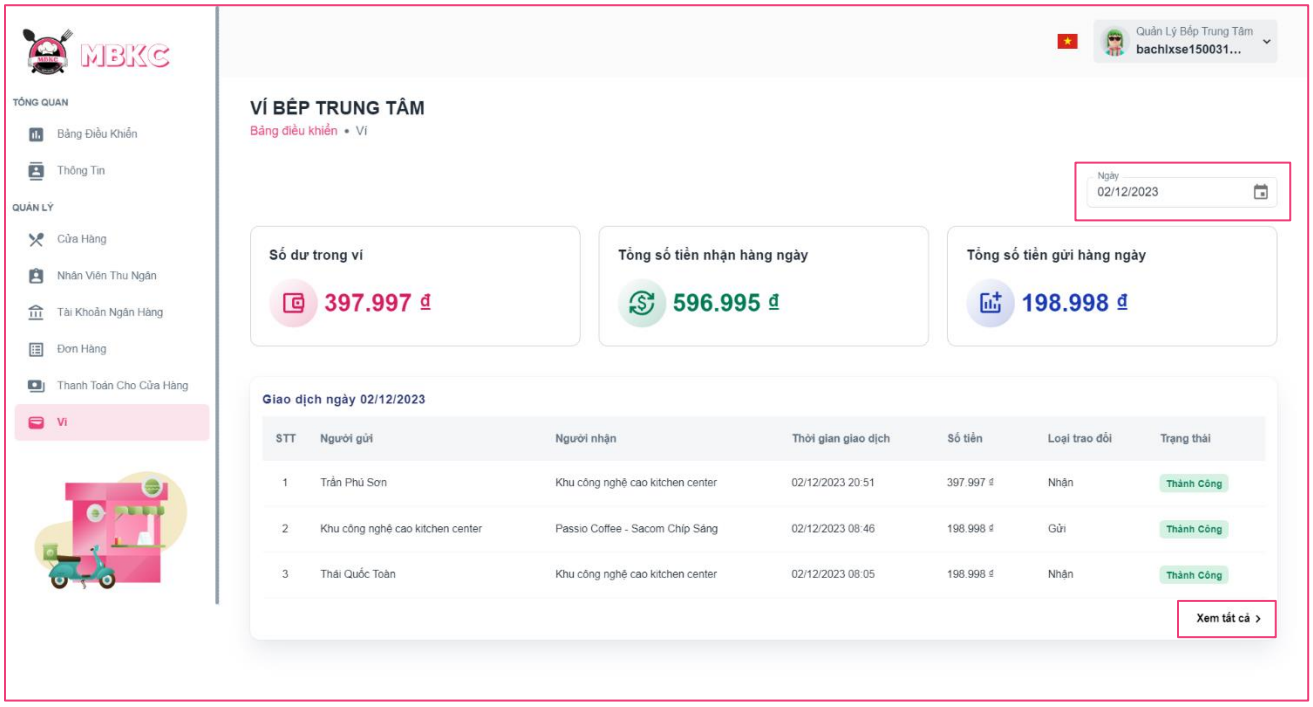


Figure 612: User Manual - View list of transactions – Step 4

+ Cashier Wallet Page

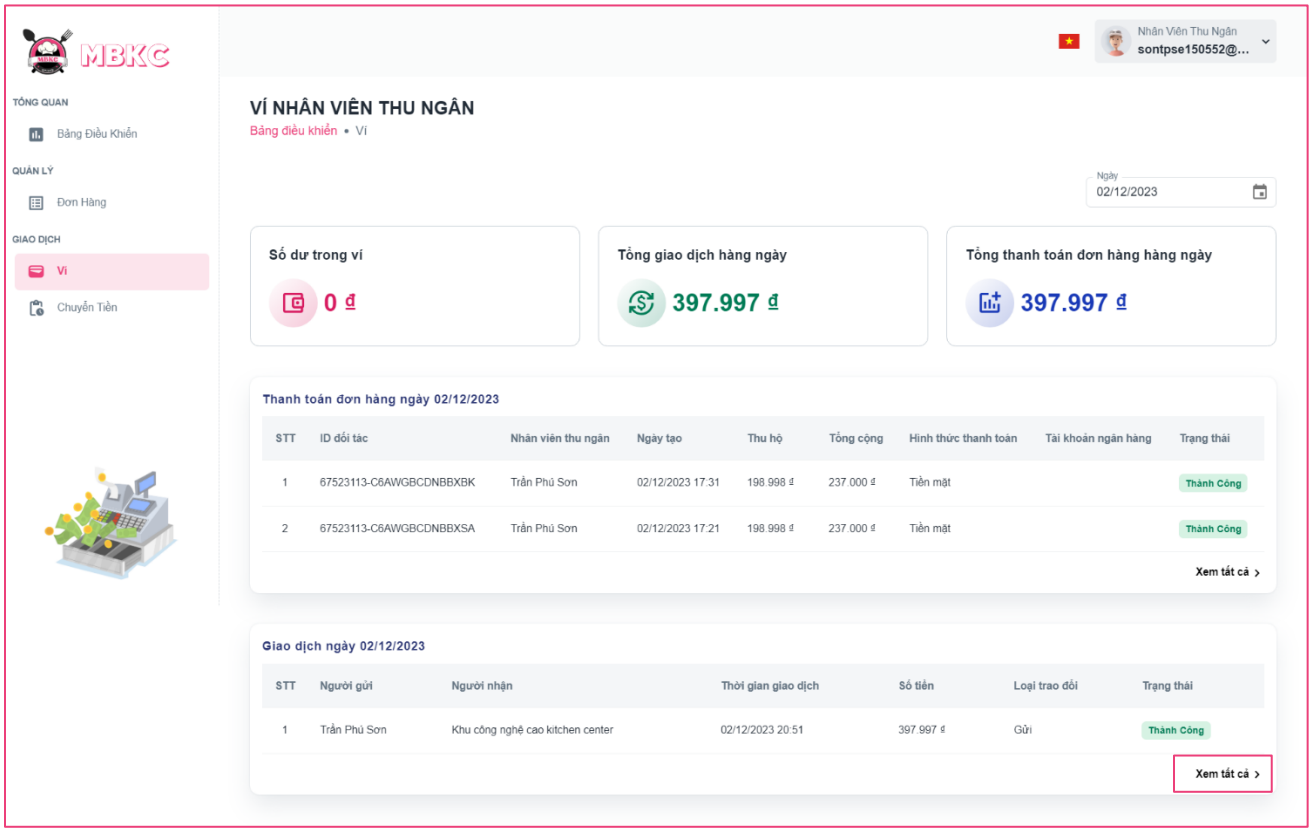


Figure 613: User Manual - View list of transactions – Step 4

Step 5: Navigate to List of Transactions Page.

Case 1: Filter by "Từ ngày" and "Đến ngày" to view a list of transactions by selected date.

+ Filter “Từ ngày” to view list of transactions from the selected date to the present.

+ Filter “Đến ngày” to view a list of transactions from the selected date to earlier.

+ Filter “Từ ngày” and "Đến ngày" to view a list of transactions from “Từ ngày” selected to “Đến ngày” selected.

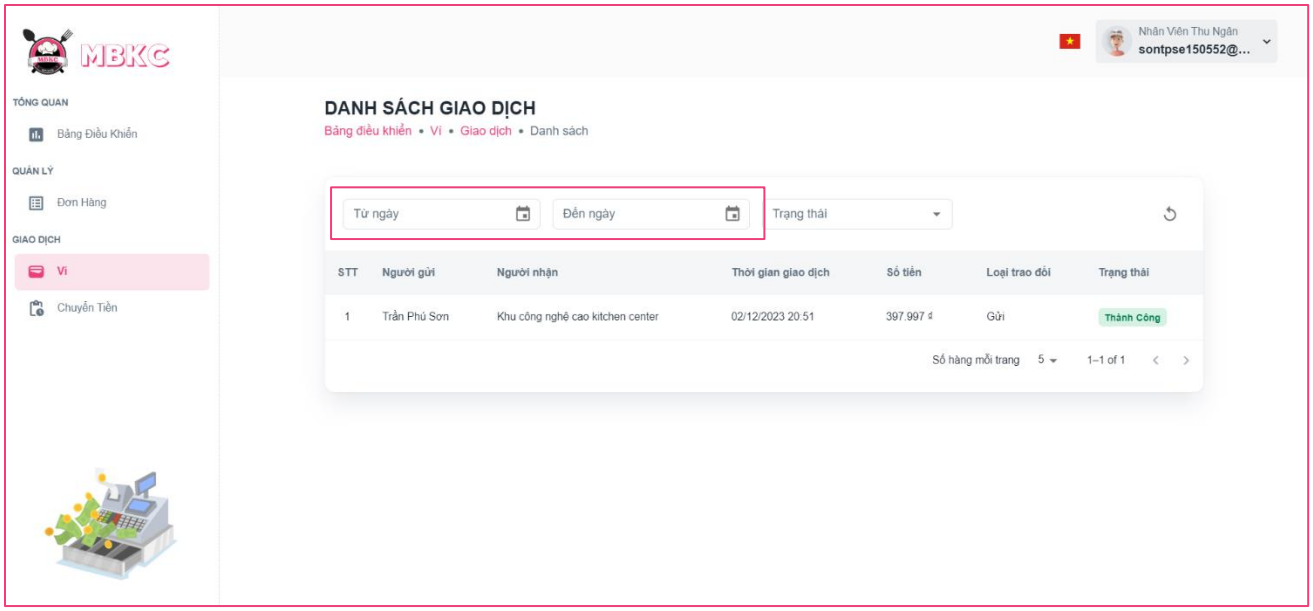


Figure 614: User Manual - View list of transactions – Step 5 – Case 1

Case 2: Filter by “Trạng thái” to view the list of transactions that have status corresponding.

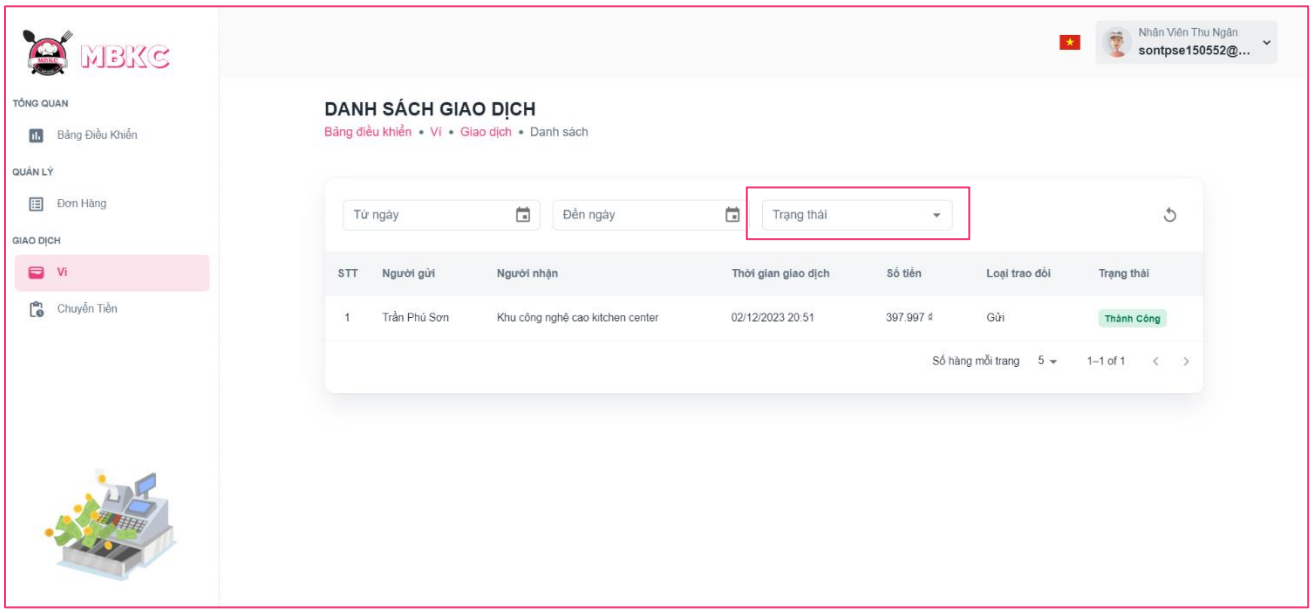


Figure 615: User Manual - View list transactions – Step 5 – Case 2

Case 3: Filter by “Loại trao đổi” to view the list of transactions that have type exchange corresponding. This case support for account of Kitchen Center Manager.

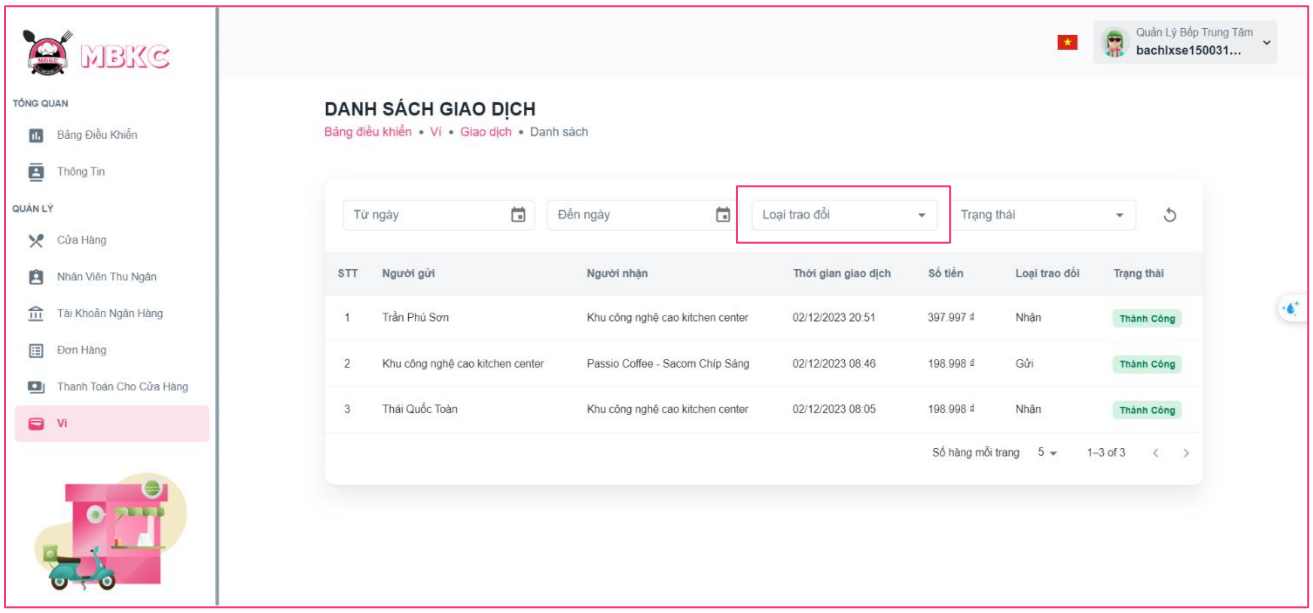


Figure 616: User Manual - View list transactions – Step 5 – Case 3

Step 6: Click on the transaction row to pop up Transaction Details Modal.

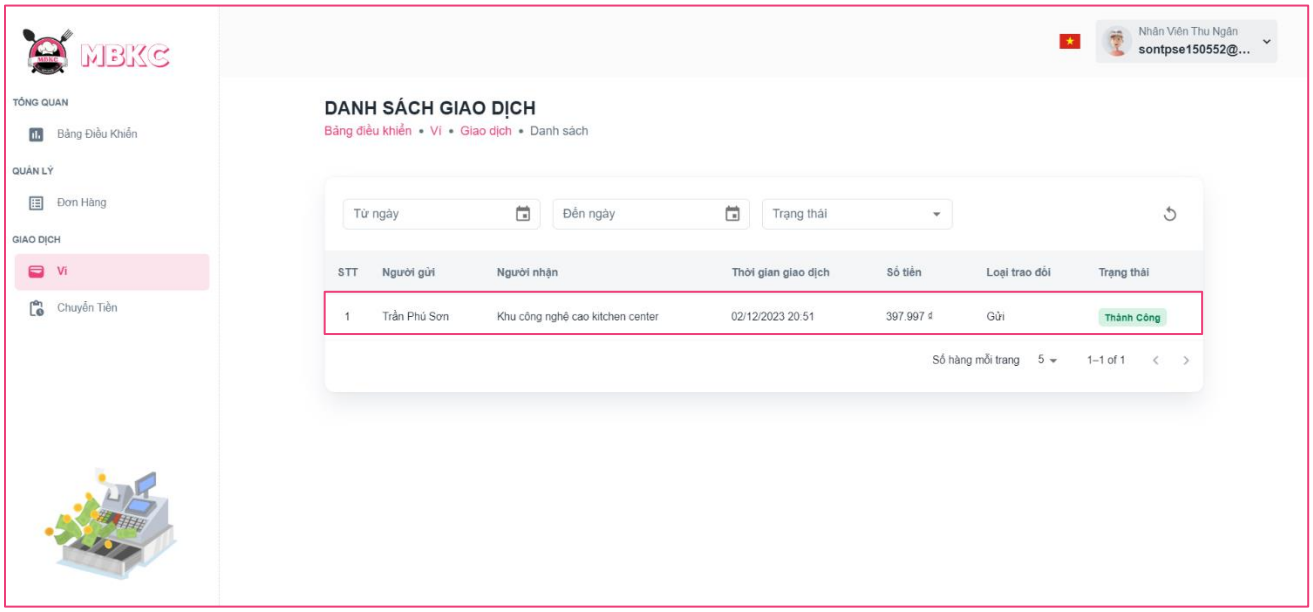


Figure 617: User Manual - View list of transactions - Step 6

Step 7: Pop up Transaction Details Modal.

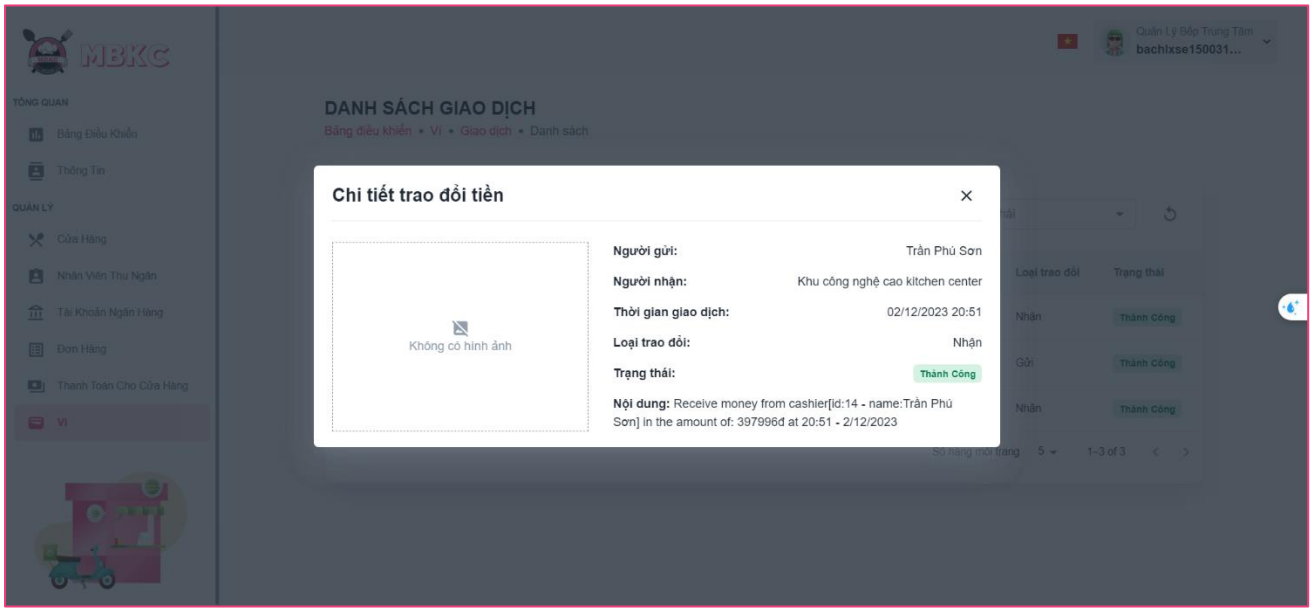


Figure 618: User Manual - View list of transactions - Step 7

3.2.14.4. View list of shipper payments

3.2.14.4.1. Description

Use this guideline to view list of shipper payments.

This web application supports view list of shipper payments with an account as a Cashier.

3.2.14.4.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Cashier.

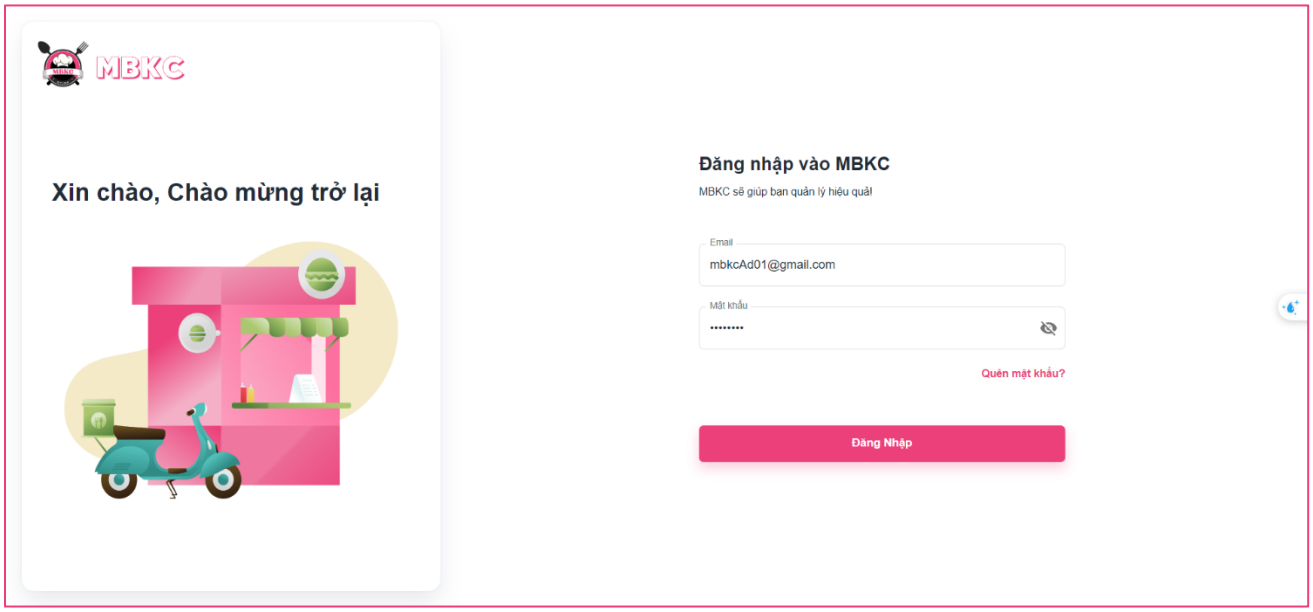


Figure 619: User Manual - View list of shipper payments – Step 1

Step 2: Click “Đăng Nhập” button.

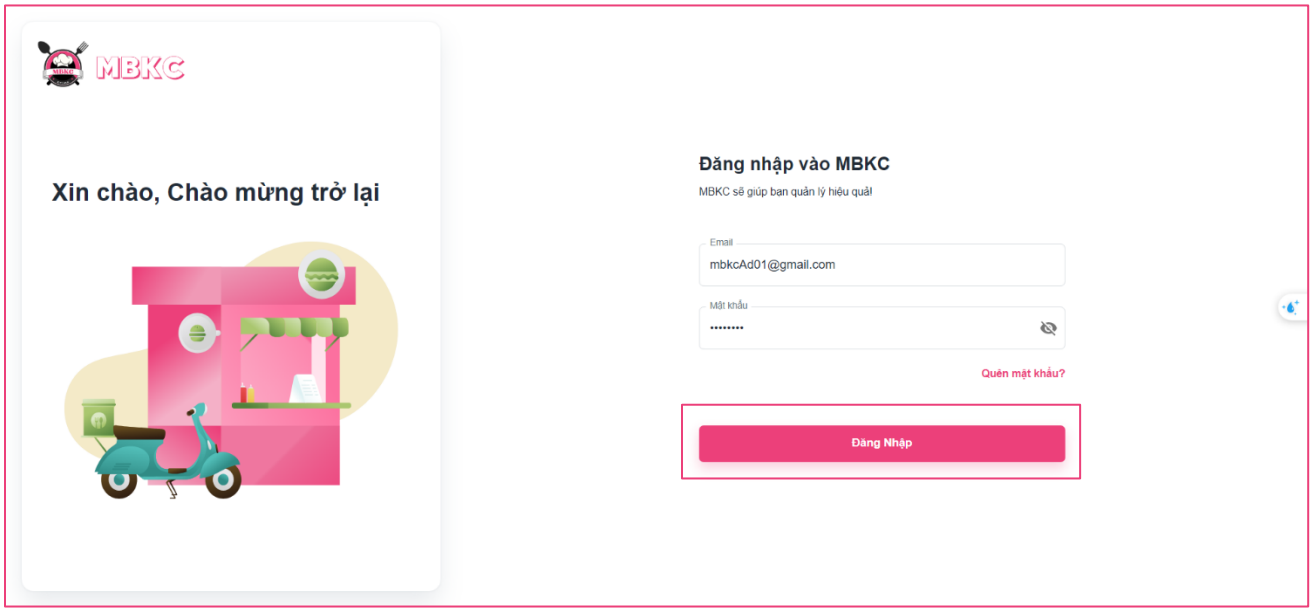


Figure 620: User Manual - View list of shipper payments – Step 2

Step 3: From Cashier Dashboard Page, choose “Ví” from the left sidebar.

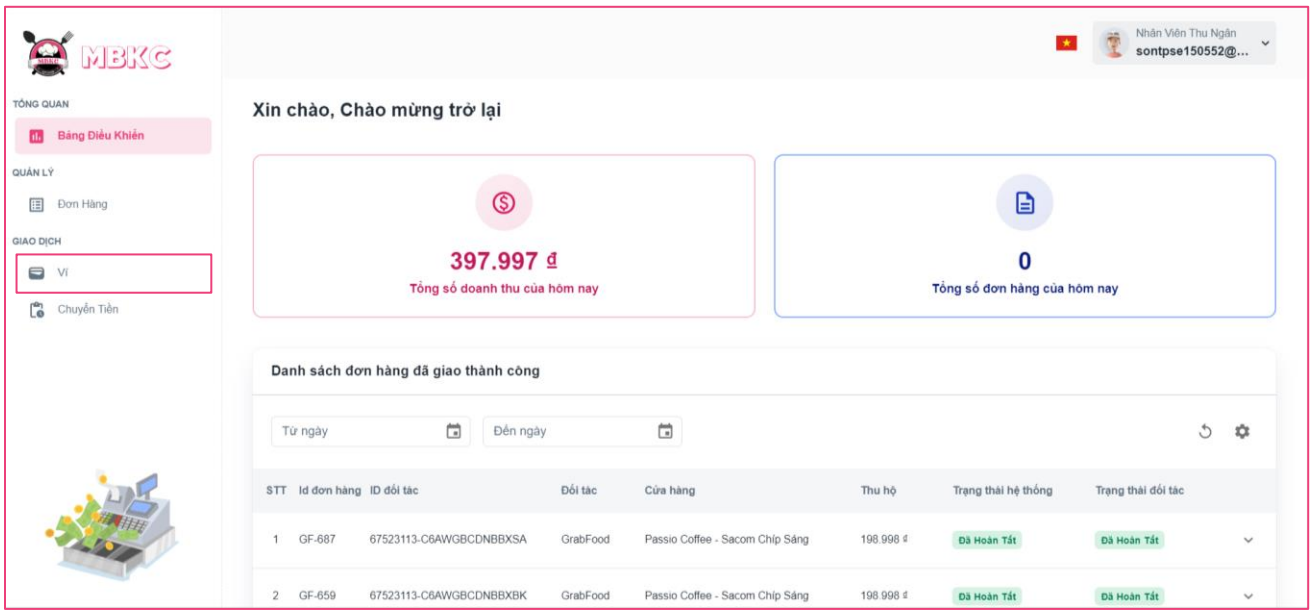


Figure 621: User Manual - View list of shipper payments – Step 3

Step 4: Press “Xem tất cả” to view list of shipper payments.

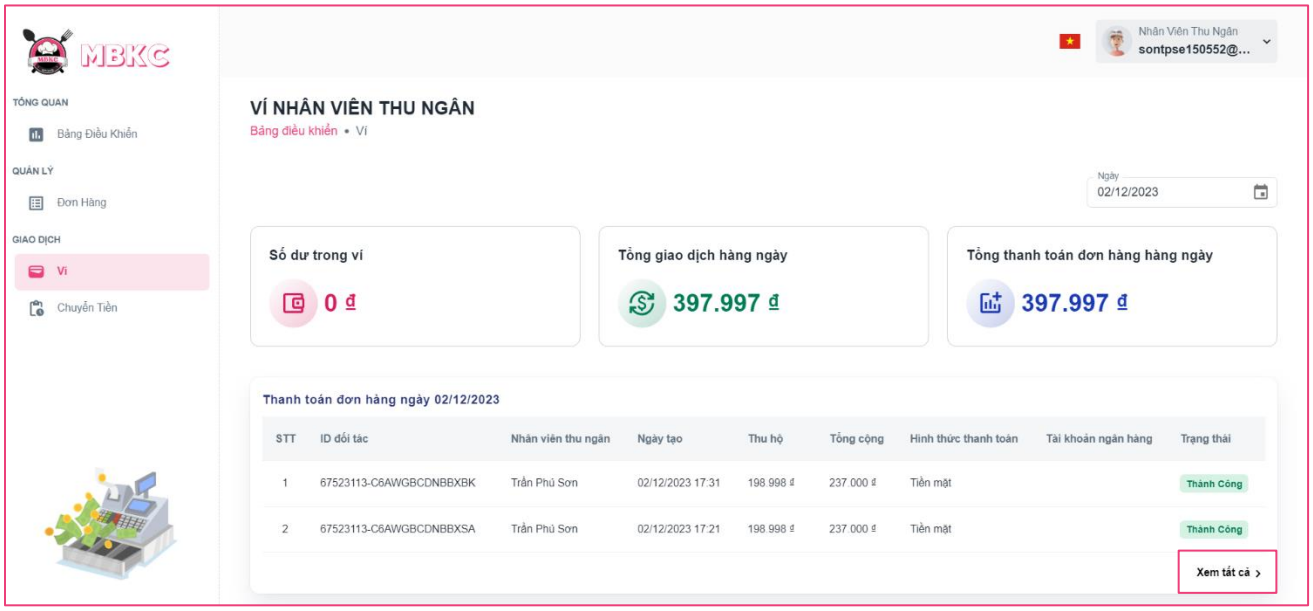


Figure 622: User Manual - View list of shipper payments – Step 4

Step 5: Navigate to List of Shipper Payments Page.

Case 1: Filter by "Từ ngày" and "Đến ngày" to view a list of shipper payments by selected date.

+ Filter “Từ ngày” to view list of shipper payments from the selected date to the present.

+ Filter “Đến ngày” to view a list of shipper payments from the selected date to earlier.

+ Filter “Từ ngày” and "Đến ngày" to view a list of shipper payments from “Từ ngày” selected to “Đến ngày” selected.

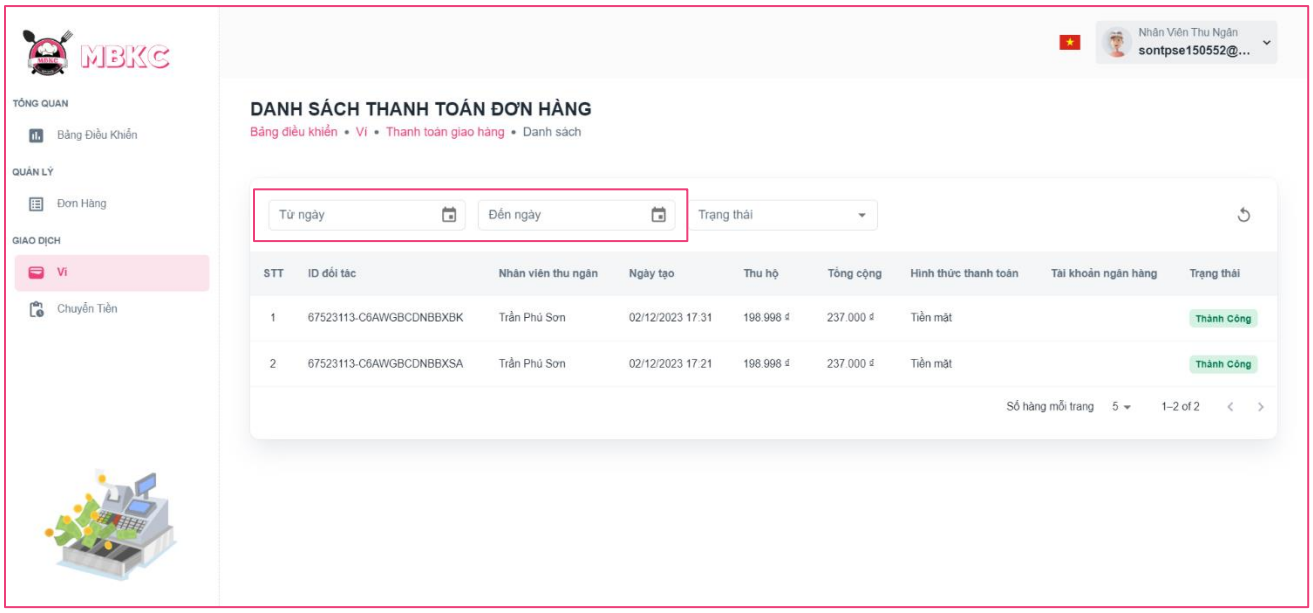


Figure 623: User Manual - View list of shipper payments – Step 5 – Case 1

Case 2: Filter by “Trạng thái” to view the list of shipper payments that have status corresponding.

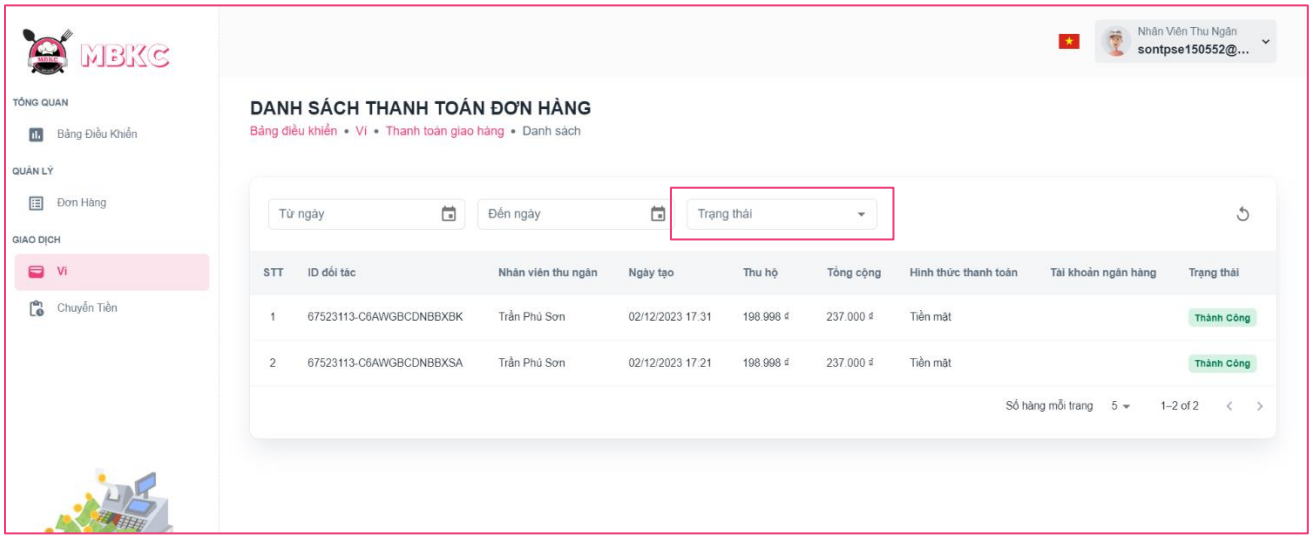


Figure 624: User Manual - View list of shipper payments – Step 5 – Case 2

Step 6: Click on the shipper payment row to pop up Shipper Payment Details Modal.

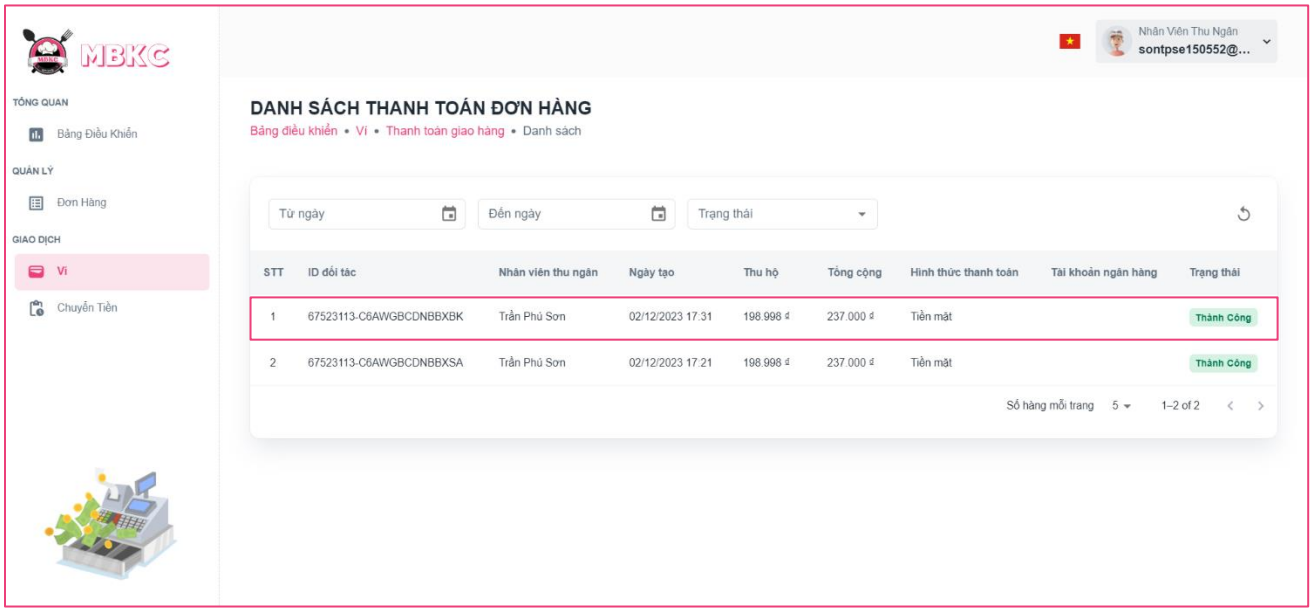


Figure 625: User Manual - View list of shipper payments - Step 6

Step 7: Pop up Shipper Payment Details Modal.

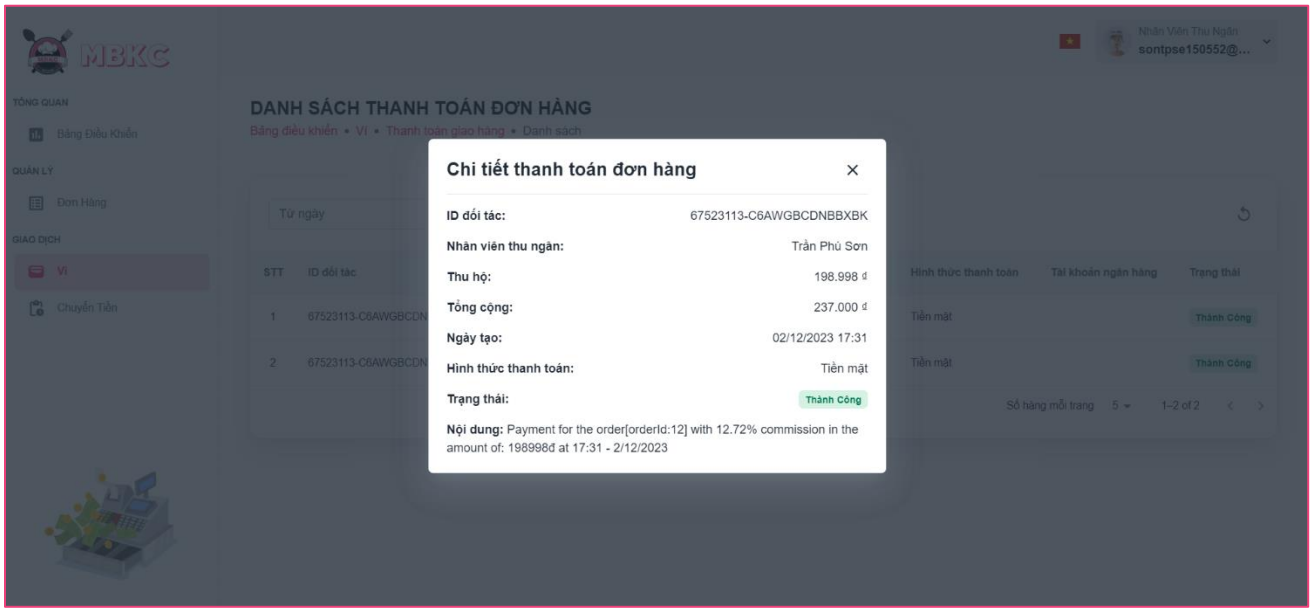


Figure 626: User Manual - View list of shipper payments - Step 7

3.2.14.5. Transfer Money to Kitchen Center

3.2.14.5.1. Description

Use this guideline to transfer money to kitchen center.

This web application supports transfer money to kitchen center with an account as a Cashier.

3.2.14.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Cashier.

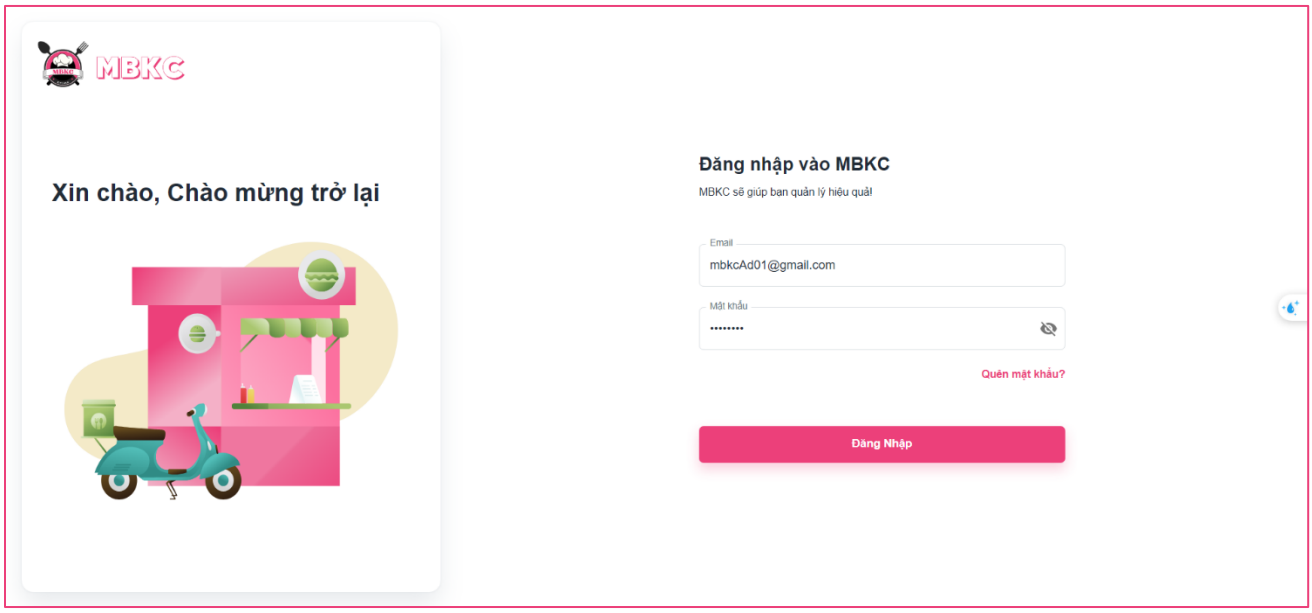


Figure 627: User Manual – Transfer Money to Kitchen Center – Step 1

Step 2: Click “Đăng Nhập” button.

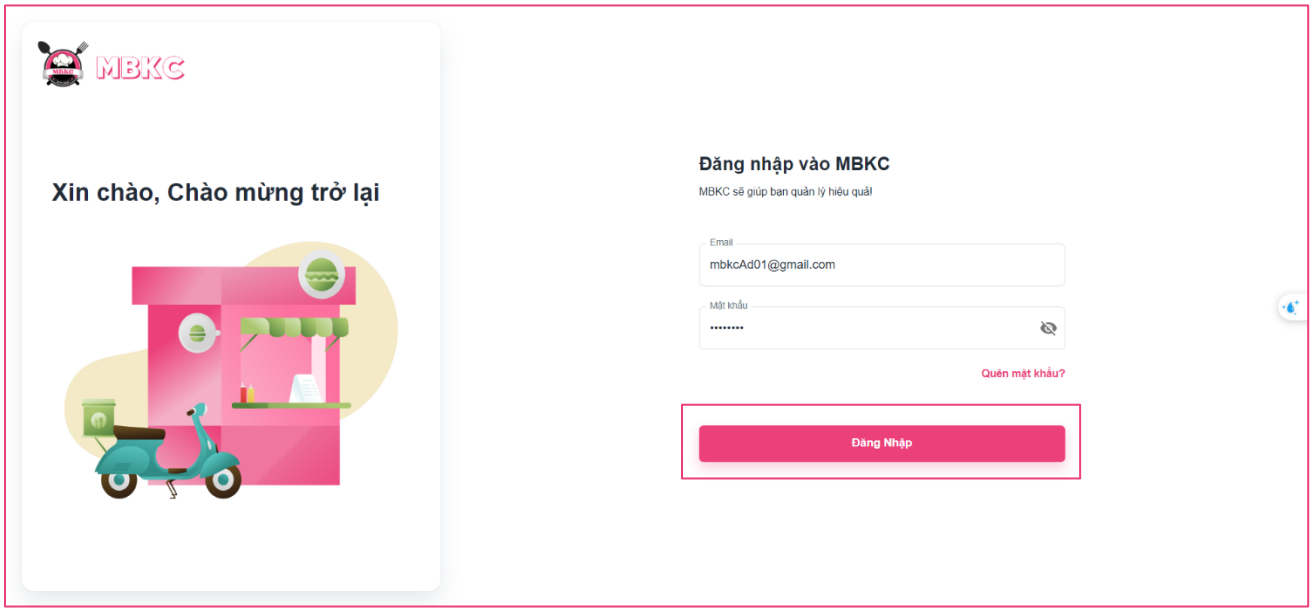


Figure 628: User Manual - Transfer Money to Kitchen Center - Step 2

Step 3: From Cashier Dashboard Page, choose “Ví” from the left sidebar.

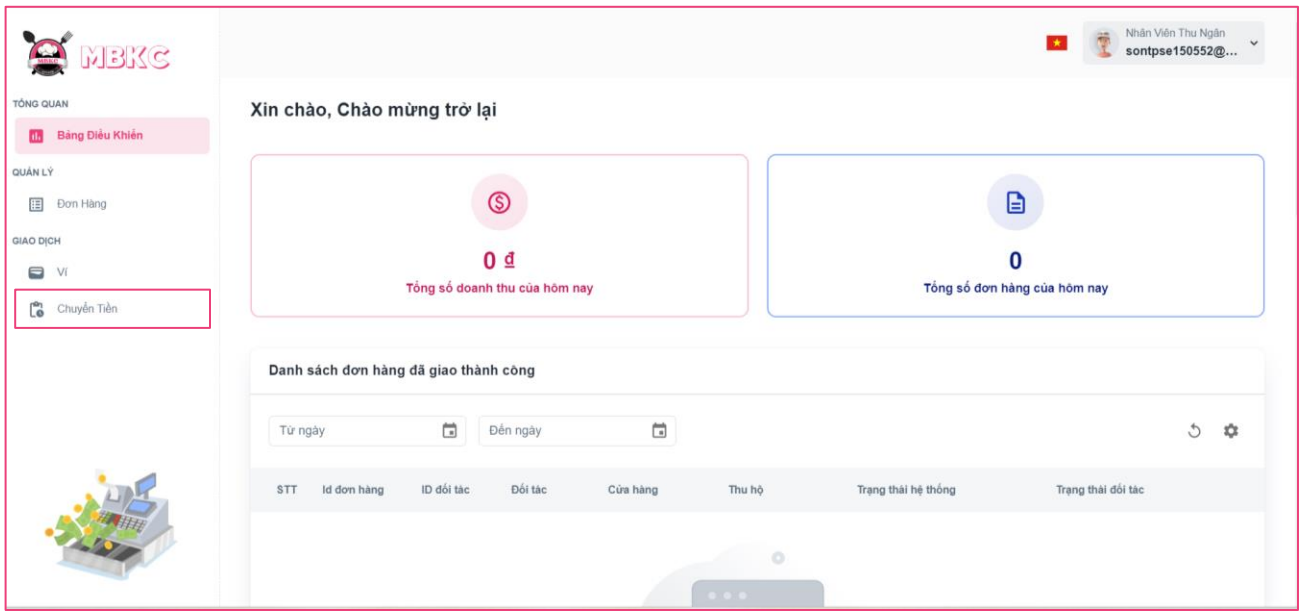


Figure 629: User Manual - Transfer Money to Kitchen Center - Step 3

Step 4: Press “Chi tiết” to show list of orders that cashier delivered today. And click “Xác nhận chuyển tiền” button.

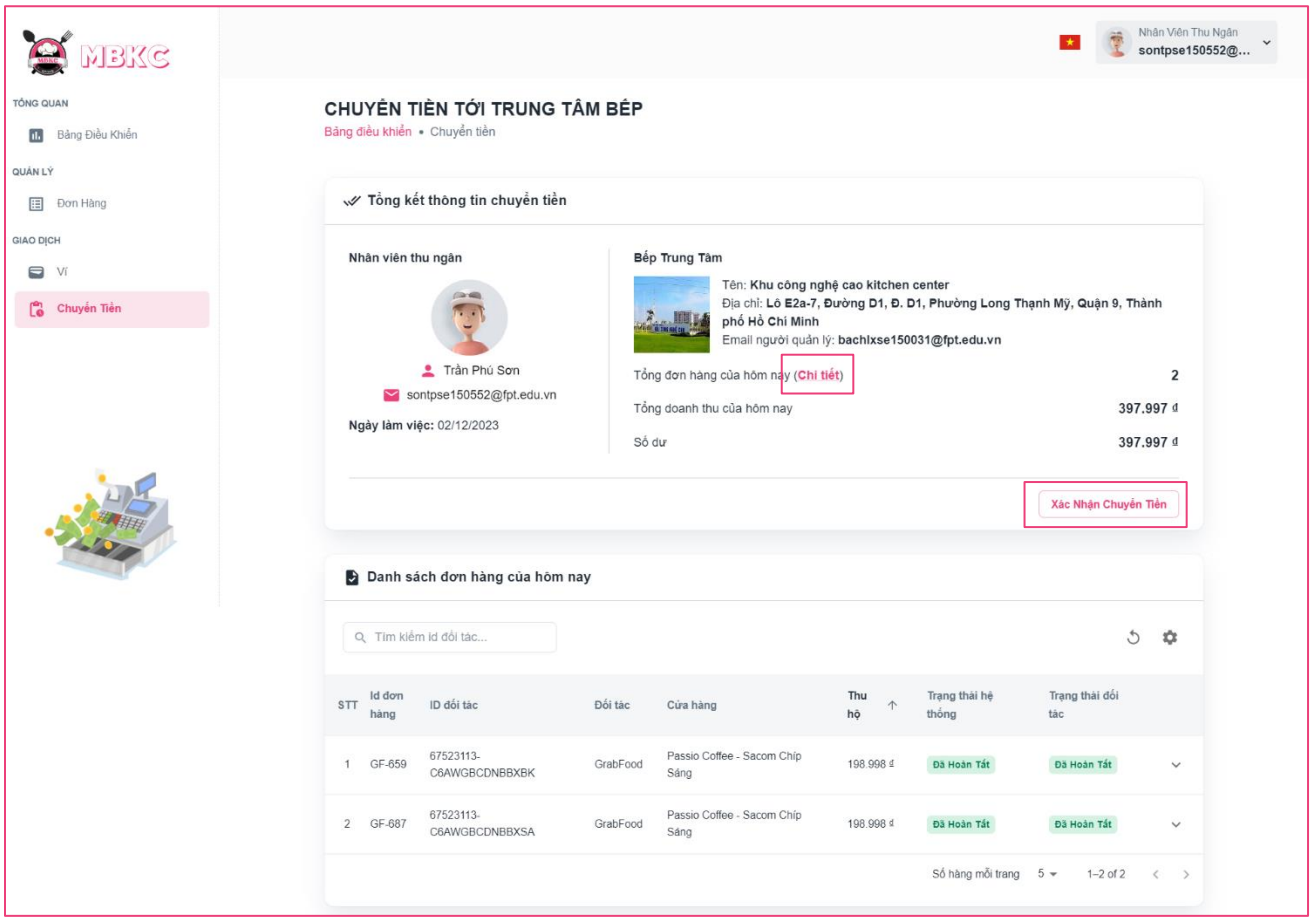


Figure 630: User Manual - Transfer Money to Kitchen Center - Step 4

Step 5: Click “Xác nhận” button.

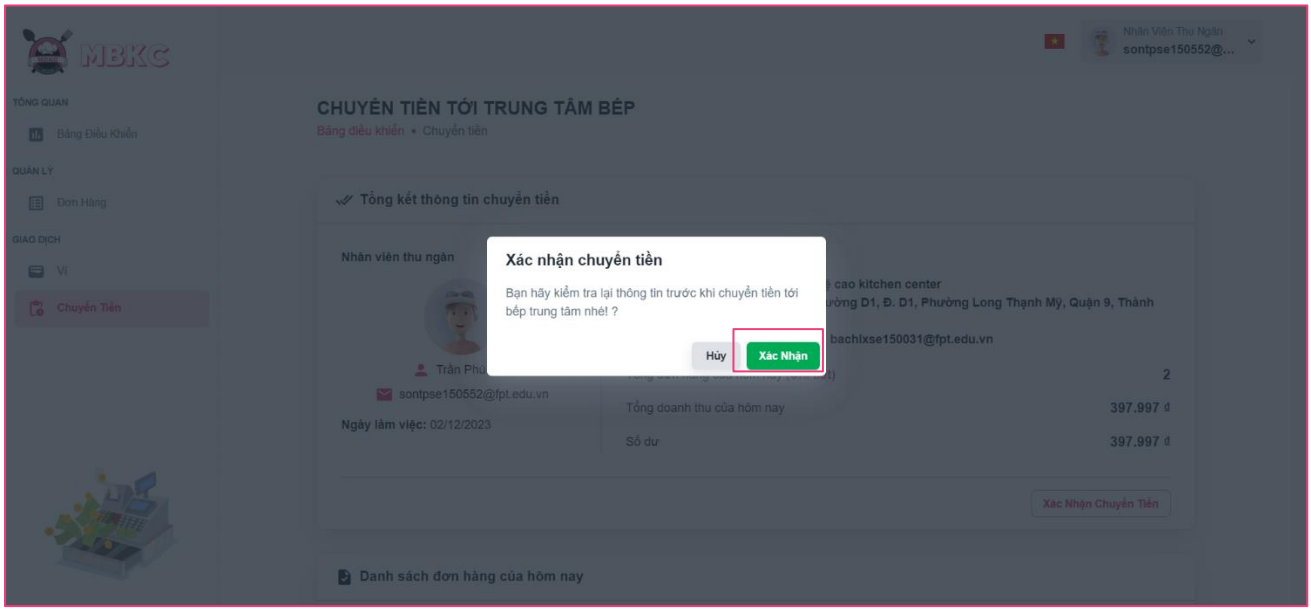


Figure 631: User Manual - Transfer Money to Kitchen Center - Step 5

Step 6: Transfer Money to Kitchen Center Page will be reloaded to show the information details updated and a toast message “Đã gửi tiền tới thành công”.

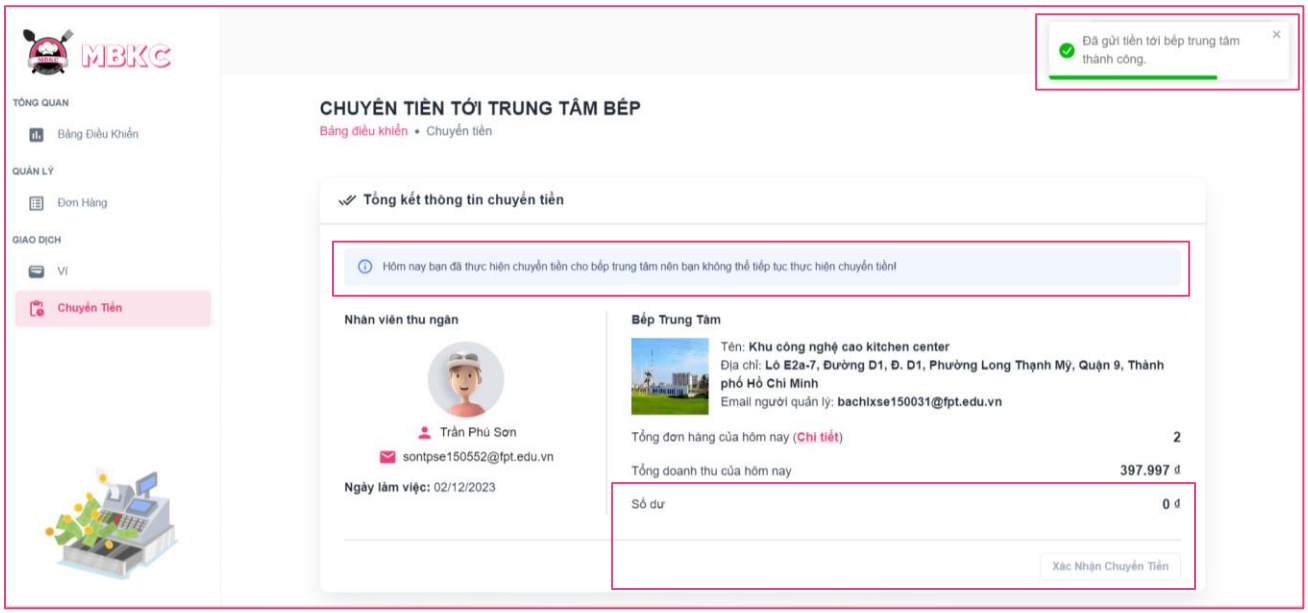


Figure 632: User Manual - Transfer Money to Kitchen Center - Step 6

3.2.14.6. Create Payment for Store

3.2.14.5.1. Description

Use this guideline to create payment for store.

This web application supports create payment for store with an account as a Kitchen Center Manager.

3.2.14.5.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account Kitchen Center Manager.

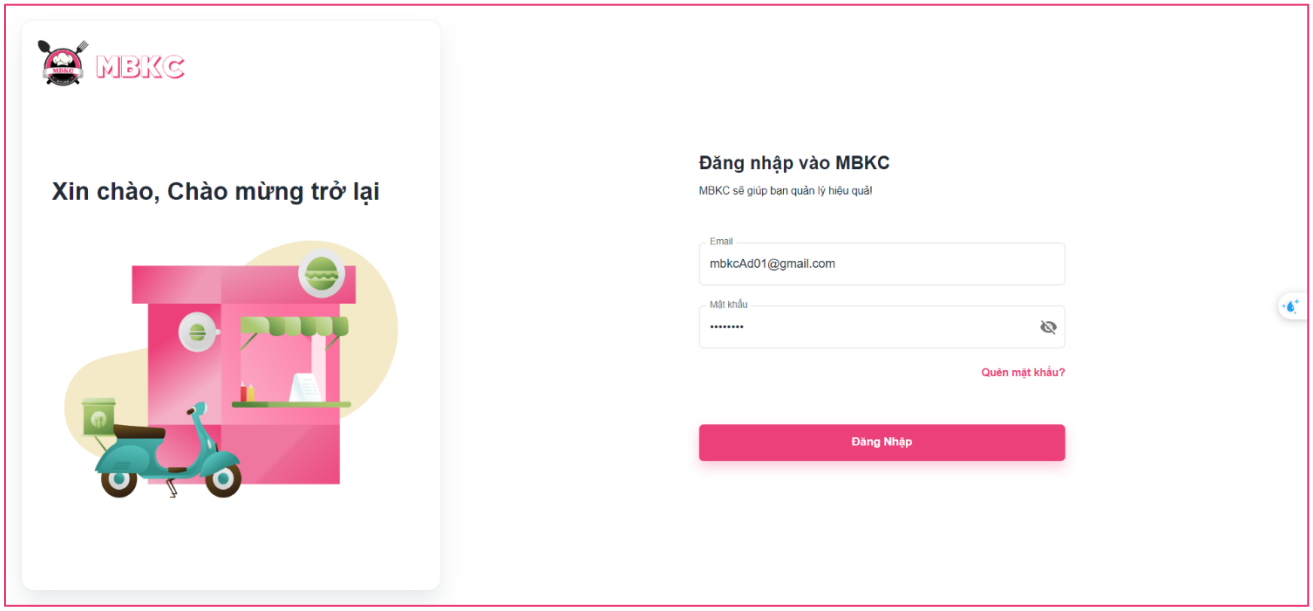


Figure 633: User Manual – Create Payment for Store – Step 1

Step 2: Click “Đăng Nhập” button.

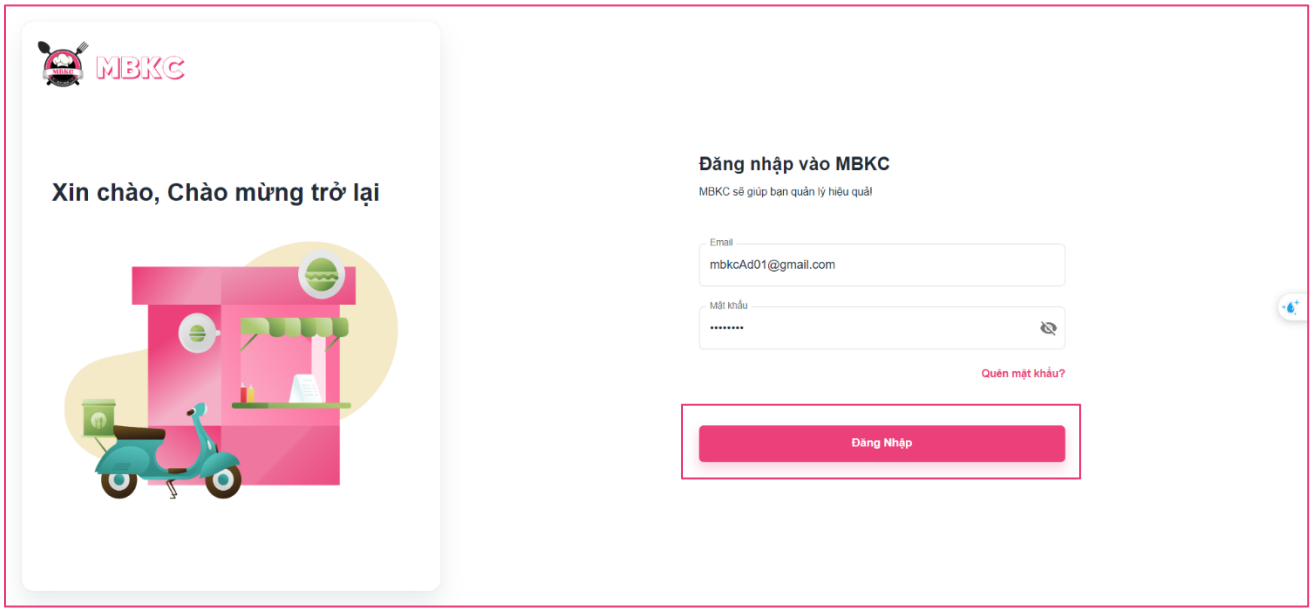


Figure 634: User Manual - Create Payment for Store - Step 2

Step 3: From Kitchen Center Dashboard Page, choose “Ví” from the left sidebar.

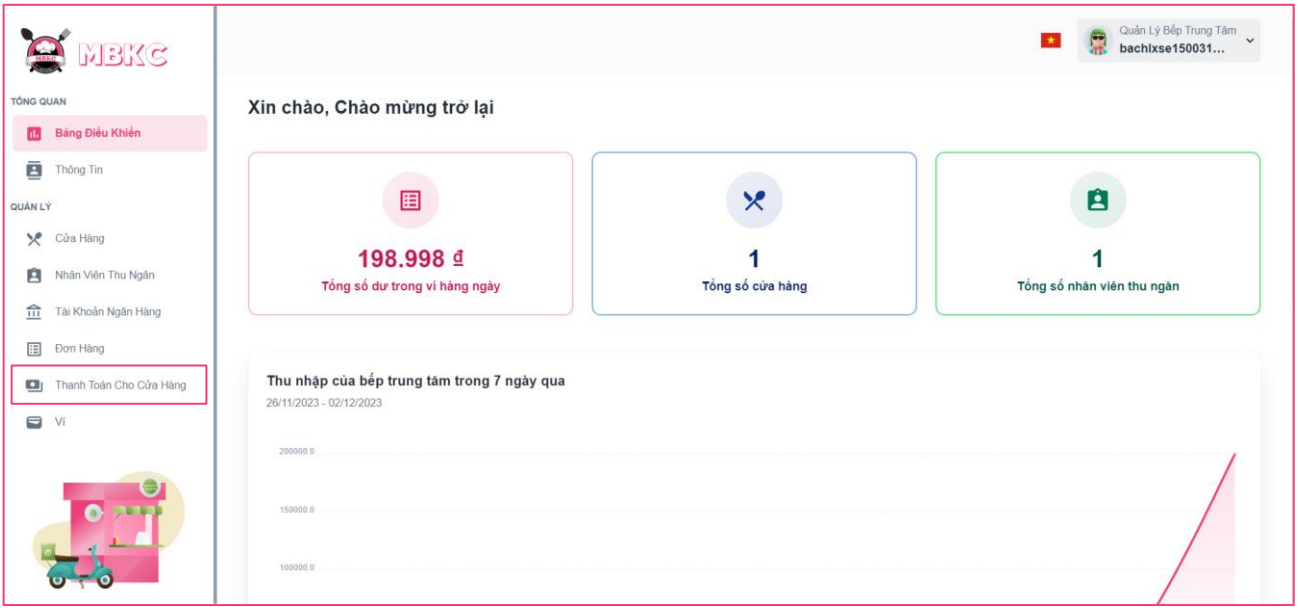


Figure 635: User Manual - Create Payment for Store - Step 3

Step 4: Click “Tạo Thanh Toán Cho Cửa Hàng” button.

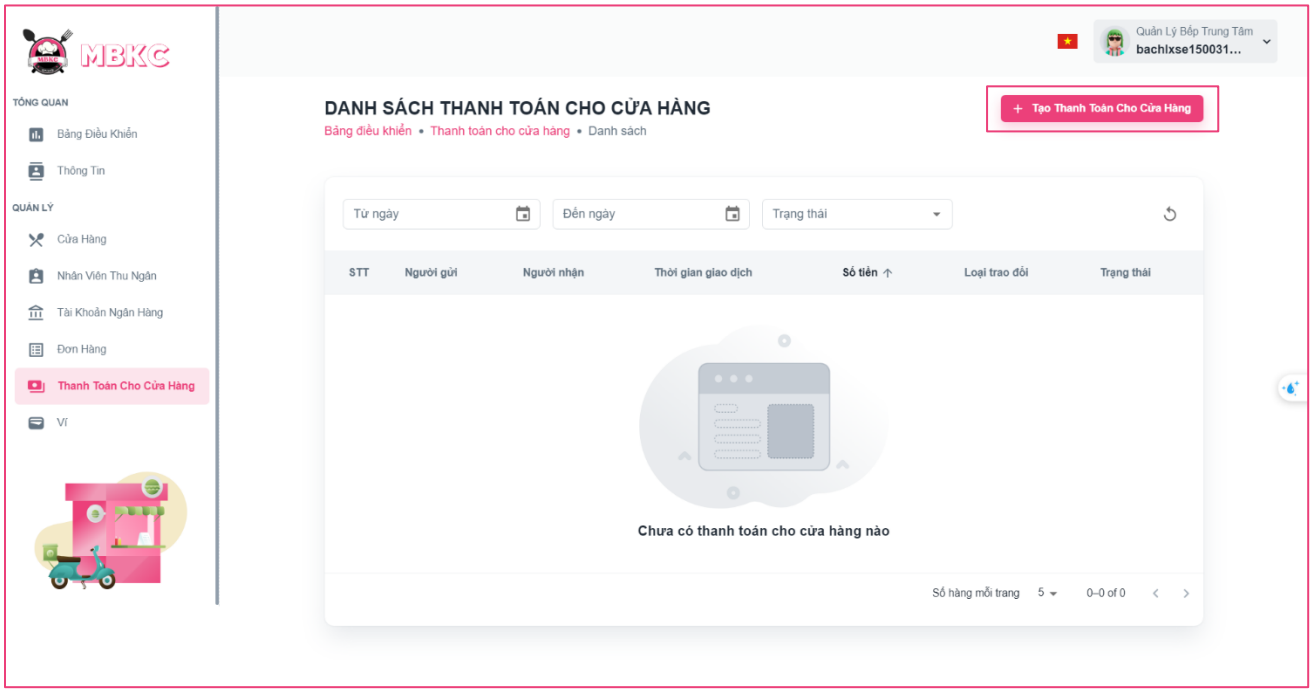


Figure 636: User Manual - Create Payment for Store - Step 4

Step 5: Filling all the required information to create payment for store including: “Hình ảnh”, “Cửa Hàng”, “Số tiền”.

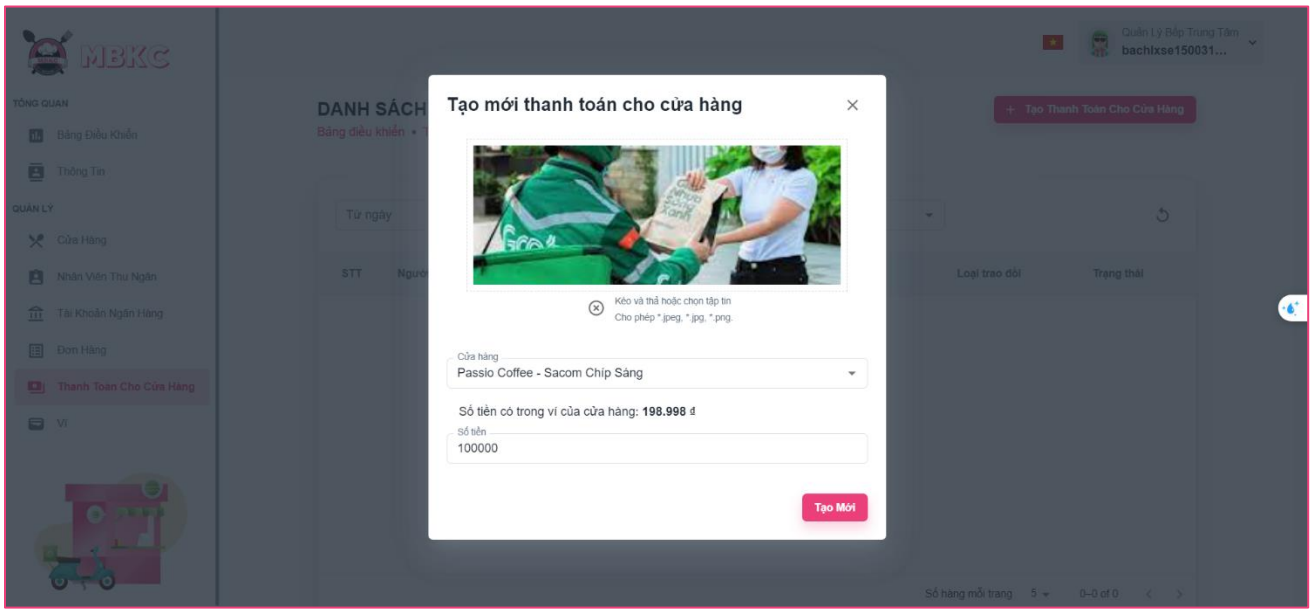


Figure 637: User Manual - Create Payment for Store - Step 5

Step 6: Click “Tạo Mới” button.

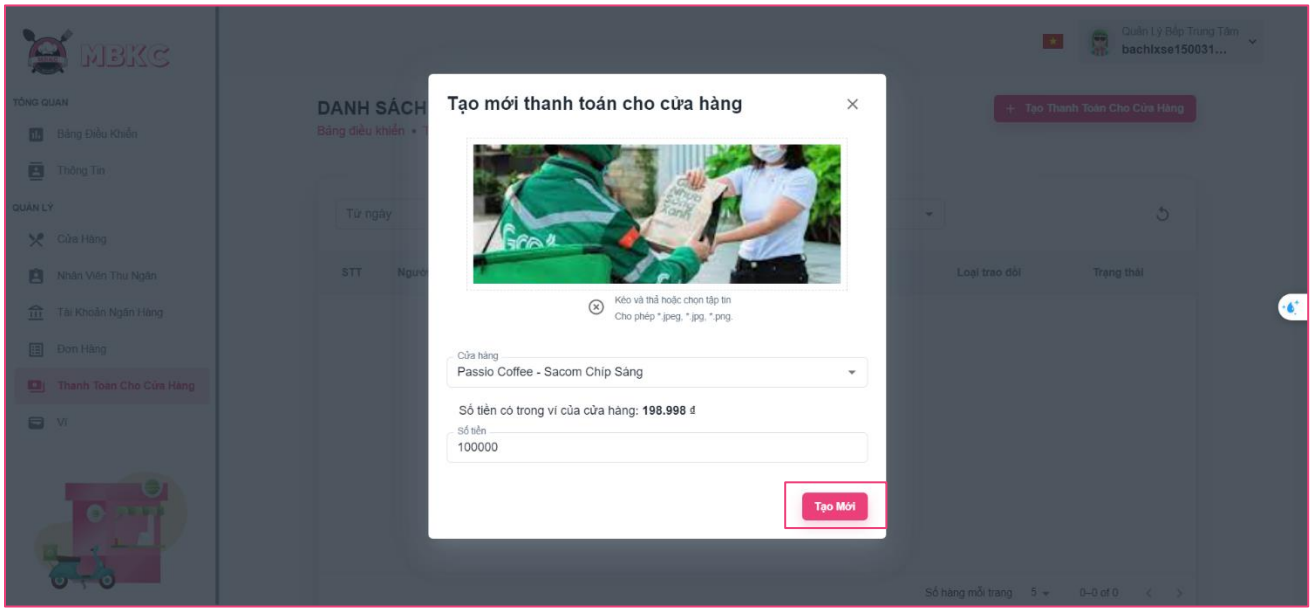


Figure 638: User Manual - Create Payment for Store - Step 5

Step 6: List of Payments for Store Page will be reloaded to show the list of payments for store and a toast message “Rút tiền cho cửa hàng thành công”.

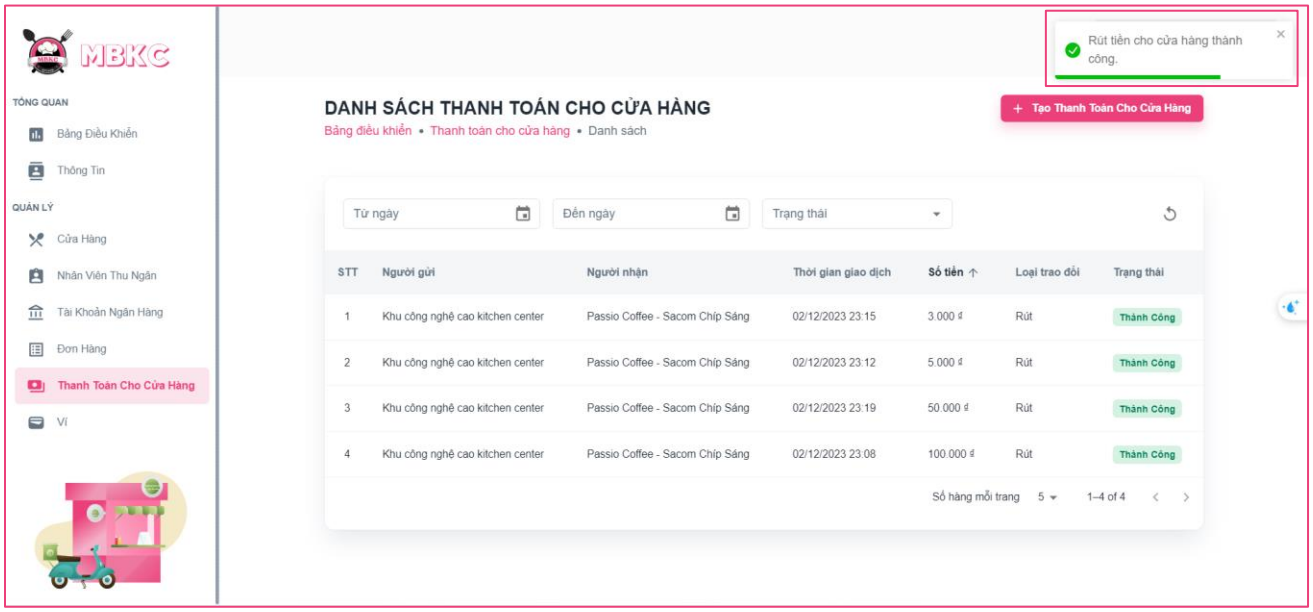


Figure 639: User Manual - Create Payment for Store - Step 6

Step 7: Click on the payment for store row to pop up Transaction Details Modal.

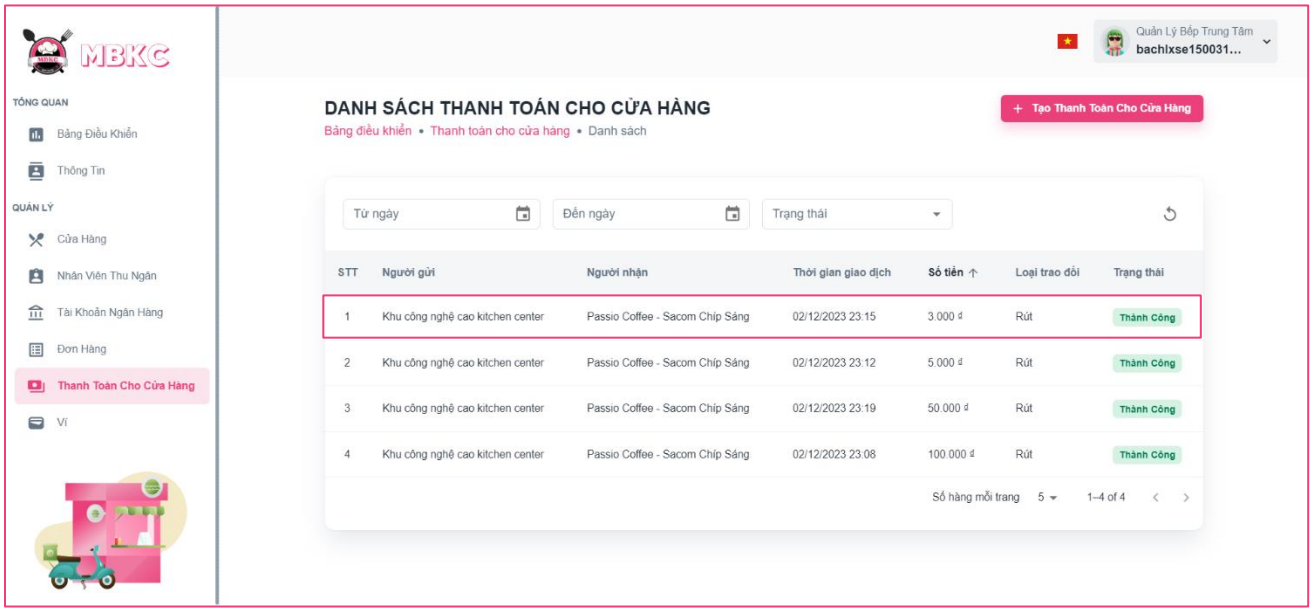


Figure 640: User Manual - Create Payment for Store - Step 7

Step 8: Pop up Transaction Details Modal.

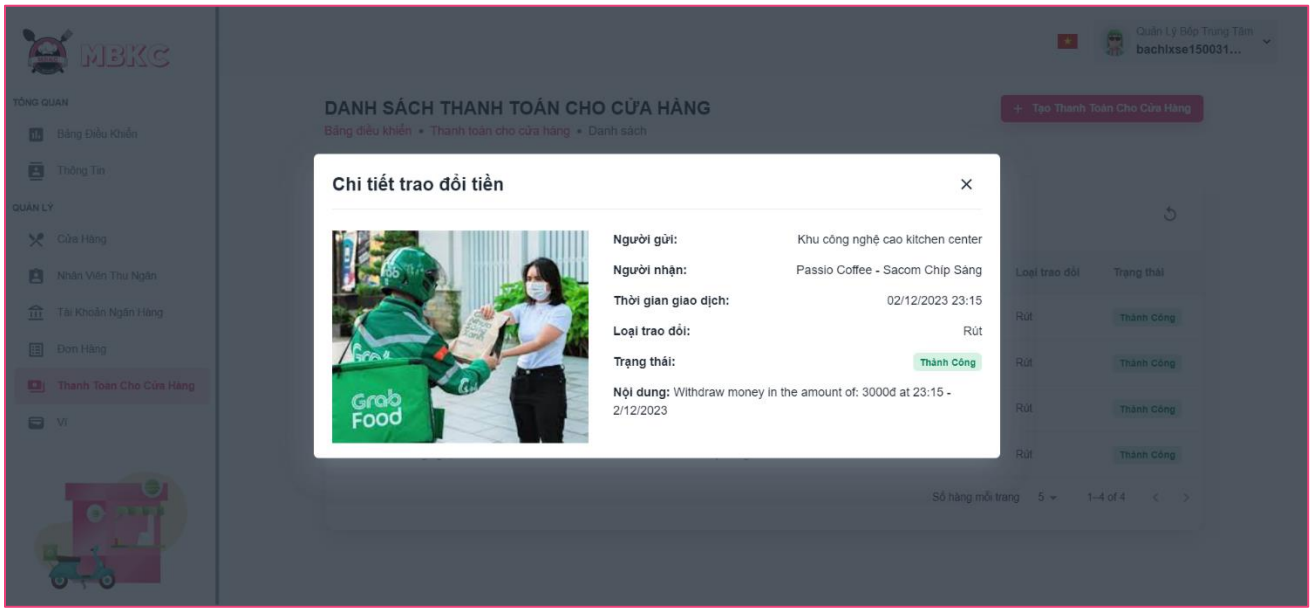


Figure 641: User Manual - Create Payment for Store - Step 8

3.2.15. Update Configuration

3.2.15.1. Description

Use this guideline to update order receiving time.

This web application supports update order receiving time with an account as a MBKC Admin.

3.2.15.2. Guide Steps

Step 1: Access MBKC at https://mbkc.vercel.app/. Then fill in “Email” and “Password” of account MBKC Admin.

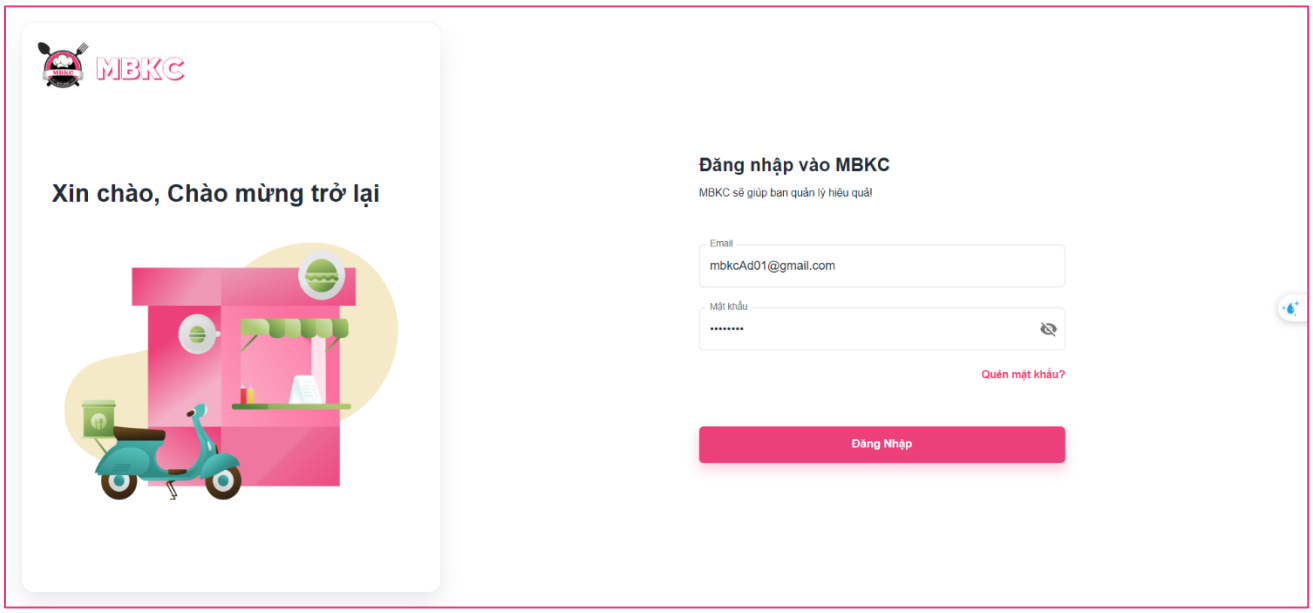


Figure 642: User Manual – Update Configuration – Step 1

Step 2: Click “Đăng Nhập” button.

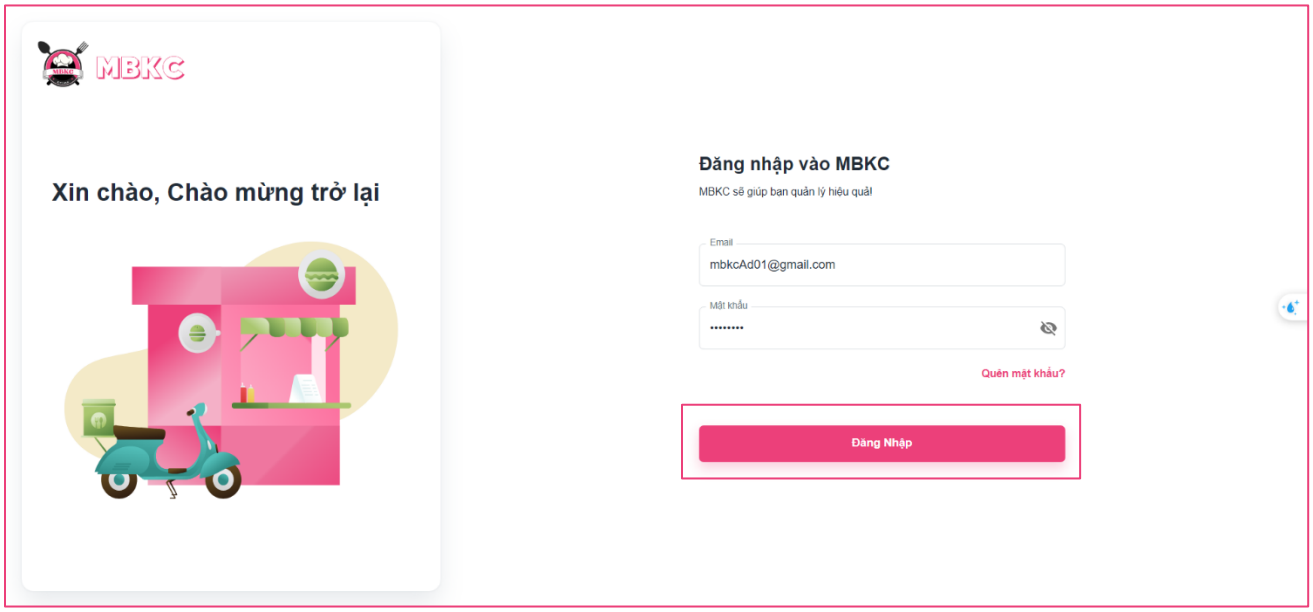


Figure 643: User Manual - Update Configuration – Step 2

Step 3: From Admin Dashboard Page, choose “Cấu hình hệ thống” from the left sidebar.

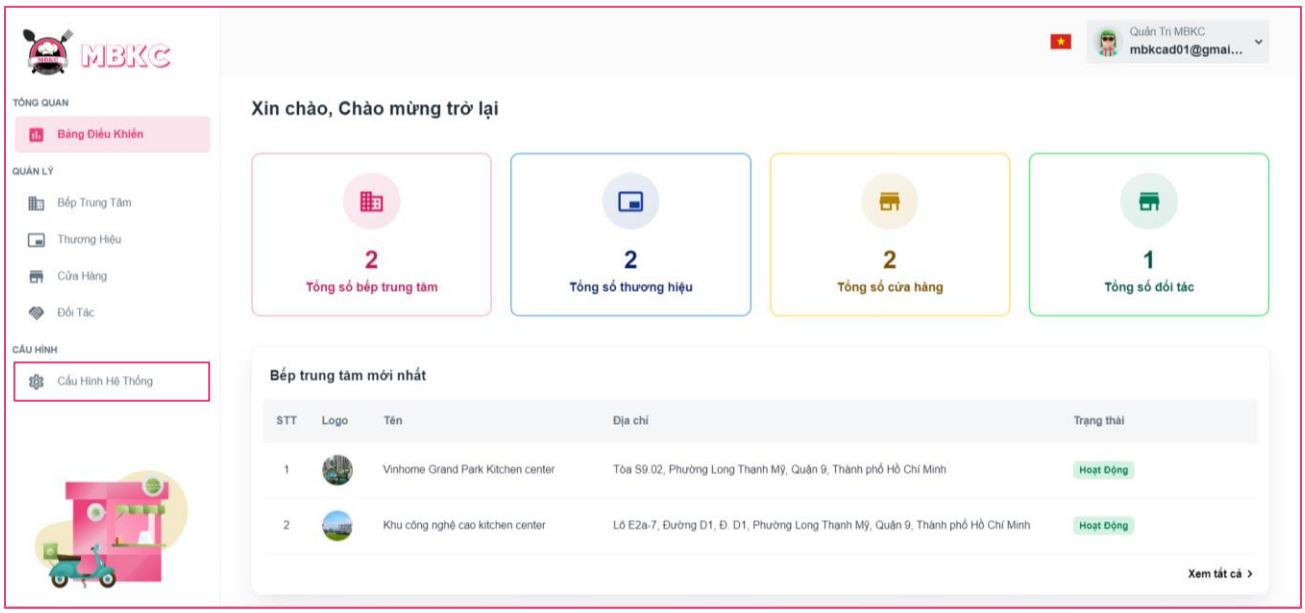


Figure 644: User Manual - Update Configuration – Step 3

Step 4: Click Arrow right icon button in box to show the configuration section.

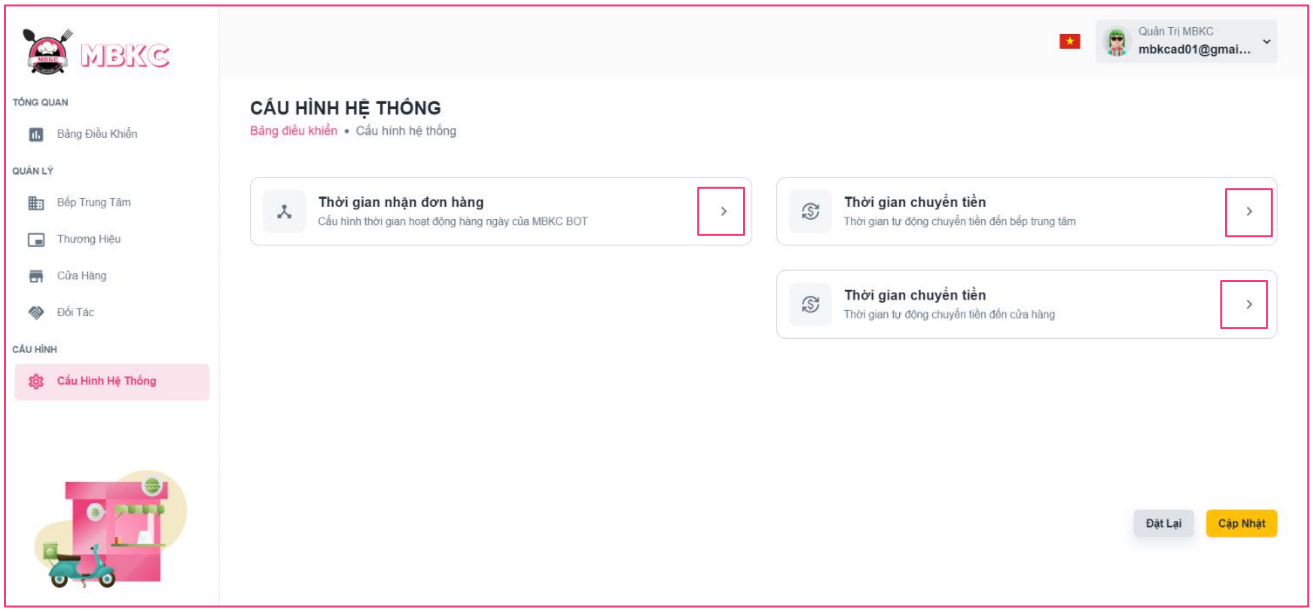


Figure 645: User Manual - Update Configuration – Step 4

Step 5: Click on switch button to turn off the disabled mode to update the configuration. And then select time in clock to update.

+ The order receiving time must end at least 10 minutes earlier than the time to automatically transfer money from the cashier's wallet to the kitchen center's wallet.

+ The time to automatically transfer money from the cashier's wallet to the kitchen center's wallet must be at least 1 hour earlier than the time to automatically transfer money from the kitchen center's wallet to the store's wallet.

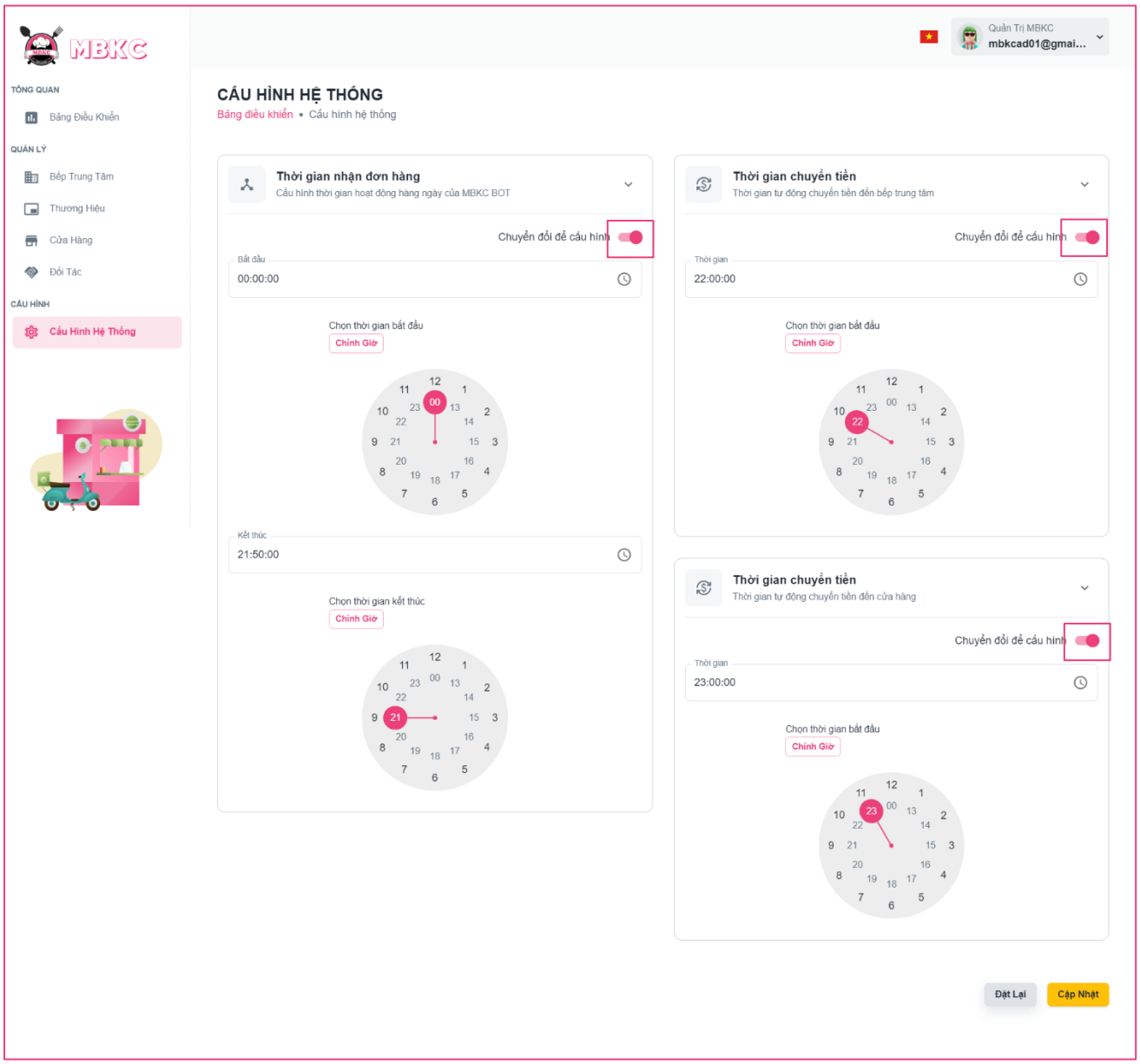


Figure 646: User Manual - Update Configuration – Step 5

Step 6: After select time, click “Cập Nhật” button.

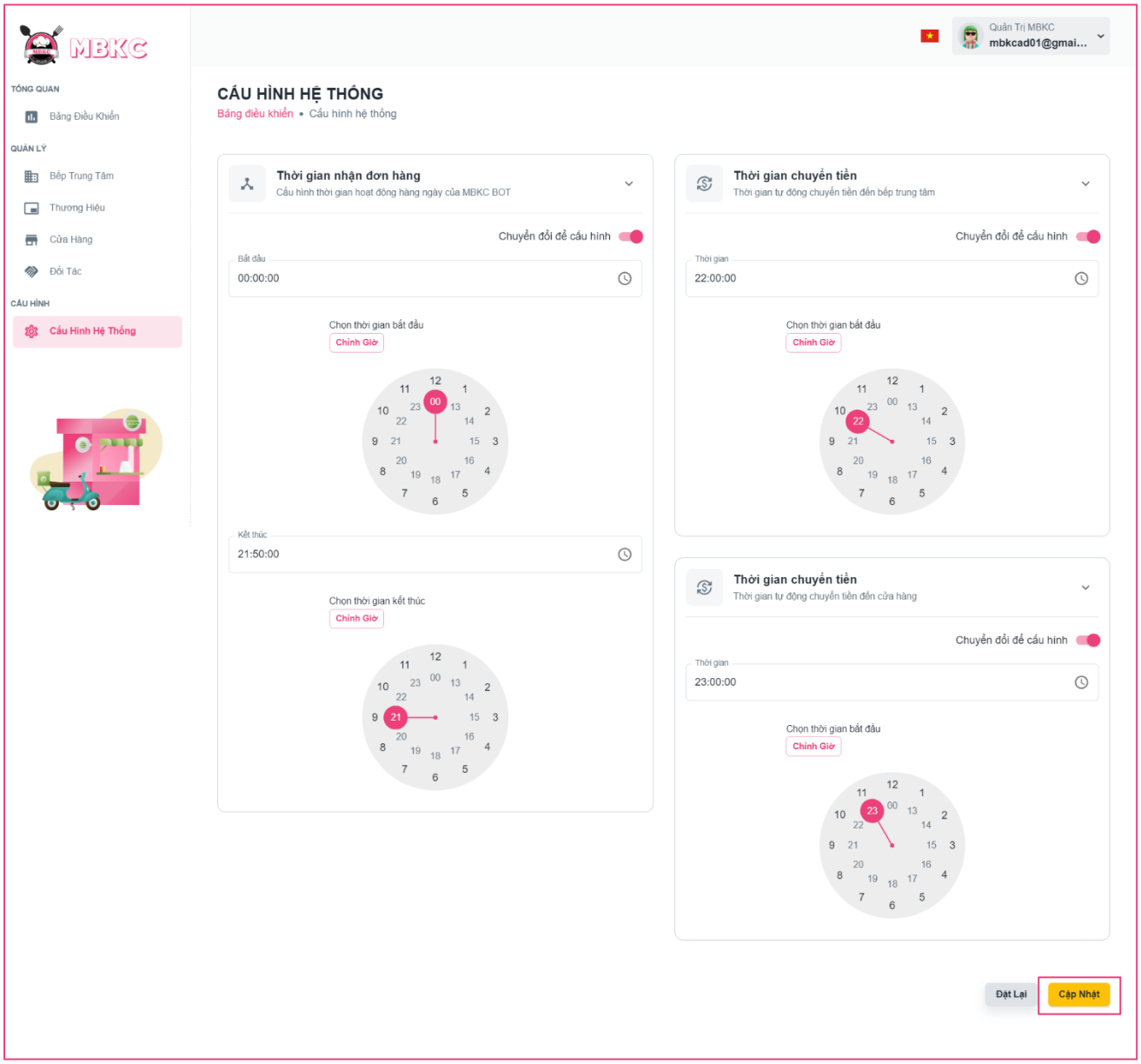


Figure 647: User Manual - Update Configuration – Step 6

Step 6: Configuration Page will be reloaded to show the configuration information updated and a toast message “Đã cập nhật cấu hình hệ thống thành công”.

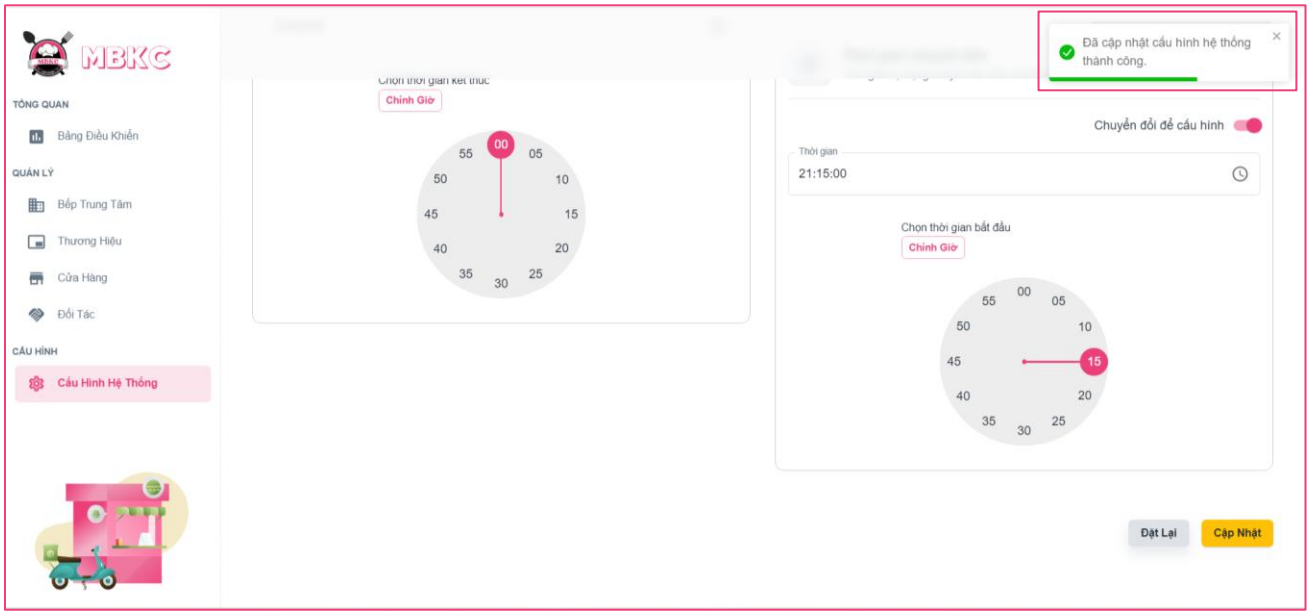


Figure 648: User Manual - Update Configuration – Step 6

3.3. Mobile Application

3.3.1. Authentication

3.3.1.1 Login

3.3.1.1.1 Description

Use this guideline to login into MBKC mobile application with an email account.

This web application supports login with email account that has been added as Store Manager.

3.3.1.1.2. Guide Steps

Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 649: User Manual - Mobile - Login - Step 1

Step 2: Click “Đăng Nhập” button.

Figure 650: User Manual - Mobile - Login - Step 2 - 1

3.3.1.2. Reset Password

3.3.1.2.1. Description

Use this guide so users can reset their email account password in mobile application when they forget their password.

This web application supports reset their email account password in mobile application that has been added as Store Manager.

3.3.1.2.2. Guide Steps

Step 1: Access MBKC mobile application. Then press “Quên mật khẩu”.

Figure 651: User Manual – Mobile - Reset Password – Step 1

Step 2: Navigate to Forgot Password Screen and then fill in “Email”.

Figure 652: User Manual - Mobile - Reset Password - Step 2

Step 3: Click “Gửi Email” button.

Figure 653: User Manual – Mobile - Reset Password - Step 3

After sending the email successfully, the user is redirected to the Verify OTP Screen with the email that sent the OTP authentication code.

Step 4: Check email in field “Email” and fill in “Mã OTP”.

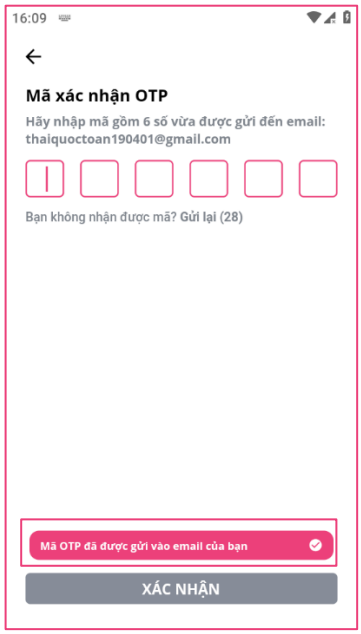


Figure 654: User Manual - Mobile - Reset Password - Step 4

Step 5: Click “Xác Minh” button.

Figure 655: User Manual – Mobile - Reset Password – Step 5

After successfully verifying the OTP code, the user is redirected to the Reset Password Page to reset the password.

Step 6: Then fill in “Mật khẩu mới” and “Nhập lại mật khẩu”.

Figure 656: User Manual - Mobile - Reset Password - Step 6

Step 7: Click “XÁC NHẬN” button.

Figure 657: User Manual - Mobile - Reset Password - Step 7

After successfully reset password, the user is redirected to the Login Page to login again.

3.3.1.3. Update Password in First Time Login

3.3.1.3.1. Description

Use this guide so users will update their password when they login for the first time with a new account.

This web application supports updating passwords when they login for the first time in mobile appliction for actors: Store Manager.

3.3.1.3.2. Guide Steps

Step 1: Access MBKC mobile application. After login successfully, navigate to Update Password Screen and fill in “Mật khẩu mới” and “Nhập lại mật khẩu”.

Figure 658: User Manual - Mobile - Update Password in First Time Login - Step 1

Step 2: Click “XÁC NHẬN” button.

Figure 659: User Manual – Mobile - Update Password in First Time Login - Step 2

After successfully update password, navigate Dashboard Screen to continue to use mobile application.

3.2.1.4. Update Password

3.2.1.4.1. Description

Use this guide so users can update their email account password when they want to change their password.

This web application supports updating passwords in mobile application for actors: Store Manager.

3.2.1.4.2. Guide Steps

Step 1: Access MBKC mobile application. After logging in successfully, click the icon user tab on the bottom tab navigation and.

Figure 660: User Manual – Mobile - Update Password - Step 1

After click icon user tab, the user is redirected to the Profile Page.

Step 2: Press “Thay đổi mật khẩu” menu item.

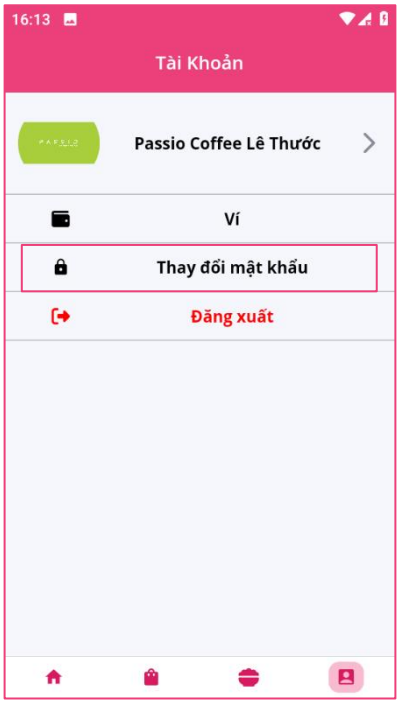


Figure 661: User Manual – Mobile - Update Password - Step 2

Step 3: Fill in “Mật khẩu” and “Nhập lại mật khẩu”, then click “XÁC NHẬN” button.

Figure 662: User Manual – Update Password – Step 3

After successfully update password, navigate to Profile Screen.

3.3.2. Category management

3.3.2.1. View list of Categories

3.3.2.1.1. Description

Use this guideline to search, reload data, filter by category type to view list of categories.

This web application supports view list of categories in mobile application with account as a Store Manager.

3.3.2.1.2. Guide Steps

Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 663: User Manual - Mobile - View list of Categories - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 664: User Manual - Mobile - View list of Categories - Step 2 - 1

Step 3: From Store Dashboard Screen, choose Food icon tab from the bottom tab.

Figure 665: User Manual – Mobile - View list of Categories - Step 3

Step 4: Search, reload data, and filter by category type to view list of categories.

Case 1: Fill in the category's name in the search box to search categories in the list.

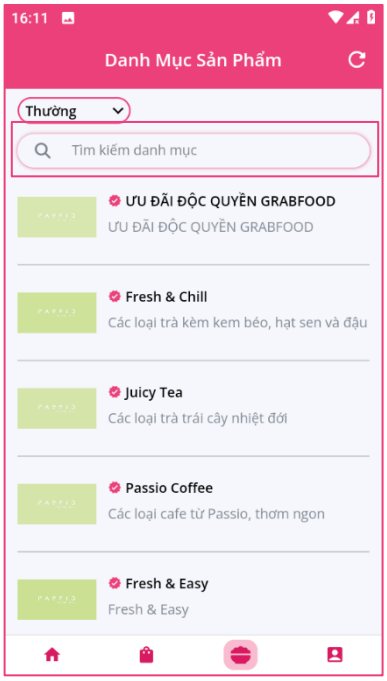


Figure 666: User Manual – Mobile - View list of Categories – Step 4 – Case 1

Case 2: Click the Reload icon button to reload the list of categories.

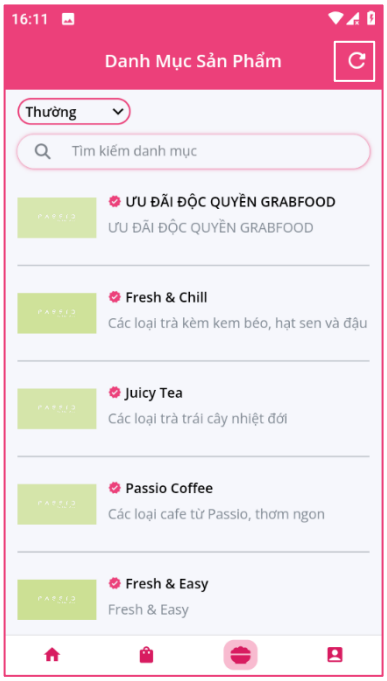


Figure 667: User Manual - Mobile - View list of Categories - Step 4 - Case 2

Case 3: Click on the box category type and select category type corresponding you want to view in list of categories.

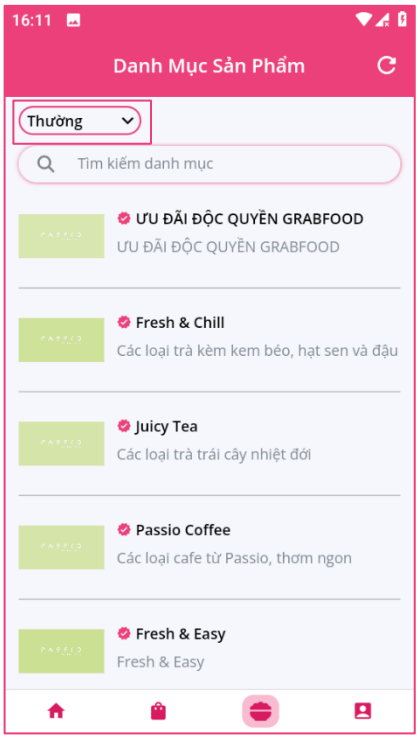


Figure 668: User Manual – Mobile - View list of Categories - Step 4 - Case 3

3.3.2.2. View list of Extra Categories

3.3.2.2.1. Description

Use this guideline to view list of extra categories.

This web application supports view list of extra categories in mobile application with account as a Store Manager.

3.3.2.2.2. Guide Steps

Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 669: User Manual - Mobile - View list of Extra Categories - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 670: User Manual - Mobile - View list of Extra Categories - Step 2 - 1

Step 3: From Store Dashboard Screen, choose Food icon tab from the bottom tab.

Figure 671: User Manual – Mobile - View list of Extra Categories - Step 3

Step 4: Click on category item in List of Categories Screen.

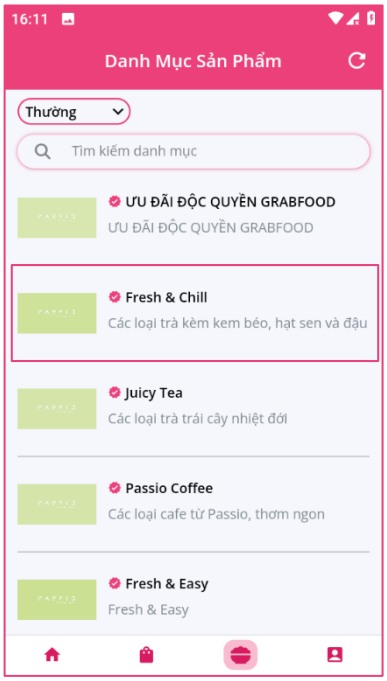


Figure 672: User Manual - Mobile - View list of Extra Categories – Step 4

Step 5: Click on “Danh mục thêm” to view list of extra categories of this category.

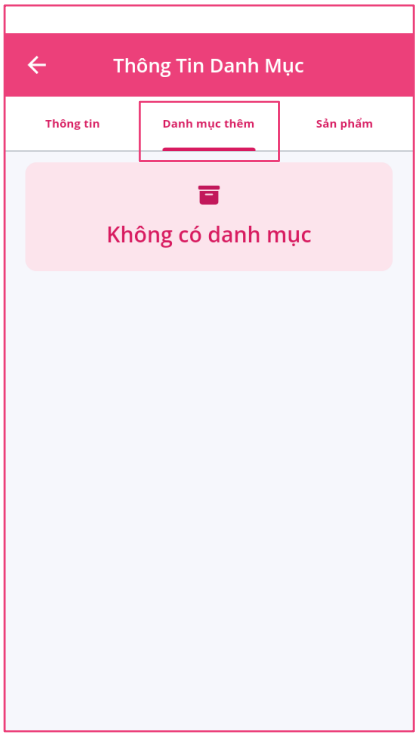


Figure 673: User Manual – Mobile - - View list of Extra Categories - Step 5

3.3.2.3. View details Category

3.3.2.3.1. Description
Use this guideline to view details category.

This web application supports view details category in mobile application with account as a Store Manager.

3.3.2.3.2. Guide Steps

Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 674: User Manual - Mobile - View details Category - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 675: User Manual - Mobile - View details Category - Step 2 - 1

Step 3: From Store Dashboard Screen, choose Food icon tab from the bottom tab.

Figure 676: User Manual - Mobile - View details Category - Step 3

Step 4: Click on category item in List of Categories Screen.

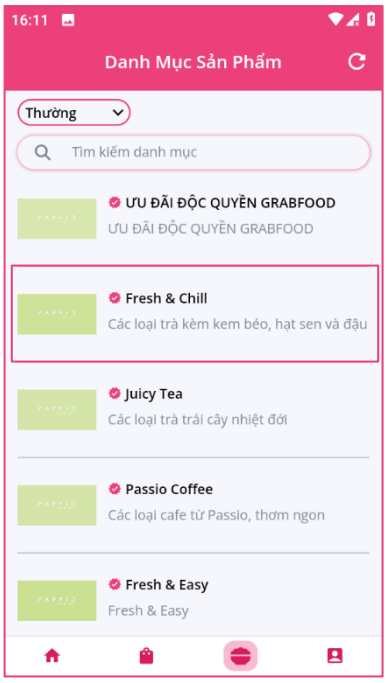


Figure 677: User Manual - Mobile - View details Category – Step 4

Step 5: Navigate to Category Details Screen to show information details of category.

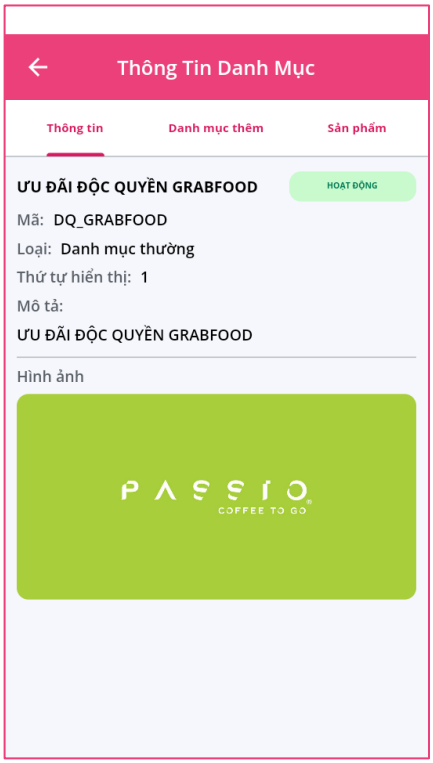


Figure 678: User Manual – Mobile - View details Category - Step 5

3.3.3. Product management

3.3.3.1. View list of Products

3.3.3.1.1. Description

Use this guideline to search, filter by product type, and filter by price to view list of products.

This web application supports view list of products in mobile application with account as a Store Manager.

3.3.3.1.2. Guide Steps

Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure679: User Manual - Mobile - View list of Products - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 680: User Manual - Mobile - View list of Products - Step 2

Step 3: From Store Dashboard Screen, choose Food icon tab from the bottom tab.

Figure 681: User Manual – Mobile - View list of Products - Step 3

Step 4: Click on category item in List of Categories Screen.

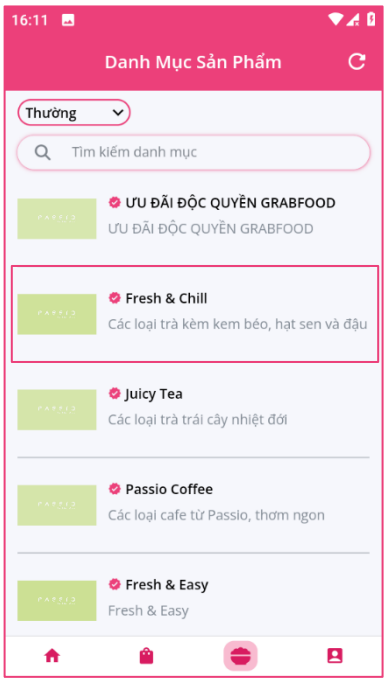


Figure 682: User Manual - Mobile - View list of Products – Step 4

Step 5: Navigate to Category Details Screen and then click “Sản phẩm” tab to view list of products of this category.

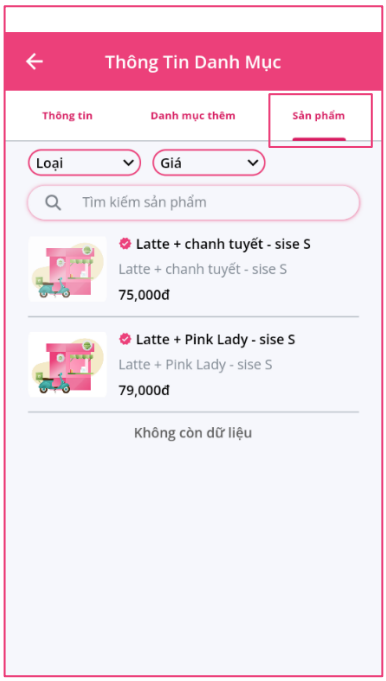


Figure 683: User Manual – Mobile - View list of Products - Step 5 – Case 2

Step 6: Search, filter by product type, and filter by price to view list of products.

Case 1: Fill in the product’s name in the search box to search products in the list.

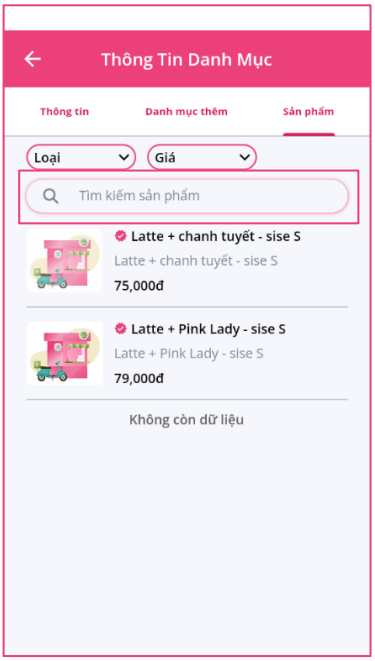


Figure 684: User Manual – Mobile - View list of Products – Step 6 – Case 1

Case 2: Click on the box “Loại” and select product type corresponding you want to view in list of products.

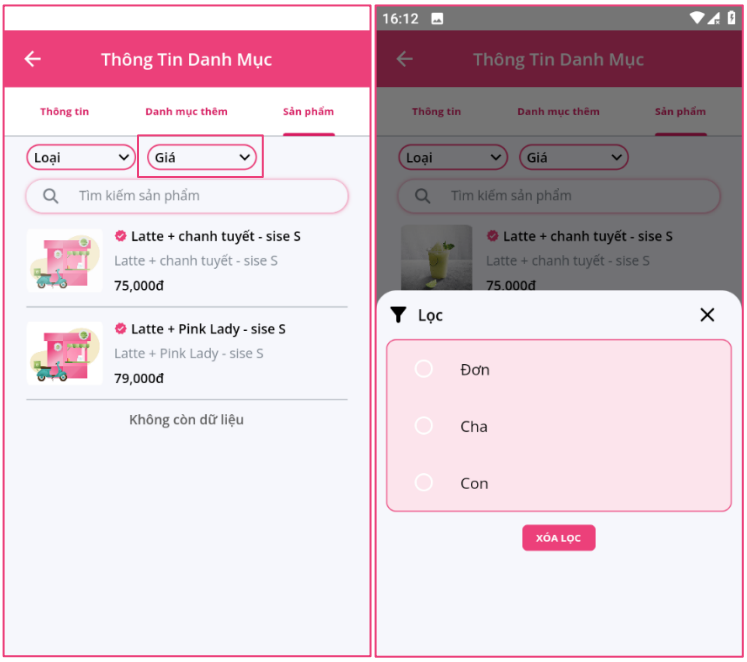


Figure 685: User Manual – Mobile - View list of Products – Step 6 – Case 2

Case 3: Click on the “Giá” and select option “Tăng dần” or “Giảm dần” of price to arrange the display order of products by price.

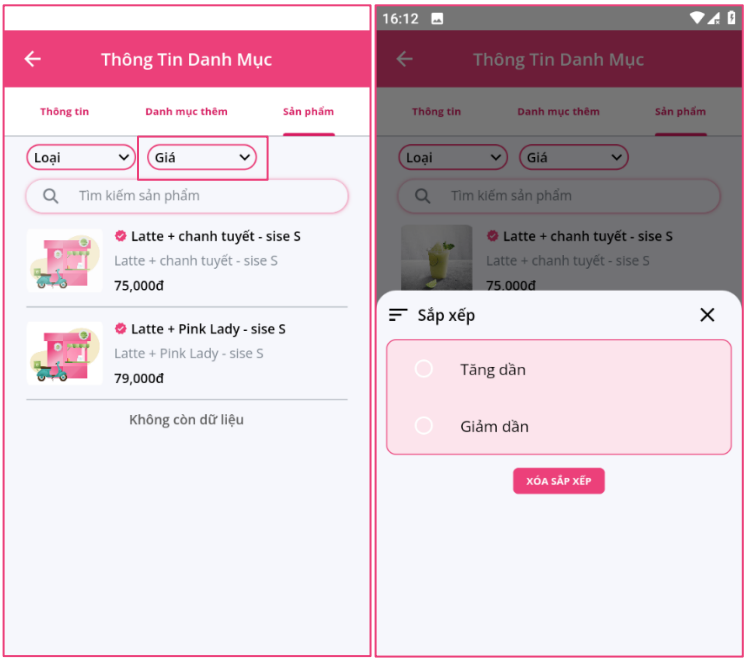


Figure 686: User Manual – Mobile - View list of Products – Step 6 – Case 3

3.3.3.2. View details Product

3.3.3.2.1. Description
Use this guideline to view details product.

This web application supports view details product in mobile application with account as a Store Manager.

3.3.3.2.2. Guide Steps
Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 687: User Manual - Mobile - View details Product - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 688: User Manual - Mobile - View details Product - Step 2

Step 3: From Store Dashboard Screen, choose Food icon tab from the bottom tab.

Figure 689: User Manual - Mobile - View details Product - Step 3

Step 4: Click on category item in List of Categories Screen.

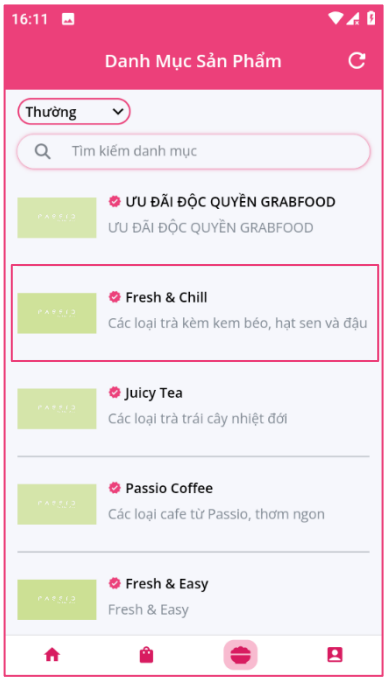


Figure 690: User Manual - Mobile - View details Product – Step 4

Step 5: Navigate to Category Details Screen to show information details of category.

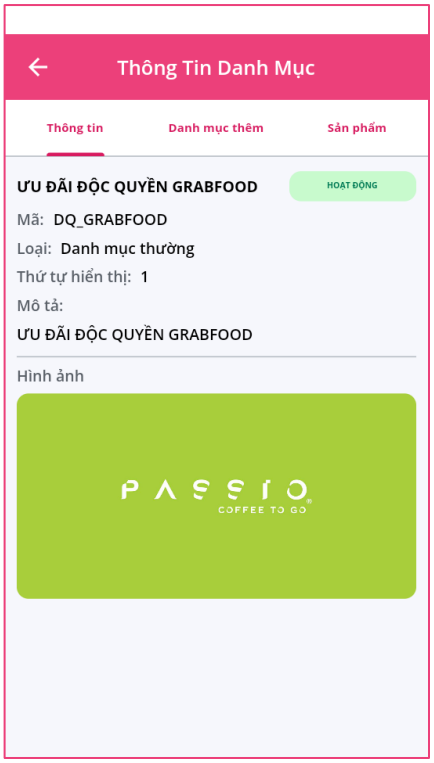


Figure 691: User Manual – Mobile - View details Product - Step 5

Step 6: Click on “Sản phẩm” to view list of products of this category.

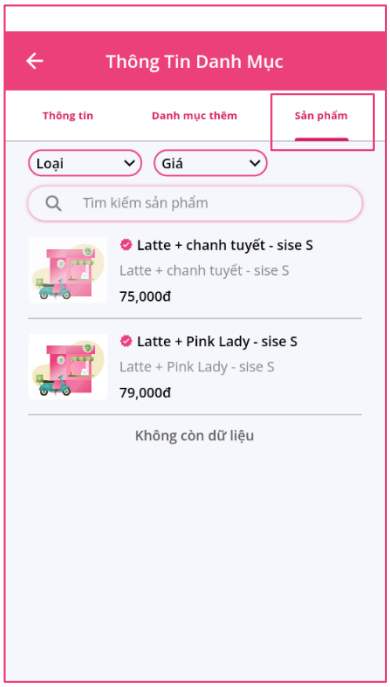


Figure 692: User Manual – Mobile - View details Product - Step 6

Step 7: Click on product item in List of Products Screen to view information detail of product.

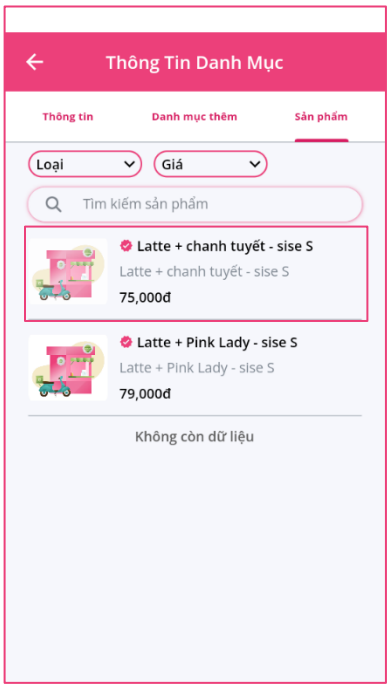


Figure 693: User Manual – Mobile - View details Product - Step 7

Step 8: Navigate to Products Details Screen to show information details of product.

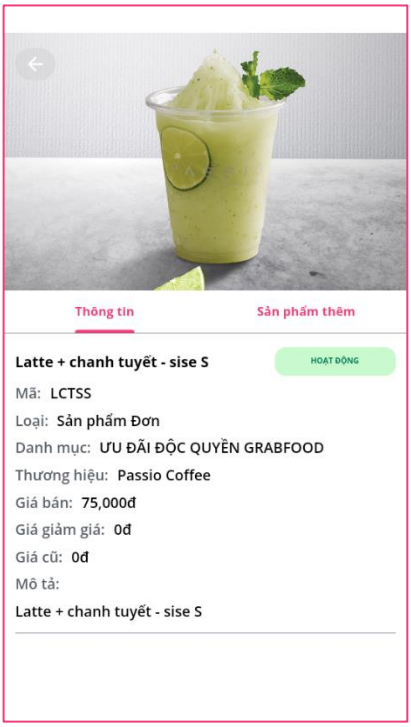


Figure 694: User Manual – Mobile - View details Product - Step 8

Step 9: Click "Sản phẩm thêm" tab to view list of extra products belong to this product.

Figure 695: User Manual – Mobile - View details Product - Step 9

3.3.4. Order management

3.3.4.1. View list of Orders

3.3.4.1.1. Description
Use this guideline to reload, and filter to view list of orders.

This web application supports view list of orders in mobile application with account as a Store Manager.

3.3.4.1.2. Guide Steps
Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 696: User Manual - Mobile - View list of Orders - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 697: User Manual - Mobile - View list of Orders - Step 2

Step 3: From Store Dashboard Screen, choose Order icon tab from the bottom tab.

Figure 698: User Manual – Mobile - View list of Orders - Step 3

Step 4: Reload, and filter to view list of orders.

Case 1: Click the Reload icon button to reload the list of orders.

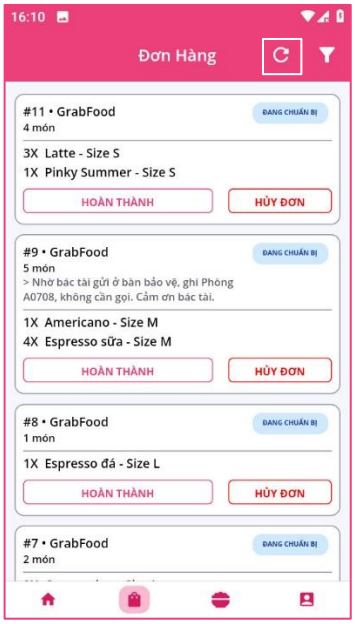


Figure 699: User Manual – Mobile - View list of Orders – Step 4 – Case 1

Case 2: Click the Filter icon button to select option filter include: “Từ ngày”, “Đến ngày”, “Trạng thái hệ thống”, “Trang thái đối tác” to view the list of orders.

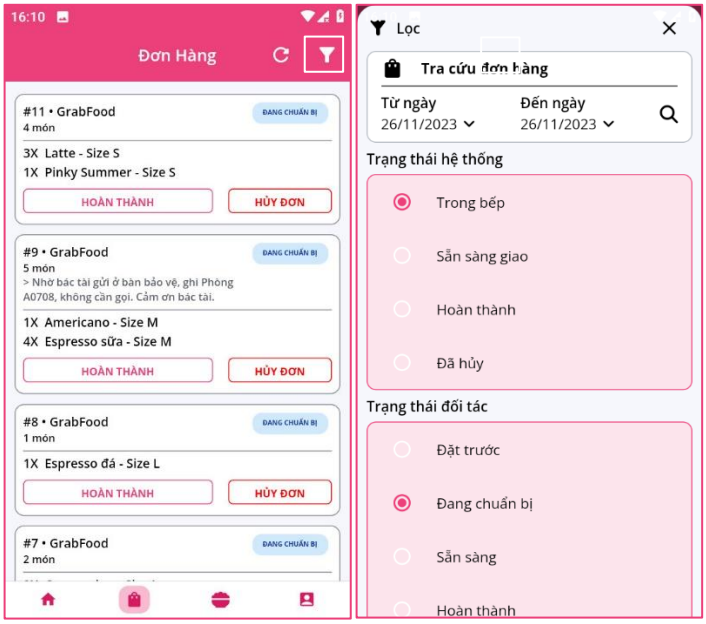


Figure 700: User Manual – Mobile - View list of Orders – Step 4 – Case 2

3.3.4.2. View details Order

3.3.4.2.1. Description
Use this guideline to view details order.

This web application supports view details order in mobile application with account as a Store Manager.

3.3.4.2.2. Guide Steps
Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 701: User Manual - Mobile - View details Order - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 702: User Manual - Mobile - View details Order - Step 2

Step 3: From Store Dashboard Screen, choose Food icon tab from the bottom tab.

Figure 703: User Manual - Mobile - View details Order - Step 3

Step 4: Click on order item in List of Orders Screen.

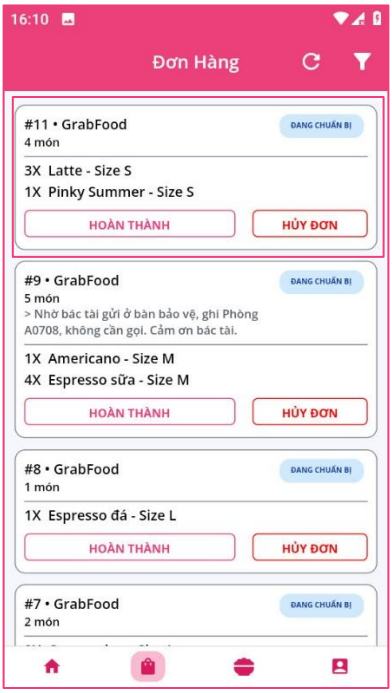


Figure 704: User Manual - Mobile - View details Order – Step 4

Step 5: Navigate to Order Details Screen to show information details of order.

Figure 705: User Manual – Mobile - View details Order - Step 5

3.3.4.3. Confirm Order to Ready

3.3.4.3.1. Description

Use this guideline to confirm order to ready.

This web application supports confirm order to ready in mobile application with account as a Store Manager.

3.3.4.3.2. Guide Steps

Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 706: User Manual - Mobile – Confirm Order to Ready - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 707: User Manual - Mobile - Confirm Order to Ready - Step 2

Step 3: From Store Dashboard Screen, choose Food icon tab from the bottom tab.

Figure 708: User Manual - Mobile - Confirm Order to Ready - Step 3

Step 4: Confirm Order to Ready.

Case 1: Confirm order to ready on List of Orders Screen.

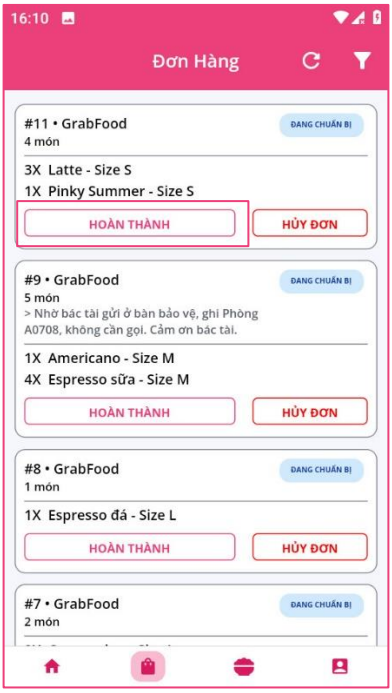


Figure 709: User Manual - Mobile - Confirm Order to Ready - Step 4 - Case 1

Case 2: Confirm order to ready on Order Detail Screen.

Figure 710: User Manual - Mobile - Confirm Order to Ready - Step 4 - Case 2

Step 5: Click “Đồng ý” button.

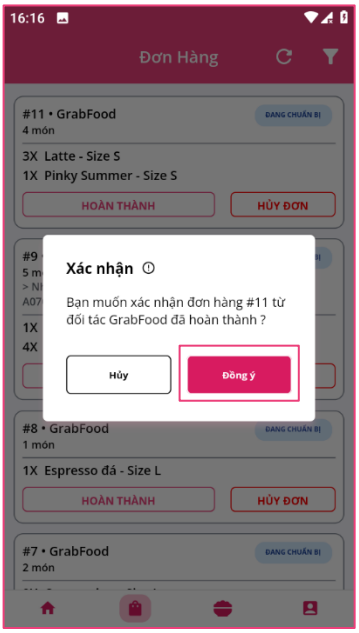


Figure 711: User Manual – Mobile - Confirm Order to Ready - Step 5

+ If you confirm order in List of Orders Screen, List Orders Screen will be reloaded and toast message update successfully.

+ If you confirm order in Order Detail Screen, Order Detail Screen will be reloaded and toast message update successfully.

3.3.4.4. Cancel Order

3.3.4.4.1. Description
Use this guideline to cancel order.

This web application supports cancel order in mobile application with account as a Store Manager.

3.3.4.4.2. Guide Steps

Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 712: User Manual - Mobile – Confirm Order to Ready - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 713: User Manual - Mobile - Confirm Order to Ready - Step 2

Step 3: From Store Dashboard Screen, choose Food icon tab from the bottom tab.

Figure 714: User Manual - Mobile - Confirm Order to Ready - Step 3

Step 4: Cancel Order.

Case 1: Cancel order on List of Orders Screen.

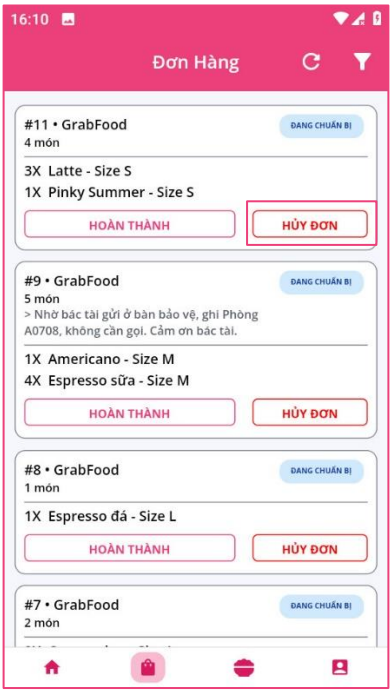


Figure 715: User Manual - Mobile - Confirm Order to Ready - Step 4 - Case 1

Case 2: Cancel order on Order Detail Screen.

Figure 716: User Manual - Mobile - Confirm Order to Ready - Step 4 - Case 2

Step 5: Click "Đồng ý" button.

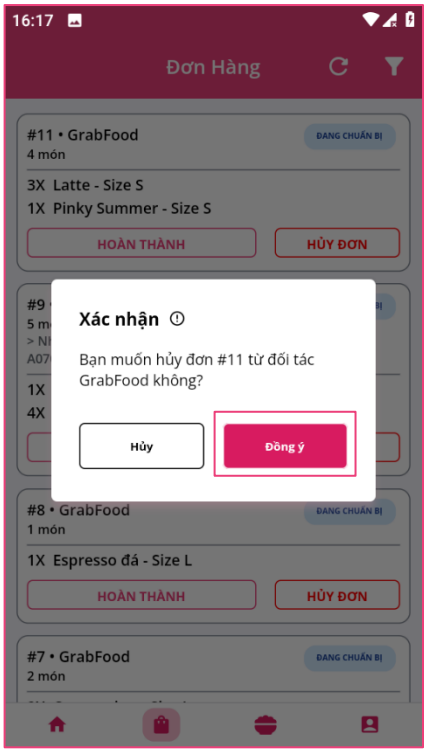


Figure 717: User Manual – Mobile - Confirm Order to Ready - Step 5

+ If you cancel order in List of Orders Screen, List Orders Screen will be reloaded and toast message update successfully.

+ If you cancel order in Order Detail Screen, Order Detail Screen will be reloaded and toast message update successfully.

3.3.3. Wallet management

3.3.3.1. View wallet

3.3.3.1.1. Description
Use this guideline to view wallet information and list of transactions.

This web application supports view wallet information and list of transactions in mobile application with account as a Store Manager.

3.3.3.1.2. Guide Steps
Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 718: User Manual - Mobile - View wallet - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 719: User Manual - Mobile - View wallet - Step 2

Step 3: From Store Dashboard Screen, choose User icon tab from the bottom tab.

Figure 720: User Manual – Mobile - View wallet - Step 3

Step 4: Press “Ví” menu item to navigate to Wallet Screen.

Figure 721: User Manual - Mobile - View wallet – Step 4

Step 5: Reload, and filter to view information of wallet.

Case 1: Click the Reload icon button to reload the information of wallet.

Figure 722: User Manual – Mobile - View wallet - Step 5 – Case 1

Case 2: Click the Filter icon button to select option filter include: “Từ ngày”, “Đến ngày”, and “Loại giao dịch” to view information of wallet.

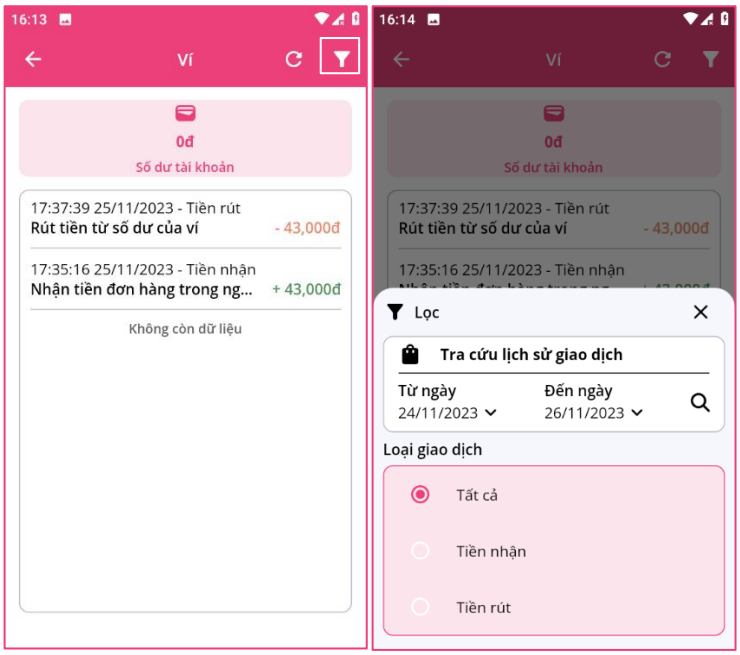


Figure 723: User Manual – Mobile - View wallet – Step 4 – Case 2

3.3.3.2. View details Transaction

3.3.3.2.1. Description

Use this guideline to view details transaction.

This web application supports view details transaction in mobile application with account as a Store Manager.

3.3.3.2.2. Guide Steps

Step 1: Access MBKC mobile application. Then fill in “Email” and “Password” of account Store Manager.

Figure 724: User Manual - Mobile - View details Transaction - Step 1

Step 2: Click “ĐĂNG NHẬP” button.

Figure 725: User Manual - Mobile - View details Transaction - Step 2

Step 3: From Store Dashboard Screen, choose User icon tab from the bottom tab.

Figure 726: User Manual - Mobile - View details Transaction - Step 3

Step 4: Press "Ví" menu item to navigate to Wallet Screen.

Figure 727: User Manual - Mobile - View details Transaction – Step 4

Step 5: Navigate to Transaction Details Screen to show information details of transaction.

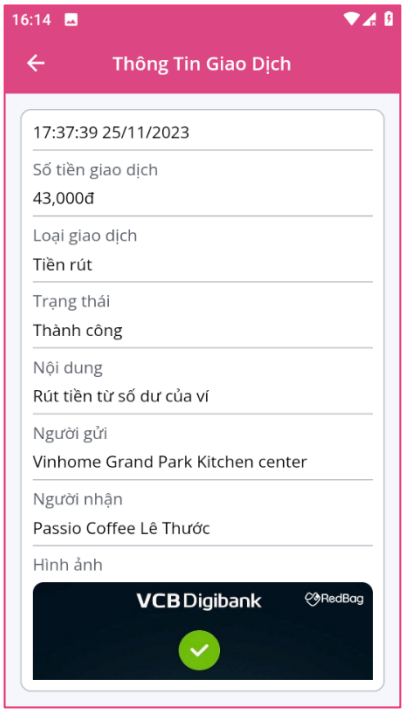


Figure 728: User Manual – Mobile - View details Transaction - Step 5