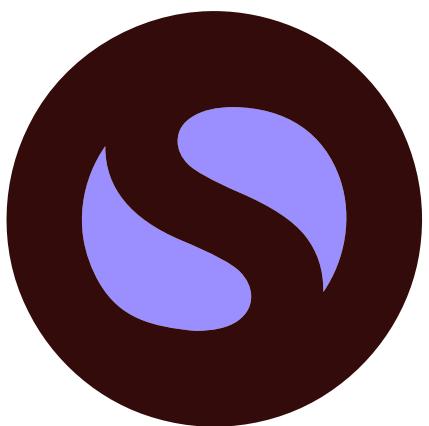
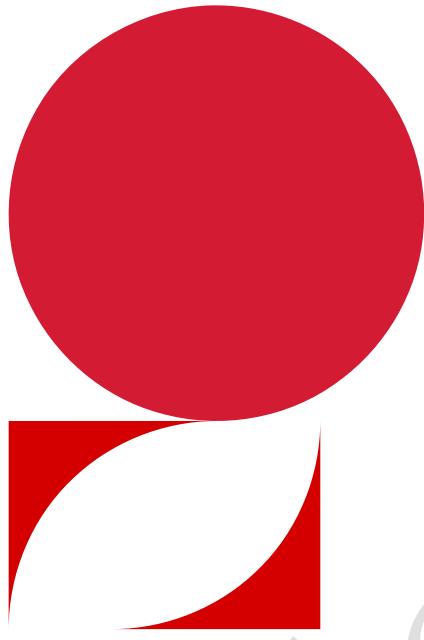


bis**terium**



ABSTRACT

With the growing trends of the internet and its usage in social networking using a platform like Facebook, Instagram etc., People seem they are diverted from their social responsibilities. They are shy and busy to involve in any social and welfare activities. With the concept of social media and social networking, The Social Service Application can be a dedicated virtual space for social workers and related enthusiasts to motivate to donate NGOs and create awareness and attention post messages, polls, petitions etc. In a nutshell, the project aims to bridge the gap between the NGOs and general people, increasing communication and work that benefits both.

This report includes the introduction to the project: Social Service Application with the scope to solve the problem domain by analysing the current scenario. The end-users are defined with a brief explanation of the project solution, similar projects, and projects comparison in this report. The considered and selected methodology along with phases are defined in the development section of this report. The Pre and Post survey are also included. SRS to document the requirements after analysis is available in the APPENDIX section. The Project development involves design figures (i.e., UML diagrams, DFDs, wireframes, prototypes) needed to construct the system. The developed system is then tested. The test plans and test reports are in the Testing and analysis section. The legal, social, and ethical issues of using the Social Service Application are also included. The overall report is then concluded with personal experience and opinion on the overall Final Year Project (FYP).

TABLE OF CONTENTS

1. INTRODUCTION	1
1.1. PROJECT DESCRIPTION	1
1.2. CURRENT SCENARIO	2
1.2.1. WORLD	2
1.2.2. NEPAL	2
1.3. PROBLEMS DOMAIN AND PROJECT AS A SOLUTION	2
1.4. AIM AND OBJECTIVES	3
1.5. STRUCTURE OF THE REPORT	4

1.5.1. BACKGROUND	4
1.5.2. DEVELOPMENT	4
1.5.3. TESTING AND ANALYSIS	5
1.5.4. CONCLUSION.....	5
2. BACKGROUND	6
2.1. ABOUT THE END-USERS	6
2.2. UNDERSTANDING THE SOLUTION	6
2.2.1. RESTFUL API.....	7
2.3. SIMILAR PROJECTS	7
2.3.1. SYSTEM A: CHANGE.ORG.....	7
2.4. COMPARISONS.....	9
3. DEVELOPMENT	11
3.1. CONSIDERED METHODOLOGY	11
3.2. SELECTED METHODOLOGY.....	12
3.3. PHASES OF METHODOLOGY	12
3.4. SURVEY RESULTS.....	13
3.4.1. PRE-SURVEY RESULTS	13
3.4.2. POST-SURVEY RESULTS.....	13
3.5. REQUIREMENT ANALYSIS	14
3.6. DESIGN	15
3.6.1. ICONOGRAPHY.....	15
3.6.2. WIREFRAME.....	16
3.6.3. PROTOTYPE.....	17
3.6.4. DATABASE DESIGN	18
3.6.5. UML DESIGN	21

3.7. IMPLEMENTATION.....	39
4. TESTING AND ANALYSIS	43
4.1. TEST PLAN.....	43
4.1.1. UNIT TESTING, TEST PLAN.....	43
4.1.2. SYSTEM TESTING, TEST PLAN	46
4.2. UNIT TESTING.....	48
4.2.1. TEST DJANGO WEB-APPLICATION	48
4.2.2. TEST RESTFUL APIs	56
4.2.3. TEST DART DATA CLASSES	61
4.3. SYSTEM TESTING	70
4.3.1. ADMIN-STAFF PORTAL TESTING.....	70
4.3.2. SASAE MOBILE APP TESTING	73
4.4. USER ACCEPTANCE TESTING	82
4.5. CRITICAL ANALYSIS	82
5. CONCLUSION.....	83
5.1. LEGAL, SOCIAL AND ETHICAL ISSUES	84
5.1.1. LEGAL ISSUES	84
5.1.2. SOCIAL ISSUES.....	84
5.1.3. ETHICAL ISSUES	84
5.2. ADVANTAGES	84
5.3. LIMITATIONS.....	85
5.4. FUTURE WORK.....	85
6. REFERENCES AND BIBLIOGRAPHY	86
7. APPENDIX.....	92
7.1. APPENDIX A: PRE-SURVEY.....	92

7.1.1. PRE-SURVEY FORM	92
7.1.2. SAMPLE OF FILLED PRE-SURVEY FORMS	94
7.1.3. PRE-SURVEY RESULT.....	96
2. APPENDIX B: POST-SURVEY.....	101
7.2.1. POST-SURVEY FORM	101
7.2.2. SAMPLE OF FILLED POST-SURVEY FORMS	103
7.2.3. POST-SURVEY RESULT.....	105
3. APPENDIX C: REQUIREMENT ANALYSIS	108
7.3.1. SOFTWARE REQUIREMENTS SPECIFICATION (SRS).....	108
4. APPENDIX D: BACKGROUND.....	115
7.4.1. UNDERSTANDING THE SOLUTION	115
7.4.2. SIMILAR PROJECTS	116
5. APPENDIX E: CONSIDERED METHODOLOGY	119
7.5.1. EXTREME PROGRAMMING	119
7.5.2. KANBAN.....	120
6. APPENDIX F: PHASES OF METHODOLOGY	121
7.6.1. BACKLOG	121
7.6.2. REQUIREMENTS	121
7.6.3. DESIGN	121
7.6.4. DEVELOPMENT	121
7.6.5. TESTING	122
7.6.6. DEPLOYMENT	122
7. APPENDIX G: SAMPLE CODES	123
7.7.1. SAMPLE CODE OF THE UI	123
7.7.2. SAMPLE CODE FOR THE TEST AUTOMATION SCRIPT	131

7.8. APPENDIX H: DESIGN	135
7.8.1. GANTT CHART	135
7.8.2. WORK BREAKDOWN STRUCTURE	136
7.8.3. ALGORITHMS & FLOWCHARTS	137
7.8.4. DATABASE DESIGN	139
7.8.5. LEVEL-1 DFDS	151
7.8.6. USE CASES.....	159
7.8.7. COMMUNICATION/COLLABORATION DIAGRAMS	164
7.8.8. SEQUENCE DIAGRAMS	174
7.8.9. WIREFRAMES	202
7.8.10. PROTOTYPES	238
7.9. APPENDIX I: IMPLEMENTATION.....	249
7.10. APPENDIX J: SCREENSHOTS OF THE SYSTEM.....	258
7.11. APPENDIX K: TEST	295
7.11.1. SYSTEM TESTING	295
7.12. APPENDIX L: USER FEEDBACK	372
7.12.1. USER FEEDBACK FORM	372
7.12.2. SAMPLE OF FILLED USER FEEDBACK FORMS	373
7.12.3. USER FEEDBACK RESULT.....	374
7.13. APPENDIX M: FUTURE WORK.....	375
7.13.1. READINGS FOR FUTURE WORK	375

TABLE OF FIGURES

Figure 1: End-Users diagrammatic representation	6
Figure 2: RESTful API Model (Hirani, 2020)	7
Figure 3: Change.org - Discover petitions (Change.org, PBC, 2021)	8
Figure 4: Change.org - Creating a Petition (Start a petition · Change.org, 2021)	9
Figure 5: Waterfall Methodology (Team, 2013).....	12
Figure 6: Application branding Icon	16
Figure 7: Wireframe - Login Screen	17
Figure 8: Prototype, Login	18
Figure 9: Prototype, Reset Password	18
Figure 10: Prototype, Register	18
Figure 11: Database Design, Abstract ERD	20
Figure 12: Use Case Diagram	22
Figure 13: Level-0 DFD (Context Diagram)	24
Figure 14: Level-1 DFD, UC1	25
Figure 15: Communication/Collaboration Diagram – UC1.....	26
Figure 16: Communication/Collaboration Diagram – UC2.....	26
Figure 17: Communication/Collaboration Diagram – UC3.....	27
Figure 18: Communication/Collaboration Diagram – UC4.....	27
Figure 19: Communication/Collaboration Diagram – UC5.....	28
Figure 20: Sequence Diagram - UC1	29
Figure 21: Sequence Diagram - UC2	30
Figure 22: Sequence Diagram - UC3	31
Figure 23: Sequence Diagram - UC4	32
Figure 24: Sequence Diagram - UC5	33
Figure 25: Django Backend Class Diagram.....	34
Figure 26: Flutter Frontend Class Diagram	35
Figure 27: Flutter Frontend Class Diagram (Contd.)	36
Figure 28: Flutter Frontend Class Diagram (Contd.)	37
Figure 29: Flutter Frontend Class Diagram (Contd.)	38
Figure 30: Flutter Frontend Class Diagram (Contd.)	39
Figure 31: Kanban Backlog	40

Figure 32: Kanban Backlog (Contd.)	40
Figure 33: Kanban Backlog (Contd.)	41
Figure 34: Kanban Backlog (Contd.)	41
Figure 35: Kanban Backlog (Contd.)	42
Figure 36: Kanban, Requirements Phase	42
Figure 37: Pre-Survey Form (Google Forms)	43
Figure 38: Unit Testing Django Web-App TC-1 Result Screenshot [ERROR]	49
Figure 39: Unit Testing Django Web-App TC-1 Screenshot [SOLUTION]	50
Figure 40: Unit Testing Django Web-App TC-1 Result Screenshot (NEW)	50
Figure 41: Unit Testing, Django Web-App TC-2 Result Screenshot	51
Figure 42: Unit Testing, Django Web-App TC-3 Result Screenshot	52
Figure 43: Unit Testing, Django Web-App TC-4 Result Screenshot	53
Figure 44: Unit Testing, Django Web-App TC-5 Result Screenshot	54
Figure 45: Unit Testing, Django Web-App TC-6 Result Screenshot	55
Figure 46: Overall Unit Testing Result: Django Web-Application	56
Figure 47: Unit Testing, RESTful API TC-1 Result Screenshot	57
Figure 48: Unit Testing, RESTful API TC-2 Result Screenshot	58
Figure 49: Unit Testing, RESTful API TC-3 Result Screenshot	59
Figure 50: Unit Testing, RESTful API TC-4 Result Screenshot	60
Figure 51: Unit Testing, RESTful API TC-5 Result Screenshot	61
Figure 52: Unit Testing, RESTful API TC-6 Result Screenshot	62
Figure 53: Unit Testing, RESTful API TC-7 Result Screenshot	63
Figure 54: Unit Testing, Dart Data Class TC-1 Result Screenshot	64
Figure 55: Unit Testing, Dart Data Class TC-2 Result Screenshot	65
Figure 56: Unit Testing, Dart Data Class TC-3 Result Screenshot	66
Figure 57: Unit Testing, Dart Data Class TC-4 Result Screenshot	67
Figure 58: Unit Testing, Dart Data Class TC-5 Result Screenshot	68
Figure 59: Unit Testing, Dart Data Class TC-6 Result Screenshot [ERROR]	69
Figure 60: Unit Testing, Dart Data Class TC-6 Screenshot [SOLUTION]	70
Figure 61: Unit Testing, Dart Data Class TC-6 Result Screenshot (NEW).....	70
Figure 62: Unit Testing, Dart Data Class TC-7 Result Screenshot	71
Figure 63: Unit Testing, Dart Data Class TC-8 Result Screenshot	72

Figure 64: Login Page with admin credentials (Before)	73
Figure 65: Admin Homepage (After)	74
Figure 66: System Testing, Sasae Mobile App TC-1	75
Figure 67: Mobile, Login Screen	76
Figure 68: Mobile, Filled Registration Form	77
Figure 69: Mobile, Login Screen with a Success Message (After)	78
Figure 70: Mobile, Profile Update Form with changed Data (Before)	79
Figure 71: Mobile, Update Profile Prompt	80
Figure 72: Mobile, General People Profile with a Success Message (After)	81
Figure 73: General People Account Delete Prompt (Before)	82
Figure 74: Mobile Login Screen with Success Message (After)	83
Figure 75: Pre-Survey Form	94
Figure 76: Pre-Survey Form Contd.....	95
Figure 77: Filled Pre-Survey Sample	96
Figure 78: Filled Pre-Survey Sample Contd.	97
Figure 79: Pre-Survey Question-1 Result	98
Figure 80: Pre-Survey Question-2 Result	98
Figure 81: Pre-Survey Question-3 Result	99
Figure 82: Pre-Survey Question-4 Result	99
Figure 83: Pre-Survey Question-5 Result	100
Figure 84: Pre-Survey Question-6 Result	100
Figure 85: Pre-Survey Question-7 Result	101
Figure 86: Pre-Survey Question-8 Result	101
Figure 87: Pre-Survey Question-9 Result	102
Figure 88: Pre-Survey Question-10 Result	102
Figure 89: Post-Survey Form	103
Figure 90: Post-Survey Form Contd.	104
Figure 91: Filled Post-Survey Form	105
Figure 92:Filled Post-Survey Form Contd.	106
Figure 93: Post-Survey, Question-1 Result	107
Figure 94: Post-Survey, Question-2 Result	107
Figure 95: Post-Survey, Question-3 Result	108

Figure 96: Post-Survey, Question-4 Result	108
Figure 97: Post-Survey, Question-5 Result	109
Figure 98: Post-Survey, Question-6 Result	109
Figure 99: Post-Survey, Question-7 Result	110
Figure 100: Post-Survey, Question-8 Result	110
Figure 101: System Architecture	118
Figure 102: Three-Tier Architecture	119
Figure 103: NRCS Website - Donate (NRCS, 2021)	120
Figure 104: Facebook page - NGO Federation of Nepal (Meta, 2021)	121
Figure 105: Extreme Programming Methodology (Team, 2021)	122
Figure 106: Kanban Methodology (Lerche-Jensen, 2019)	123
Figure 107: Old Gantt Chart	139
Figure 108: New Gantt Chart (Because of Deadline Extension)	139
Figure 109: Work Breakdown Structure (WBS)	140
Figure 110: Auto Login Activity (Session-based)	141
Figure 111: Upvote/Downvote Activity	142
Figure 112: Database Design, Generated Concrete ERD	153
Figure 113: Level-1 DFD, UC2	154
Figure 114: Level-1 DFD, UC3	154
Figure 115: Level-1 DFD, UC4	154
Figure 116: Level-1 DFD, UC5	154
Figure 117: Level-1 DFD, UC6	155
Figure 118: Level-1 DFD, UC7	155
Figure 119: Level-1 DFD, UC8	156
Figure 120: Level-1 DFD, UC9	156
Figure 121: Level-1 DFD, UC10	157
Figure 122: Level-1 DFD, UC11	157
Figure 123: Level-1 DFD, UC12	157
Figure 124: Level-1 DFD, UC13	158
Figure 125: Level-1 DFD, UC14	158
Figure 126: Level-1 DFD, UC15	158
Figure 127: Level-1 DFD, UC16	159

Figure 128: Level-1 DFD, UC17	159
Figure 129: Level-1 DFD, UC18	159
Figure 130: Level-1 DFD, UC19	160
Figure 131: Level-1 DFD, UC20	160
Figure 132: Level-1 DFD, UC21	161
Figure 133: Level-1 DFD, UC22	161
Figure 134: Level-1 DFD, UC23	162
Figure 135: Communication/Collaboration Diagram – UC6	168
Figure 136: Communication/Collaboration Diagram – UC7	169
Figure 137: Communication/Collaboration Diagram – UC8	169
Figure 138: Communication/Collaboration Diagram – UC8	169
Figure 139: Communication/Collaboration Diagram – UC10	170
Figure 140: Communication/Collaboration Diagram – UC11	170
Figure 141: Communication/Collaboration Diagram – UC12	171
Figure 142: Communication/Collaboration Diagram – UC13	171
Figure 143: Communication/Collaboration Diagram – UC14	172
Figure 144: Communication/Collaboration Diagram – UC15	172
Figure 145: Communication/Collaboration Diagram – UC16	173
Figure 146: Communication/Collaboration Diagram – UC17	173
Figure 147: Communication/Collaboration Diagram – UC18	174
Figure 148: Communication/Collaboration Diagram – UC19	174
Figure 149: Communication/Collaboration Diagram – UC20	175
Figure 150: Communication/Collaboration Diagram – UC21	175
Figure 151: Communication/Collaboration Diagram – UC22	176
Figure 152: Communication/Collaboration Diagram – UC23	176
Figure 153: Sequence Diagram - UC6 (Create)	177
Figure 154: Sequence Diagram - UC6 (Update).....	178
Figure 155: Sequence Diagram – UC6 (View)	179
Figure 156: Sequence Diagram – UC6 (Delete) <i>applies to all kinds of post types.</i>	180
Figure 157: Sequence Diagram – UC7 (Create)	181
Figure 158: Sequence Diagram – UC7 (Update)	182
Figure 159: Sequence Diagram – UC7 (View)	183

Figure 160: Sequence Diagram – UC8 (Create)	184
Figure 161: Sequence Diagram – UC8 (Update)	185
Figure 162: Sequence Diagram – UC8 (View)	186
Figure 163: Sequence Diagram – UC9	187
Figure 164: Sequence Diagram – UC10	188
Figure 165: Sequence Diagram – UC11	189
Figure 166: Sequence Diagram – UC12	190
Figure 167: Sequence Diagram – UC13	191
Figure 168: Sequence Diagram – UC14	192
Figure 169: Sequence Diagram – UC15 (Create)	193
Figure 170: Sequence Diagram – UC15 (Delete)	194
Figure 171: Sequence Diagram – UC15 (Update)	195
Figure 172: Sequence Diagram – UC15 (View)	196
Figure 173: Sequence Diagram – UC16	197
Figure 174: Sequence Diagram – UC17	198
Figure 175: Sequence Diagram – UC18	199
Figure 176: Sequence Diagram – UC19	200
Figure 177: Sequence Diagram – UC20	201
Figure 178: Sequence Diagram – UC21	202
Figure 179: Sequence Diagram – UC22	203
Figure 180: Sequence Diagram – UC23	204
Figure 181: Web Login	205
Figure 182: Web Forgot Password	205
Figure 183: Web Change Password	206
Figure 184: Web Admin Home	206
Figure 185: Web Staff Home	207
Figure 186: Web Reported Post Review	207
Figure 187: Web Staff Tab	208
Figure 188: Web NGO Tab	208
Figure 189: Web People Tab	209
Figure 190: Web Create Staff	210
Figure 191: Web Create NGO	211

Figure 192: Web Update NGO	212
Figure 193: Web Update Staff and People	212
Figure 194: Web View Staff and People	213
Figure 195: Web View NGO	214
Figure 196: Mobile Login	215
Figure 197: Mobile Forgot Password	216
Figure 198: Mobile Password Change	217
Figure 199: Mobile Home Tab	218
Figure 200: Mobile Post Tab	219
Figure 201: Mobile Post Types	220
Figure 202: Mobile NGO Tab	221
Figure 203: Mobile NGO Details	222
Figure 204: Mobile People User Register	223
Figure 205: Mobile Settings.....	224
Figure 206: Mobile Notifications.....	225
Figure 207: Mobile User Profile	226
Figure 208: Wireframe – Reset Password Screen (Updated)	227
Figure 209: Wireframe – General People Register Screen (Updated)	228
Figure 210: Wireframe - Posts Screen (Updated)	229
Figure 211: Wireframe – Different Types of Post Screen (Updated)	230
Figure 212: Wireframe – Post Type Widgets (Updated)	231
Figure 213: Wireframe – Post-Create Screen (Updated)	232
Figure 214: Wireframe – NGOs Screen (Updated)	233
Figure 215: Wireframe – NGO Detail Screen (Updated)	234
Figure 216: Wireframe – NGO Donation Screen (Updated)	235
Figure 217: Wireframe –General People Profile Screen (Updated)	236
Figure 218: Wireframe – Change Password Screen (Updated)	237
Figure 219: Wireframe – Update General People Screen (Updated)	238
Figure 220: Wireframe - Notification Screen (Updated)	239
Figure 221: Wireframe - Logout Screen (Updated)	240
Figure 222: Prototype, Posts	241
Figure 223: Prototype, Post a Post	241

Figure 224: Prototype, Profile Screen	241
Figure 225: Prototype, Account Delete	242
Figure 226: Prototype, Change Password	242
Figure 227: Prototype, User's Posts.....	242
Figure 228: Prototype, Update Profile	243
Figure 229: Prototype, NGOs	243
Figure 230: Prototype, NGOs filtration	243
Figure 231: Prototype, NGO Profile	244
Figure 232: Prototype, NGO Donation	244
Figure 233: Prototype, NGO's Posts	244
Figure 234: Prototype, Poll Post	245
Figure 235: Prototype, Request Post	245
Figure 236: Prototype, Normal Post	245
Figure 237: Prototype, Notifications.....	246
Figure 238: Prototype, Clear Notifications	246
Figure 239: Prototype, Settings.....	246
Figure 240: Kanban, Design Phase	247
Figure 241: Communication/Collaboration Diagram on Draw.io	248
Figure 242: Sequence Diagram using PlantUML	248
Figure 243: Class Diagram on PyCharm IDE.....	249
Figure 244: Icon Design on Inkscape	249
Figure 245: Kanban, Development Phase (Admin Staff Portal)	250
Figure 246: Project's Backend Dependencies	251
Figure 247: Kanban, Development Phase (RESTful APIs)	251
Figure 248: Staff/Admin Home Feature Implementation as per Kanban	252
Figure 249: Code Error during Development	253
Figure 250: Kanban, Continuous Development and Testing	253
Figure 251: Kanban, Development Phase (Sasae Mobile App)	254
Figure 252: Kanban, Deployment Phase	255
Figure 253: System: Admin-Staff Portal, log in	256
Figure 254: System: Admin-Staff Portal, Password Change	256
Figure 255: System: Admin-Staff Portal, Logout	257

Figure 256: System: Admin-Staff Portal, Password Reset	257
Figure 257: System: Admin-Staff Portal, Password Reset Confirmation	258
Figure 258: System: Admin-Staff Portal, Admin Home	258
Figure 259: System, Admin-Staff Portal, Staff Home	259
Figure 260: System: Admin-Staff Portal, List of Staff	259
Figure 261: System: Admin-Staff Portal, List of NGOs.....	260
Figure 262: System: Admin-Staff Portal, List of General People	260
Figure 263: System: Admin-Staff Portal, List of Reported Posts	261
Figure 264: System: Admin-Staff Portal, Reported Post Detail	261
Figure 265: System: Admin-Staff Portal, List of Reviewed Reports	262
Figure 266: System: Admin-Staff Portal, Reviewed Report Detail	262
Figure 267: System: Admin-Staff Portal, NGO Detail	263
Figure 268: System: Admin-Staff Portal, NGO Detail Update	263
Figure 269: System: Sasae App, log in	264
Figure 270: System: Sasae App, Password Reset	265
Figure 271: System: Sasae App, General People Registration	266
Figure 272: System: Sasae App, List of Posts	267
Figure 273: System: Sasae App, Post a Post	268
Figure 274: System: Sasae App, Post a Normal Post	269
Figure 275: System: Sasae App, Post a Poll Post	270
Figure 276: System: Sasae App, Post a Request Post	271
Figure 277: System: Sasae App, Normal Post	272
Figure 278: System: Sasae App, Request Post	273
Figure 279: System: Sasae App, Poll Post	274
Figure 280: System: Sasae App, General People Profile	275
Figure 281: System: Sasae App, Profile Update	276
Figure 282: System: Sasae App, General People Account Deletion	277
Figure 283: System: Sasae App, Change Password.....	278
Figure 284: System: Sasae App, User's Post	279
Figure 285: System: Sasae App, Post Edit or Delete	280
Figure 286: System: Sasae App, Post Update	281
Figure 287: System: Sasae App, List of NGOs	282

Figure 288: System: Sasae App, NGOs Filtration by Field of Work	283
Figure 289: System: Sasae App, Search NGO(s)	284
Figure 290: System: Sasae App, NGO Profile Screen	285
Figure 291: System: Sasae App, Donation to NGO	286
Figure 292: System: Sasae App, Khalti Payment as Donation	287
Figure 293: System: Sasae App, Notifications	288
Figure 294: System: Sasae App, Setting	289
Figure 295: System: Sasae App, Logout	290
Figure 296: System: Sasae App, About App	291
Figure 297: System: Sasae App, App Licenses	292
Figure 298: Staffs Page	295
Figure 299: Filled Staff Registration Form (Before)	296
Figure 300: Staff Profile Page (After)	297
Figure 301: Staff Update Form with Changed Field Data (Before)	298
Figure 302: Confirm Update Prompt	298
Figure 303: Staff Profile Page (After)	299
Figure 304: Staff Delete Prompt (Before)	300
Figure 305: Staffs Page (After)	300
Figure 306: NGOs Page	301
Figure 307: NGO Registration Page (Before)	302
Figure 308: NGO Profile Page (After).....	303
Figure 309: NGO Update Page with changed Field Data (Before)	304
Figure 310: NGO Profile Page (After).....	305
Figure 311: NGO Delete Prompt (Before).....	306
Figure 312: NGOs Page (After)	306
Figure 313: General People Page	307
Figure 314: General People Profile	308
Figure 315: Update General People Form with changed Field Data (Before)	309
Figure 316: General People Profile Page (After)	310
Figure 317: General People Delete Prompt (Before)	311
Figure 318: General People Page (After)	311
Figure 319: Reported Posts Page (before)	312

Figure 320: Reviewing the Reported Post	313
Figure 321: Reported Post Page (After).....	313
Figure 322: Reviewed Reports Page	315
Figure 323: Read Reviewed Post Page	316
Figure 324: Logout Prompt (Before)	317
Figure 325: Successful Logout Message Page (After)	318
Figure 326: Password Change Page with filled fields (Before)	319
Figure 327: Password Change Successful Message Page (After)	320
Figure 328: Mobile, Login Screen with filled credentials (Before).....	321
Figure 329: Mobile, Posts Screen (After)	322
Figure 330: Mobile, Change Password Modal Bottom Sheet with filled data (Before)	324
Figure 331: Mobile, General People Profile (After)	325
Figure 332: Mobile, View NGOs.....	327
Figure 333: Mobile, NGO Profile	328
Figure 334: Mobile, NGOs Filtration by Field of Work (Before)	330
Figure 335: Mobile, Updated NGOs List (After)	331
Figure 336: Mobile, NGO Donation with filled Amount (Before)	333
Figure 337: Mobile, Khalti Payment Screen.....	334
Figure 338: Mobile, NGO Profile with Success Message (After)	335
Figure 339: Mobile, Create Post Screen with filled Data (Before)	337
Figure 340: Mobile, User Posts Tab (After)	338
Figure 341: Mobile, Edit-Delete Post Modal Sheet	339
Figure 342: Mobile, Post Update Form Screen with changed Data (Before)	340
Figure 343: Mobile, User's Posts Tab (After)	341
Figure 344: Mobile Post Delete Dialog (Before)	342
Figure 345: Mobile, User's Post Tab with Success Message (After)	343
Figure 346: Mobile, Posts Page	345
Figure 347: Mobile, Normal Post Screen	346
Figure 348: Mobile, Poll Post Screen	347
Figure 349:Mobile, Request Post Screen	348
Figure 350: Mobile, Normal Post Screen (Before)	350
Figure 351: Mobile, Normal Post Screen (After)	351

Figure 352: Mobile, Poll Post Screen (Before)	352
Figure 353: Mobile, Poll Post Screen (After)	353
Figure 354: Mobile, Request Post Screen (Before)	354
Figure 355:Mobile, Request Post Participate Prompt	355
Figure 356: Mobile, Request Post Screen (After)	356
Figure 357: Mobile, Normal Post Upvoted (Before)	358
Figure 358: Mobile, Post Reaction Push Notification. (After)	359
Figure 359: Mobile, Notification Page (After)	360
Figure 360: Web, Review Reported Posts (Before).....	361
Figure 361: Mobile, Push Notification for Reported Post Remove (After)	362
Figure 362: Mobile, Notification Page (After)	363
Figure 363: Mobile, Setting Page	365
Figure 364: Mobile, Logout Prompt (Before).....	366
Figure 365: Mobile, Login Screen (After)	367
Figure 366: User Feedback Form	368
Figure 367: Filled User Feedback Form Sample	369
Figure 368: User Feedback, Question-2 Result	370
Figure 369: User Feedback, Question-3 Result	370
Figure 370: User Feedback, Question-4 Result	371
Figure 371: User Feedback, Question-5 Result	371

TABLE OF TABLES

Table 1: Data Dictionary, User Table	19
Table 2: Actor & Use-Cases in Tabular View with Use-Case ID (UC)	23
Table 3: Unit Testing, Django Web-App Test Plans	44
Table 4: Unit Testing, RESTful API Test Plans	46
Table 5: Unit Testing, Dart Data Class Test Plans	46
Table 6: System Testing, Admin-Staff Portal Test Plans	47
Table 7: System Testing, Sasae Mobile App Test Plans	48
Table 8: Unit Testing, Django Web-App TC-1	49
Table 9: Unit Testing, Django Web-App TC-2	51
Table 10: Unit Testing, Django Web-App TC-3	52

Table 11: Unit Testing, Django Web-App TC-4	53
Table 12: Unit Testing, Django Web-App TC-5	54
Table 13: Unit Testing, Django Web-App TC-6	55
Table 14: Unit Testing, RESTful API TC-1	57
Table 15: Unit Testing, RESTful API TC-2	58
Table 16: Unit Testing, RESTful API TC-3	59
Table 17: Unit Testing, RESTful API TC-4	60
Table 18: Unit Testing, RESTful API TC-5	61
Table 19: Unit Testing, RESTful API TC-6	62
Table 20: Unit Testing, RESTful API TC-7	63
Table 21: Unit Testing, Dart Data Class TC-1	64
Table 22: Unit Testing, Dart Data Class TC-2	65
Table 23: Unit Testing, Dart Data Class TC-3	66
Table 24: Unit Testing, Dart Data Class TC-4	67
Table 25: Unit Testing, Dart Data Class TC-5	68
Table 26: Unit Testing, Dart Data Class TC-6	69
Table 27: Unit Testing, Dart Data Class TC-7	71
Table 28: Unit Testing, Dart Data Class TC-8	72
Table 29: System Testing, Admin-Staff Portal TC-1	73
Table 30: Data Dictionary, PeopleUser Table	144
Table 31: Data Dictionary, Staff Table	145
Table 32: Data Dictionary, NGOUser Table	147
Table 33: Data Dictionary, Bank Table	147
Table 34: Data Dictionary, Post Table	148
Table 35: Data Dictionary, PostNormal Table	149
Table 36: Data Dictionary, PostPoll Table	150
Table 37: Data Dictionary, PostRequest Table	151
Table 38: Data Dictionary, PollOption Table	152
Table 39: Data Dictionary, Report Table	152
Table 40: High-Level UC1 Description	162
Table 41: High-Level UC2 Description	162
Table 42: High-Level UC3 Description	163

Table 43: High-Level UC4 Description	163
Table 44: High-Level UC5 Description	163
Table 45: High-Level UC6 Description	163
Table 46: High-Level UC7 Description	164
Table 47: High-Level UC8 Description	164
Table 48: High-Level UC9 Description	164
Table 49: High-Level UC10 Description	164
Table 50: High-Level UC11 Description	165
Table 51: High-Level UC12 Description	165
Table 52: High-Level UC13 Description	165
Table 53: High-Level UC14 Description	165
Table 54: High-Level UC15 Description	166
Table 55: High-Level UC16 Description	166
Table 56: High-Level UC17 Description	166
Table 57: High-Level UC18 Description	166
Table 58: High-Level UC19 Description	166
Table 59: High-Level UC20 Description	167
Table 60: High-Level UC21 Description	167
Table 61: High-Level UC22 Description	167
Table 62: High-Level UC23 Description	167
Table 63: System Testing, Admin-Staff Portal TC-2	293
Table 64: Login page with Staff Credentials (Before).....	293
Table 65: Staff Homepage (After)	294
Table 66: System Testing, Admin-Staff Portal TC-3	295
Table 67: System Testing, Admin-Staff Portal TC-4	301
Table 68: System Testing, Admin-Staff Portal TC-5	307
Table 69: System Testing, Admin-Staff Portal TC-6	312
Table 70: System Testing, Admin-Staff Portal TC-7	314
Table 71: System Testing, Admin-Staff Portal TC-8	317
Table 72: System Testing, Admin-Staff Portal TC-9	319
Table 73: System Testing, Sasae Mobile App TC-2.....	320
Table 74: System Testing, Sasae Mobile App TC-3.....	323

Table 75: System Testing, Sasae Mobile App TC-4.....	326
Table 76: System Testing, Sasae Mobile App TC-5.....	329
Table 77: System Testing, Sasae Mobile App TC-6.....	332
Table 78: System Testing, Sasae Mobile App TC-7.....	336
Table 79: System Testing, Sasae Mobile App TC-8.....	344
Table 80: System Testing, Sasae Mobile App TC-9.....	349
Table 81: System Testing, Sasae Mobile App TC-10.....	357
Table 82: System Testing, Sasae Mobile App TC-11.....	361
Table 83: System Testing, Sasae Mobile App TC-12.....	364

TABLE OF ABBREVIATIONS

CRUD	Create, Read, Update and Delete
API	Application Programming Interface
REST	Representational State Transfer
JSON	JavaScript Object Notation
HTTP	HyperText Transfer Protocol
CSRF	Cross-Site Request Forgery
INTERNET	Interconnected Network
NGO	Non-Governmental Organization
UML	Unified Modelling Language
WAN	Wide Area Network
LAN	Local Area Network
ODBC	Open Database Connectivity
UI/UX	User Interface/User Experience
MVC	Model View Controller
MVVM	Model-View-ViewModel
MVT	Model View Template

1. INTRODUCTION

1.1. PROJECT DESCRIPTION

In the final year of graduation, an undergraduate is encouraged and tasked with a Final Year Project (FYP). The final year project involves the development and documentation in a behavioural and structured manner of any field related project of choice that compiles the overall practical knowledge and learning outcome from the research and completed courses. An individual student is kept under the supervision of external and internal supervisors throughout the end of project completion with the expected outcome of a full-stack application mostly.

Technological and infrastructural advancement has long proven human intelligence to be the superior and successor of other living species on earth with dominance. One of the exponentially up-rising web-based platforms, social media has changed the way people live their lives. It has been a major source of news, user-generated content (i.e., photos, videos) and information, featuring communication channels between the users and beyond social and local boundaries (Mohsin, 2021). Social media is generic based on user interaction with the service. The usability varies among people. From a wide range of categories, some people are using social media to empower and liberate other people, while others are using it to promote social cohesion, change and development. The voluntary group of those people can form a Non-Governmental Organization or NGO to execute social works. Operating independently of the government, NGOs depend upon contribution and funding to provide aid and development.

Currently, multiple social media (i.e., Facebook, Twitter, Instagram) are active on the internet, but there is no dedicated platform for NGOs and people with common interests. The lack of dedicated social media for social work enthusiasts led to the motivation for the Final Year Project to create **Social Service Application**. The final project's product name is **Sasae**. The end-users of the mobile application are general people and NGOs. Social Service Application bridges the communication gap and improves awareness between the NGOs and people to execute social work more effectively and efficiently. Like social media, the application liberates sharing thoughts, ideas, or any information in the form of a post.

1.2. CURRENT SCENARIO

1.2.1. WORLD

There are an estimated ten million (non-governmental organizations) NGOs in the world. The number of people donating to non-governmental organizations (NGOs) increased by 1.4 billion in 2014, up from 1.2 billion in 2011. By 2030, the number is expected to reach 2.5 billion. The value of volunteering is estimated to be \$23.07 per hour. As a result, the value of the 7.7 billion hours of volunteer labour accomplished by 62.6 million Americans, or 25.4% of the adult population, in 2013 was \$173 billion. Nongovernmental organizations (NGOs) are thought to make it simple to contribute to positive social change by 80% of people around the world. (International Businesses Standards Organization (IBSO), 2015).

1.2.2. NEPAL

Because of the country's political instability, Nepal is currently going through a particularly difficult era in its development. The Nepalese government's acceptance of this reality has resulted in a large expansion in the number of NGOs in recent years, despite the country's protracted conflict and political instability making their work more challenging (Iris Kobek and Ram Pratap Thapa, 2004). The NGOs appear to be less active since they are unaware of their surroundings, which may be alleviated by using a shared platform.

1.3. PROBLEMS DOMAIN AND PROJECT AS A SOLUTION

The exponential growth of the digital world has increased social disengagement among people by addiction. People have different natures, natures of beliefs, visions, ideologies, norms, values etc. A sense of social justice exists among people, provoking social/welfare work. People are being self-busy such that almost no one cares about others, decreasing socialization. People have become socially passive and isolated such that those who wish to change are timid and unconcerned are hollow. Most of them lack a psyche of social justice, human rights, collective responsibilities, and respect. Social work has been hard to recognize. It is not because one cannot or does not want to help others but rather due to communication difficulties or autism. NGOs are difficult to reach, and their work is frequently overlooked. The shortfall of people's interaction, network, unity, and collaboration has resulted in a shortage of social work and activity. Although social media exists, it is more of a general

platform where individuals of all types congregate. In a word, the problem scenario is that there is no specific app for social work enthusiasts and NGOs.

Social media connect people from all around the world, regardless of geography or time. With similar usability to social media, the project enables NGOs and people to interact with each other and allows them to create, share and/or exchange information and awareness. Deploying this project online on the web might be a very way to alleviate existing gaps, concerns, and problems among social workers. The project provides the below feature solutions:

- Improve accessibility to NGOs
- Enable Online donation to NGOs
- Publish Shared programmes, campaigns, and other social activities

1.4. AIM AND OBJECTIVES

This project aims to bridge the communication and interaction gap among individuals and non-governmental organizations (NGOs) by offering a centralized networking platform via a mobile application for sharing, notifying, and organizing welfare programs, campaigns, and events.

The following are the objectives for achieving the project's goal and completing it:

- Conduct a comprehensive study, research and analysis of the requirements and overall aspects of this project
- Learn the Kanban framework to plan, monitor, manage and control the project effectively and efficiently.
- Analyse the possible risks, threats and issues that might arise during the project execution and plan contingency plans accordingly to handle them
- Break down the project into multiple phases of prioritized activities that can be iterated easily.
- Build proper plans and an optimal project schedule in terms of time and risk.
- Perform group surveys and interpret feedback to understand the feasibility and usability of the project and project outcome.

- Understand Three-tier/client-server architecture.
- Enrol in HTTP and RESTful API related crash course to swiftly understand methods, URI/URL, parameters, protocols, status code, headers, JSON/form body, and other components.
- Learn Django, Django REST and Flutter framework by getting through official documentation, YouTube videos, blogs, and articles.
- Acquire a good grasp on widgets, session management, state management, RESTful API calls, middleware, serialization, pagination, form, CSRF, tokens, ORM, template, route, exceptions, validations, parsers, etc.
- Learn Google's material design principle and understand UI/UX guidelines and components from the official documentation.
- Build the APIs and Web-based application for User management and report post handling by admin and staff.
- Build the Mobile-based application like social media only for social work enthusiasts (i.e., general people and NGOs)
- Validate the project's development completion by testing (i.e., Unit testing, System testing) the whole application.
- Deploy the backend in Heroku and mobile application in third part Appstore like APKPure or APKMirror
- Compile all the work activities into project documentation under the supervisors' guidance and feedback.

1.5. STRUCTURE OF THE REPORT

1.5.1. BACKGROUND

The background section summarizes the important theoretical concepts and algorithms as solutions implemented in the project's development phase. The end-users of the project outcome are also mentioned in this section, including some similar projects and comparisons.

1.5.2. DEVELOPMENT

The development section includes a list of considered methodologies with a brief description. The selected methodology along with the phases is also described in this section.

This section also includes survey results, requirement analysis, design, and Implementation subsections with explanations.

1.5.3. TESTING AND ANALYSIS

This section includes test plans for Unit and System testing with lists of multiple actual testing done after the completion of the project's development phase. Lastly, the testing is critically analysed in this section.

1.5.4. CONCLUSION

This section includes a summary of the report and project results. It also briefs the legal, ethical, and social issues and concerns on the project's usability. Nonetheless, System advantages, limitations and future work is also briefed in this section.

2. BACKGROUND

2.1. ABOUT THE END-USERS

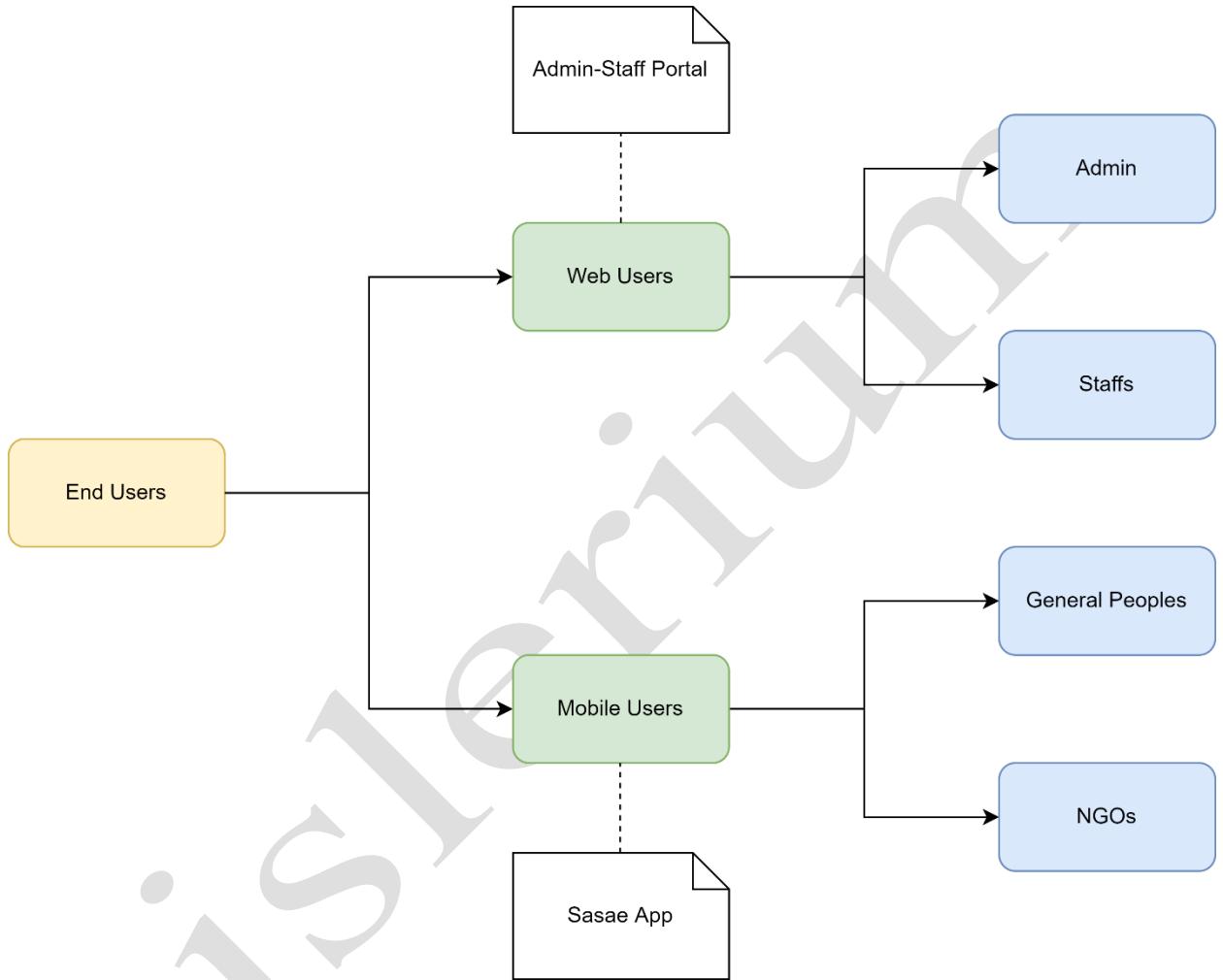


Figure 1: End-Users diagrammatic representation

This project involves multiple end-users based on the type of application (i.e., Web, Mobile) used. The NGOs and general peoples are the end-users of the Sasae app (i.e., Android Mobile Application) that is a part of the project outcome, while the admin and staffs are endusers of the Admin-Staff Portal.

2.2. UNDERSTANDING THE SOLUTION

Further Explanation is available in Appendix: [[7.4.1. UNDERSTANDING THE SOLUTION](#)].

2.2.1. RESTFUL API

A RESTful API (aka. REST API) is a web application programming interface (Web API) that adheres to the REST architectural style's constraints and allows users to interact with RESTful Web services (Red Hat, 2020). The Django Application provides the RESTful web services to the client by transferring a resource state's representation upon request via RESTful API. The representation or information is delivered via HTTP with the body in JSON format. The flutter mobile application is the REST client that consumes RESTful resources transferred by the Django Backend server upon HTTP request.

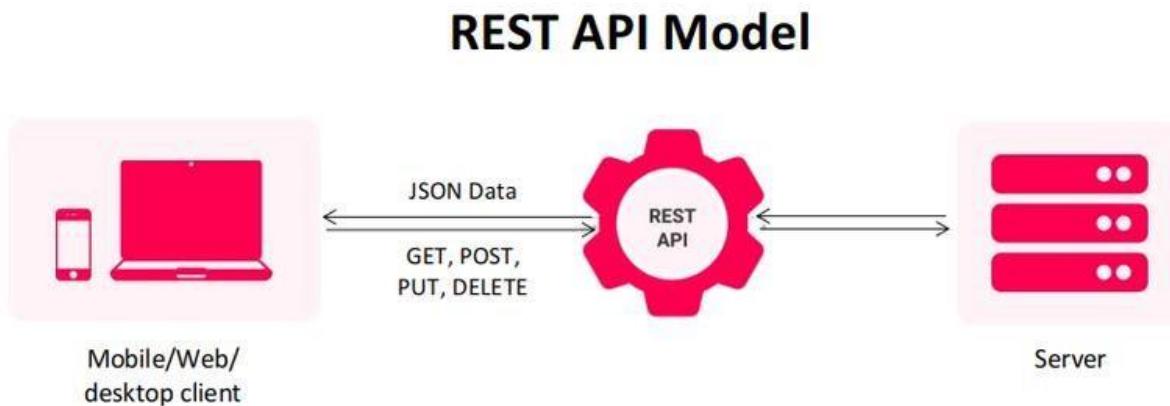


Figure 2: RESTful API Model (Hirani, 2020)

2.3. SIMILAR PROJECTS

2.3.1. SYSTEM A: CHANGE.ORG

With over 329 million users worldwide, Change.org is the world's biggest open-petition platform for social change allowing people to raise the action on the existing issues. It features 18 local teams from different countries and is available in 12 languages. The public Benefit Corporation (PBC) and a charity foundation made Change.org have a global impact. Change.org has become one of the most powerful activist platforms. Their strategy in creating a system to responsively decide any problems and issues is to create a commonly accessible platform. The platform will empower ordinary people to be campaigners, engage them for contribution and compel decision-makers to respond to public demands (Change.org, PBC, 2019). The change.org allows users to freely browse petitions and create a petition. The petition creation involves choosing related issues,

SASAE

writing the petition title, choosing recipient(s), explaining the problem(s) to be solved and adding a photo or video. The petition information is shown in a list with a photo, petition title, problem description and progress bar of total signed by a goal in a card layout.

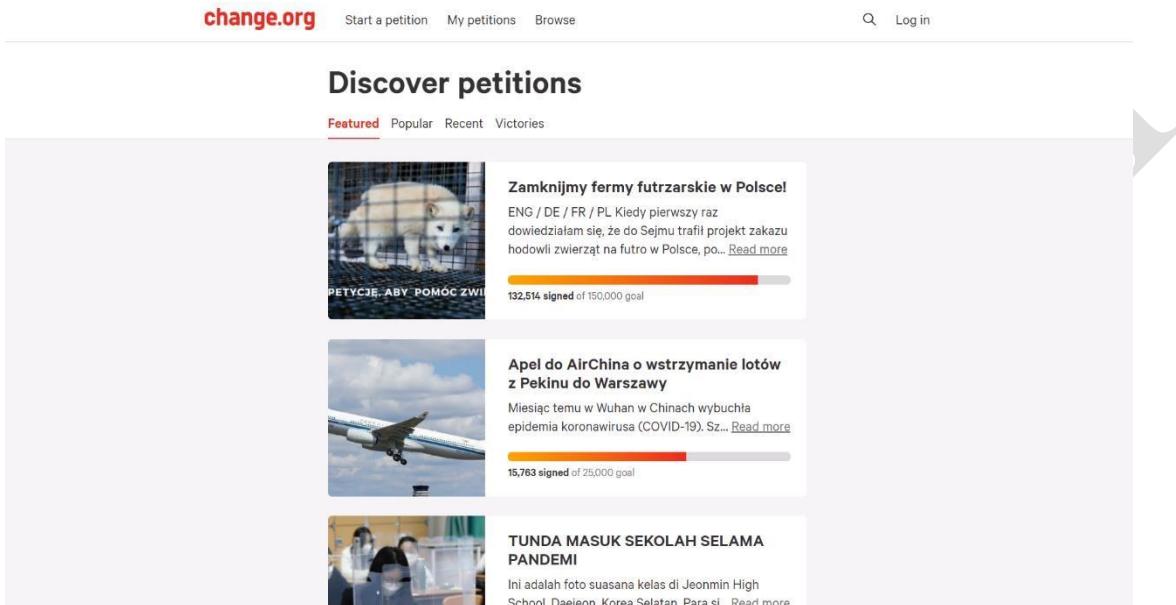


Figure 3: Change.org - Discover petitions (Change.org, PBC, 2021)

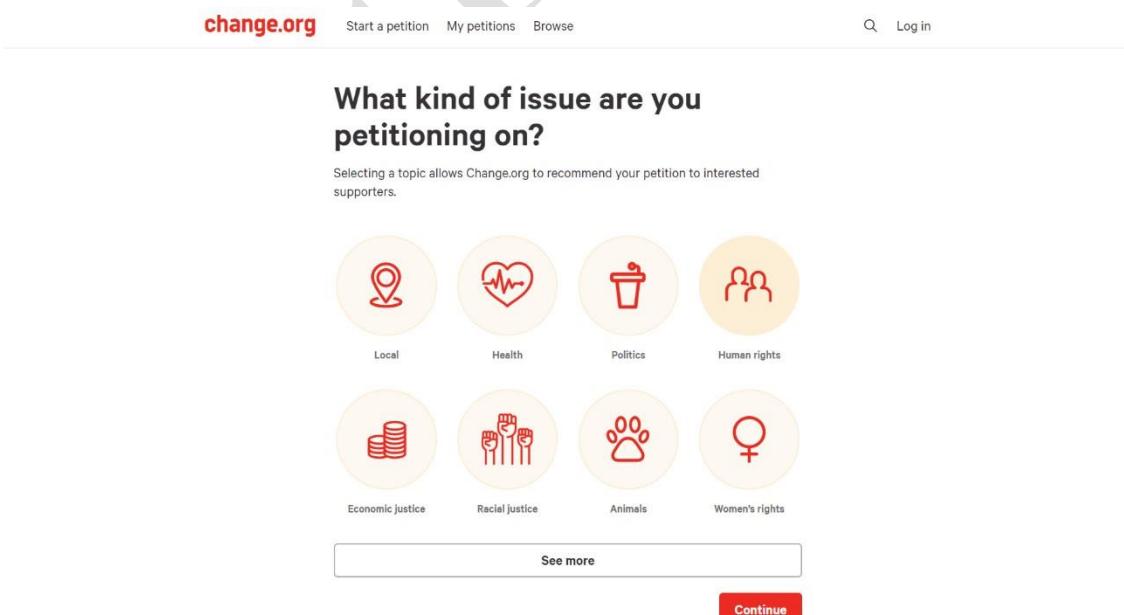


Figure 4: Change.org - Creating a Petition (Start a petition · Change.org, 2021)

Other Similar Projects are available in Appendix: [[7.4.2. SIMILAR PROJECTS](#)].

2.4. COMPARISONS

SN	Features	System			
		A	B	C	This
1	Normal Post	✗	✓	✓	✓
2	Poll Post	✗	✗	✓	✓
3	Petition Request Post	✓	✗	✗	✓
4	Join Request Post	✗	✗	✗	✓
	Anonymous Post	✗	✗	✗	✓
5	Online Donation to NGO	✗	✓	✗	✓
	Poke NGO in Post	✗	✗	✓	✓
6	React Post	✗	✗	✓	✓
7	User management	✗	✗	✓	✓
	Report Post	✓	✗	✓	✓
	Reported Post Handling	✓	✗	✓	✓
8	User Authentication and Authorization	✓	✗	✓	✓
9	List view of Registered NGOs	✗	✗	✓	✓
	Search and Filter NGO(s)	✗	✗	✓	✓
10	App Theme (Light/Dark/System)	✗	✗	✓	✓
	In-App Notifications	✗	✗	✓	✓

Table 1: Similar Systems' Features Comparison

From the above comparison table, Social Service Application can be concluded as the most innovative project by the above-listed features among the other comparable projects. Change.org is specifically made only for petitions. The user needs to log in to create or/and sign a petition. In comparison to this project, Change.org only features petition posts, petition signs, report posts, reported post handling and authentication. Red Cross Society is an International Non-Government Organization (INGO) and has regional websites with content & activities related only to its organization. In comparison to this project, the Nepal Red Cross Society (NRCS) only features normal posts and donations exclusive to them. The world's most widely used social media: Facebook, has most of the project features in comparison but lacks request posts, anonymous posts, and online NGO donations. Hence, the project outcome is feature-rich as compared to the abovementioned active projects. Unlike other systems, the social service application won't have any compromise on the features mentioned in the above comparison table. It will be a social media platform for General people and NGOs with enthusiasm for social work.

3. DEVELOPMENT

3.1. CONSIDERED METHODOLOGY

3.1.1. Waterfall Methodology

Waterfall methodology follows a linear approach to managing software development projects by completing prioritized and sorted tasks' groups (aka phases). Each completed phase is the successor of the previous. The flow of the completed phase is linear like water which cannot be flowed or iterated back to the previous one (Sherman, 2015). A traditional waterfall methodology has 5 phases: Requirement Analysis, System Design, Design Implementation, Verification and Testing, System Deployment, Software Maintenance etc.

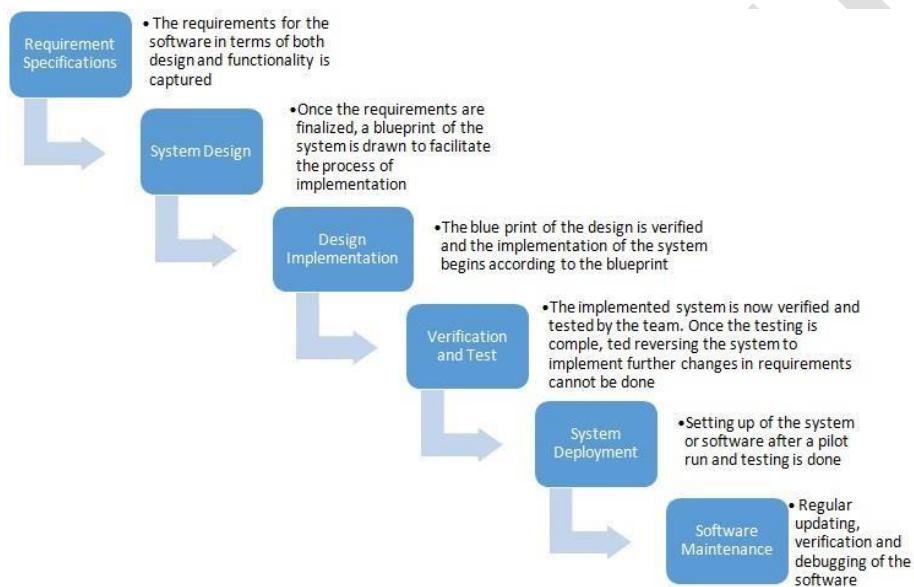


Figure 5: Waterfall Methodology (Team, 2013)

The reason to consider the waterfall methodology for managing this project is below:

- It has a simple and intuitive progression structure with a clear, defined set of steps that is progressed linearly, leaving a completer and more polished project outcome with easy progress measurement.
- With clear, stable, and committed goals and end deliverables from the start, the risk of deviation or bogging down during project work is eliminated (Team, 2022).

Other considered Methodologies are available in Appendix [[7.5. APPENDIX E: CONSIDERED METHODOLOGY](#)].

3.2. SELECTED METHODOLOGY

One of the most widely used agile frameworks in the industries (not just limited to IT): Kanban is selected to execute the Social Service Application project after analysing multiple software development methodologies and frameworks' benefits and drawbacks. The traditional waterfall is best for software projects that involve a linear fashion where the outcome is never iterated back for changes. For adaptive, simultaneous, and future workflows needed for this project, the waterfall is deficient. The project does not involve many iterations during the development. XP is an iterative agile method and needs on-site customer interaction which is needless for a current project that does not involve any clients. Some of the main reasons to select Kanban over all other considered methodologies are:

- No compulsorily on-site custom/client interaction is needed.
- Progress visualization.
- Short iterations.
- No coding standards are needed.
- Changes allowed at any time [BEST].

3.3. PHASES OF METHODOLOGY

As per David in his 'Blue Book', Kanban is not considered to be software development or a project management methodology. It does not make any statements about how software or software project is planned and developed/implemented. Even if Kanban is not a management methodology, it can be applied on top of any process or methodology like Scrum, XP, traditional waterfall etc. It enhances the work process over time, reduces cycle time, and improves flow, enabling continuous delivery of features, goods, or services (Digité, 2022).

Below are the suitable phases that can be used in the current project with Kanban:

All the phases are listed in Appendix: [\[7.6. APPENDIX F: PHASES OF METHODOLOGY\]](#)

3.4. SURVEY RESULTS

3.4.1. PRE-SURVEY RESULTS

The visualized data from the survey result is available at [[7.1. APPENDIX A: PRESURVEY](#)] provides details on social work, NGOs and involved technologies. Almost all the people taking this survey are from student and teaching professions as per the Survey Question-1 bar chart. Survey Question-2 shows the popularity of the word ‘Social Service’ where 90% claims to have heard it. Most people have been involved in social work and activities but not in NGOs as per Survey Question-3 and 4. From Survey Question 5, It is found that most people think NGOs are active and responsive. NGOs are most visible and accessed on social media, specifically Facebook as per Survey Questions 6 and 7. Few people think NGOs are responsive toward their requests and messages as per Survey Question 8.

The analysis of Survey Question-9 and 10 assures the feasibility of this project. Most people believe the current project idea to be practically beneficial and are interested to use the deployed application.

3.4.2. POST-SURVEY RESULTS

The post-survey was taken in a close group of 10 people. Those people were given a day to beta-test the app. One of the main reasons to perform the post-Survey form was to test the usability of the application in the real world. The Post-Survey form and result are available under [[7.2. APPENDIX B: POST-SURVEY](#)] From the survey result of question 1, all the people are IT, students, by profession. They all believe that the Social Service Application project can positively impact the social work/welfare activities as per the result of question 2. Almost people would like to donate money to NGOs and the least people would like to post petitions and polls through this app in future as per the result of question 3 result. In this project, the donation feature is best as per the peoples’ votes in the question 4 result. Almost people believe the public, and NGOs would take the greatest benefit from using the app as per the question 5 result. The average overall project features rating is around 4.5 as per the question 6 result. Most people are very satisfied with this project as per the question 7 result and are likely to recommend this project to others as per the question 8 result. From the overall analysis of survey results, the app has positive feedback, and the donation feature is the most-liked feature of the app.

3.5. REQUIREMENT ANALYSIS

Requirement Engineering often called Requirement Analysis, Requirements Gathering, or Requirements capture in software engineering is a crucial part of the Software Development Life Cycle (SDLC) to achieve successful project development. It is the process of defining user expectations for a new or changes software. The task to determine the actual needs or conditions for a new or changed product/project, identify the potentially conflicting requirements, analyse, validate, manage, and document software or system requirements are all included in requirement analysis.

The requirement is gathered by identifying the personal/customer's needs, evaluating the system feasibility, performing technical analysis, determining functions per system elements, establishing project schedule and limitations, and creating system definitions (i.e., UML models) (Visual Paradigm, 2022). The gathered requirements are documented in the Software Requirements Specification (SRS) document and are available in the Appendix sub-section: [[7.3.1. SOFTWARE REQUIREMENTS SPECIFICATION \(SRS\)](#)].

3.6. DESIGN

3.6.1. ICONOGRAPHY



Figure 6: Application branding Icon

The icon has the first character ‘S’ from the branding name ‘Sasae’. The colour of the logo: Violet symbolizes spirituality, wisdom, creativity, and sensitivity. It urges people to blend emotional clarity with mental stability or physical harmony. It inspires, motivates, uplifts and balances selfless love and packs a powerful punch with its confidence and prestigious reputation.

The violet colour has positive traits like encouragement and consideration (Olesen, 2022).

3.6.2. WIREFRAME

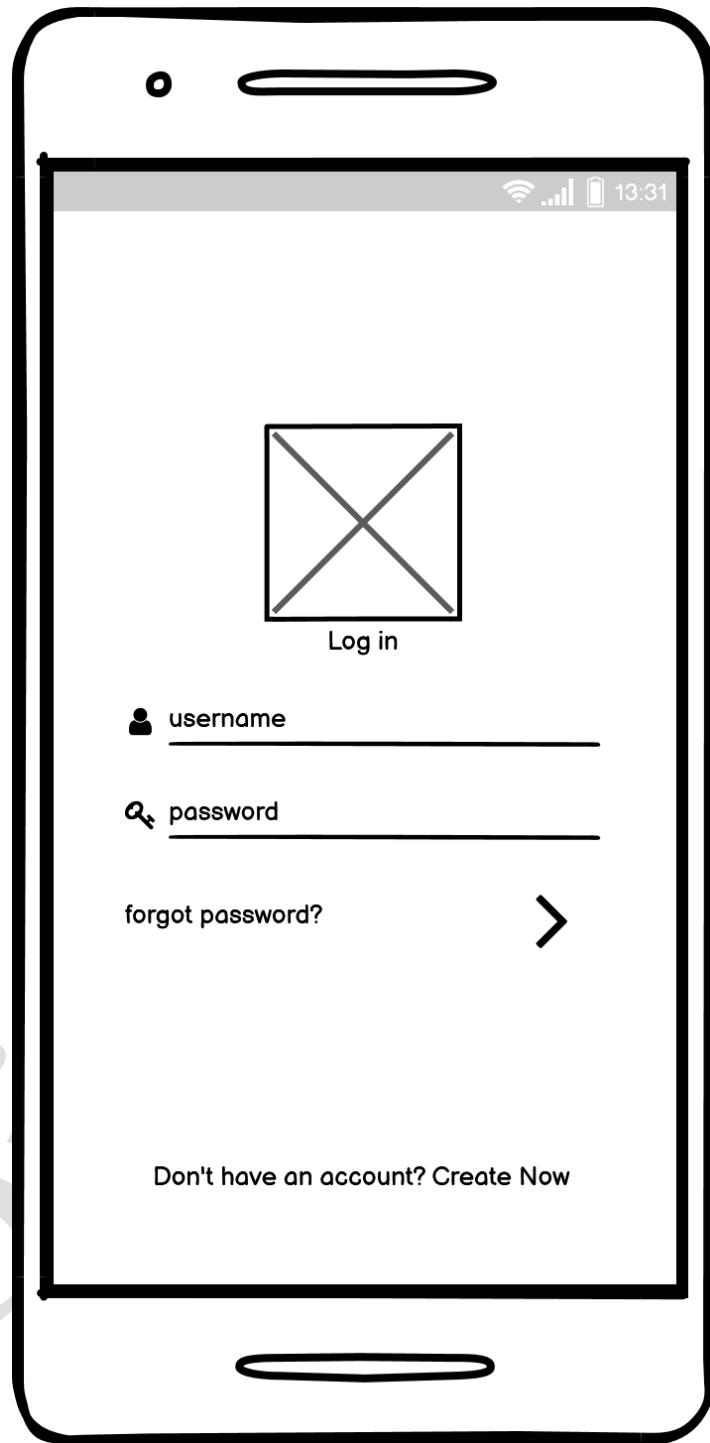


Figure 7: Wireframe - Login Screen

The Wireframes are available in the Appendix section: [\[7.8.9. WIREFRAMES\]](#).

SASAE

3.6.3. PROTOTYPE

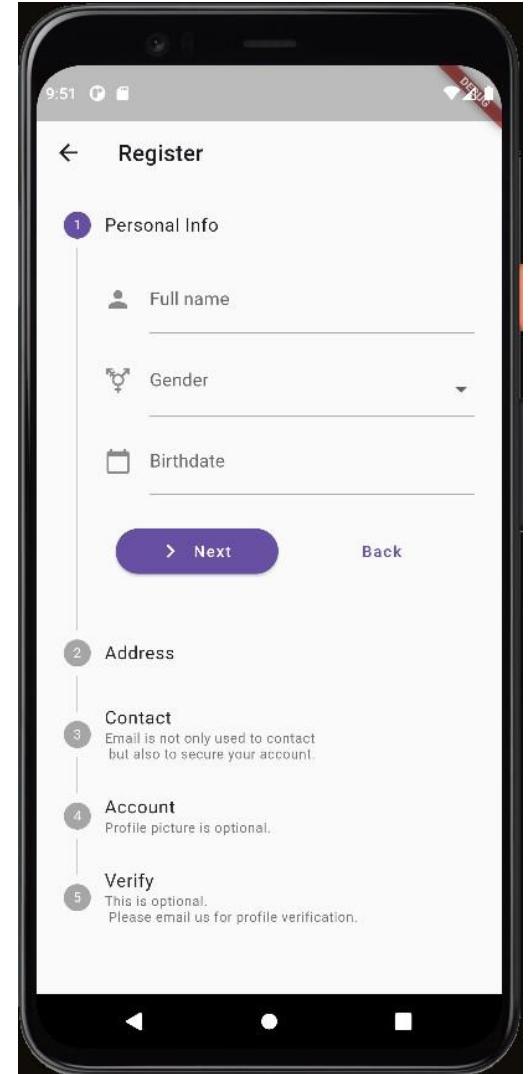
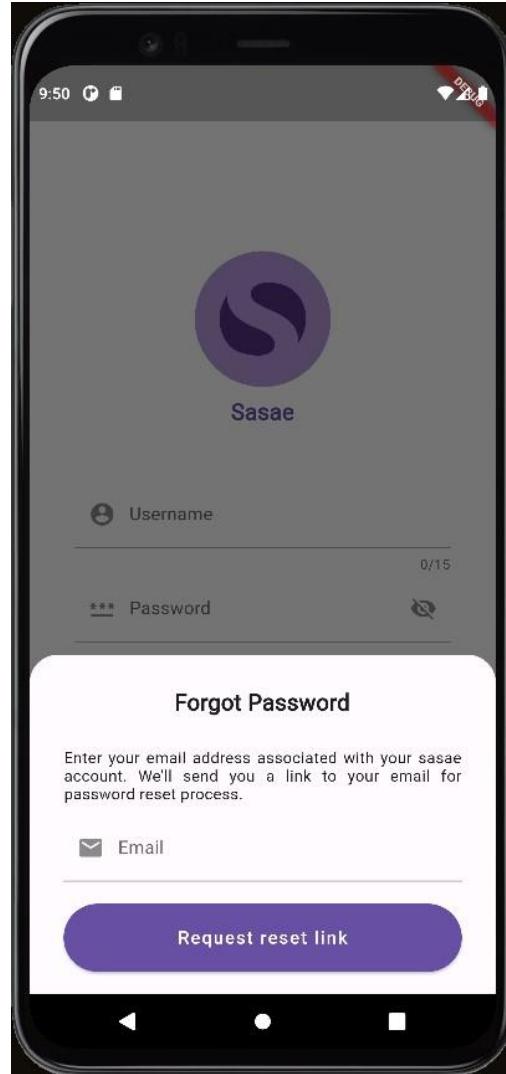
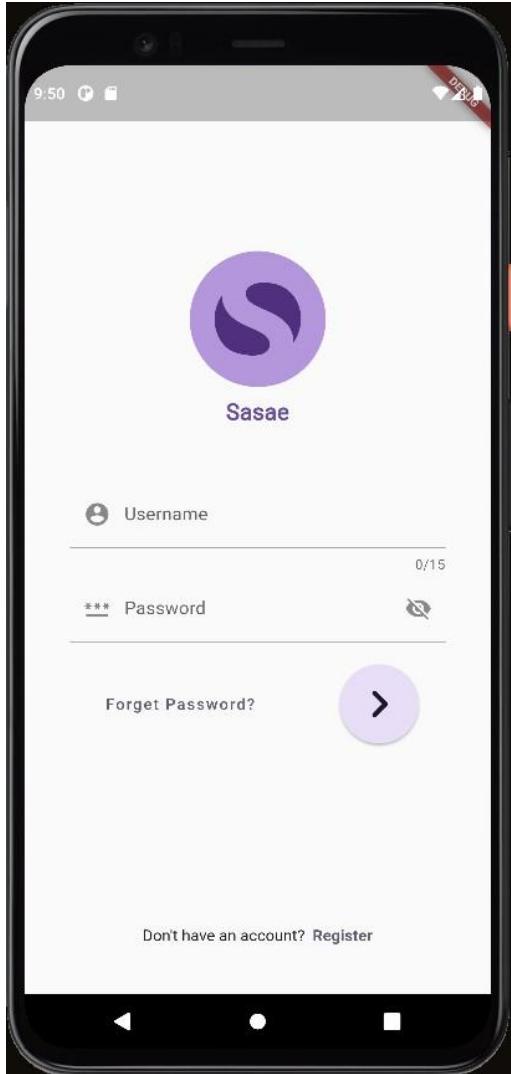


Figure 8: Prototype, Login Figure 9: Prototype, Reset Password Figure 10: Prototype, Register
The Prototypes are available in Appendix: [\[7.8.10. PROTOTYPES\]](#).

3.6.4. DATABASE DESIGN

3.6.4.1. DATA DICTIONARY

3.6.4.1.1. USER

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		Primary Key			To store an id that uniquely identifies the User	123
user_name	VARCHAR	150	NOT NULL			To store the unique username	Bishal69
password	VARCHAR	128	NOT NULL			To store the password	B#23vdg#45f&
email	VARCHAR	254	NOT NULL			To store the email of the user	Bishal69@gmail.com
is_staff	BOOL		NOT NULL			To store the staff status	FALSE
is_active	BOOL		NOT NULL			To store the active status	TRUE
date_joined	DATETIME		NOT NULL			To store the registration date of the user	2012-12-09

Table 1: Data Dictionary, User Table The

Data Dictionaries are available in Appendix: [[7.8.4.1. DATA DICTIONARY](#)].

3.6.4.2. ENTITY RELATIONAL DIAGRAM 3.6.4.2.1.

ABSTRACT ERD

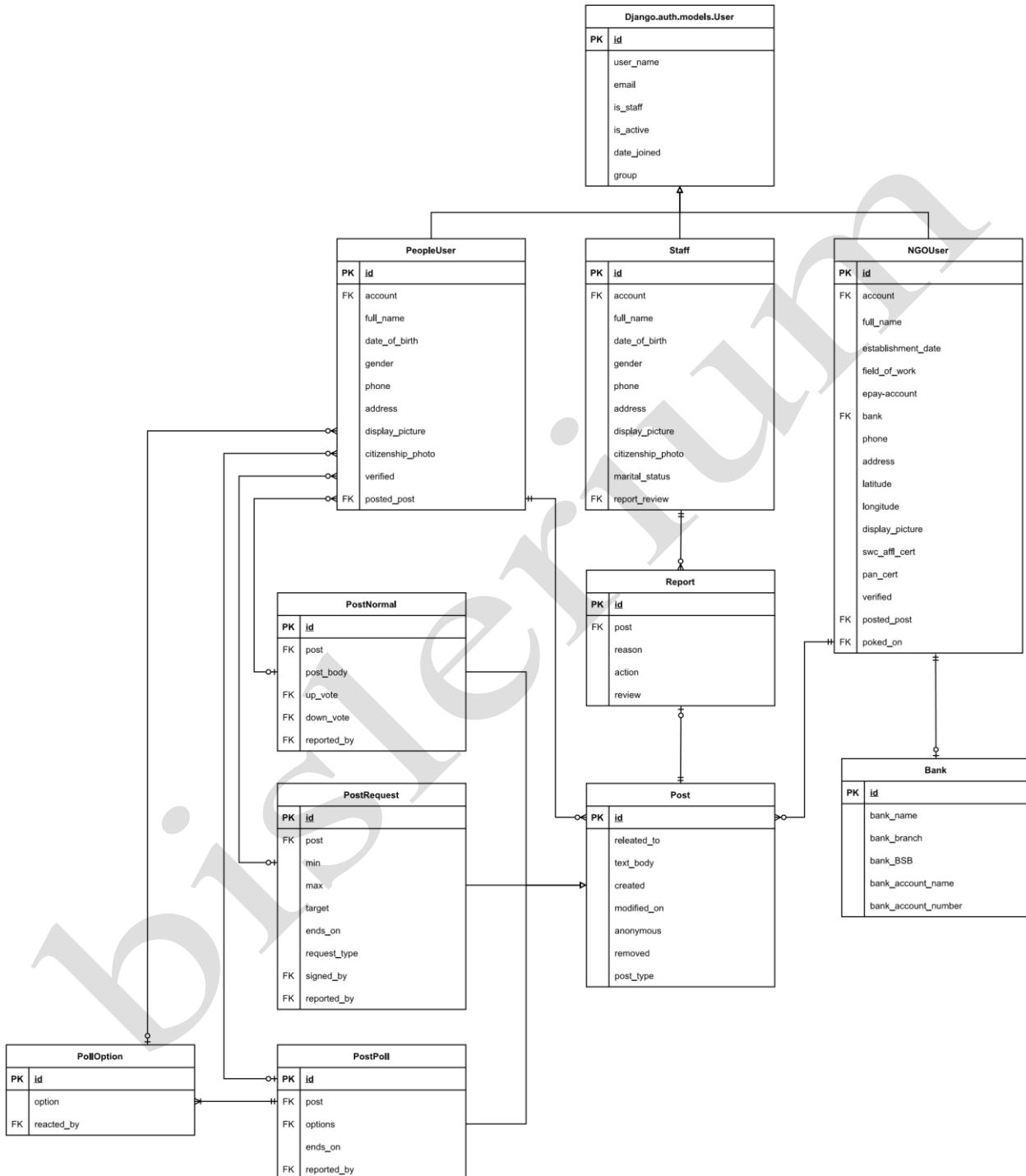


Figure 11: Database Design, Abstract ERD

3.6.4.2.2. CONCRETE ERD

The concrete ERD generated using the DBeaver application is available in [[7.8.4.2. CONCRETE ERD](#)].

bisleri.umm

3.6.5. UML DESIGN

3.6.5.1. USE CASE DIAGRAM

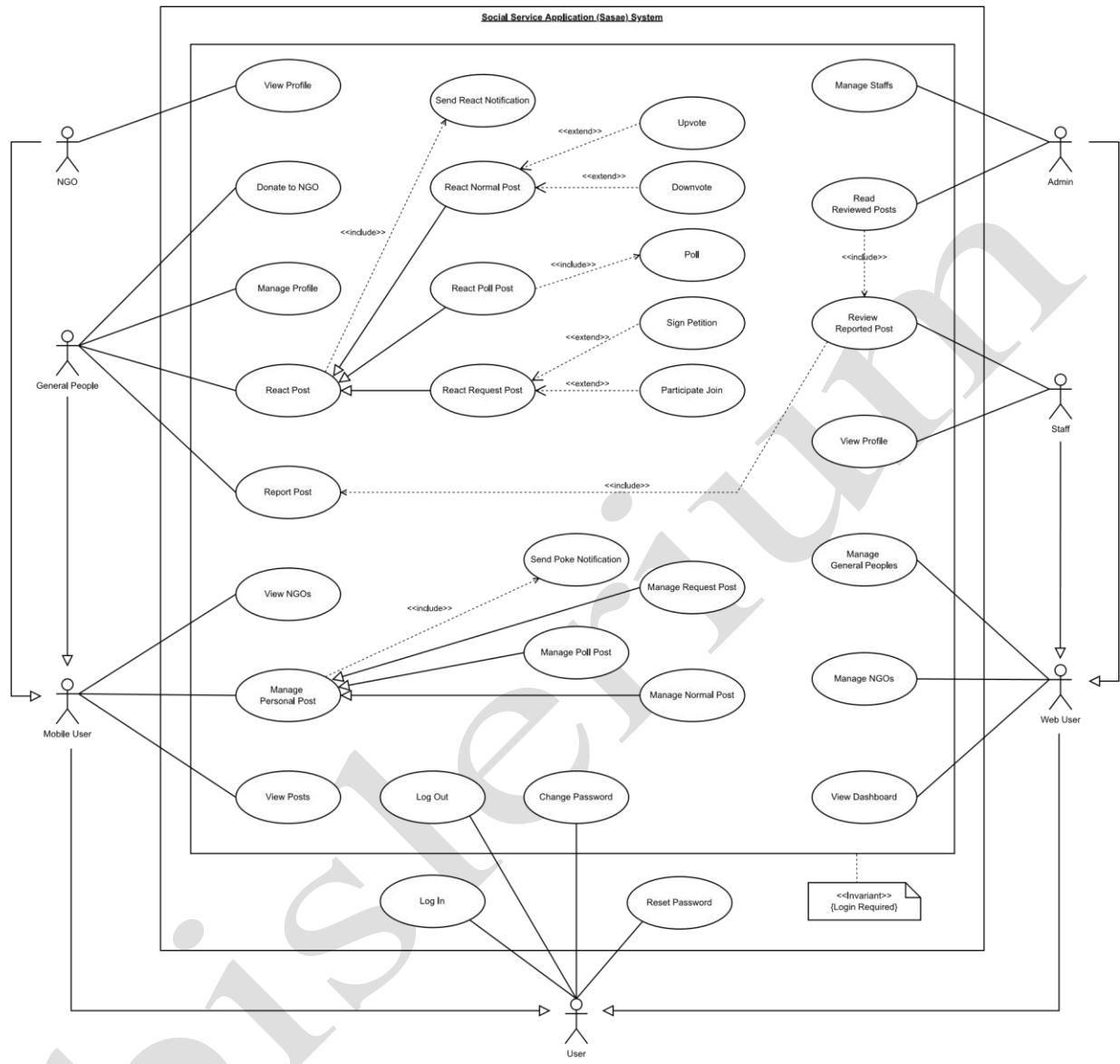


Figure 12: Use Case Diagram

The system is used by 4 types of users: General People, NGOs, Staff, and Admin that can be grouped by their common tasks. The General People and NGOs can be grouped as Mobile Users since both can interact with the same use cases: view NGOs, manage personal posts and view posts. The Staff and Admin can be grouped as Web Users. Since both can interact same use cases: manage general people, manage NGOs, view dashboard etc. The Web User and Mobile User can be grouped as User for common use cases: login/logout and change/reset password.

Use Case ID	Actor	Extends Actor	Use Case
UC1	User		Log in
UC2			Log out
UC3			Change Password
UC4			Reset Password
UC5	Mobile User	User	View NGOs
UC6			Manage Normal Post
UC7			Manage Poll Post
UC8			Manage Request Post
UC9			View Posts
UC10	Web User	User	Manage General Peoples
UC11			Manage NGOs
UC12			View Dashboard
UC13	NGO	Mobile User	View Profile
UC14	General People	Mobile User	Donate to NGO
UC15			Manage Profile
UC16			React Normal Post
UC17			React Poll Post
UC18			React Request Post
UC19			Report Post
UC20	Staff	Web User	View Profile
UC21			Review Reported Post
UC22	Admin	Web User	Manage Staffs
UC23			Read Reviewed Posts

Table 2: Actor & Use-Cases in Tabular View with Use-Case ID (UC)

The Extended Level Use-Case Descriptions are available in Appendix [[7.8.6.1. HIGH-LEVEL USE CASE DESCRIPTION](#)].

3.6.5.2. DATA FLOW DIAGRAM (DFD)

3.6.5.2.1. CONTEXT DIAGRAM (LEVEL-0)

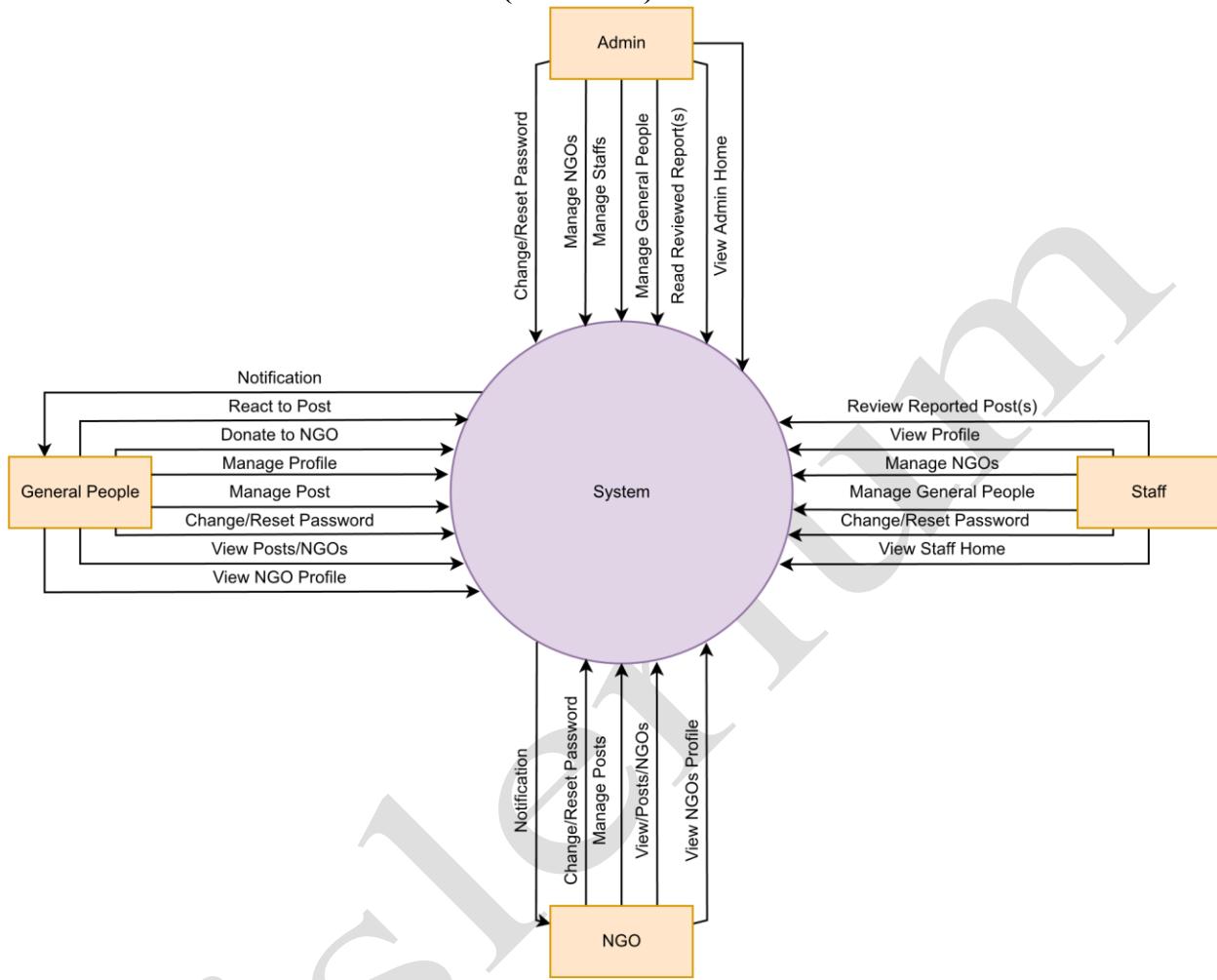


Figure 13: Level-0 DFD (Context Diagram)

The project system is a combination of two frontend applications and a backend. The two fronted applications are mobile and an Admin-Staff portal. The overall user can log in, log out, and change/reset their password. The Mobile User can perform various tasks like managing posts, receiving notifications, and viewing posts/NGOs/NGO profiles. Additionally, The General People can react to posts, donate to NGOs, manage /her profile etc. The web user can commonly manage NGOs and people's profiles and view their home/dashboard. The Staff additionally can review the reported post and the admin can manage staff along with viewing the reviewed reports, using the system.

3.6.5.2.2. LEVEL-1 DFD

3.6.5.2.2.1. UC1

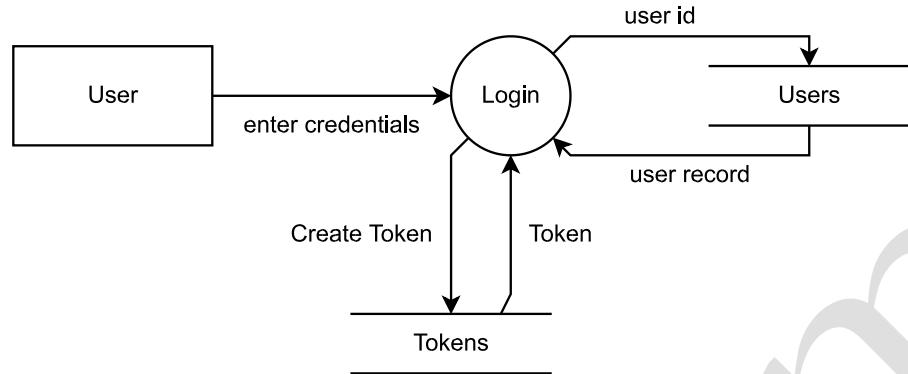


Figure 14: Level-1 DFD, UC1

The level-1 DFDs are available in Appendix: [[7.8.5. LEVEL-1 DFDS](#)].

3.6.5.3. COMMUNICATION/COLLABORATION DIAGRAM

3.6.5.3.1. UC1

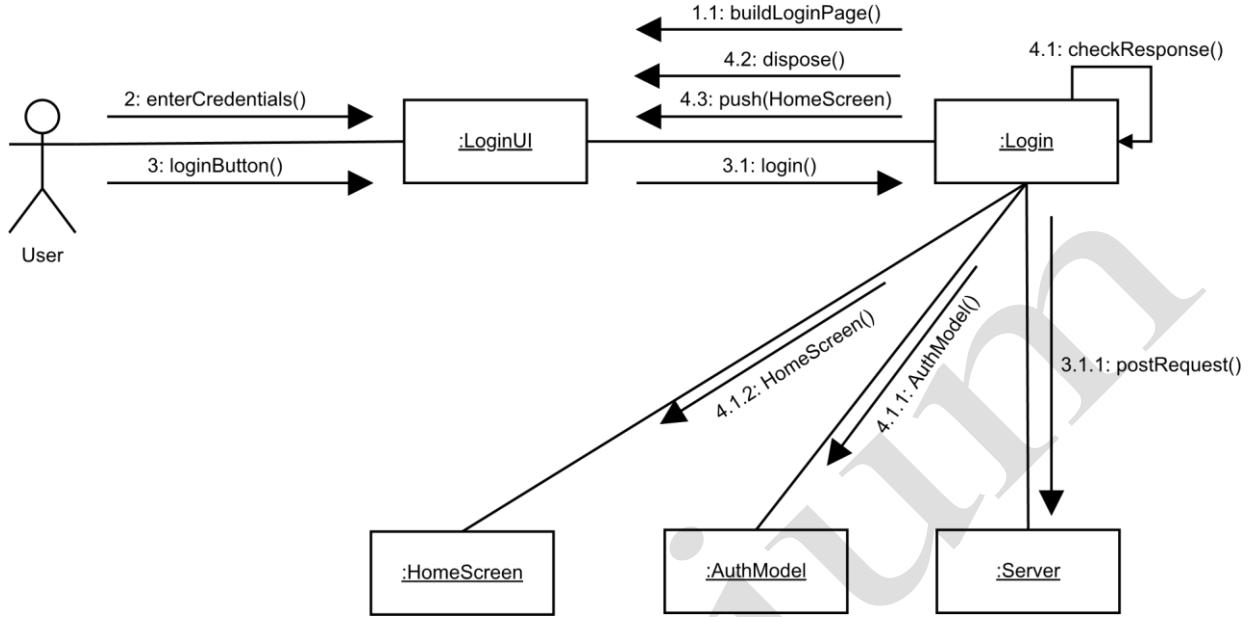


Figure 15: Communication/Collaboration Diagram – UC1

3.6.5.3.2. UC2

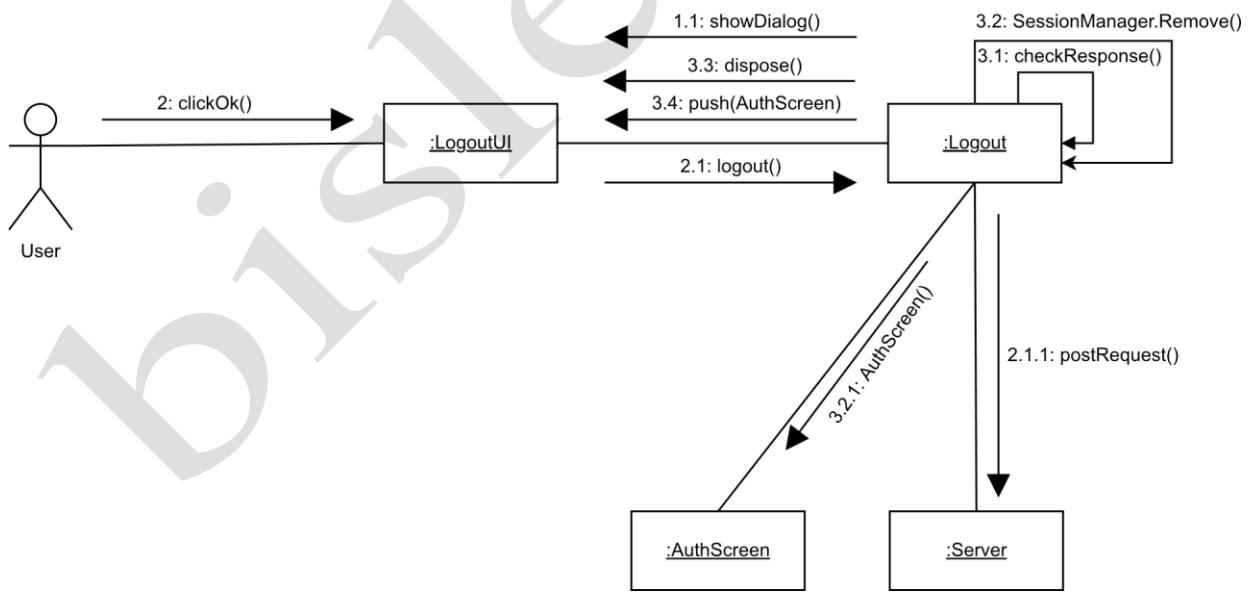


Figure 16: Communication/Collaboration Diagram – UC2

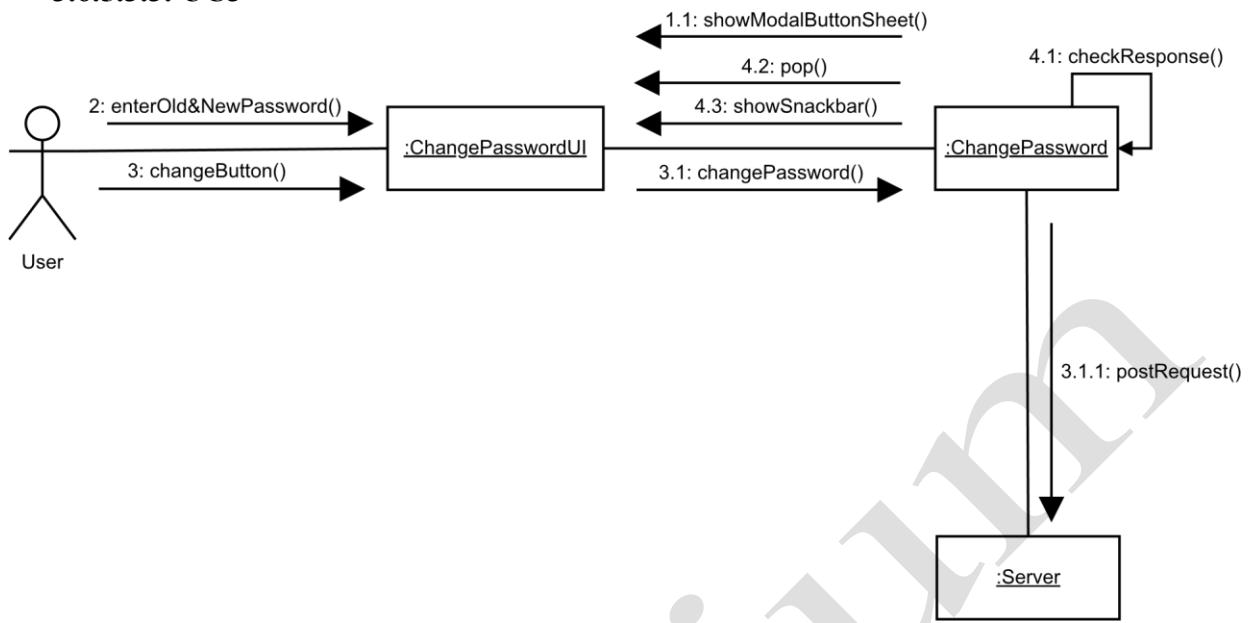
3.6.5.3.3. UC3

Figure 17: Communication/Collaboration Diagram – UC3

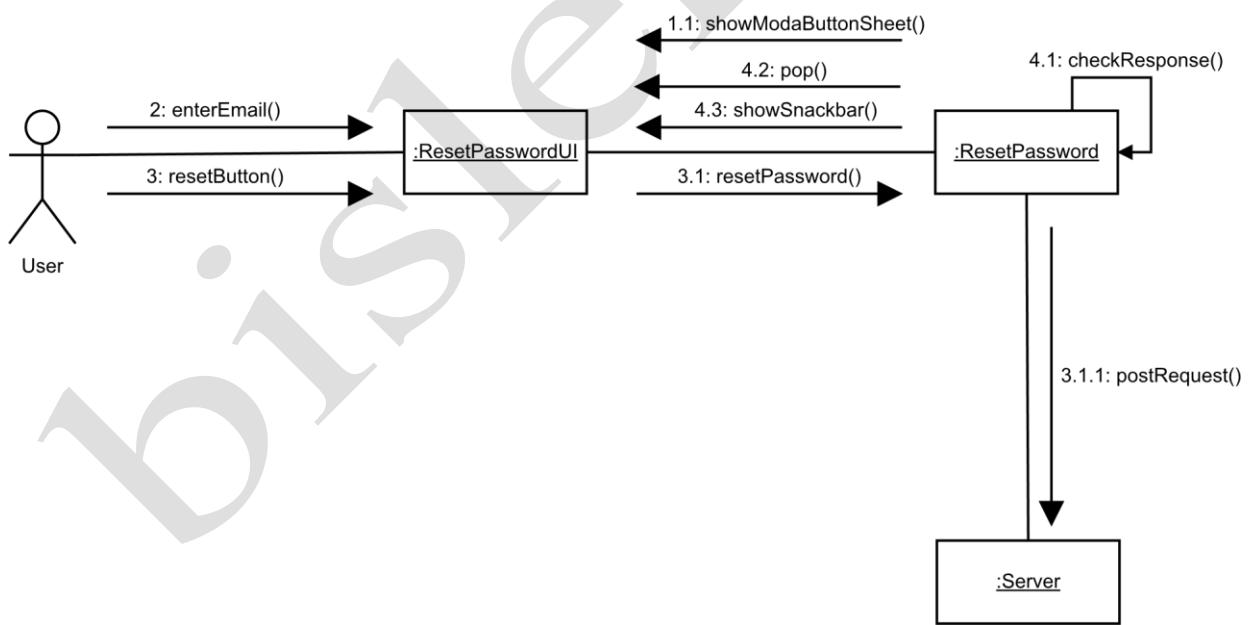
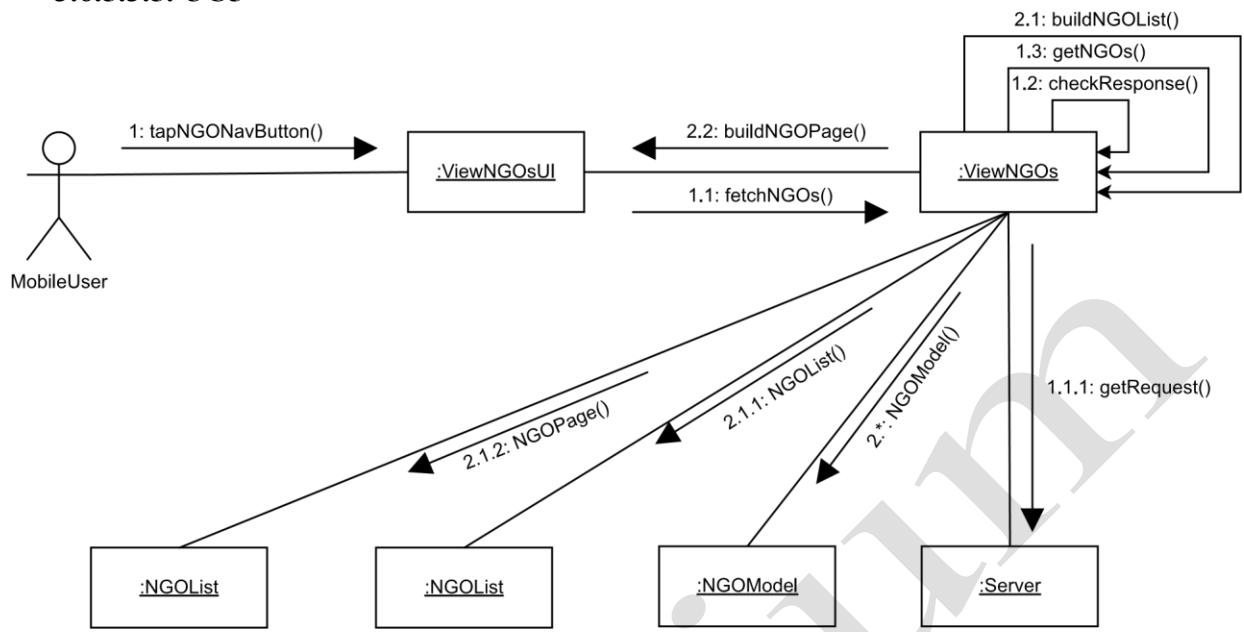
3.6.5.3.4. UC4

Figure 18: Communication/Collaboration Diagram – UC4

3.6.5.3.5. UC5*Figure 19: Communication/Collaboration Diagram – UC5*

The Communication/Collaboration Diagrams are available in Appendix [7.8.7. COMMUNICATION/COLLABORATION DIAGRAMS].

3.6.5.4. SEQUENCE DIAGRAM

3.6.5.4.1. UC1

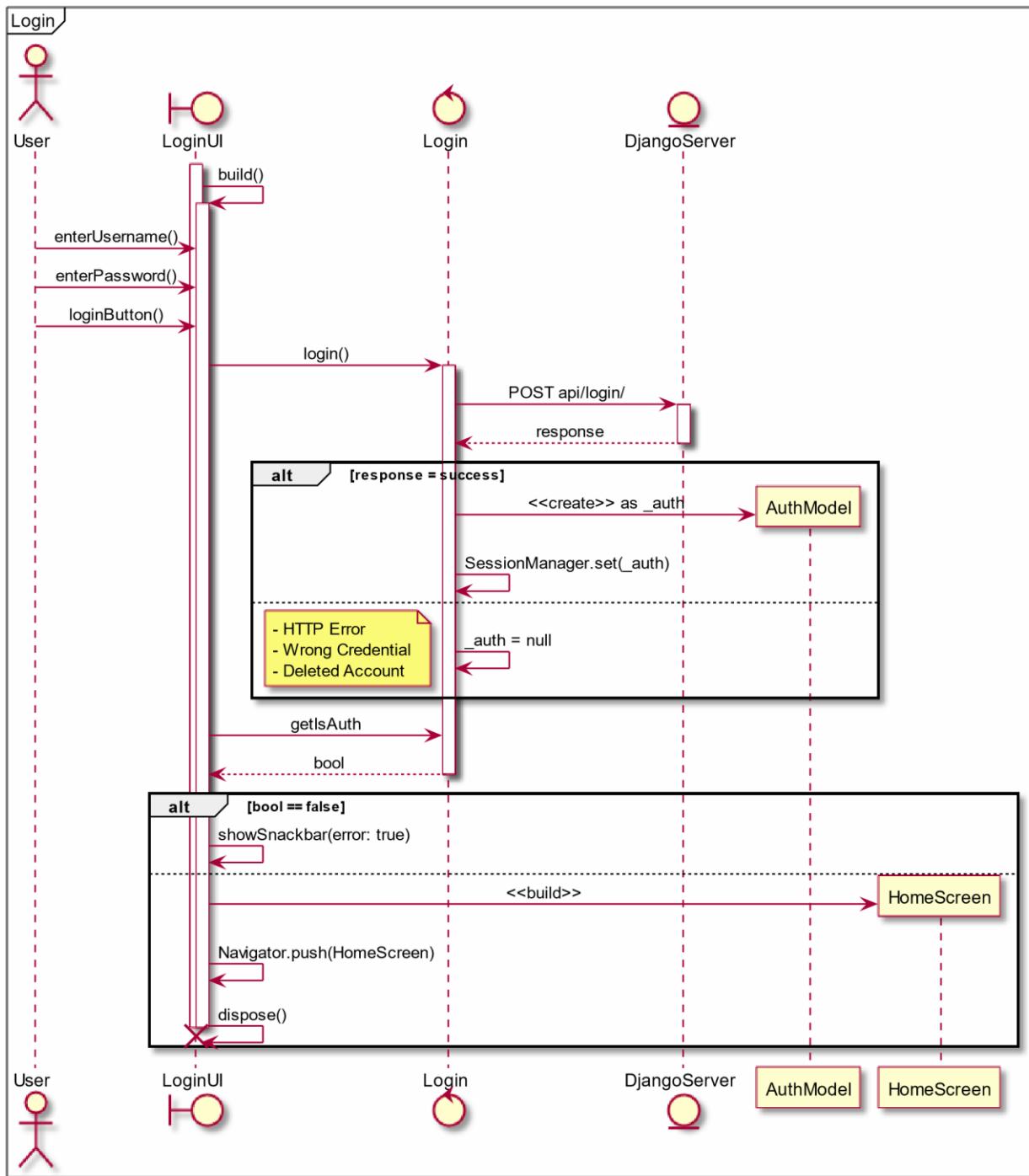


Figure 20: Sequence Diagram - UC1

3.6.5.4.2. UC2

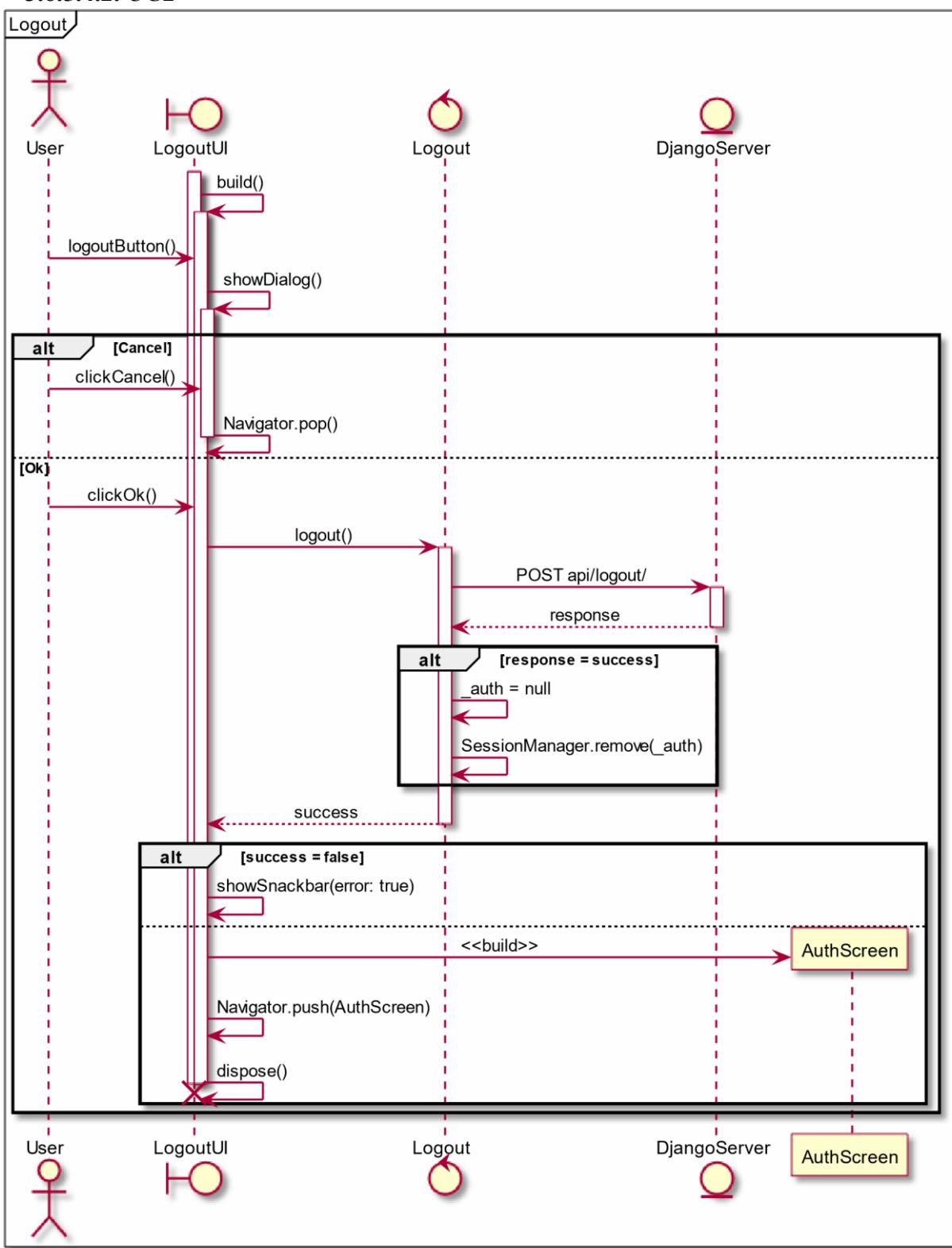
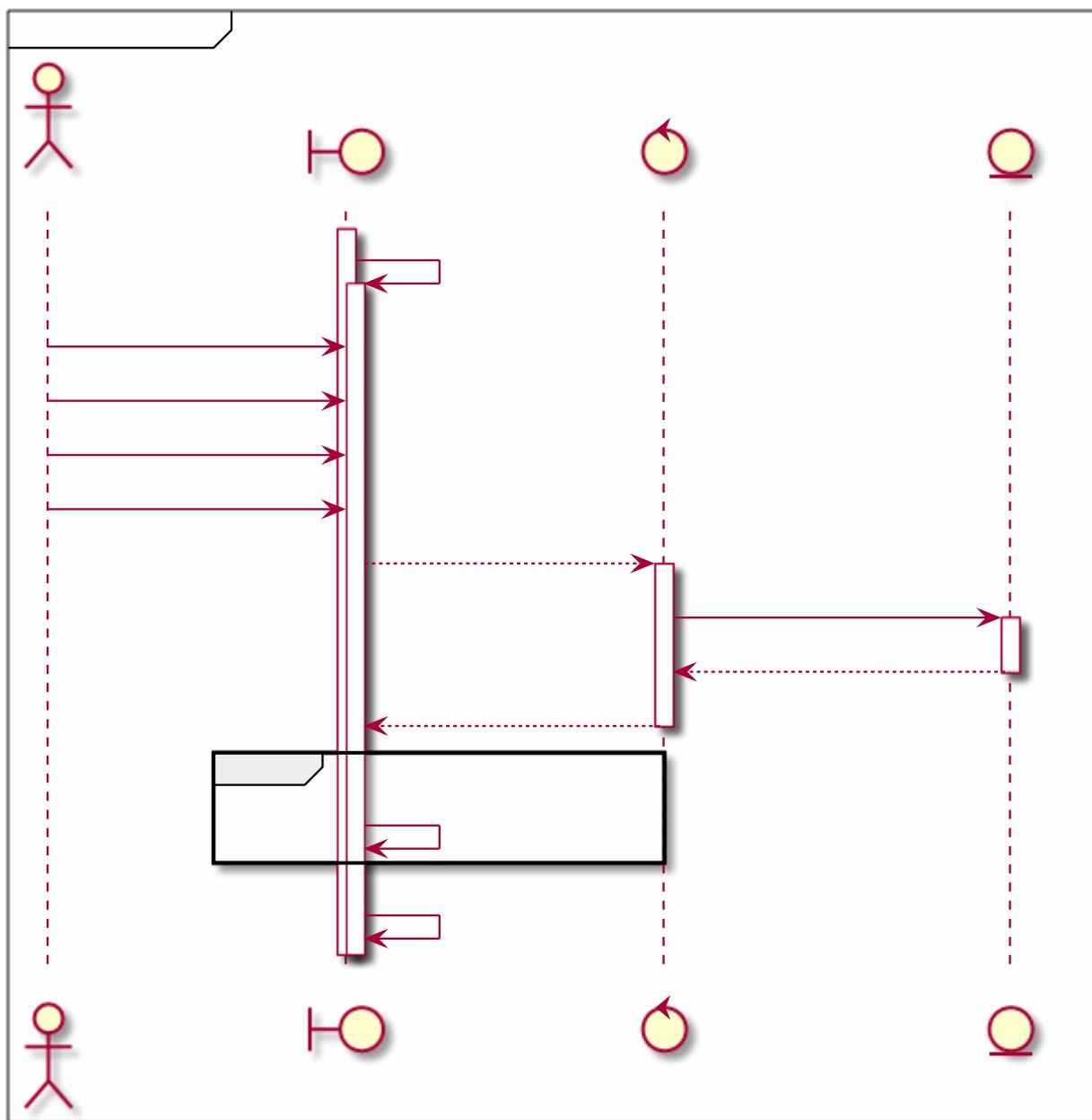


Figure 21: Sequence Diagram - UC2

3.6.5.4.3. UC3*Figure 22: Sequence Diagram - UC3*

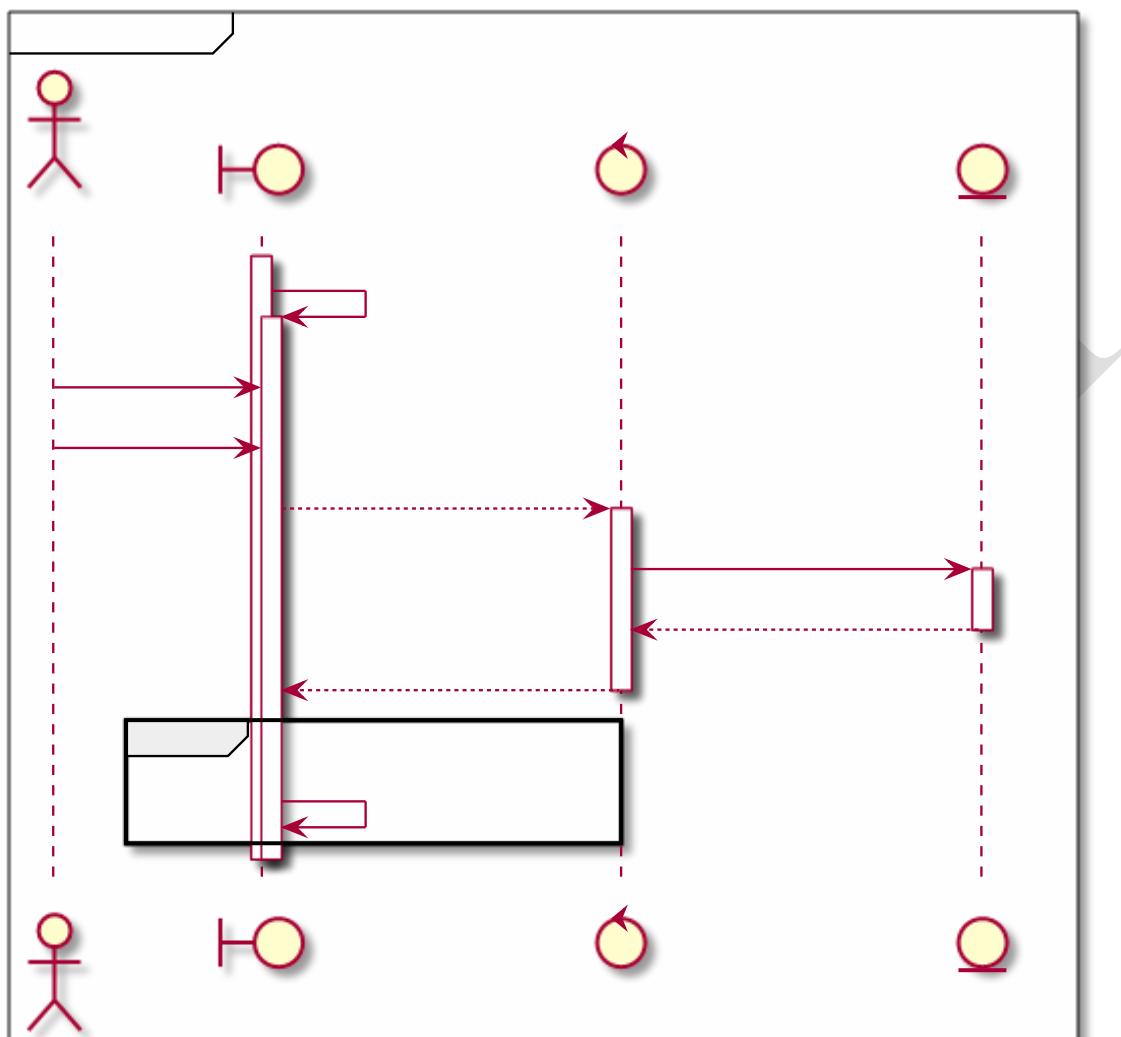
3.6.5.4.4. UC4

Figure 23: Sequence Diagram - UC4

3.6.5.4.5. UC5

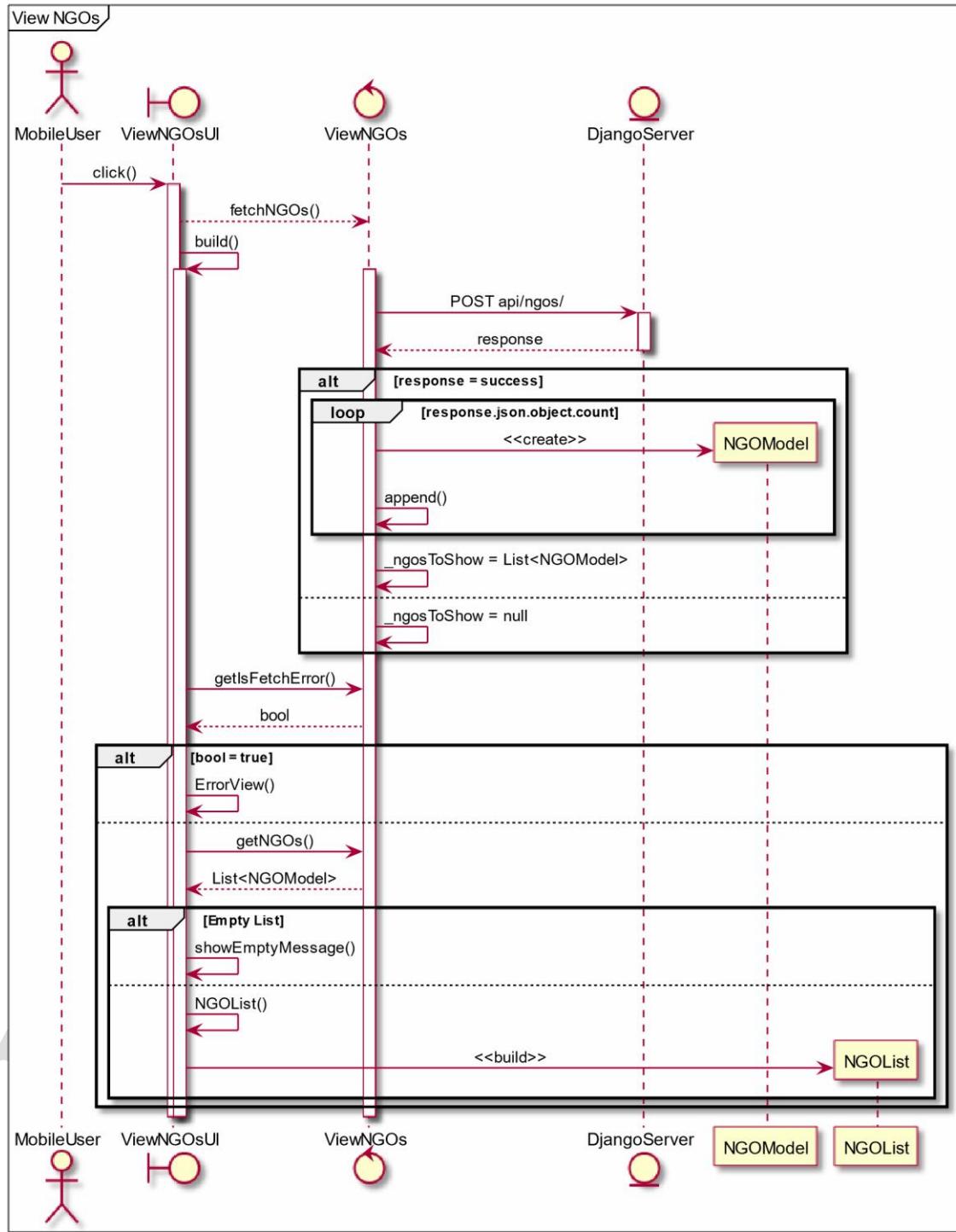


Figure 24: Sequence Diagram - UC5

The Sequence Diagrams are available in Appendix: [[7.8.8. SEQUENCE DIAGRAM](#)].

3.6.5.5. CLASS DIAGRAM

3.6.5.5.1. BACKEND (DJANGO/PYTHON)

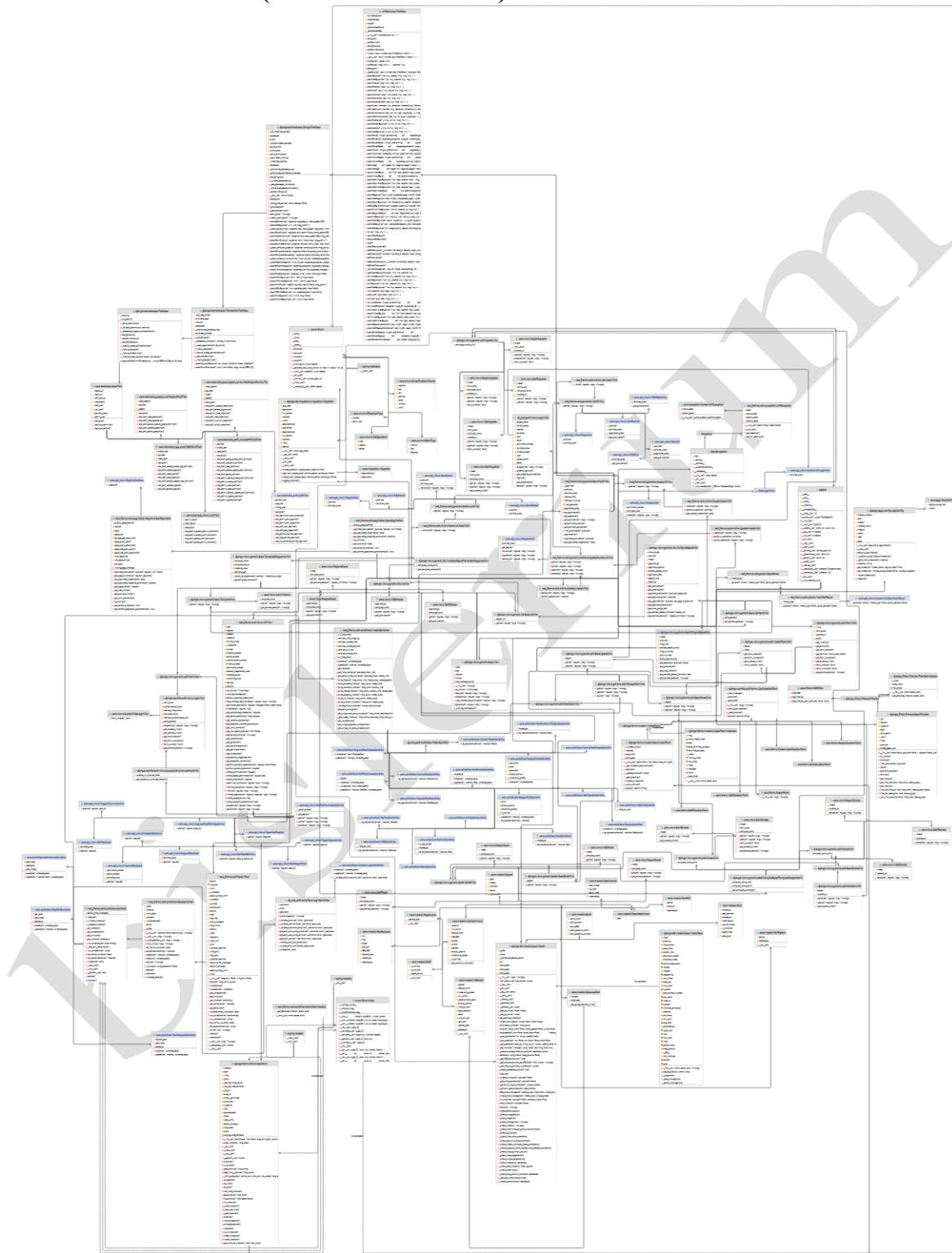


Figure 25: Django Backend Class Diagram

3.6.5.5.2. FRONTEND (FLUTTER/DART)

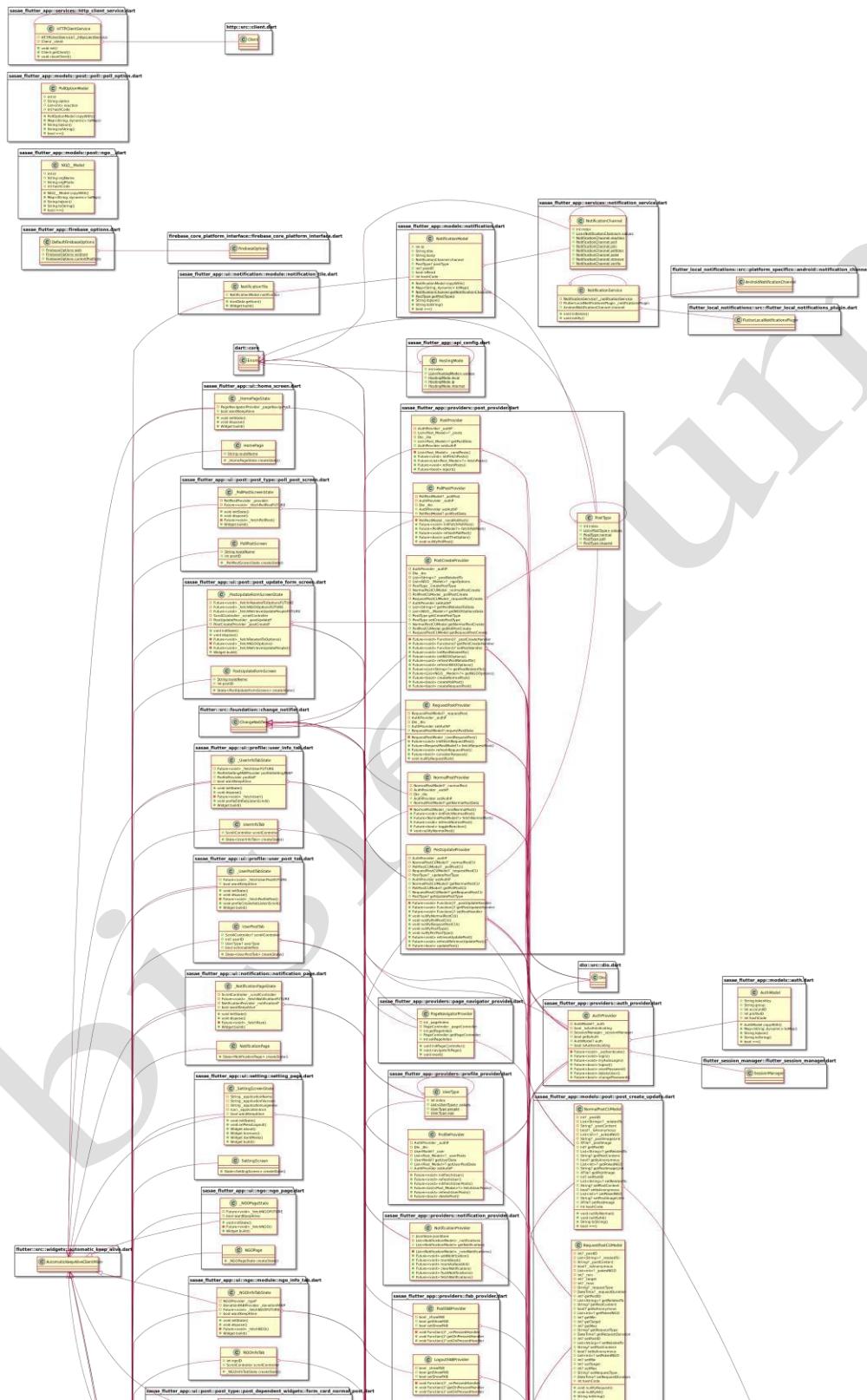


Figure 26: Flutter Frontend Class Diagram

SASAE

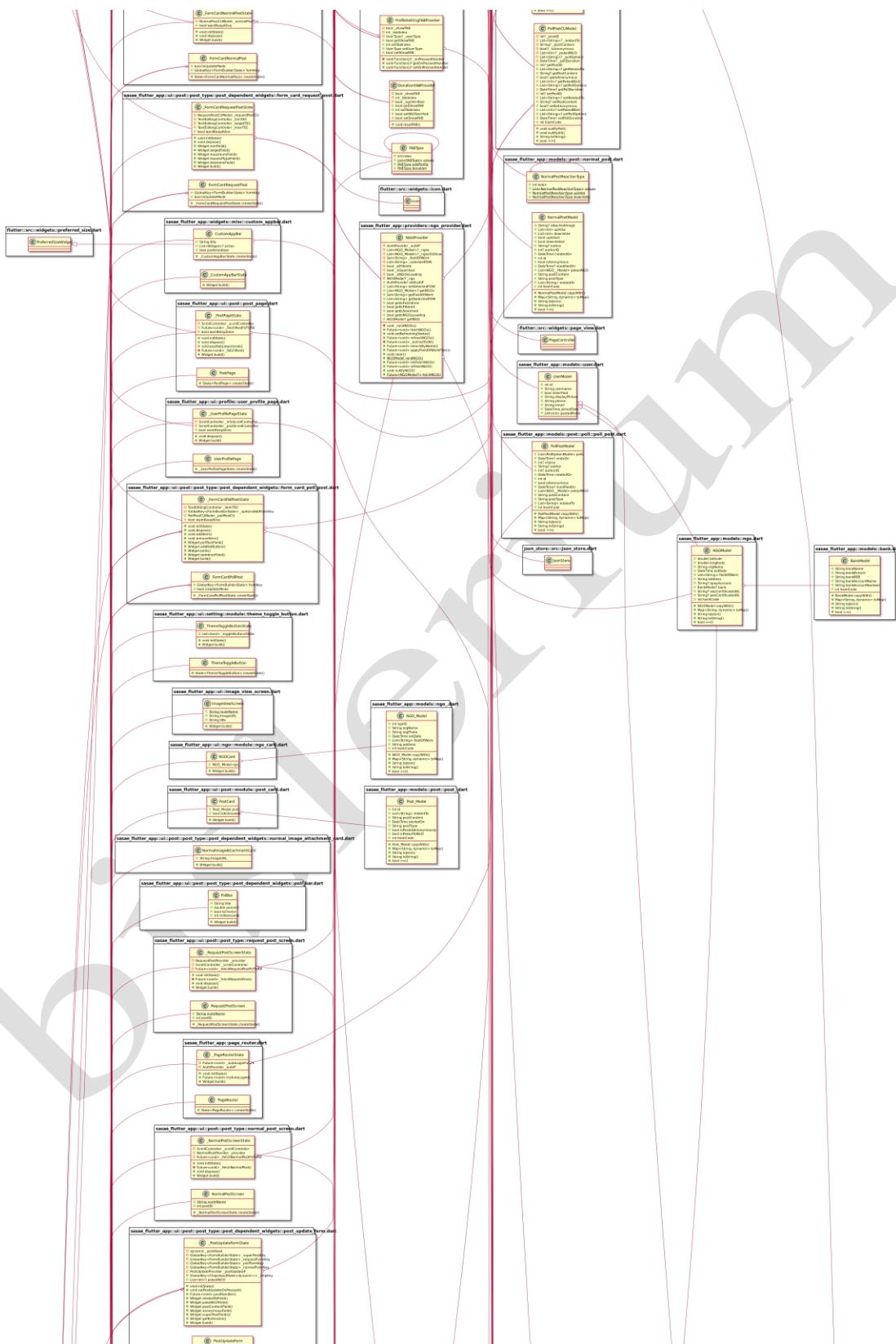


Figure : Flutter Frontend Class Diagram (Contd.)

SASAE

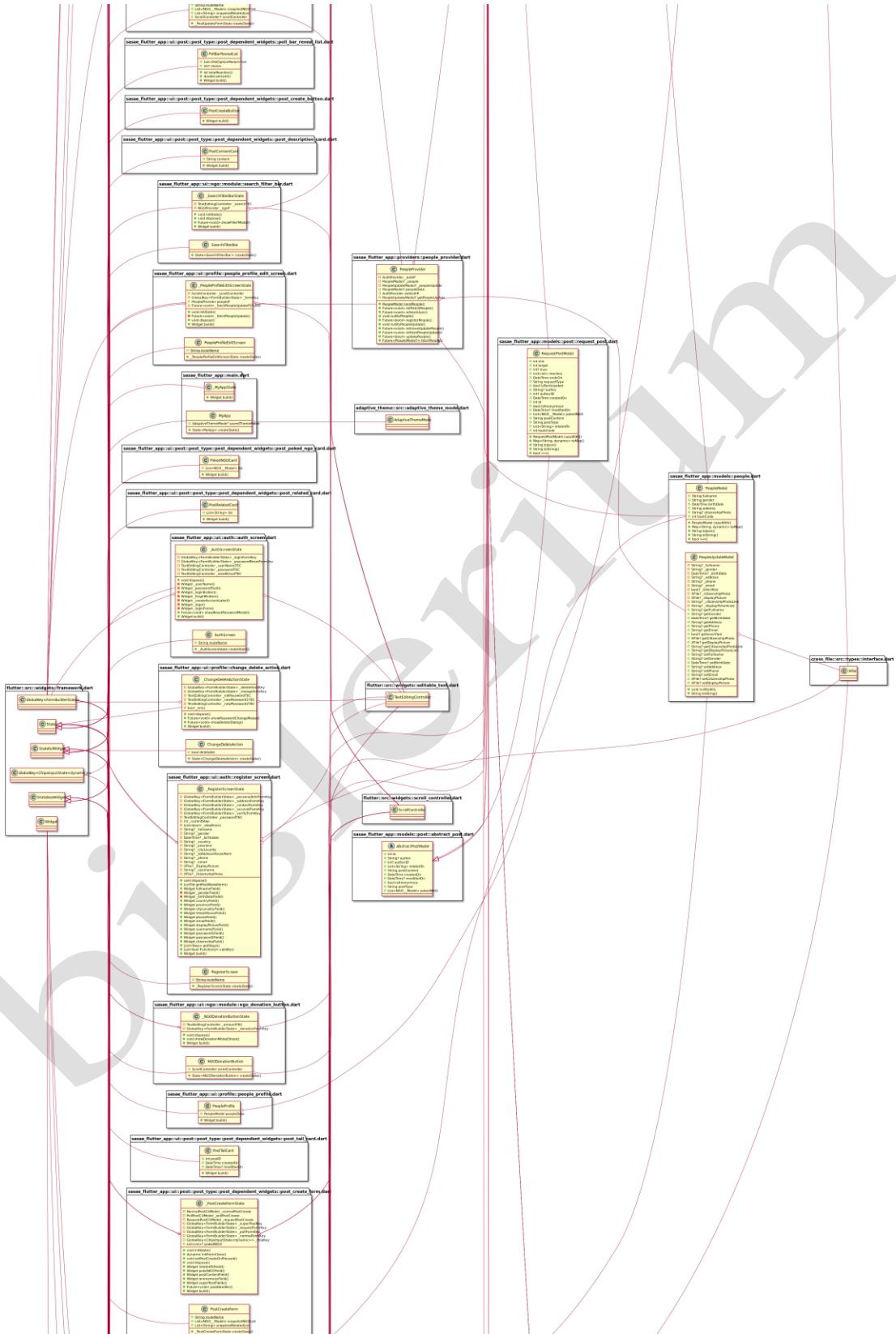


Figure : Flutter Frontend Class Diagram (Contd.)

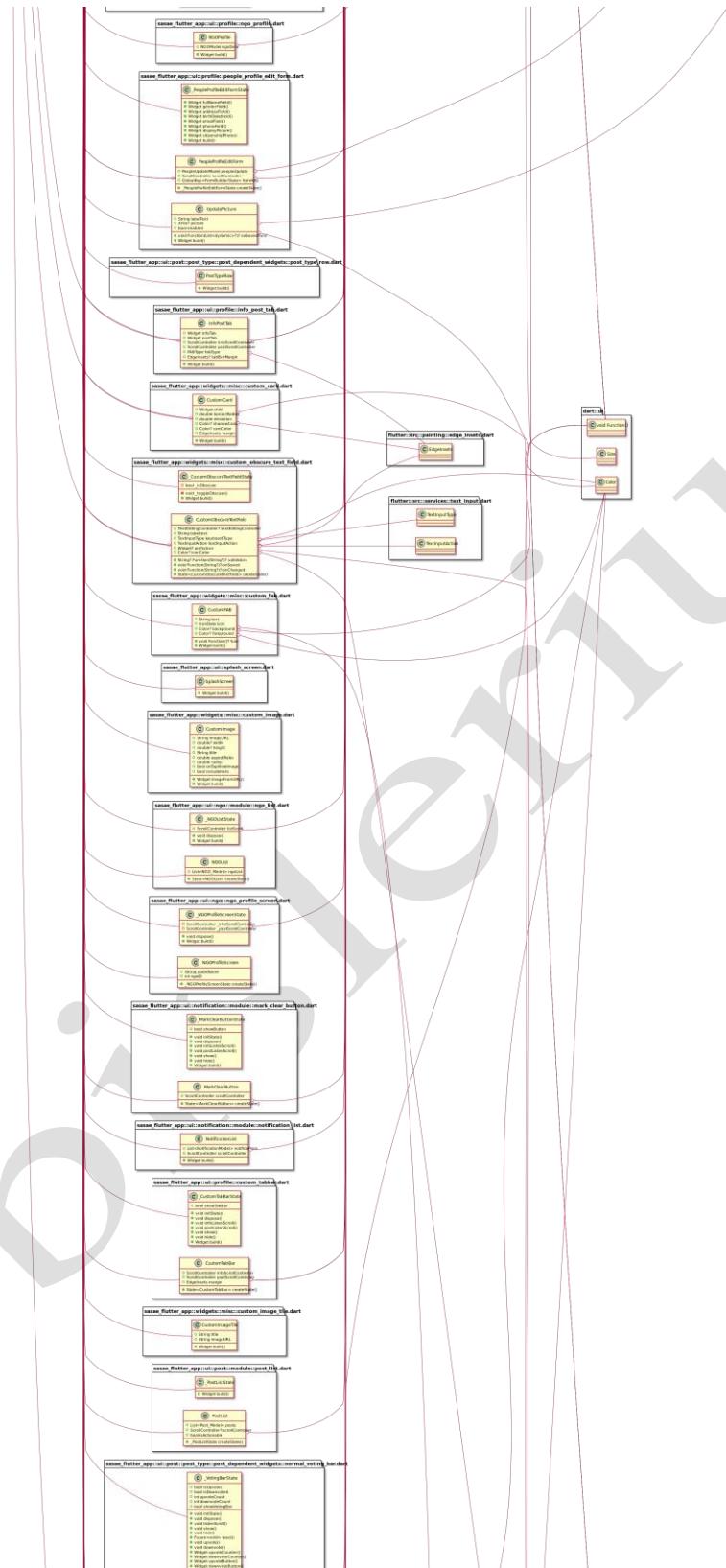


Figure : Flutter Frontend Class Diagram (Contd.)

SASAE

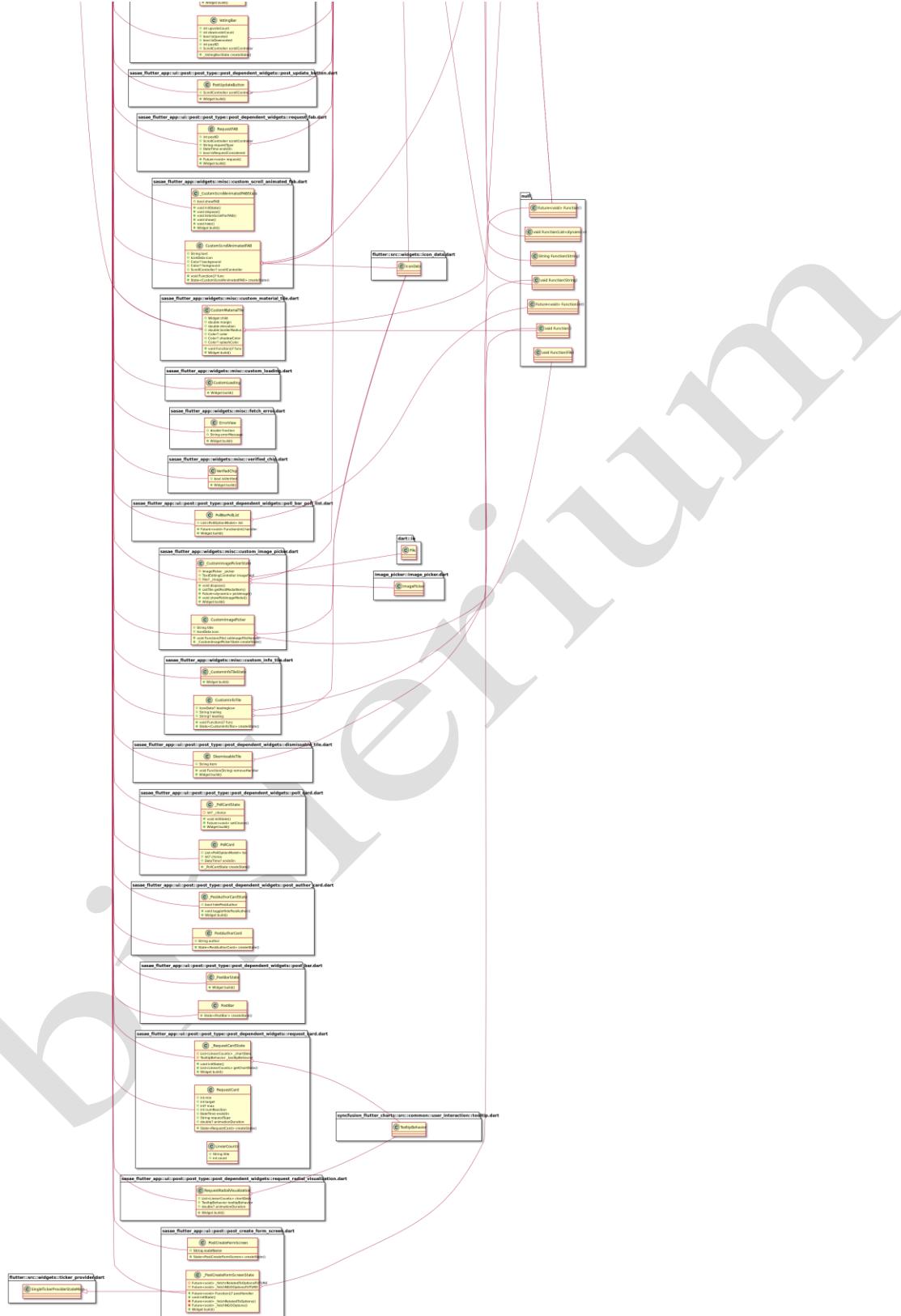


Figure 30: Flutter Frontend Class Diagram (Contd.)

3.7. IMPLEMENTATION

The Kanban framework was precisely followed to manage and implement the overall project. This implementation section mainly focuses on the build or development phase of the methodology. Kanban is different from other alternative methodologies like Extreme Programming and traditional waterfall. The project is split into groups of smaller tasks which are arranged in a queue of cards and are considered a Kanban backlog.



Figure 31: Kanban Backlog

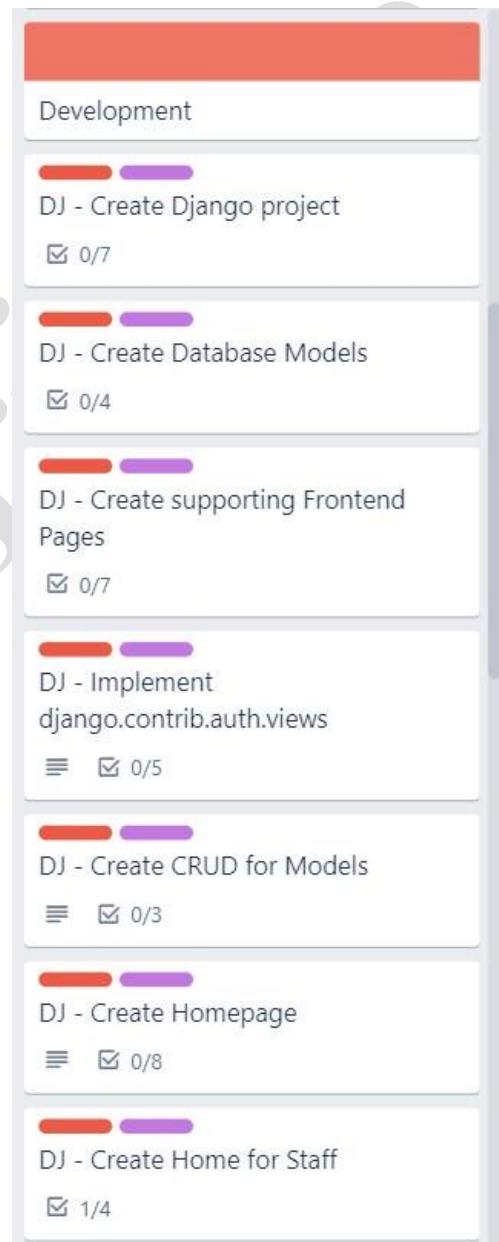


Figure 32: Kanban Backlog (Contd.)

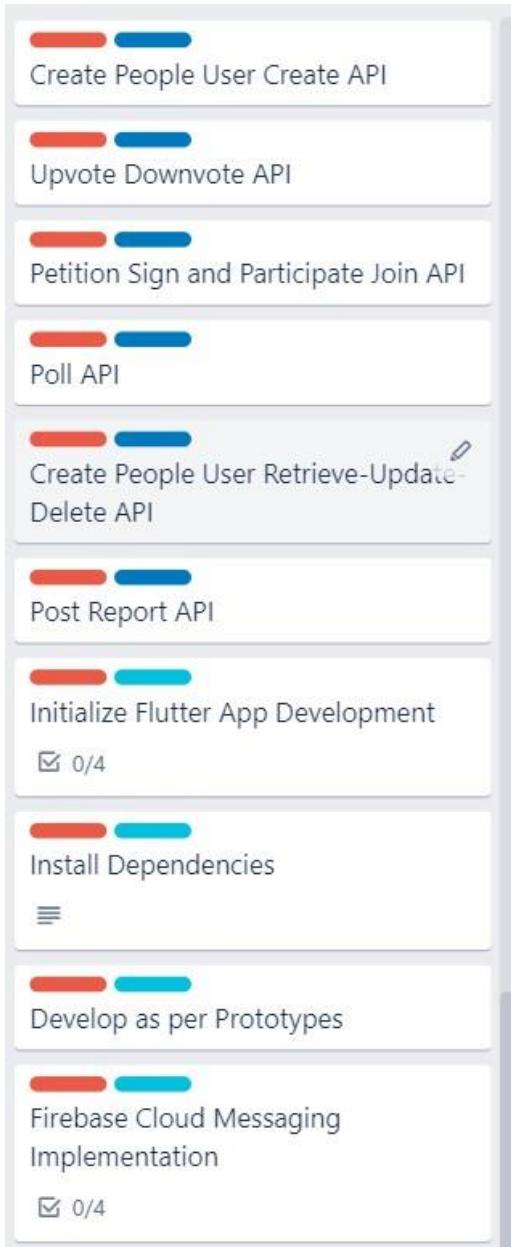


Figure 33: Kanban Backlog (Contd.)

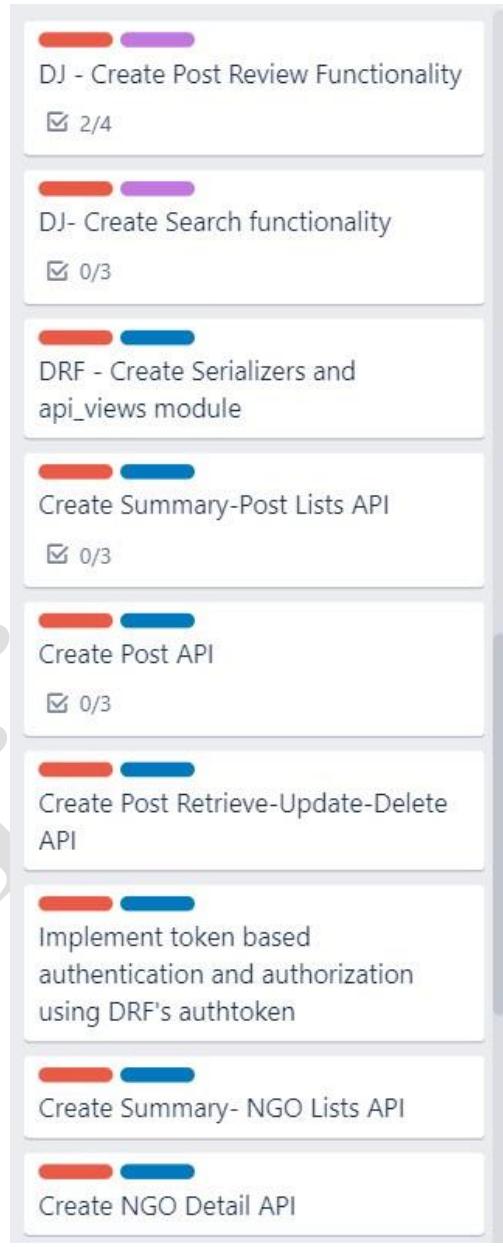


Figure 34: Kanban Backlog (Contd.)

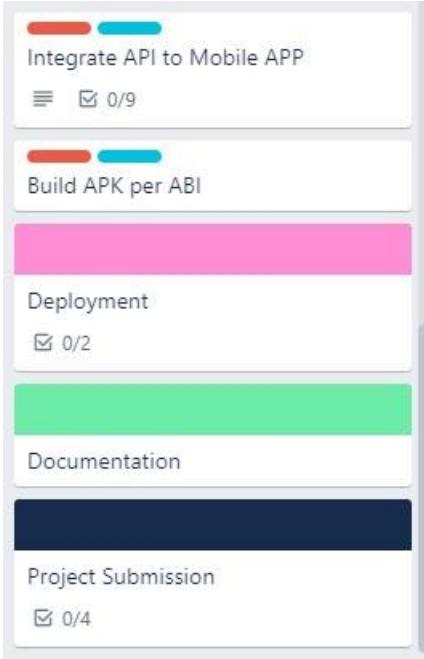


Figure 35: Kanban Backlog (Contd.)

For each progress on task completion, a new card is pulled to a proper column and is focused on the next accomplishment. This process as a requirement is considered and followed during the project development.

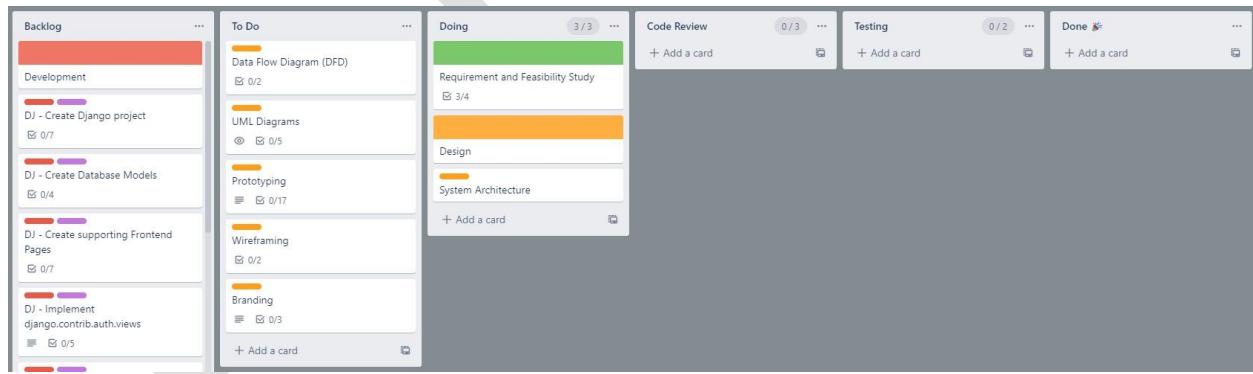


Figure 36: Kanban, Requirements Phase

Initially, the Requirement and Feasibility Study has carried out to understand and weigh the overall project's needs and viability. From the Kanban board, the 'Requirement and Feasibility

Study' labelled card is pulled and dragged to the 'Doing' column section. The feasibility of the project was studied by conducting and analysing the survey form and survey result respectively. The Survey result is in [7.1. APPENDIX A: PRE-SURVEY]. The requirement is then gathered and documented in System Requirement Specification (SRS) available in [7.3.1. SOFTWARE REQUIREMENTS SPECIFICATION (SRS)]. The card is then pulled and dragged to the 'Done' column section.

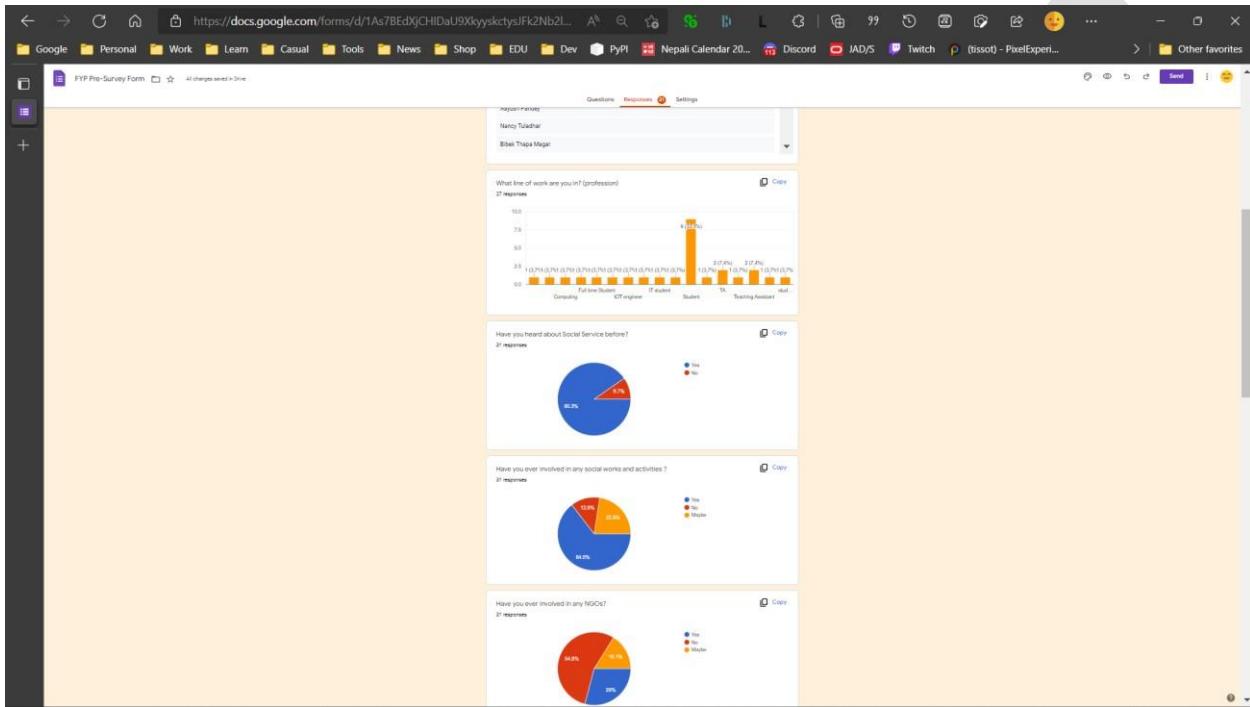


Figure 37: Pre-Survey Form (Google Forms)

Further Explanations are available in Appendix: [7.9. APPENDIX I: IMPLEMENTATION].

4. TESTING AND ANALYSIS

4.1. TEST PLAN

4.1.1. UNIT TESTING, TEST PLAN

4.1.1.1. TEST PLAN FOR DJANGO WEB-APPLICATION

ID	Test Plan Description
TC-1	Test to sequentially log into the Admin-Staff Portal with credentials of Admin, Staff, NGO, General People and Unknown.
TC-2	Test to load the Admin and Staff Dashboard/Home from Admin-Staff Portal from both Admin and Staff accounts.
TC-3	Test to perform CRUD operations on NGO from relevant rendered pages.
TC-4	Test to Read, Update and Write (RUD) operations on General People from relevant rendered pages.
TC-5	Test to access, read, and review the Post Report from both Admin and Staff accounts.
TC-6	Test to perform CRUD operations on Staff from relevant rendered pages, using both Admin and Staff accounts.

Table 3: Unit Testing, Django Web-App Test Plans

4.1.1.2. TEST PLAN FOR RESTFUL API

ID	API Collection	Request Type	API Endpoints	Test Plan Description
TC-1	Auth	POST	api/login/	Test for authenticating credential.
		POST	api/password/change/	Test for changing password.
		GET	api/user/verify/	Test for verifying the user's authentication token.
		POST	api/password/reset/	Test for resetting password.

SASAE

TC-2	Post	POST	api/post/<post_id>/report/	Test for reporting any post.
		GET	api/posts/	Test for fetching a list of posts.
		GET	api/post/<post_id>/	Test for fetching Post's details.
		DELETE	api/post/<post_id>/delete/	Test for deleting any post.
		GET	api/post/<post_id>/detail/	Test for fetching update details of any post.
		GET	api/post/relatedto/	Test for fetching related to options.
		GET	api/(ngo people)/<user_id>/posts/	Test for fetching Mobile User's posted posts.
TC-3	Normal Post	POST	api/post/normal/	Test for posting a Normal Post.
		POST	api/post/<post_id>/upvote/	Test for upvoting a Normal Post.
		POST	api/post/<post_id>/downvote/	Test for downvoting a Normal Post.
		PUT	api/post/<post_id>/update/	Test for Updating an existing Normal Post.
TC-4	Poll Post	POST	api/post/poll/	Test for posting a Poll Post.
		POST	api/post/<post_id>/poll/<option_id>/	Test for polling an option of Poll Post.
		PUT	api/post/<post_id>/update	Test for Updating an existing Poll Post.
TC-5	Request Post	POST	api/post/request/	Test for posting Request Post.
		POST	api/post/<post_id>/participate/	Test for involving in Request Post.
		PUT	api/post/<post_id>/update/	Test for Updating an existing Request Post.
TC-6	NGO	GET	api/ngo/<ngo_id>/	Test for fetching NGO's details.
		GET	api/post/ngos/	Test for fetching NGO options.
		GET	api/ngos/	Test for fetching a list of NGOs.

TC-7	General People	POST	api/people/add	Test for registering a General People.
		PUT	api/people/update	Test for Updating an existing General People account.
		GET	api/people/detail/	Test for fetching the logged in General People's details.
		DELETE	api/people/delete/	Test for deleting the logged in General People.

Table 4: Unit Testing, RESTful API Test Plans

4.1.1.3. TEST PLAN FOR DART DATA CLASS

ID	Test Plan Description
TC-1	Test for deserializing API response to AuthModel Instance used for Session Management.
TC-2	Test for deserializing API response to Bank Model Instance used to show banking details for donation.
TC-3	Test for deserializing API response to NGOModel Instance used for screening the NGO profile.
TC-4	Test for deserializing API response to NormalPostModel Instance used for screening Normal Post details.
TC-5	Test for deserializing API response to NotificationModel Instance used for screening Notification details.
TC-6	Test for deserializing API response to PeopleModel Instance used for screening General People profile.
TC-7	Test for deserializing API response to PollPostModel Instance used for screening Poll Post details.
TC-8	Test for deserializing API response to RequestPostModel Instance used for screening Request Post details.

Table 5: Unit Testing, Dart Data Class Test Plans

4.1.2. SYSTEM TESTING, TEST PLAN

4.1.2.1. TEST PLAN FOR ADMIN-STAFF PORTAL

ID	Test Plan Description
TC-1	Test for logging in with Admin Credentials.
TC-2	Test for logging in with Staff Credentials.
TC-3	Test for performing CRUD operations on Staff resources.
TC-4	Test for performing CRUD operations on NGO resources.
TC-5	Test for performing RUD operations on General People resource.
TC-6	Test for reviewing a Reported Post.
TC-7	Test for reading Reviewed Reports(s).
TC-8	Test for logging out from the portal.
TC-9	Test for changing the password of the logged-in user.

Table 6: System Testing, Admin-Staff Portal Test Plans

4.1.2.2. TEST PLAN FOR SASAE MOBILE APP

ID	Test Plan Description
TC-1	Test for performing CRUD operations on General People Resource.
TC-2	Test for logging in to the app with General People credentials.
TC-3	Test for changing the password of a logged-in user.
TC-4	Test for viewing a list of NGOs or an individual NGO detail.
TC-5	Test for filtering NGOs based on Field of Work.

SASAE

TC-6	Test for NGO Donation using Khalti payment gateway
TC-7	Test for performing CRUD Operations on Post, regardless of its type.
TC-8	Test for viewing a list of Posts or an individual Post detail.
TC-9	Test for reacting on a Post.
TC-10	Test for pushing a notification to the author of a post that has been reacted.
TC-11	Test for pushing a notification to the author of a post that has been removed.
TC-12	Test for logging out of the app.

Table 7: System Testing, Sasae Mobile App Test Plans

4.2. UNIT TESTING

4.2.1. TEST DJANGO WEB-APPLICATION

4.2.1.1. TEST AUTH PAGE

Test Case	TC-1
Objective	<ul style="list-style-type: none"> - Check if Admin and Staff can log in/log out or not. - Check if an NGO, a General People, or an unauthorised individual can log in or not.
Action	Run the test_auth.py script by executing the below command in the terminal: <i>python manage.py test core.tests.test_auth</i>
Expected Result	Test failed: 0, passed: 6 of 8 tests
Actual Result	Test failed: 2, passed: 6 of 8 tests
Conclusion	Test Unsuccessful

Table 8: Unit Testing, Django Web-App TC-1

```

    password = User.objects.create_user(**self.credentials)
    new_general = User.objects.create_user(**self.general_user)
    new_general.groups.add(self.general_group)
    new_general.save()

    def test_login_form_can_render(self):
        response = self.client.get(self.login_url)
        self.assertEqual(response.status_code, 200)
        self.assertTemplateUsed(response, 'core/account/login.html')

    def test_admin_can_login(self):
        response = self.client.post(self.login_url, self.admin_user, follow=True)
        self.assertTrue(response.context['user'].is_active)

    def test_admin_can_logout(self):
        response = self.client.post(self.logout_url, follow=True)
        self.assertFalse(response.context['user'].is_active)

    def test_staff_can_login(self):
        response = self.client.post(self.login_url, self.staff_user, follow=True)

```

Run: Test: core.tests.test_auth

Tests failed: 2, passed: 6 of 8 tests – 2 sec 792 ms

Test	Status
test_admin_can_login	✓
test_admin_can_logout	✗
test_general_cannot_login	✗
test_login_form_can_render	✓
test Ngo cannot login	✗
test_staff_can_login	✓
test_staff_can_logout	✗
test_un_authorize_cannot_login	✗

Figure 38: Unit Testing Django Web-App TC-1 Result Screenshot [ERROR]

[SOLUTION] LoginView class was extended to a CustomWebLoginView to override the form_valid function. Inside the form_valid function, code logic to restrict NGO and General People was added. The code snippet of the solution is below:

```
class CustomWebLoginView(LoginView):

    def form_valid(self, form):
        user = form.get_user()
        if not user.is_superuser and user.groups.first().name in ['General', 'NGO']:
            raise PermissionDenied
        return super().form_valid(form)
```

Figure 39: Unit Testing Django Web-App TC-1 Screenshot [SOLUTION] The test was re-run afterwards. The new test result screenshot is below:

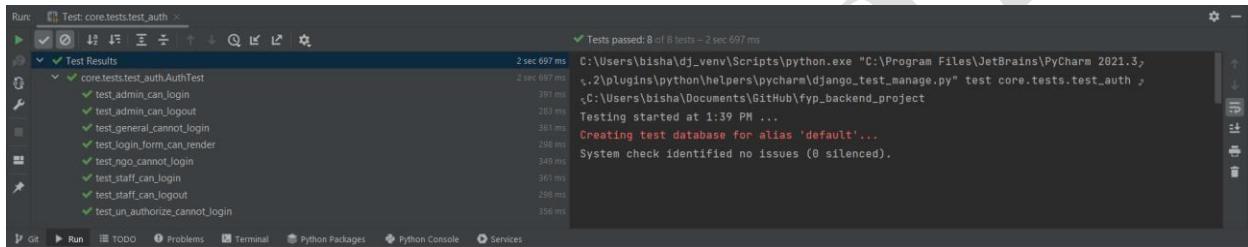


Figure 40: Unit Testing Django Web-App TC-1 Result Screenshot (NEW)

4.2.1.2. TEST HOME PAGES

Test Case	TC-2
Objective	- Check if Admin and Staff can access admin home and staff home or not.
Action	Run the test_home.py script by executing the below command in the terminal: <i>python manage.py test core.tests.test_home</i>
Expected Result	Test failed: 0, passed: 4 of 4 tests
Actual Result	Test failed: 0, passed: 4 of 4 tests
Conclusion	Test Successful

Table 9: Unit Testing, Django Web-App TC-2

SASAE

```

3
4
5 class AuthTest(BaseTest):
6
7     def setUp(self):
8         super().setUp()
9         self.staff_home = reverse('staff-home')
10        self.admin_home = reverse('admin-home')
11
12    def test_admin_access_admin_home(self):
13        self.login_as_admin()
14        response = self.client.get(self.admin_home)
15        self.assertEqual(response.status_code, 200)
16        self.assertTemplateUsed(response, 'core/admin/admin-home.html')
17
18    def test_admin_access_staff_home(self):
19        self.login_as_admin()
20        response = self.client.get(self.staff_home)
21        self.assertEqual(response.status_code, 200)
22        self.assertTemplateUsed(response, 'core/extensions/403-page.html')

```

Run: Test: core.tests.test_home-X

Tests passed: 4 of 4 tests – 956 ms

Test Results

- core.tests.test_home.AuthTest
 - test_admin_access_admin_home
 - test_admin_access_staff_home
 - test_staff_access_admin_home
 - test_staff_access_staff_home

C:\Users\bisha\dj_venv\Scripts\python.exe "C:\Program Files\JetBrains\PyCharm 2021.3, v.2\plugins\python\helpers\pycharm\django_test_manage.py" test core.tests.test_home ; C:\Users\bisha\Documents\GitHub\fyp_backend_project

Testing started at 1:53 PM ...
Creating test database for alias 'default'...
System check identified no issues (0 silenced).

Figure 41: Unit Testing, Django Web-App TC-2 Result Screenshot

4.2.1.3. TEST NGO CRUD PAGES

Test Case	TC-3
Objective	<ul style="list-style-type: none"> - Check if Staff can perform CRUD operations on NGO or not.
Action	<p>Run the test Ngo_crud.py script by executing the below command in the terminal:</p> <p><i>python manage.py test core.tests.test Ngo_crud</i></p>
Expected Result	Test failed: 0, passed: 4 of 4 tests
Actual Result	Test failed: 0, passed: 4 of 4 tests
Conclusion	Test Successful

Table 10: Unit Testing, Django Web-App TC-3

SASAE

The screenshot shows the PyCharm IDE interface. The code editor displays `test_ngo_crud.py` with several test methods: `test_staff_create Ngo`, `test_staff_read Ngo`, and `test_staff_read_ngos`. The run tool window at the bottom shows the test results: 7 tests passed in 2 seconds. The command used was `python manage.py test core.tests.test_ngo_crud`.

```

def test_staff_create Ngo(self):
    self.login_as_staff()
    response = self.client.post(self.create Ngo, self.payload, follow=True)
    self.assertEqual(response.status_code, 200)
    self.assertEqual(NGOUser.objects.filter(account__username=self.payload['username']).count(), 1)
    self.assertTemplateUsed(response, 'core/ngo/ngo-read.html')

def test_staff_read Ngo(self):
    self.login_as_staff()
    response = self.client.get(get_path(DBOperation.read, ngo.pk))
    self.assertEqual(response.status_code, 200)
    self.assertTemplateUsed(response, 'core/ngo/ngo-read.html')

def test_staff_read_ngos(self):
    self.login_as_staff()
    response = self.client.get(self.read_ngos)
    self.assertEqual(response.status_code, 200)

```

Figure 42: Unit Testing, Django Web-App TC-3 Result Screenshot

4.2.1.4. TEST GENERAL PEOPLE RUD PAGES

Test Case	TC-4
Objective	- Check if Staff can perform RUD operations on General People or not.
Action	Run the <code>test_people_rud.py</code> script by executing the below command in the terminal: <code>python manage.py test core.tests.test_people_rud</code>
Expected Result	Test failed: 0, passed: 5 of 5 tests
Actual Result	Test failed: 0, passed: 5 of 5 tests
Conclusion	Test Successful

Table 11: Unit Testing, Django Web-App TC-4

SASAE

The screenshot shows the PyCharm IDE interface. The code editor displays `test_people_rud.py` with the following content:

```

self.assertTemplateUsed(response, 'core/user/people-read.html')

def test_staff_access_update_people_form(self):
    self.login_as_staff()
    response = self.client.get(get_path(DBOperation.update, self.people.pk))
    self.assertEqual(response.status_code, 200)
    self.assertTemplateUsed(response, 'core/user/people-update.html')

def test_staff_update_people(self):
    self.login_as_staff()
    self.payload.pop('account')
    self.payload['address'] = 'Shaktapur'
    self.payload['is_verified'] = True
    response = self.client.post(get_path(DBOperation.update, pk=self.people.pk), self.payload, follow=True)
    new = PeopleUser.objects.filter(pk=self.people.pk).first()
    self.assertEqual(response.status_code, 200)
    self.assertEqual(new.address, self.payload['address'])
    self.assertEqual(new.is_verified, self.payload['is_verified'])
    self.assertTemplateUsed(response, 'core/user/people-read.html')

def test_staff_delete_people(self):
    self.login_as_staff()
    response = self.client.delete(get_path(DBOperation.delete, pk=self.people.pk), follow=True)

```

The run tool window at the bottom shows the following output:

```

Tests passed: 5 of 5 tests - 7 sec 551 ms
C:\Users\bisha\DJ_venv\Scripts\python.exe "C:\Program Files\JetBrains\PyCharm 2021.3,
s.2\plugins\python\helpers\pycharm\django_test_manage.py" test core.tests.test_people_rud ,
c:\Users\bisha\Documents\GitHub\fyp_backend_project
Testing started at 1:55 PM ...
Creating test database for alias 'default'...
System check identified no issues (0 silenced).

```

Figure 43: Unit Testing, Django Web-App TC-4 Result Screenshot

4.2.1.5. TEST POST REPORT REVIEW PAGE

Test Case	TC-5
Objective	- Check if Admin and Staff can read, access and review the report or not.
Action	Run the <code>test_post_report_review.py</code> script by executing the below command in the terminal: <code>python manage.py test core.tests.test_post_report_review</code>
Expected Result	Test failed: 0, passed: 6 of 6 tests
Actual Result	Test failed: 0, passed: 6 of 6 tests
Conclusion	Test Successful

Table 12: Unit Testing, Django Web-App TC-5

SASAE

```

Project: fyp_backend_project
File: core/tests/test_post_report_review.py
  def test_admin_review_report(self):
      self.login_as_admin()
      response = self.client.post(reverse('review-report', kwargs={'pk': self.report.id}), data=self.payload)
      self.assertEqual(response.status_code, 200)
      self.assertTemplateUsed(response, 'core/report/report-review-read.html')

  def test_staff_review_report(self):
      self.login_as_staff()
      response = self.client.post(reverse('review-report', kwargs={'pk': self.report.id}), data=self.payload,
                                  follow=True)
      self.assertEqual(response.status_code, 200)
      new = Report.objects.get(pk=self.report.id)
      self.assertEqual(new.is_reviewed, True)
      self.assertEqual(new.reason, self.payload['reason'])
      self.assertEqual(new.action, self.payload['action'])
      if self.payload['action'] == Report.ACTION[0][0]:
          self.assertEqual(new.post.is_removed, True)
      if self.payload['action'] == Report.ACTION[1][0]:
          self.assertEqual((new.post.people_posted_post_rn or new.post.ngo_posted_post_rn).first().account.is_active,
                          False)

Run: Test: core.tests.test_post_report_review
  ✓ Tests passed: 6 of 6 tests – 1 sec 999 ms
  ✓ 1 sec 999 ms C:\Users\bisha\dj_venv\Scripts\python.exe "C:\Program Files\JetBrains\PyCharm 2021.3,
  ✓ 2\plugins\python\helpers\pycharm\django_test_manage.py" test core.tests,
  ✓ .test_post_report_review C:\Users\bisha\Documents\GitHub\fyp_backend_project
  ✓ Testing started at 1:56 PM ...
  ✓ Creating test database for alias 'default'...
  ✓ System check identified no issues (0 silenced).

```

Figure 44: Unit Testing, Django Web-App TC-5 Result Screenshot

4.2.1.6. TEST STAFF CRUD PAGES

Test Case	TC-6
Objective	- Check if Admin and Staff can perform CRUD operations on Staff or not.
Action	Run the <code>test_staff_crud.py</code> script by executing the below command in the terminal: <code>python manage.py test core.tests.test_staff_crud</code>
Expected Result	Test failed: 0, passed: 12 of 12 tests
Actual Result	Test failed: 0, passed: 12 of 12 tests
Conclusion	Test Successful

Table 13: Unit Testing, Django Web-App TC-6

SASAE

The screenshot shows the PyCharm IDE interface with the following details:

- Project Structure:** The project is named "fyp_backend_project". It contains a "core" package with "migrations", "static", and "templates" sub-directories. There are also "tests" and "api" packages.
- Code Editor:** The code editor displays a Python test file, `test_staff_crud.py`, which contains several test methods for the `StaffCRUDTest` class. The methods include `test_staff_read_staffs`, `test_staff_access_update_staff_form`, `test_staff_delete_staff`, and others.
- Run Tab:** The "Run" tab shows the command used to run the tests: `C:\Program Files\JetBrains\PyCharm 2021.3\bin\plugins\python\helpers\pycharm\django_test_manage.py test core.tests.test_staff_crud`.
- Toolbars and Status Bar:** The top bar includes standard PyCharm icons like file operations and search. The status bar at the bottom right shows "Tests passed: 12 of 12 tests - 3 sec 124 ms".
- Bottom Panel:** The bottom panel displays the "Test Results" section, which lists all 12 test cases with their execution times. The results are all marked with a green checkmark, indicating they passed.

Figure 45: Unit Testing, Django Web-App TC-6 Result Screenshot

4.2.1.7. OVERALL TEST RESULT

Test: core.tests: 42 total, 42 passed		14.87 s
		Collapse Expand
core.tests.test_auth.AuthTest		2.70 s
test_can_admin_login	passed	394 ms
test_can_admin_logout	passed	304 ms
test_can_general_login	passed	350 ms
test_can_ngo_login	passed	358 ms
test_can_staff_login	passed	367 ms
test_can_staff_logout	passed	291 ms
test_can_un_authorize_login	passed	354 ms
test_login_view_get_request	passed	285 ms
core.tests.test_home.AuthTest		928 ms
test_admin_access_admin_home	passed	248 ms
test_admin_access_staff_home	passed	230 ms
test_staff_access_admin_home	passed	224 ms
test_staff_access_staff_home	passed	226 ms
core.tests.test Ngo_crud.NGOCRUDTest		3.35 s
test_staff_access_create ngo_form	passed	362 ms
test_staff_access_update ngo_form	passed	1.50 s
test_staff_create ngo	passed	307 ms
test_staff_delete ngo	passed	333 ms
test_staff_read ngo	passed	309 ms
test_staff_read ngos	passed	223 ms
test_staff_update ngo	passed	317 ms
core.tests.test_people_rud.PeopleCRUDTest		2.91 s
test_staff_access_update people_form	passed	325 ms
test_staff_delete people	passed	297 ms
test_staff_read people	passed	307 ms
test_staff_read peoples	passed	288 ms
test_staff_update people	passed	1.70 s
core.tests.test_post_report_review.PostReportReviewTest		1.87 s
test_admin_access_reports	passed	315 ms
test_admin_read_report	passed	336 ms
test_admin_review_report	passed	300 ms
test_staff_access_reports	passed	295 ms
test_staff_read_report	passed	306 ms
test_staff_review_report	passed	316 ms
core.tests.test_staff_crud.StaffCRUDTest		3.11 s
test_admin_access_create staff_form	passed	251 ms
test_admin_access_update staff_form	passed	273 ms
test_admin_create staff	passed	326 ms
test_admin_delete staff	passed	258 ms
test_admin_read staff	passed	229 ms
test_admin_read_staffs	passed	251 ms
test_admin_update staff	passed	383 ms
test_staff_access_create staff_form	passed	228 ms
test_staff_access_update staff_form	passed	220 ms
test_staff_delete staff	passed	225 ms
test_staff_read_own_self	passed	228 ms
test_staff_read_staffs	passed	234 ms

Generated by PyCharm on 4/23/22, 12:53 AM

Figure 46: Overall Unit Testing Result: Django Web-Application

4.2.2. TEST RESTFUL APIs

4.2.2.1. TEST AUTH APIs

Test Case	TC-1
Objective	- Check if APIs for Login, Password Change, Password Reset, and Token Verification are working or not.
Action	Run the Collection Runner over the Auth folder in Postman.
Expected Result	Test failed: 0, passed: 4 of 4 tests
Actual Result	Test failed: 0, passed: 4 of 4 tests
Conclusion	Test Successful

Table 14: Unit Testing, RESTful API TC-1

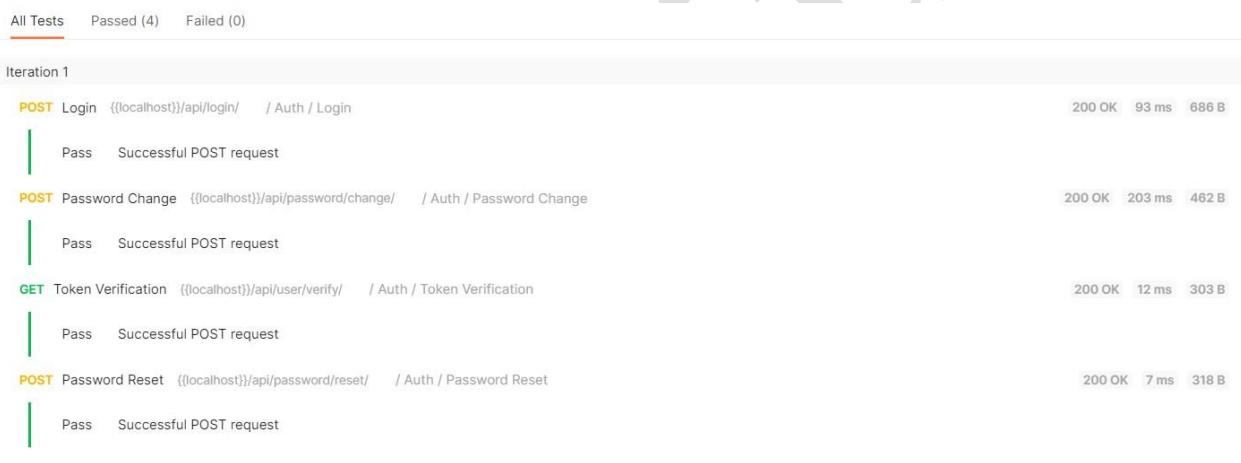


Figure 47: Unit Testing, RESTful API TC-1 Result Screenshot

4.2.2.2. TEST POST APIs

Test Case	TC-2
Objective	- Check if APIs for Post Report, Posts List, Post Detail by ID, Delete Post, Post Update Details, Related To and User Posts are working or not.
Action	Run the Collection Runner over the Post folder in Postman.
Expected Result	Test failed: 0, passed: 7 of 7 tests

SASAE

Actual Result	Test failed: 0, passed: 7 of 7 tests
Conclusion	Test Successful

Table 15: Unit Testing, RESTful API TC-2

All Tests	Passed (7)	Failed (0)
Iteration 1		
<code>POST</code> Report post {{localhost}}/api/post/13/report/ / Post / Report post		200 OK 27 ms 306 B
Pass Successful POST request		
<code>GET</code> posts {{localhost}}/api/posts/ / Post / posts		200 OK 76 ms 16.492 KB
Pass Successful POST request		
<code>GET</code> post {{localhost}}/api/post/73/ / Post / post		200 OK 28 ms 1.291 KB
Pass Successful POST request		
<code>DELETE</code> Delete post {{localhost}}/api/post/90/delete/ / Post / Delete post		204 No Content 25 ms 262 B
Pass Successful POST request		
<code>GET</code> Retrieve to Update Post {{localhost}}/api/post/47/detail/ / Post / Retrieve to Update Post		200 OK 28 ms 527 B
Pass Successful POST request		
<code>GET</code> relatedTo {{localhost}}/api/post/relatedto / Post / relatedTo		200 OK 10 ms 907 B
Pass Successful POST request		
<code>GET</code> User Posts {{localhost}}/api/ngo/1/posts/ / Post / User Posts		200 OK 20 ms 1.991 KB
Pass Successful POST request		

Figure 48: Unit Testing, RESTful API TC-2 Result Screenshot

4.2.2.3. TEST NORMAL POST APIs

Test Case	TC-3
Objective	- Check if APIs for CRUD operations and reaction (i.e., Upvote, Downvote) on Normal Post is working or not.
Action	Run the Collection Runner over the Normal Post folder in Postman.
Expected Result	Test failed: 0, passed: 4 of 4 tests
Actual Result	Test failed: 0, passed: 4 of 4 tests
Conclusion	Test Successful

Table 16: Unit Testing, RESTful API TC-3

SASAE

All Tests Passed (4) Failed (0)

Iteration 1

Method	Endpoint	Description	Status	Time	Size
POST	Post normal post {{localhost}}/api/post/normal/	/ Post / Normal Post / Post normal post	201 Created	1596 ms	575 B
		Pass Successful POST request			
POST	Upvote normal post {{localhost}}/api/post/92/upvote/	/ Post / Normal Post / Upvote normal post	200 OK	42 ms	304 B
		Pass Successful POST request			
POST	Downvote normal post {{localhost}}/api/post/92/downvote/	/ Post / Normal Post / Downvote normal post	200 OK	46 ms	306 B
		Pass Successful POST request			
PUT	Update normal post {{localhost}}/api/post/26/update/	/ Post / Normal Post / Update normal post	200 OK	41 ms	328 B
		Pass Successful POST request			

Figure 49: Unit Testing, RESTful API TC-3 Result Screenshot

4.2.2.4. TEST POLL POST API

Test Case	TC-4
Objective	- Check if APIs for CRUD operations and Poll reaction on Poll Post is working or not.
Action	Run the Collection Runner over the Poll Post folder in Postman.
Expected Result	Test failed: 0, passed: 3 of 3 tests
Actual Result	Test failed: 0, passed: 3 of 3 tests
Conclusion	Test Successful

Table 17: Unit Testing, RESTful API TC-4

All Tests Passed (3) Failed (0)

Iteration 1

Method	Endpoint	Description	Status	Time	Size
POST	Post poll post {{localhost}}/api/post/poll/	/ Post / Poll Post / Post poll post	201 Created	6079 ms	950 B
		Pass Successful POST request			
POST	Poll poll post http://127.0.0.1:8000/api/post/88/poll/80/	/ Post / Poll Post / Poll poll post	200 OK	35 ms	304 B
		Pass Successful POST request			
PUT	Update Poll Post http://127.0.0.1:8000/api/post/59/update/	/ Post / Poll Post / Update Poll Post	200 OK	39 ms	328 B
		Pass Successful POST request			

Figure 50: Unit Testing, RESTful API TC-4 Result Screenshot

4.2.2.5. TEST REQUEST POST APIs

Test Case	TC-5
Objective	- Check if APIs for CRUD operations and reaction (i.e., Sign, Participate) on Request Post is working or not.
Action	Run the Collection Runner over the Post Request folder in Postman.
Expected Result	Test failed: 0, passed: 3 of 3 tests
Actual Result	Test failed: 0, passed: 3 of 3 tests
Conclusion	Test Successful

Table 18: Unit Testing, RESTful API TC-5

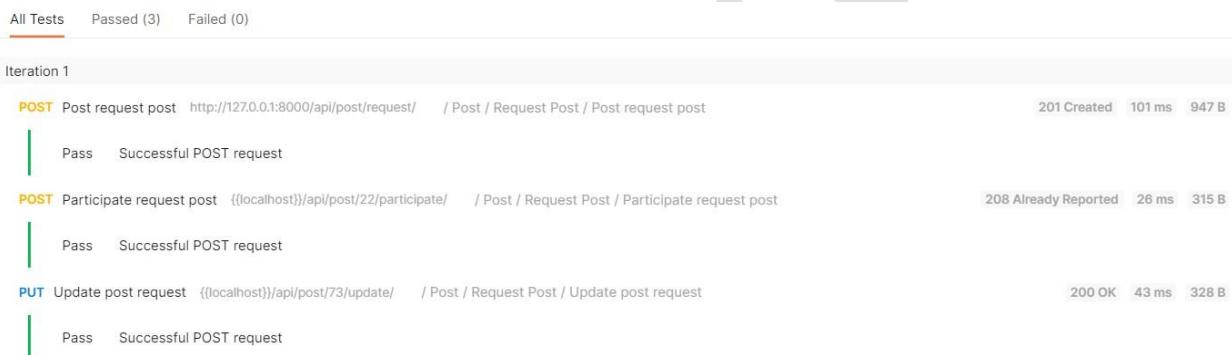


Figure 51: Unit Testing, RESTful API TC-5 Result Screenshot

4.2.2.6. TEST NGO APIs

Test Case	TC-6
Objective	- Check if APIs for NGO Detail by ID, NGO Options, and NGOs List are working or not.
Action	Run the Collection Runner over the NGO folder in Postman.
Expected Result	Test failed: 0, passed: 3 of 3 tests
Actual Result	Test failed: 0, passed: 3 of 3 tests
Conclusion	Test Successful

Table 19: Unit Testing, RESTful API TC-6

All Tests	Passed (3)	Failed (0)
Iteration 1		
<code>GET</code> Get NGO	<code>localhost</code> /api/ngo/2 / NGO / Get NGO	200 OK 17 ms 1.095 KB
Pass	Successful POST request	
<code>GET</code> NGO Options	<code>localhost</code> /api/post/ngos / NGO / NGO Options	200 OK 13 ms 662 B
Pass	Successful POST request	
<code>GET</code> List of NGOs	<code>localhost</code> /api/ngos / NGO / List of NGOs	200 OK 22 ms 1.059 KB
Pass	Successful POST request	

Figure 52: Unit Testing, RESTful API TC-6 Result Screenshot

4.2.2.7. TEST GENERAL PEOPLE APIs

Test Case	TC-7
Objective	- Check if APIs for CRUD operations on General People are working or not.
Action	Run the Collection Runner over the General People folder in Postman.
Expected Result	Test failed: 0, passed: 4 of 4 tests
Actual Result	Test failed: 0, passed: 4 of 4 tests
Conclusion	Test Successful

Table 20: Unit Testing, RESTful API TC-7

SASAE

All Tests	Passed (5)	Failed (0)
Iteration 1		
POST Create People User {{localhost}}/api/people/add/ / General People / Create People User		
Pass	Successful POST request	
PUT Update People User {{localhost}}/api/people/update/ / General People / Update People User		201 Created 171 ms 307 B
Pass	Successful POST request	
Pass	Successful POST request	
GET Detail People User {{localhost}}/api/people/detail/ / General People / Detail People User		200 OK 26 ms 669 B
Pass	Successful POST request	
DELETE Delete people user {{localhost}}/api/people/delete/ / General People / Delete people user		204 No Content 25 ms 269 B
Pass	Successful POST request	

Figure 53: Unit Testing, RESTful API TC-7 Result Screenshot

4.2.3. TEST DART DATA CLASSES

4.2.3.1. TEST AUTHMODEL

Test Case	TC-1
Objective	- Check if the API response is properly deserialized to an AuthModel data instance.
Action	Run the test by selecting <code>test\auth_model_test.dart</code> and clicking the run icon corresponding to the right of it.
Expected Result	Test failed: 0, passed: 7 of 7 tests
Actual Result	Test failed: 0, passed: 7 of 7 tests
Conclusion	Test Successful

Table 21: Unit Testing, Dart Data Class TC-1

SASAE

```
auth_model_test.dart x
Filter (e.g. text, exclude, @tag)
7/7 tests passed (100%)
> o integration_test/auth_test.dart
> o integration_test/post_test.dart
✓ v authauth_model_test.dart 7/7 passed: 35ms
    (setUpAll) 18ms
    (tearDownAll) 2.0ms
    ✓ JSON de-serialized to AuthModel Instance 4.0ms
    ✓ TokenKey casted 3.0ms
    ✓ Group casted 3.0ms
    ✓ AccountID casted 2.0ms
    ✓ ProfileID casted 3.0ms
    > o testbank_model_test.dart 8/8 passed
    > o testingo_model_test.dart 21/21 passed
    > o testnormal_post_model_test.dart 17/17 passed
    > o testnotification_model_test.dart 10/10 passed
    > o testpeople_model_test.dart 16/16 passed
    > o testpoll_post_model_test.dart 0/14 passed
    > o testrequest_post_model_test.dart
    > o testwidget_test.dart

test > auth_model_test.dart > main > test("JSON de-serialized to AuthModel Instance")
You, 12 hours ago | 1 author (You)
1 import 'package:faker/faker.dart';
2 import 'package:sasae_flutter_app/models/auth.dart';
3 import 'package:test/test.dart';
4
5 Run | Debug
6 void main() {
7     late Map<String, dynamic> json;
8     late AuthModel authModel;
9
10    setupAll() {
11        json = {
12            'key': faker.jwt.secret,
13            'group': faker.randomGenerator.element(['NGO', 'General']),
14            'account_id': faker.randomGenerator.integer(2000),
15            'profile_id': faker.randomGenerator.integer(2000)
16        };
17    }
18    Run | Debug
19    test('JSON de-serialized to AuthModel Instance', () {
20        authModel = AuthModel.fromAPIResponse(json);
21        expect(authModel, isA<AuthModel>());
22    });
23    Run | Debug
24    test('Tokenkey casted', () {
25        expect(authModel.tokenKey, json['key']);
26    });
27    Run | Debug
28    test('Group casted', () {
29        expect(authModel.group, json['group']);
30    });
31    Run | Debug
32    test('AccountID casted', () {
33        expect(authModel.accountID, json['account_id']);
34    });
35
```

Figure 54: Unit Testing, Dart Data Class TC-1 Result Screenshot

4.2.3.2. TEST BANKMODEL

Test Case	TC-2
Objective	- Check if the API response is properly deserialized to a BankModel data instance.
Action	Run the test by selecting <code>test\bank_model_test.dart</code> and clicking the run icon corresponding to the right of it.
Expected Result	Test failed: 0, passed: 8 of 8 tests
Actual Result	Test failed: 0, passed: 8 of 8 tests
Conclusion	Test Successful

Table 22: Unit Testing, Dart Data Class TC-2

SASAE

The screenshot shows the Android Studio interface during unit testing. On the left, a tree view displays various test files and their results. The file `bank_model_test.dart` is selected, showing 8/8 tests passed (100%). On the right, the code for `bank_model_test.dart` is shown with several test cases highlighted in green. The first test case is:

```

void main() {
    late Map<String, dynamic> json;
    late BankModel bankModel;

    setUpAll(() {
        json = {
            'bank_name': faker.company.name(),
            'bank_branch': faker.address.city(),
            'bank_BSB': int.parse(faker.randomGenerator.numberOfLength(6)).toString(),
            'bank_account_name': faker.company.name(),
            'bank_account_number': int.parse(faker.randomGenerator
                .numberOfLength(faker.randomGenerator.integer(16, min: 9)))
                .toString()
        };
    });

    test('JSON deserialized to BankModel instance', () {
        bankModel = BankModel.fromJson(json);
        expect(bankModel, isA<BankModel>());
    });

    test('Bank name casted', () {
        expect(bankModel.bankName, json['bank_name']);
    });

    test('Bank branch casted', () {
        expect(bankModel.bankBranch, json['bank_branch']);
    });

    test('Bank BSB casted', () {
        expect(bankModel.bankBSB, json['bank_BSB']);
    });
}

```

The status bar at the top right indicates "Exited".

Figure 55: Unit Testing, Dart Data Class TC-2 Result Screenshot

4.2.3.3. TEST NGOMODEL

Test Case	TC-3
Objective	- Check if the API response is properly deserialized to an NGOModel data instance.
Action	Run the test by selecting <code>test\ngo_model_test.dart</code> and clicking the run icon corresponding to the right of it.
Expected Result	Test failed: 0, passed: 21 of 21 tests
Actual Result	Test failed: 0, passed: 21 of 21 tests
Conclusion	Test Successful

Table 23: Unit Testing, Dart Data Class TC-3

SASAE

Figure 56: Unit Testing, Dart Data Class TC-3 Result Screenshot

4.2.3.4. TEST NORMALPOSTMODEL

Test Case	TC-4
Objective	<ul style="list-style-type: none">- Check if the API response is properly deserialized to a NormalPostModel data instance.
Action	Run the test by selecting <code>test\NormalPostModelTest.dart</code> and clicking the run icon corresponding to the right of it.
Expected Result	Test failed: 0, passed: 17 of 17 tests
Actual Result	Test failed: 0, passed: 17 of 17 tests
Conclusion	Test Successful

Table 24: Unit Testing, Dart Data Class TC-4

SASAE

The screenshot shows the Android Studio interface with two main panes. The left pane displays a tree view of test results under the 'TESTING' tab, showing 17/17 tests passed (100%) across various files like 'integration_test_auth_test.dart', 'testbank_model_test.dart', etc. The right pane shows the code for 'normal_post_model_test.dart' and a detailed log of test cases. The log includes:

- ✓ JSON serialized to BankModel Instance
- ✓ Normal Post ID casted
- ✓ Related-to casted
- ✓ Post Content casted
- ✓ Created-on casted
- ✓ Modified-on casted
- ✓ Is-Anonymous casted
- ✓ Post Type casted
- ✓ Poked NGO casted
- ✓ Author name casted
- ✓ Author ID casted
- ✓ Post-Image casted
- ✓ Up-vote casted
- ✓ Down-vote casted
- ✓ Up-voted casted

The log concludes with 'Exited'.

Figure 57: Unit Testing, Dart Data Class TC-4 Result Screenshot

4.2.3.5. TEST NOTIFICATIONMODEL

Test Case	TC-5
Objective	- Check if the API response is properly deserialized to a NotificationModel data instance.
Action	Run the test by selecting <code>test/notification_model_test.dart</code> and clicking the run icon corresponding to the right of it.
Expected Result	Test failed: 0, passed: 10 of 10 tests
Actual Result	Test failed: 0, passed: 10 of 10 tests
Conclusion	Test Successful

Table 25: Unit Testing, Dart Data Class TC-5

SASAE

The screenshot shows a Flutter development environment with three main panes. The left pane displays a tree view of test results with 10/10 tests passed. The middle pane contains the source code for `notification_model_test.dart`, which includes several test cases using the `Faker` library to generate JSON data and verify its deserialization into `NotificationModel` instances. The right pane provides a summary of the test results, indicating that all JSON fields have been successfully casted to their respective types.

```

TESTING
Filter (e.g. text, exclude, @tag)
10/10 tests passed (100%)
> o integration_test\auth_test.dart
> o integration_test\post_test.dart
> o test\bank_model_test.dart 7/7 passed: 35ms
> o test\ngrok_model_test.dart 2/21 passed: 114ms
> o test\normal_post_model_test.dart 17/17 passed: 89ms
> o test\notification_model_test.dart 10/10 passed: 59ms
  o (setUpAll) 27ms
  o (tearDownAll) 2.0ms
  o JSON serialized to NotificationModel Instance 9.0...
  o Notification ID casted 3.0ms
  o Title casted 4.0ms
  o Body casted 4.0ms
  o Channel casted 3.0ms
  o Post type casted 2.0ms
  o Post ID casted 2.0ms
  o Is-Read casted 3.0ms
> o test\people_model_test.dart 16/16 passed
> o test\poll_post_model_test.dart 0/14 passed
> o test\request_post_model_test.dart
> o test\widget_test.dart

notification_model_test.dart x
void main() {
  late Map<String, dynamic> json;
  late NotificationModel notificationModel;

  setUpAll() {
    bool _ = faker.randomGenerator.boolean();
    json = {
      'id': faker.Faker().integer(1000),
      'title': fa,
      'body': faker.packageOf('src/faker.dart').in(''),
      'channel': faker.randomGenerator.element(NotificationChannel.values).r,
      'post_type':
        ? faker.randomGenerator.element(PostType.values).name : null,
      'post_id': _ ? faker.randomGenerator.integer(1000) : null,
      'isRead': faker.randomGenerator.boolean(),
    };
  }

  test('JSON serialized to NotificationModel Instance', () {
    notificationModel = NotificationModel.fromMapResponse(json);
    expect(notificationModel, isA<NotificationModel>());
  });

  test('Notification ID casted', () {
    expect(notificationModel.id, isA<int>());
  });

  test('Title casted', () {
    expect(notificationModel.title, json['title']);
  });

  test('Body casted', () {
    expect(notificationModel.body, json['body']);
  });
}

Filter (e.g. text, exclude)
sasae_flutter_app x
✓ JSON serialized to NotificationModel instance
✓ Notification ID casted
✓ Title casted
✓ Body casted
✓ Channel casted
✓ Post type casted
✓ Post ID casted
✓ Is-Read casted
Exited

```

Figure 58: Unit Testing, Dart Data Class TC-5 Result Screenshot

4.2.3.6. TEST PEOPLEMODEL

Test Case	TC-6
Objective	- Check if the API response is properly deserialized to a PeopleModel data instance.
Action	Run the test by selecting <code>test\people_model_test.dart</code> and clicking the run icon corresponding to the right of it.
Expected Result	Test failed: 0, passed: 7 of 7 tests
Actual Result	Test failed: 0, passed: 7 of 7 tests
Conclusion	Test Successful

Table 26: Unit Testing, Dart Data Class TC-6

SASAE

```

test > people.model_test.dart > main > test['JSON deserialized to PeopleModel instance']
  18   gender : faker.randomGenerator.element(['Male', 'Female', 'LGBIQ+']),
  19   'date_of_birth': Jiffy(faker.date.dateTime(maxYear: 2010, minYear: 1900))
  20     .format('yyyy-MM-dd'),
  21   'address': faker.address.streetAddress(),
  22   'phone': Faker.phoneNumber.us(),
  23   'email': Faker.internet.email(),
  24   'posted_post': Set<int>.of(List.generate(
  25     faker.randomGenerator.integer(250),
  26     (index) => faker.randomGenerator.integer(3000)).toList(), // List.generate // Set.of
  27   'date_joined': Jiffy(faker.date.dateTime(maxYear: 2010, minYear: 1900))
  28     .format("yyyy-MM-dd'T'HH:mm:ss"),
  29   'isVerified': isVerified
  30     ? faker.image.image(width: 800, height: 600, random: true)
  31     : null,
  32   );
  33 });
  34 }

Run | Debug
test['JSON deserialized to PeopleModel instance', () {
  final peopleModel = PeopleModel.fromJson(json);
  expect(peopleModel, isA());
}];

Run | Debug
test['Account ID casted', () {
  final peopleModel = PeopleModel.fromJson(json);
  expect(peopleModel.id, json['id']);
}];

Run | Debug
test['Is-verified casted', () {
  final peopleModel = PeopleModel.fromJson(json);
  expect(peopleModel.isVerified, json['is_verified']);
}];

Run | Debug
test['Display picture casted', () {
  final peopleModel = PeopleModel.fromJson(json);
  expect(peopleModel.displayPicture, json['display_picture']);
}];

Run | Debug
test['Username casted', () {
  final peopleModel = PeopleModel.fromJson(json);
  expect(peopleModel.username, json['username']);
}];

Run | Debug
test['Joined date casted', () {
  final peopleModel = PeopleModel.fromJson(json);
  expect(peopleModel.dateJoined, json['date_joined']);
}];

Run | Debug
test['Citizenship photo casted', () {
  final peopleModel = PeopleModel.fromJson(json);
  expect(peopleModel.citizenshipPhoto, json['citizenship_photo']);
}];
  
```

Figure 59: Unit Testing, Dart Data Class TC-6 Result Screenshot [ERROR]

[SOLUTION] Rewrite the `fromJson` instance method to properly extract the fields from JSON and cast them properly to the related instance fields. The code snippet of the solution is below:

```

factory PeopleModel.fromJson(Map<String, dynamic> map) {
  return PeopleModel(
    id: map['id']?.toInt() ?? 0,
    isVerified: map['is_verified'] ?? false,
    displayPicture: map['display_picture'] ?? '',
    username: map['username'] ?? '',
    fullname: map['full_name'] ?? '',
    gender: map['gender'] ?? '',
    birthDate: Jiffy(map['date_of_birth'], 'yyyy-MM-dd').dateTime,
    address: map['address'] ?? '',
    phone: map['phone'] ?? '',
    email: map['email'] ?? '',
    postedPosts: List<int>.from(map['posted_post']),
    joinedDate: Jiffy(map['date_joined'], "yyyy-MM-dd'T'HH:mm:ss").dateTime,
    citizenshipPhoto: map['citizenship_photo'],
  ); // PeopleModel
}
  
```

You, last month • implemented Image view screen and multi-user base...

Figure 60: Unit Testing, Dart Data Class TC-6 Screenshot [SOLUTION] The test was re-run afterwards. The new test result screenshot is below:

```

    ✓ test\people_model_test.dart 16/16 passed: 85ms
      ✓ (setUpAll) 39ms
      ✓ (tearDownAll) 3.0ms
      ✓ JSON serialized to PeopleModel Instance 12ms
      ✓ Account ID casted 3.0ms
      ✓ Is-verified casted 3.0ms
      ✓ Display picture casted 3.0ms
      ✓ Username casted 2.0ms
      ✓ Fullname casted 2.0ms
      ✓ Gender casted 2.0ms
      ✓ Birthdate casted 4.0ms
      ✓ Address casted 2.0ms
      ✓ Phone casted 2.0ms
      ✓ Email casted 1.0ms
      ✓ Posted posts casted 2.0ms
      ✓ Joined date casted 2.0ms
      ✓ Citizenship photo casted 3.0ms
  
```

Figure 61: Unit Testing, Dart Data Class TC-6 Result Screenshot (NEW)

4.2.3.7. TEST POLLPOSTMODEL

Test Case	TC-7
Objective	- Check if the API response is properly deserialized to a PollPostModel data instance.
Action	Run the test by selecting <code>test\poll_post_model_test.dart</code> and clicking the run icon corresponding to the right of it.
Expected Result	Test failed: 0, passed: 16 of 16 tests
Actual Result	Test failed: 0, passed: 16 of 16 tests
Conclusion	Test Successful

Table 27: Unit Testing, Dart Data Class TC-7

SASAE

The screenshot shows the Android Studio interface with two main panes. The left pane displays a tree view of test results under the 'TESTING' tab, indicating 16/16 passed. The right pane shows the source code for 'poll_post_model_test.dart' and its corresponding test cases. The test cases include assertions for JSON deserialization and various casting operations. A status bar at the bottom right indicates 'Exited'.

```

TESTING
Filter (e.g. text, exclude, @tag)
Running tests, 16/16 passed (100%)
  o integration_testauth_test.dart
  o integration_testpost_test.dart
  o testbank_model_test.dart 7/7 passed: 35ms
  o testbygo_mode_test.dart 2/21 passed: 114ms
  o testnormal_post_model_test.dart 17/17 passed: 89ms
  o testnotification_model_test.dart 10/10 passed: 59ms
  o testpeople_model_test.dart 16/16 passed: 85ms
  o testpoll_post_model_test.dart 16/16 passed: 135ms
    o (setUpAll) 60ms
    o (tearDownAll) 3.0ms
    o JSON serialized to BankModel instance 18ms
    o Normal Post ID casted 5.0ms
    o Related-to casted 5.0ms
    o Post Content casted 5.0ms
    o Created-on casted 4.0ms
    o Modified-on casted 4.0ms
    o Is-Anonymous casted 2.0ms
    o Post Type casted 4.0ms
    o Poked NGO casted 4.0ms
    o Author name casted 5.0ms
    o Author ID casted 3.0ms
    o Poll-post ends-on casted 4.0ms
    o Up-vote casted 5.0ms
    o Down-vote casted 4.0ms
  o testrequest_post_model_test.dart
  o testwidget_test.dart

poll_post_model_test.dart
  main > test('JSON serialized to BankModel Instance')
  ...
  "post_content": faker.lorem.sentences(rand.nextInt(20 - 3) + 3).join(''),
  ...
  "modified_on": Jiffy(faker.date.dateTime(maxYear: 2010, minYear: 1980))
  ...
  "is_anonymous": faker.randomGenerator.boolean(),
  ...
  "poked ngo": list.generate(
    faker.randomGenerator.integer(8, min: 0),
    (index) => {
      ...
      "id": faker.randomGenerator.integer(1000),
      ...
      "full_name": faker.company.name(),
      ...
      "display_picture": faker.image.image(random: true),
      ...
    });
  ...
  "author": Faker.person.name(),
  "author_id": faker.randomGenerator.integer(1000),
  "post poll": {
    ...
    "id": Faker.randomGenerator.integer(1000),
    "ends_on": Jiffy(faker.date.dateTime(maxYear: 2010, minYear: 1980))
    ...
    "choice": choice,
    "options": pollOptions,
    ...
  };
  ...
);
Run | Debug
test('JSON serialized to BankModel Instance', () {
  pollPost = PollPostModel.fromAPIResponse(json);
  expect(pollPost, isA<PollPostModel>());
});

Run | Debug
test('Normal Post ID casted', () {
  expect(pollPost.id, json['id']);
});

Run | Debug
test('Related-to casted', () {
  expect(pollPost.relatedTo, json['related_to']);
});

```

Figure 62: Unit Testing, Dart Data Class TC-7 Result Screenshot

4.2.3.8. TEST REQUESTPOSTMODEL

Test Case	TC-8
Objective	- Check if the API response is properly deserialized to a RequestPostModel data instance.
Action	Run the test by selecting <i>testrequest_post_model_test.dart</i> and clicking the run icon corresponding to the right of it.
Expected Result	Test failed: 0, passed: 20 of 20 tests
Actual Result	Test failed: 0, passed: 20 of 20 tests
Conclusion	Test Successful

Table 28: Unit Testing, Dart Data Class TC-8

SASAE

```

TESTING
Filter (e.g. text, exclude, @tag)
20/20 tests passed (100%)
> o integration_test\auth_test.dart
> o integration_test\post_test.dart
> o test\auth_model_test.dart 7/7 passed: 35ms
> o test\bank_model_test.dart 2/21 passed: 114ms
> o test\ngo_model_test.dart 2/21 passed: 114ms
> o test\normal_post_model_test.dart 17/17 passed: 89ms
> o test\notification_model_test.dart 10/10 passed: 59ms
> o test\people_model_test.dart 16/16 passed: 85ms
> o test\poll_post_model_test.dart 16/16 passed: 135ms
> o test\request_post_model_test.dart 20/20 passed: 101ms
  (setUpAll) 44ms
  (tearDownAll) 2.0ms
  JSON deserialized to RequestModel Instance 12ms
  Normal Post ID casted 3.0ms
  Related-to casted 3.0ms
  Post Content casted 2.0ms
  Created-on casted 2.0ms
  Modified-on casted 2.0ms
  Is-Anonymous casted 1.0ms
  Post Type casted 3.0ms
  Poked NGO casted 3.0ms
  Author name casted 2.0ms
  Author ID casted 2.0ms
  Minimum participation casted 3.0ms
  Target participation casted 3.0ms
  Maximum participation casted 4.0ms
  Request-post ends-on casted 2.0ms
  Is-Participated casted 2.0ms
  Participation casted 2.0ms
  Request-type casted 3.0ms
> o test\widget_test.dart

request_post_model_test.dart x
test > request_post_model_test.dart > main > test[JSON deserialized to RequestModel Instance]
      ...
      "display_picture": faker.image.image(random: true),
      ...
    },
    "author": faker.person.name(),
    "author_id": faker.randomGenerator.integer(1000),
    "post_request": {
      "id": faker.randomGenerator.integer(1000),
      "min": min,
      "max": max,
      "target": target,
      "ends_on": jiffy(faker.date.datetime(maxYear: 2010, minYear: 1900))
        .format("yyyy-MM-dd'T'HH:mm:ss"),
      "request_type": faker.randomGenerator.element(['Join', 'Petition']),
      "reacted_by": faker.randomGenerator
        .numbers(1500, faker.randomGenerator.integer(1500)),
      "is_participated": faker.randomGenerator.boolean()
    };
  });
}

Run | Debug
test('JSON deserialized to RequestModel Instance', () {
  requestPostModel = RequestPostModel.fromAPIResponse(json);
  expect(requestPostModel, isA<RequestPostModel>());
});

Run | Debug
test('Normal Post ID casted', () {
  expect(requestPostModel.id, json['id']);
});

Run | Debug
test('Related-to casted', () {
  expect(requestPostModel.relatedTo, json['related_to']);
});

Run | Debug
test('Post Content casted', () {
  expect(requestPostModel.postContent, json['post_content']);
});

```

Figure 63: Unit Testing, Dart Data Class TC-8 Result Screenshot

4.3. SYSTEM TESTING

4.3.1. ADMIN-STAFF PORTAL TESTING

4.3.1.1. TEST ADMIN LOGIN

Test Case	TC-1
Objective	To test whether the admin can log in or not
Action	Enter the admin credentials in the login screen and click the login button.
Expected Result	On login, the user must be redirected to the Admin dashboard/home.
Actual Result	The user was redirected to the Admin dashboard/home.
Conclusion	Test Successful

Table 29: System Testing, Admin-Staff Portal TC-1

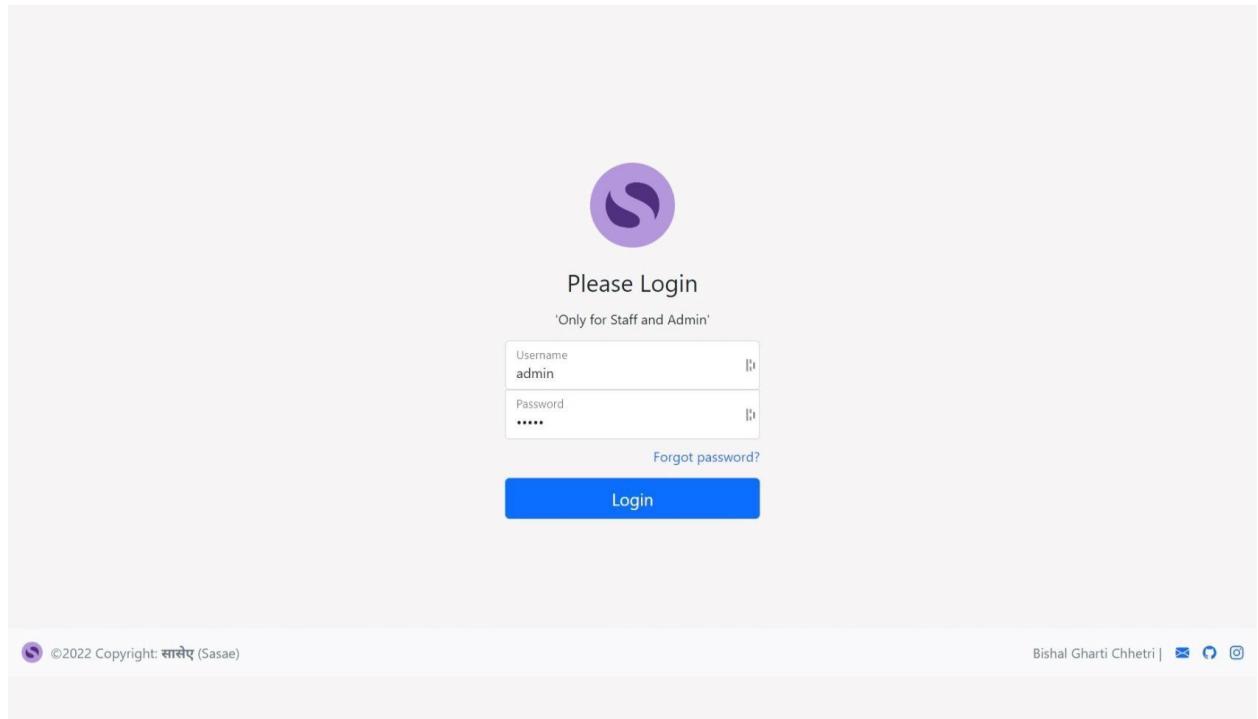


Figure 64: Login Page with admin credentials (Before)

SASAE

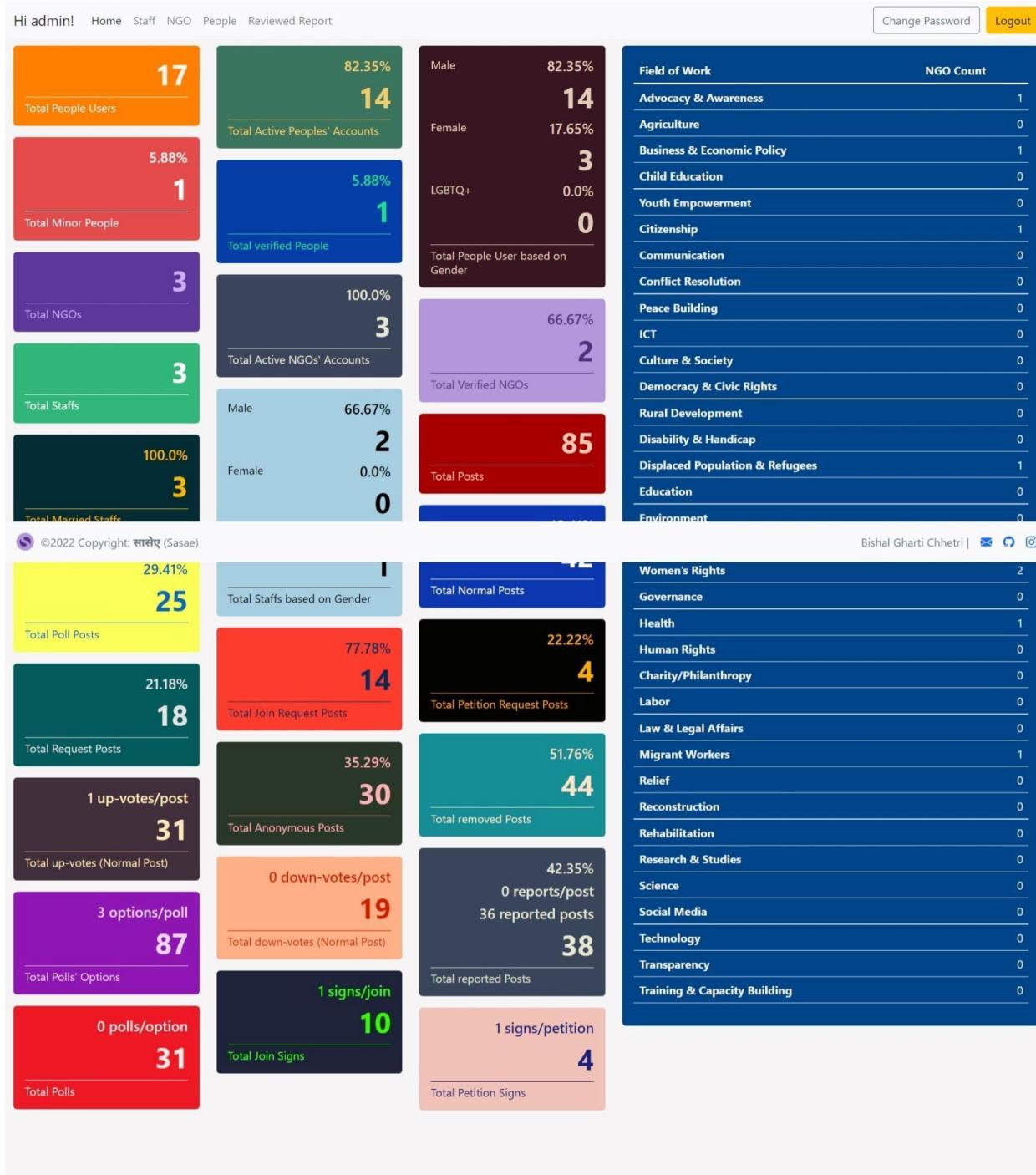


Figure 65: Admin Homepage (After)

Further System Testing of the Admin-Staff Portal is available in Appendix: [7.11.1.1. ADMIN-STAFF PORTAL TESTING].

4.3.2. SASAE MOBILE APP TESTING

4.3.2.1. TEST MANAGE GENERAL PEOPLE

Test Case	TC-1
Objective	To test General People Management (CRUD)
Action	<ul style="list-style-type: none"> - Click on the ‘Register’ button from the Login screen. Fill out the Registration form with related data and click the ‘Register’ button. - Login required for next operations. - Click the ‘Profile’ nav-item to view General People as a personal profile. - Click ‘Update Profile’ to navigate to a Form screen. Then fill in the changes required fields and the ‘Done’ button afterwards. - Click the ‘Person Remove’ icon that is in the top-left corner. A prompt dialogue box will appear. Solve the slider captcha and click confirm to delete the personal General People account.
Expected Result	User must perform all CRUD operations on General People Resource.
Actual Result	The user performs CRUD operations on General People successfully.
Conclusion	Test Successful

Figure 66: System Testing, Sasae Mobile App TC-1

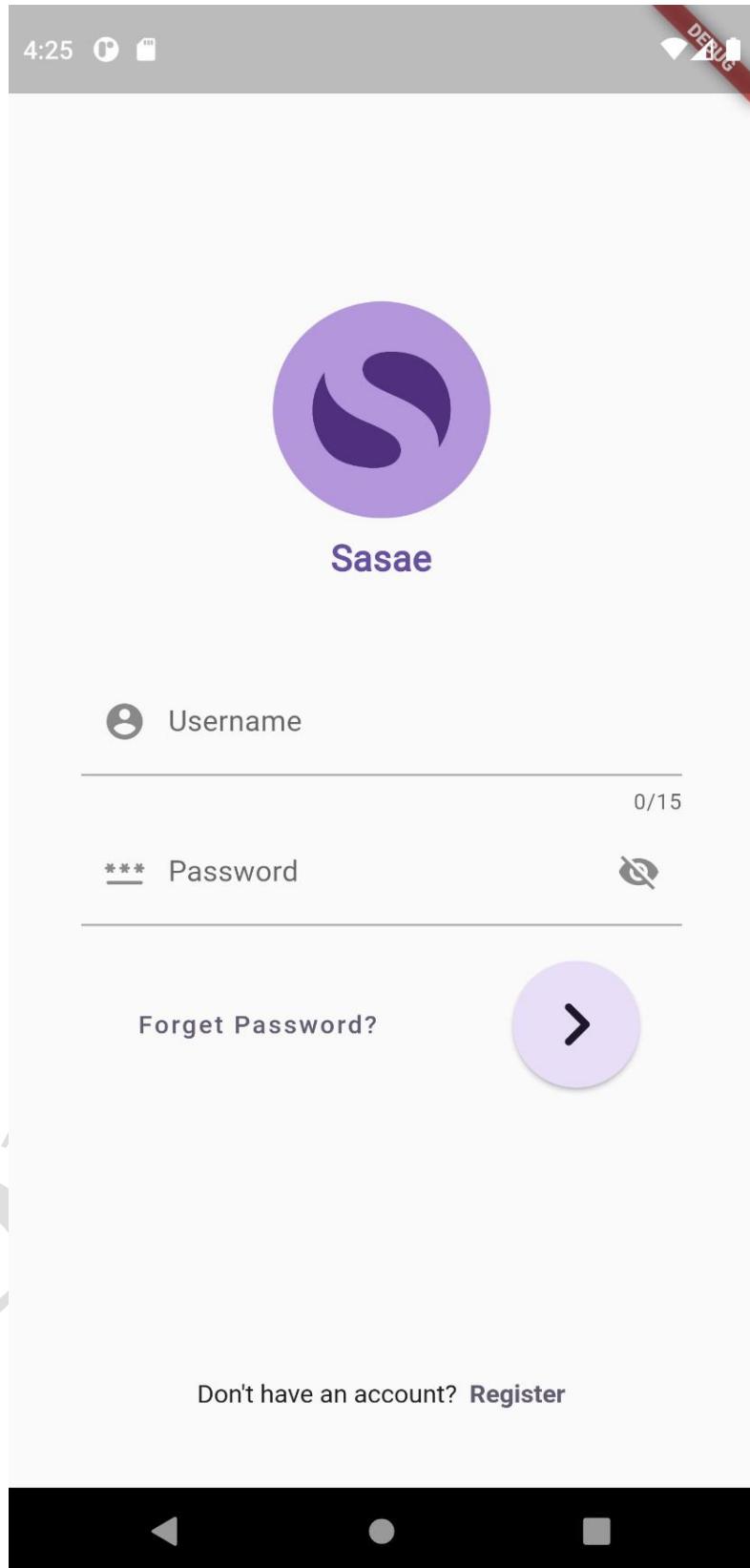


Figure 67: Mobile, Login Screen

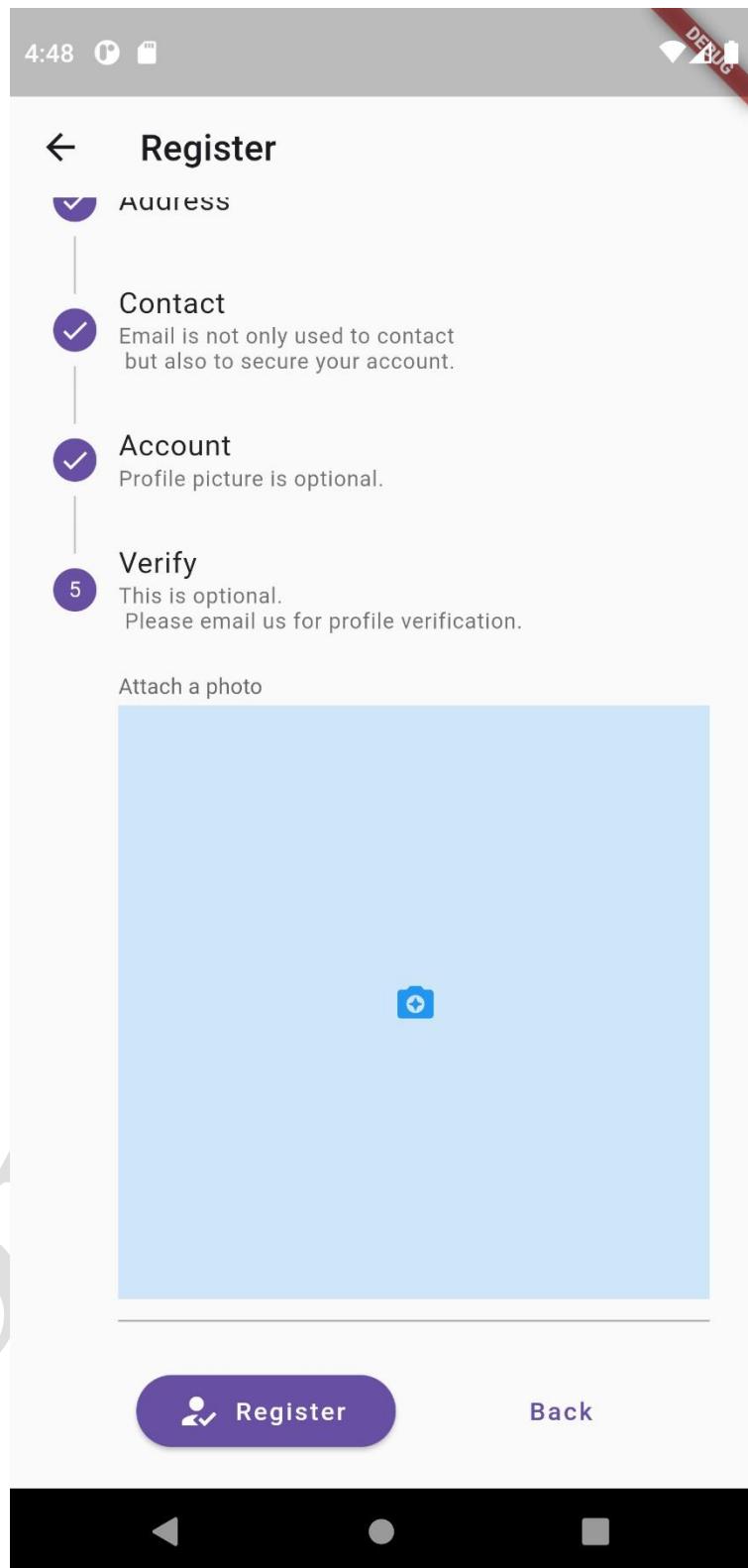


Figure 68: Mobile, Filled Registration Form

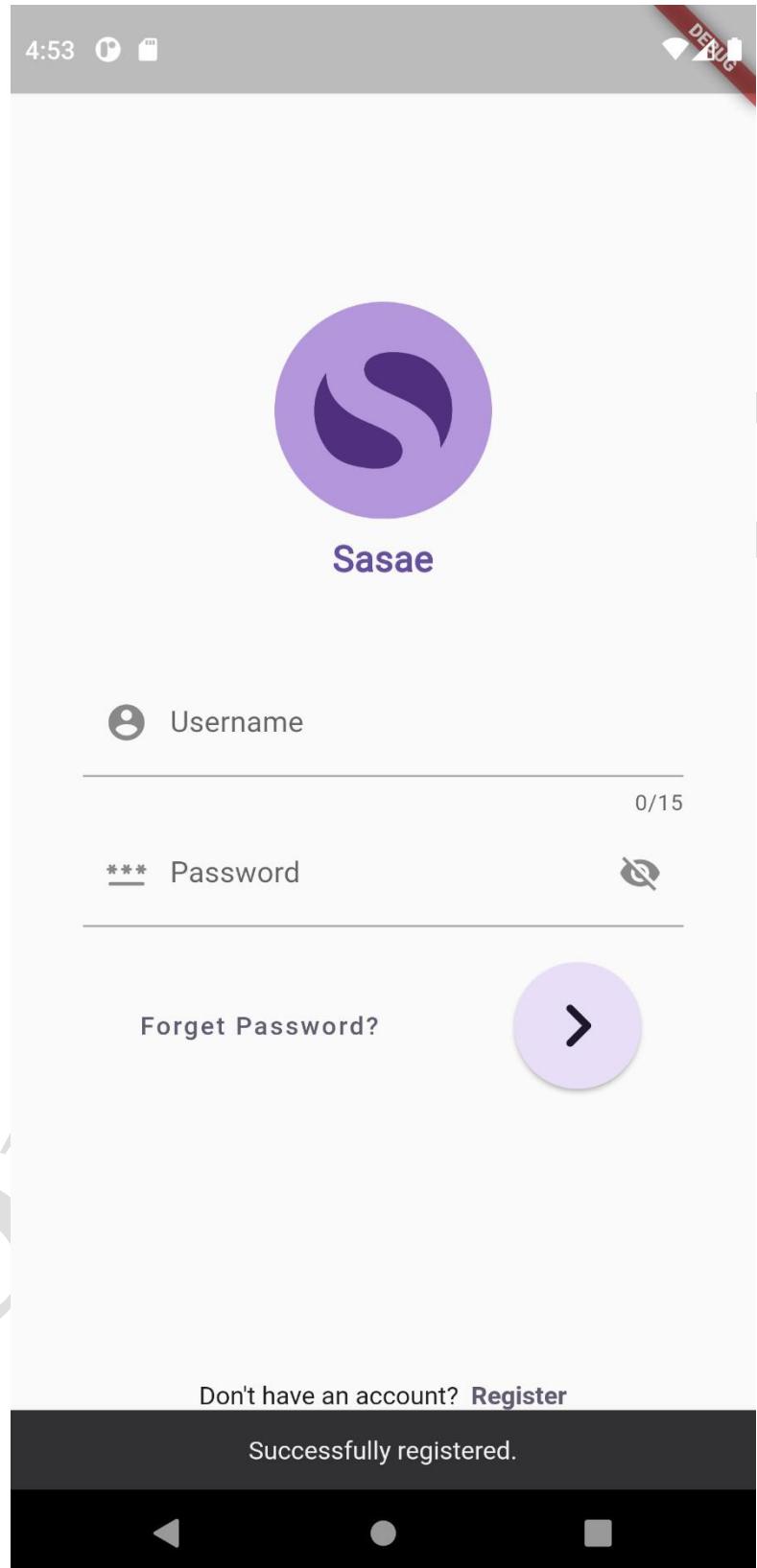


Figure 69: Mobile, Login Screen with a Success Message (After)

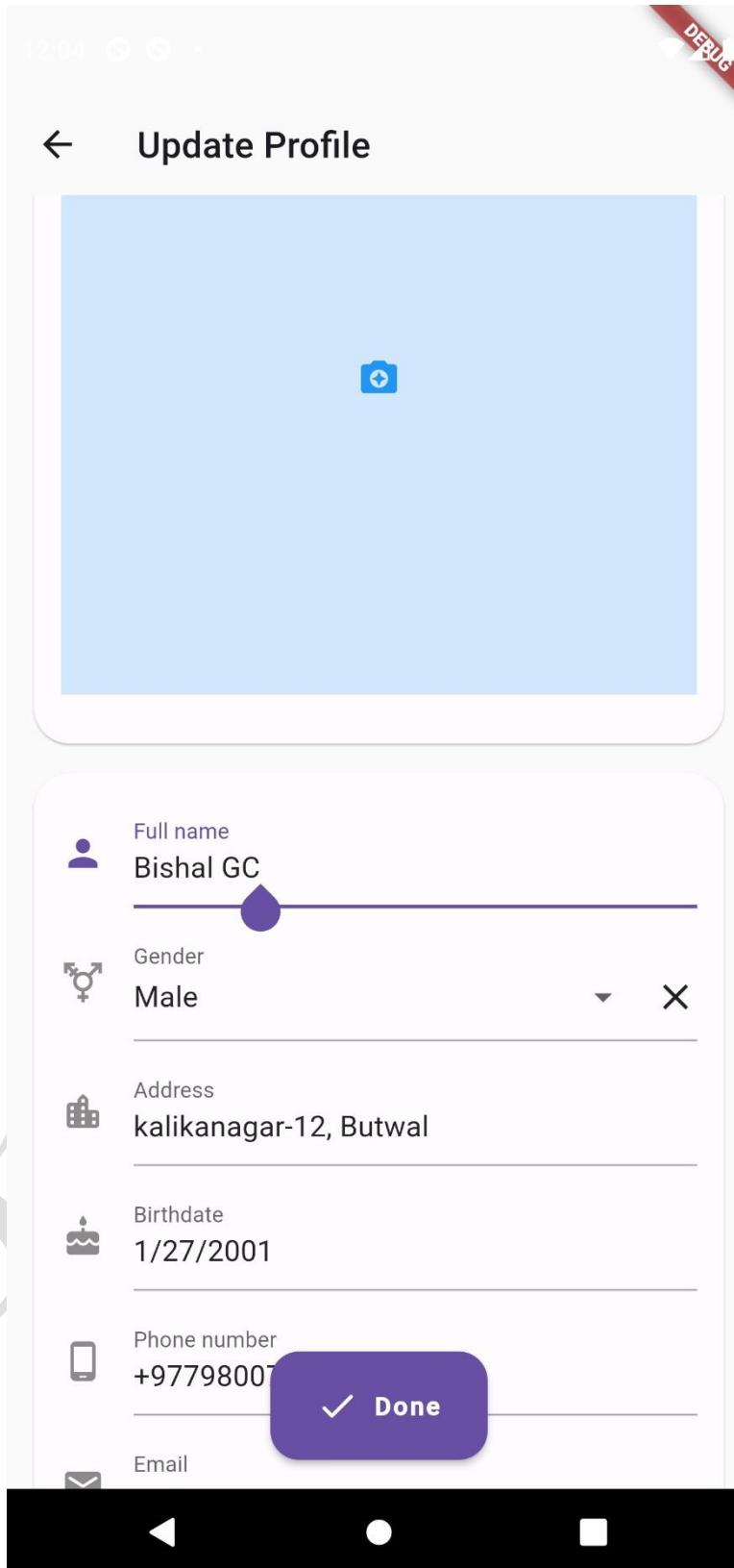


Figure 70: Mobile, Profile Update Form with changed Data (Before)

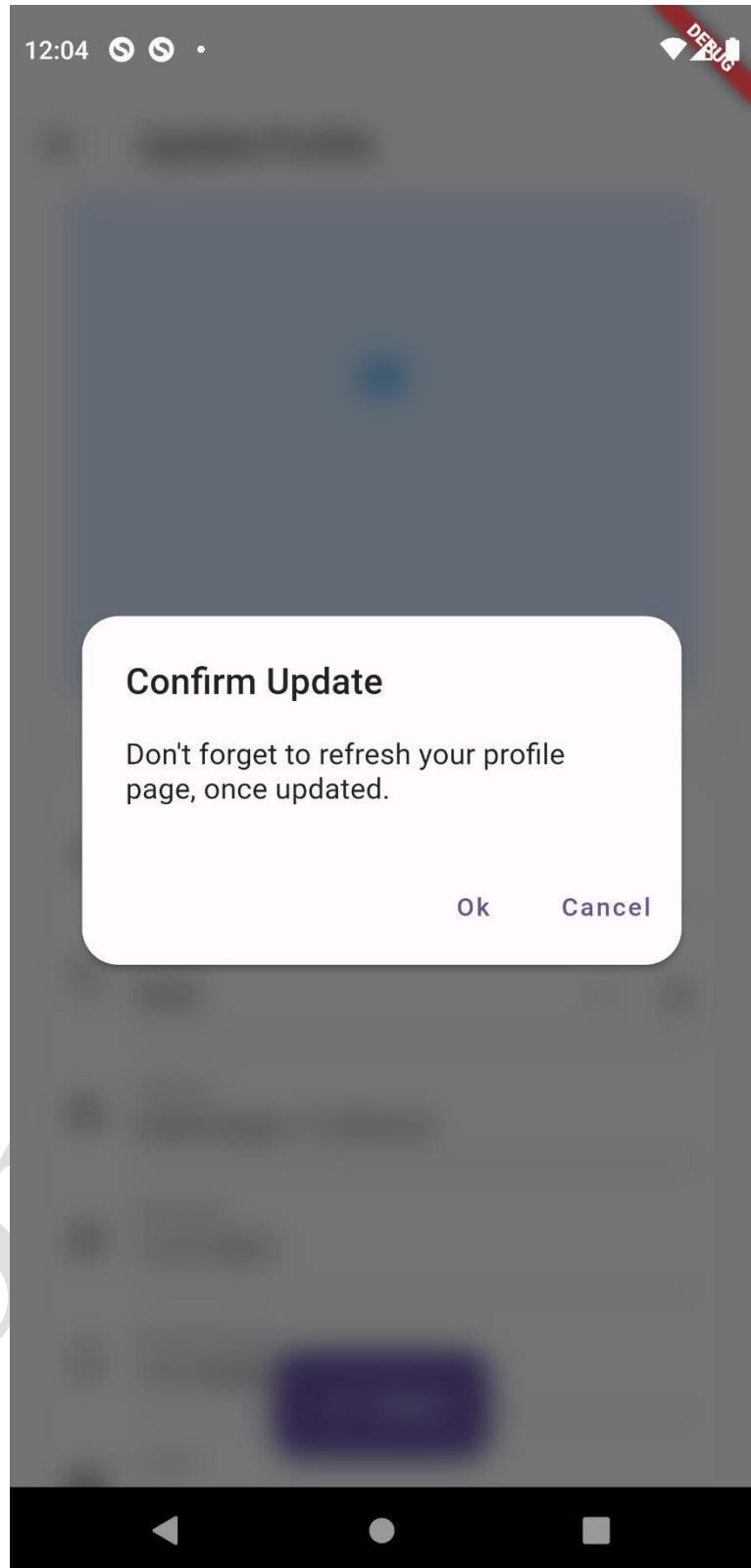


Figure 71: Mobile, Update Profile Prompt

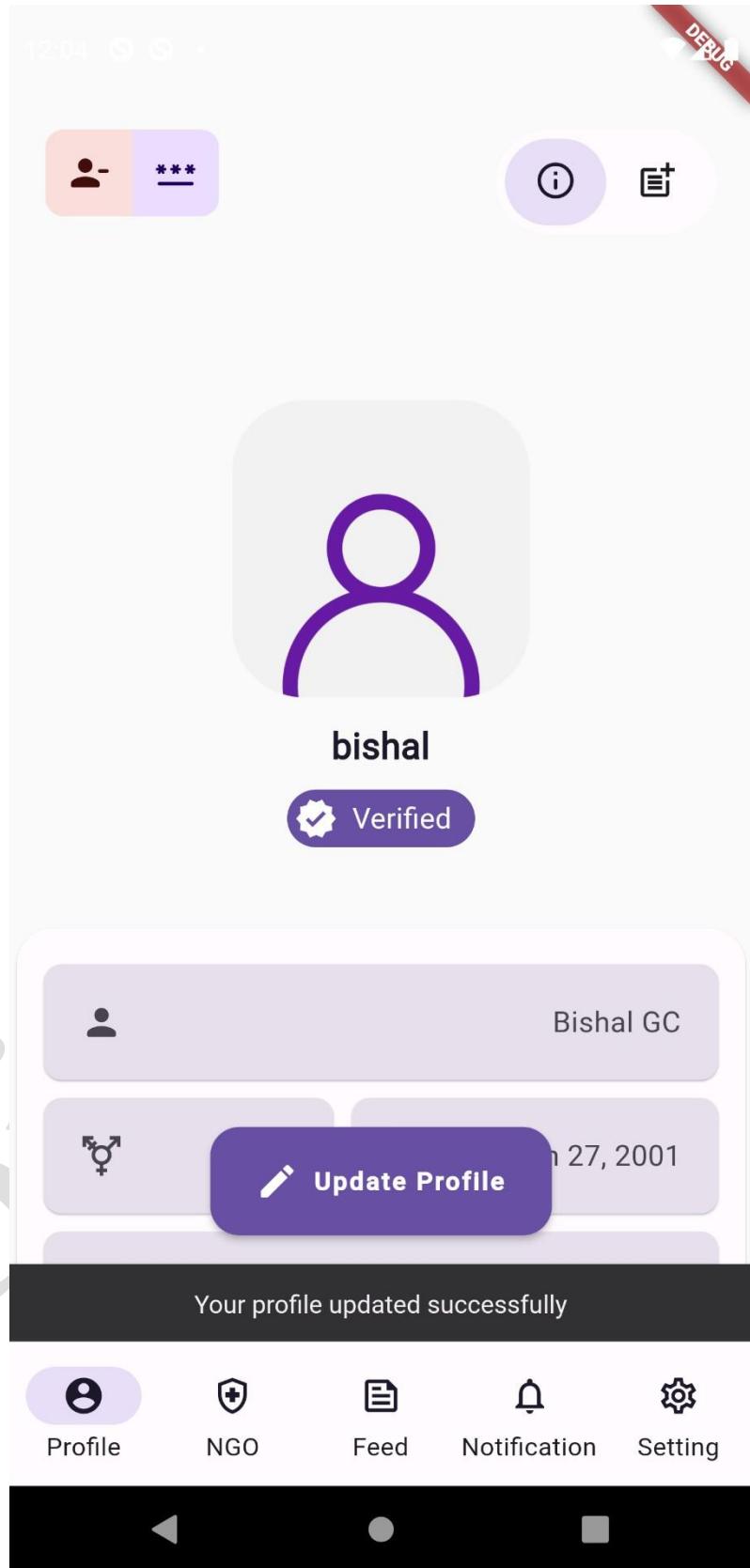


Figure 72: Mobile, General People Profile with a Success Message (After)

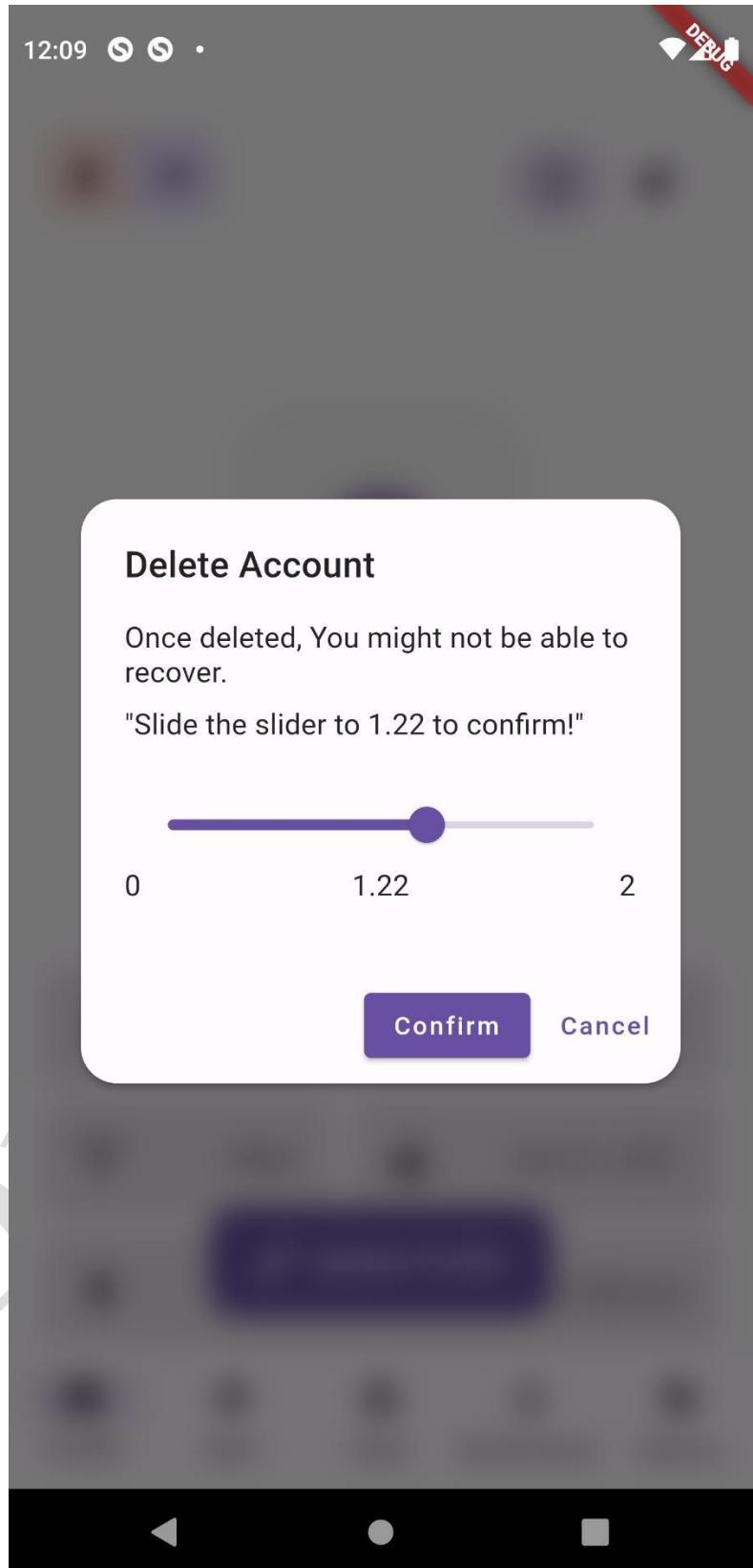


Figure 73: General People Account Delete Prompt (Before)

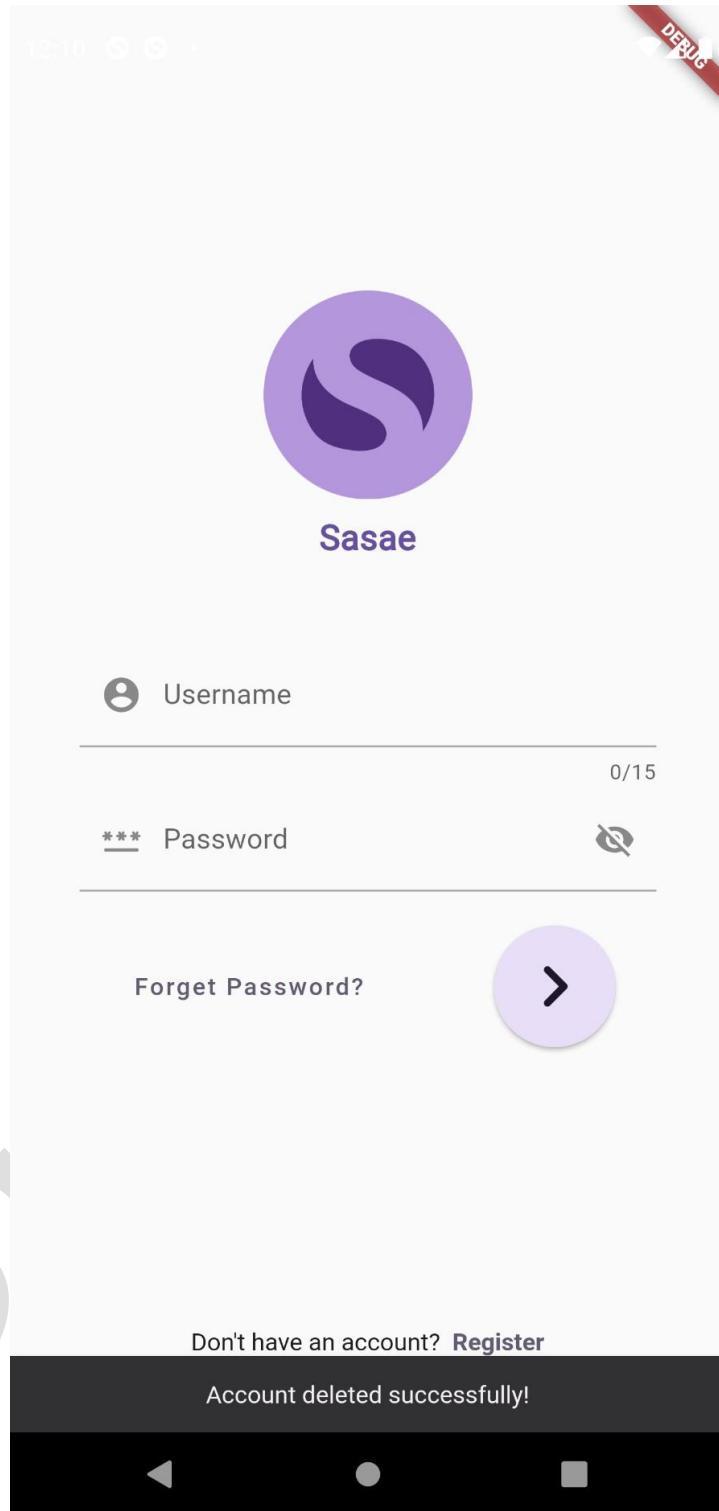


Figure 74: Mobile Login Screen with Success Message (After)

Further System Testing of Sasaе Mobile App is available in Appendix: [[7.11.1.2. SASAE MOBILE APP TESTING](#)].

4.4. USER ACCEPTANCE TESTING

User Acceptance Testing (UAT), often known as beta or end-user testing, is the process of a user or client testing software to see if it can be accepted. This is the last stage of testing after the functional, system, and regression tests have been completed. The major goal of this testing is to ensure that the software meets the company's needs. End-users who are familiar with the business need to perform this validation (SOFTWARETESTINGHELP, 2022). The UAT is carried out in a close group of 10 people. The app was handed to them for beta testing and asked to fill out the User Feedback Form. From the form analysis, the app has positive feedback.

The User Feedbacks are available in Appendix: [\[7.12. APPENDIX L: USER FEEDBACK\]](#).

4.5. CRITICAL ANALYSIS

Testing aims to find any potential issues and or problems that exist in the underdevelopment code and helps to lower the risk that could lead to bigger problems, costing more time and money. Before the actual development of the Admin-Staff portal and Dart Data classes, Test cases and Unit test code were written to apply and learn the concept of Test-Driven Development (TDD). It is found that a test acts as a required condition that needs to be fulfilled by writing code to pass it. TDD leverages simple, small, and bug-free functional code, avoiding duplication.

During the development of RESTful APIs using the Django REST framework, it was hard to develop APIs that could handle the multi-part form data. The existing parser needed to be overridden for properly parsing the multi-part form data containing JSON payload(s) and binary files like images. The positive part of using Django and Django REST Framework is that it reduced the time, complexity, and effort of RESTful APIs development. Even with such a positive part, A more enterprise-level Backend framework like ASP.NET Core or Spring is needed for better scalability and request handling. The RESTful APIs were tested in postman by using collection runner. All the unit tests were then executed in corresponding environments and the tests were all passed. Hence, there were no unit/function errors as per the unit test results.

After unit tests were successful, System testing was carried out to assure the quality of the applications by functionality, performance, installation, security, reliability, load and stability, usability, and scalability. The system testing was performed without prior knowledge of the

application's internal workings. During the system testing of the Admin-Staff Portal, performance problems, poor scalability, and poor usability due to irresponsible design were discovered. One of the solutions is to rebuild the whole application using any frontend JS libraries/frameworks like React, Vue or Angular along with a better CSS framework like Tailwind. This solution is expensive in terms of time and effort. Anyway, the carried system testing passed the functional needs.

System testing was also carried out for mobile applications developed using flutter. A few of the most repeating semantic errors faced were related to viewports, set state, null check, and render box. A proper understanding of those topics was needed to easily solve the errors. Other than that, the UIs were not responsive to the data changes made by the User operations. The provider package was used to segment the UI and the code logic. But it has some limitations. It is hard to call a method of a provider instance from another provider instance. Since the code logic as a function is stored in a group of provider classes, it is hard to make UI changes from an instance method invoked from another Provider instance. Maybe, it is a design requirement made by the flutter team. One of the solutions is to use the BLOC (Business Logic Component) pattern using a Bloc State Management Library. It provides better handling and managing of state than the Providers. The installation was errorless. The application was reliable and scalable like Legos. The authentication made the app secure. The app was usable and app UI was smooth with negligible performance lags and the application met all the functional requirements. Overall, the system testing was successful.

5. CONCLUSION

Briefing the overall project, the development part of FYP was pleasurable, while the report was quite tedious. Without much prior knowledge on frameworks to be used in FYP: Flutter for mobile fronted and Django backend, mass effort on research was required and done for getting core knowledge in lesser time. The journey through the development phase was the most intriguing and full of pits of bugs and errors. Sometimes, even a single minor bug obstructed progress for a complete day. Throughout the development, research was carried out on articles, reports, webpages, videos, books, and samples. It was fruitful to acquire and update ideas, concepts, philosophies, and pieces of knowledge by the end of the development. Testing was quite straight as development was completed without any noticeable bugs, issues and/or problems, but multiple required enhancement and required changes were discovered which is set as future work. The

documentation was quite tedious because of various requirements like drawing multiple UML diagrams, a description of each diagram, etc. set by the supervisors. Anyway, following the Kanban framework, the Social Service Application project was completed successfully.

5.1. LEGAL, SOCIAL AND ETHICAL ISSUES

5.1.1. LEGAL ISSUES

Some of the legal issues that may arise from using the Social Service Application are:

- Privacy violation due to exposing personal data like email, phone, PAN certificate etc.
- Defamation of an individual from post content.
- Intellectual property infringement from using or posting third party's images and trademark etc. without appropriate permission.
- Criminal Activities by using the app for threatening, bullying, stalking, and harassing others online through the post.

5.1.2. SOCIAL ISSUES

Some of the social issues that may arise from using the Social Service Application are:

- Addiction from increased usage of the app.
- Cyberbullying through posts causes psychological problems in victims.
- Hate and Criticism of others based on what they post.
- Racism from the racist post.

5.1.3. ETHICAL ISSUES

Some of the social issues that may arise from using the Social Service Application are:

- Harassment through the post.
- Discriminating post content.
- Abuse and exploitation of features like Anonymous Post and Report Post.
- Corporate Espionage: Staff can misuse the Admin-Staff portal and data from it.

5.2. ADVANTAGES

The perks and advantages of using the deployed Social Service Application are below:

- Promotes charity/philanthropy by allowing users to donate in-app.
- Encourages individuals on contributing to social work through the post.

- Shy and privacy-focused people can post welfare-related content and activities anonymously.
- Empowers ordinary people to be powerful social workers/campaigners.
- Facilitates social programmes and campaigns easier and more effective to create and deliver using posts.
- Assist and Support people or animals in need by creating awareness and attention posts.
- Promotes competitiveness between NGOs for increased and effective social works.

5.3. LIMITATIONS

The Limitations that currently exist in the Social Service Application are below:

- General people can only be registered from the app.
- The in-app NGO filtration is limited to the Field of Work.
- The post recommendation system is unavailable.
- The payment gateway is limited to Khalti.
- Upload and download of an image lack image compression.
- Multiple Image upload in a single Normal Post is not allowed.
- Missing Background Notification handler.

5.4. FUTURE WORK

Along with overcoming the above limitations, the Future works needed to improve and further enhance the Social Service Application Project are below:

- Migrate the Backend Code from Django framework to ASP.NET Core.
- Migrate the database from MySQL to PostgreSQL.
- Rebuild the admin-Staff portal with React JS.
- Add Google or Open Street Map in Join Request Post and NGO address field.
- Addition of a new Charity Request Post type for fundraising functionality.
- Embed Comment feature in Post.
- Addition of Chat-feature.
- Refactor the existing codes, URLs, and RESTful APIs.
- Improve the Application's existing UI/UX.
- Localize the app to support different languages and regions.
- Make the App responsive to any screen size and Adaptive to any platform (i.e., IOS).
- Deploy the mobile app in Apple App Store and Google Play Store.

- Deploy the backend in AWS Lambda.

The readings for future work are available in Appendix: [[7.13.1 READINGS FOR FUTURE WORK](#)].

6. REFERENCES AND BIBLIOGRAPHY

- Change.org, PBC, 2019. *2020_Impact+Report_Change_EN_final.pdf*. [Online] Available at: https://static.change.org/brand-pages/impact/reports/2020/2020_Impact+Report_Change_EN_final.pdf [Accessed 13 12 2021].
- Change.org, PBC, 2021. *Change.org*. [Online] Available at: <https://www.change.org/petitions> [Accessed 13 12 2021].
- Digité, 2022. *What Is Kanban? An Overview Of The Kanban Method*. [Online] Available at: <https://www.digité.com/kanban/what-is-kanban/> [Accessed 25 04 2022].
- Editor, K., 2021. *The SIX Basic Steps of Software Development by KitelyTech*. [Online] Available at: <https://kitelytech.com/six-basic-steps-software-development/#:~:text=Stages%20of%20the%20Software%20Development%20Lifecycle&text=These%20stages%20are%20planning%2C%20analysis,a%20predictable%20and%20manageable%20way.> [Accessed 25 04 2022].
- Education, I. C., 2020. *What is Three-Tier Architecture | IBM*. [Online] Available at: <https://www.ibm.com/cloud/learn/three-tier-architecture> [Accessed 25 04 2022].
- Forbes Media LLC, 2021. *Facebook (FB)*. [Online] Available at: <https://www.forbes.com/companies/facebook/?sh=12056b114193> [Accessed 13 12 2021].
- Hirani, V., 2020. *10 Major Practices to Execute in REST API Development*. [Online]

Available at: <https://www.mindinventory.com/blog/best-practices-rest-api-development/>
[Accessed 25 04 2022].

bisleriunm

Hyytiälä, H., 2011. *The Kanban method* - Reaktor. Available at: <https://www.reaktor.com/blog/the-kanban-method/> [Accessed 24 11 2021].

International Businesses Standards Organization (IBSO), 2015. *Facts and Stats about NGOs Worldwide*. [Online]

Available at: <https://www.standardizations.org/bulletin/?p=841> [Accessed 13 12 2021].

Iris Kobek and Ram Pratap Thapa, 2004. *12.04.doc.* [Online]

Available at: <http://library.fes.de/pdf-files/iez/50226.pdf> [Accessed 13 2021 2021].

Kanbanize, 2022. *Kanban Backlog: The Least Explored Corner of Kanban*. [Online]

Available at: [https://kanbanize.com/blog/kanban-backlog/#:~:text=What%20is%20the%20Purpose%20of,Do%E2%80%9D%20stage%20\(column\).](https://kanbanize.com/blog/kanban-backlog/#:~:text=What%20is%20the%20Purpose%20of,Do%E2%80%9D%20stage%20(column).)

[Accessed 25 04 2022].

Kukhnavets, P., 2018. *Disadvantages and Advantages in Extreme Programming*. [Online]

Available at: <https://hygger.io/blog/disadvantages-and-advantages-of-extreme-programming/> [Accessed 22 4 2022].

Lerche-Jensen, S., 2019. *Scrum Academy :: International Kanban Master Foundation - The 6 main*

Practices of Kanban Method. [Online]

Available at: <https://www.scrum.as/academy.php?show=4&chapter=5> [Accessed 24 11 2021].

Mbaabu, O., 2021. *Front End vs Back End in Web Development | Engineering Education (EngEd)*

Program | Section. [Online]

Available at: <https://www.section.io/engineering-education/front-end-vs-back-end-in-webdevelopment/>

[Accessed 13 12 2021].

Meta, 2021. *About Facebook Pages | Facebook Business Help Centre*. Available at: <https://www.facebook.com/business/help/461775097570076> [Accessed 13 12 2021].

Meta, 2021. *NGO Federation of Nepal | Facebook*. [Online] Available at: <https://www.facebook.com/NFNCentral> [Accessed 13 12 2021].

Mohsin, M., 2021. *10 Social Media Statistics You Need to Know in 2021 [Infographic]*. [Online] Available at: <https://www.oberlo.com/blog/social-media-marketing-statistics> [Accessed 21 4 2022].

NRCS, 2021. *Donate to NRCS | Nepal Red Cross Society*. [Online] Available at: <https://nrcs.org/donate-to-nrcs/> [Accessed 13 12 2021].

NRCS, 2021. *Introduction | Nepal Red Cross Society*. [Online] Available at: <https://nrcs.org/about-nrcs/> [Accessed 13 12 2021].

Olesen, J., 2022. *Violet Color Meaning: The Color Violet Symbolizes Wisdom and Sensitivity - Color Meanings*. [Online] Available at: <https://www.color-meanings.com/violet-color-meaning-the-color-violet/> [Accessed 23 04 2022].

Pastorino, M., 2021. *Frontend vs Backend: What's The Difference?*. [Online] Available at: <https://www.pluralsight.com/blog/software-development/front-end-vs-back-end> [Accessed 13 12 2021].
Red Hat, 2020. *What is a REST API?*. [Online] Available at: <https://www.redhat.com/en/topics/api/what-is-a-rest-api> [Accessed 25 04 2022].

Sherman, R., 2015. *Waterfall Methodology - an overview | ScienceDirect Topics*. [Online] Available at: <https://www.sciencedirect.com/topics/computer-science/waterfall-methodology> [Accessed 24 11 2021].

Siderova, S., 2019. *The Top 10 Benefits of Kanban* | Nave. Available at: <https://getnave.com/blog/kanban-benefits/> [Accessed 25 11 2021].

SOFTWARETESTINGHELP, 2022. *What is User Acceptance Testing (UAT): A Complete Guide.* [Online]

Available at: <https://www.softwaretestinghelp.com/what-is-user-acceptance-testing-uat/> [Accessed 25 04 2022].

Start a petition · Change.org, 2021. *Start a petition · Change.org.* [Online] Available at: <https://www.change.org/start-a-petition> [Accessed 13 12 2021].

Statista, 2021. • *Facebook MAU worldwide 2021* | Statista. [Online] Available at: <https://www.statista.com/statistics/264810/number-of-monthly-active-facebookusers-worldwide/> [Accessed 13 12 2021].

Sussex, U. o., 2021. *Final year project information for student.* [Online] Available at: <https://www.sussex.ac.uk/ei/internal/forstudents/informatics/undergraduate/finalyearprojects/informationforstudents> [Accessed 18 April 2022].

Team, L. C., 2021. *What Is the Extreme Programming Methodology? | Lucidchart Blog.* [Online] Available at: <https://www.lucidchart.com/blog/what-is-extreme-programming> [Accessed 24 11 2021].

Team, L. C., 2022. *The Pros and Cons of Waterfall Methodology | Lucidchart Blog.* [Online] Available at: <https://www.lucidchart.com/blog/pros-and-cons-of-waterfall-methodology> [Accessed 21 04 2022].

Team, M. S., 2013. *Waterfall Model Meaning & Definition | MBA Skool.* [Online] Available at: <https://www.mba-skool.com/business-concepts/it-and-systems/8658-waterfall->

model.html

[Accessed 24 11 2021].

Visual Paradigm, 2022. *Requirement Analysis Techniques*. [Online] Available at: <https://www.visual-paradigm.com/guide/requirements-gathering/requirementanalysistechniques/#:~:text=Requirement%20Analysis%2C%20also%20know,n%20as,requirements%20 gathering%20or%20requirements%20capturing>.

[Accessed 25 04 2022].

Wells, D., 2013. *Extreme Programming: A Gentle Introduction*. [Online] Available at: <http://www.extremeprogramming.org/> [Accessed 24 11 2021].

7. APPENDIX

7.1. APPENDIX A: PRE-SURVEY

7.1.1. PRE-SURVEY FORM

The screenshot shows a Google Forms survey titled "Social service Application Form [Pre-Survey Form]". The form is designed to collect information from social-work enthusiasts. It includes fields for Name, Email, profession, and responses to questions about previous experience with Social Services, NGOs, and their perception of NGOs' responsiveness. The form is currently in edit mode, as indicated by the "Send" button and other editing tools visible at the top.

Questions **Responses** (31) **Settings**

Social service Application Form [Pre-Survey Form]

A centralized Networking platform for social-work enthusiasts

This form is automatically collecting email addresses for Islington College users. [Change settings](#)

Enter your Name *

Short-answer text

Enter your Email *

Short-answer text

What line of work are you in? (profession)

Short-answer text

Have you heard about Social Service before? *

Yes

No

Have you ever involved in any social works and activities? *

Yes

No

Maybe

Have you ever involved in any NGOs? *

Yes

No

Maybe

What do you think about NGO(s)? Are they truly active and responsive towards their duties and responsibilities per their field(s) of work? Rate it.

1 2 3 4 5 6 7

Figure 75: Pre-Survey Form

SASAE

Have you ever came across any NGOs in any social media platform? *

Yes
 No
 Maybe

If Yes, NGOs in what social media platform?

Facebook
 Twitter
 Instagram
 YouTube
 Reddit
 LinkedIn
 Other...

Are NGOs approachable and responsive to people's message(s) and request(s)? *

Yes
 No
 Maybe

What do you think about a platform specifically made for social work with easy NGO's accessibility? Rate its usability? *

1 2 3 4 5 6 7

Are there any similar app, you came across to? *

Yes
 No
 Maybe

If yes, Name them.

Short-answer text

Figure 76: Pre-Survey Form Contd.

SASAE

7.1.2. SAMPLE OF FILLED PRE-SURVEY FORMS

The screenshot shows a Google Forms survey titled "FYP Pre-Survey Form". The interface indicates 31 responses have been submitted. The survey consists of several questions:

- Question 1:** "Enter your Name *". Response: Rupa Shrestha.
- Question 2:** "Enter your Email *". Response: np01cp4a190277@islingtoncollege.edu.np
- Question 3:** "What line of work are you in? (profession)". Response: [Blank]
- Question 4:** "Have you heard about Social Service before? *". Response: Yes (selected)
- Question 5:** "Have you ever involved in any social works and activities ? *". Response: Maybe (selected)
- Question 6:** "Have you ever involved in any NGOs? *". Response: No (selected)

Figure 77: Filled Pre-Survey Sample

SASAE

What do you think about NGO(s)? Are they truly active and responsive towards their duties and responsibilities per their field(s) of work? Rate it. *

1 2 3 4 5 6 7

Have you ever came across any NGOs in any social media platform? *

Yes
 No
 Maybe

If Yes, NGOs in what social media platform?

Facebook
 Twitter
 Instagram
 YouTube
 Reddit
 LinkedIn
 Other: _____

Are NGOs approachable and responsive to people's message(s) and request(s)? *

Yes
 No
 Maybe

What do you think about a platform specifically made for social work with easy NGO's accessibility? Rate its usability? *

1 2 3 4 5 6 7

Are there any similar app, you came across to? *

Yes
 No
 Maybe

If yes, Name them.

Submitted 10/12/2021, 20:33

Figure 78: Filled Pre-Survey Sample Contd.

7.1.3. PRE-SURVEY RESULT

What line of work are you in? (profession)

27 responses

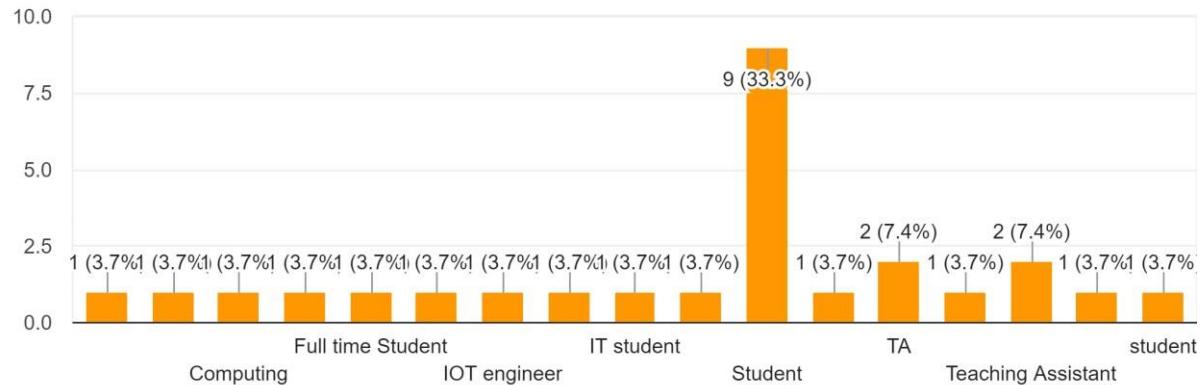


Figure 79: Pre-Survey Question-1 Result

Have you heard about Social Service before?

31 responses

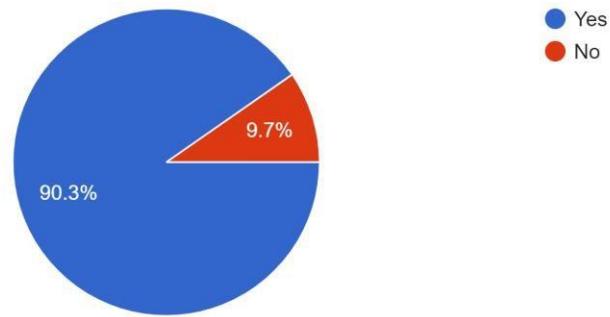


Figure 80: Pre-Survey Question-2 Result

Have you ever involved in any social works and activities ?

31 responses

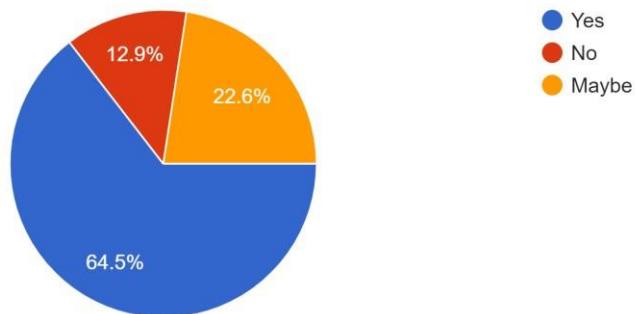


Figure 81: Pre-Survey Question-3 Result

Have you ever involved in any NGOs?

31 responses

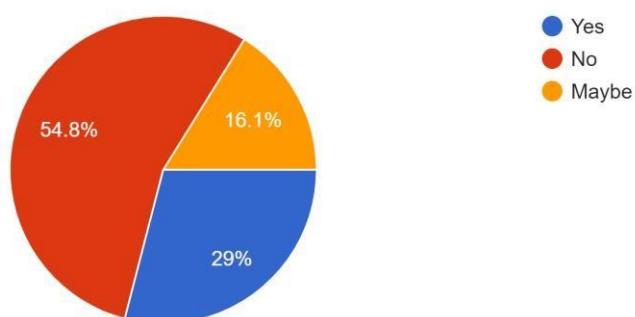


Figure 82: Pre-Survey Question-4 Result

SASAE

What do you think about NGO(s)? Are they truly active and responsive towards their duties and responsibilities per their field(s) of work? Rate it.

31 responses

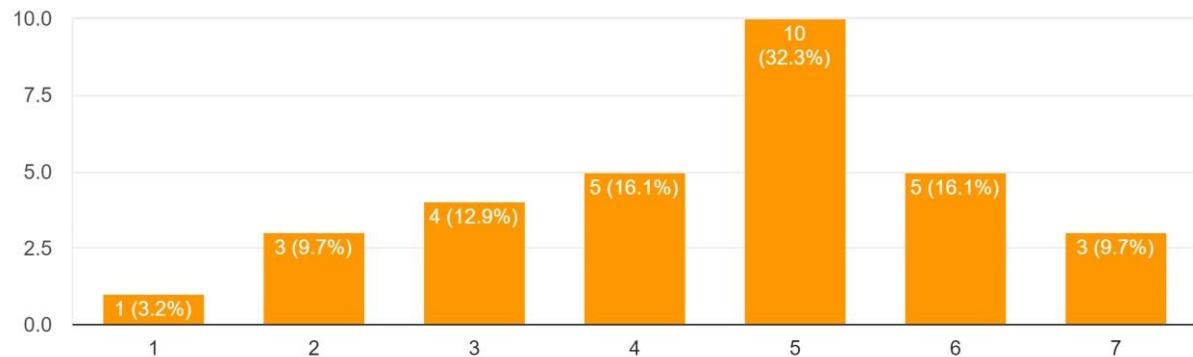


Figure 83: Pre-Survey Question-5 Result

Have you ever came across any NGOs in any social media platform?

31 responses

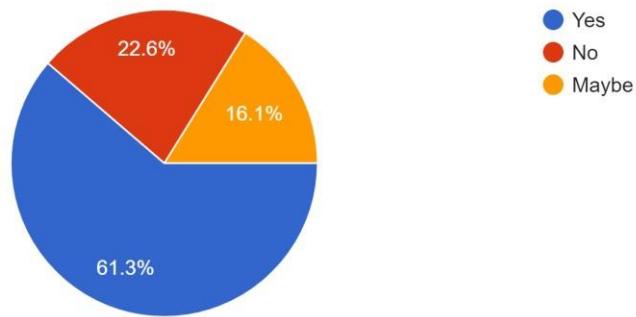


Figure 84: Pre-Survey Question-6 Result

If Yes, NGOs in what social media platform?

23 responses

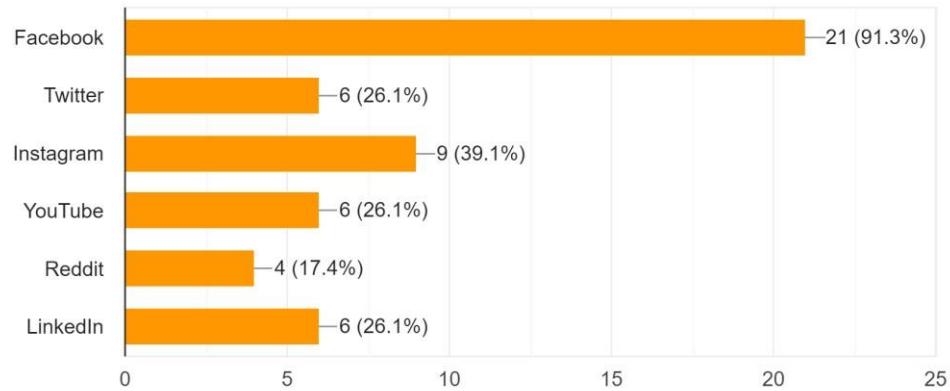


Figure 85: Pre-Survey Question-7 Result

Are NGOs approachable and responsive to people's message(s) and request(s)?

31 responses

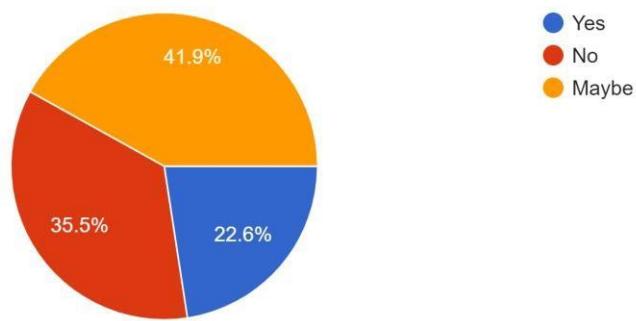


Figure 86: Pre-Survey Question-8 Result

SASAE

What do you think about a platform specifically made for social work with easy NGO's accessibility? Rate its usability?

31 responses

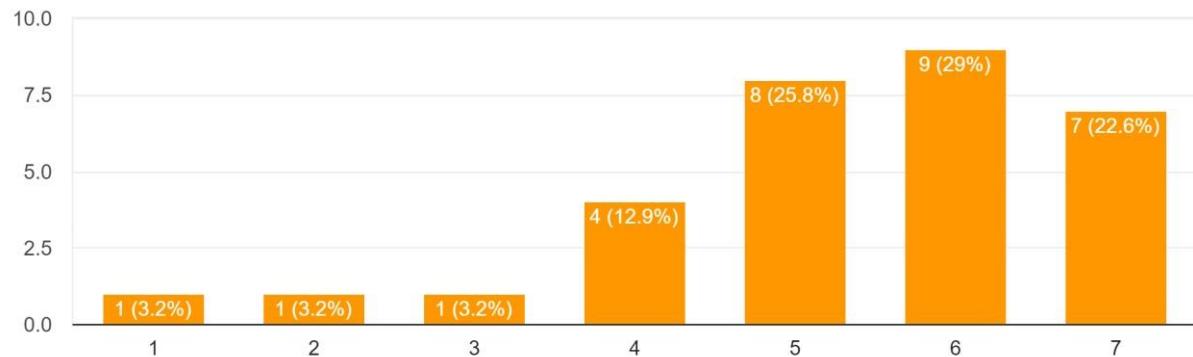


Figure 87: Pre-Survey Question-9 Result

Are there any similar app, you came across to?

31 responses

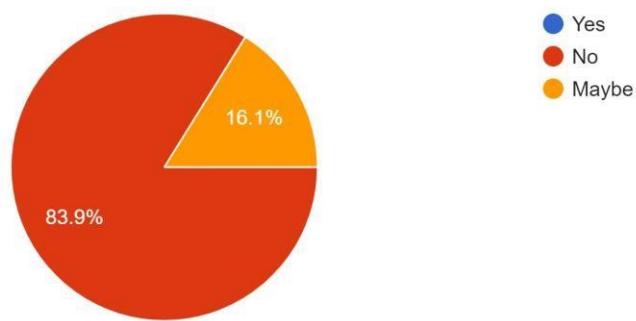


Figure 88: Pre-Survey Question-10 Result

7.2. APPENDIX B: POST-SURVEY

7.2.1. POST-SURVEY FORM

Social Service Application [Post-Survey Form]

A social media for social-welfare enthusiasts

This form is automatically collecting email addresses for Islington College users. [Change settings](#)

Image title

Enter your Email *

Short-answer text

What is your profession? *

- IT Student
- Other...

Will the project impact social-work/welfare related activities? *

- Positively Yes
- Negatively Yes
- No
- Maybe

How would you like to use this project in future? *

- Donate to NGO(s)
- Get NGO information
- Post Awareness Content
- Post Petitions
- Post Polls
- Post to Aware NGOs
- Surf through Posts and React to them

Figure 89: Post-Survey Form

SASAE

Select the features, you think the best in this project? *

Post Anonymously
 Poke NGO(s)
 In-App Notification(s)
 Filter NGOs by Field of Work
 Donation to NGO(s)
 Normal Post
 Poll Post
 Petition Post
 Report Post
 Click to Show Post Author

Where do you think the greatest benefit would be? *

Public
 NGO
 Other...

Please rate the overall project features? *

1 2 3 4 5
Pool ○ ○ ○ ○ ○ Excellent

How satisfied are you with the project? *

Very Satisfied
 Satisfied
 Neutral
 Dissatisfied
 Very Dissatisfied

How likely are you to recommend this project to others? *

1 2 3 4 5
Not at All ○ ○ ○ ○ ○ Extremely

Figure 90: Post-Survey Form Contd.

7.2.2. SAMPLE OF FILLED POST-SURVEY FORMS

The screenshot shows a Google Form titled "Social Service Application [Post-Survey Form]". The form has the following sections:

- Header:** Summary, Question, Individual. A sidebar indicates 3 responses and that accepting responses is turned on.
- Section 1:** "Responses cannot be edited". It includes a header "Social Service Application [Post-Survey Form]", a sub-header "A social media for social-welfare enthusiasts", and a note that the respondent's email (np01cp4a190018@islingtoncollege.edu.np) was recorded on submission. It also includes a required field note.
- Section 2:** Placeholder for "Enter your Email *". The value np01cp4a190018@islingtoncollege.edu.np is shown.
- Section 3:** Placeholder for "What is your profession? *". The value "IT Student" is selected from a radio button group. There is also a field for "Other:".
- Section 4:** Placeholder for "Will the project impact social-work/welfare related activities? *". The value "Positively Yes" is selected from a radio button group. Other options include "Negatively Yes", "No", and "Maybe".
- Section 5:** Placeholder for "How would you like to use this project in future? *". The following checkboxes are checked:
 - ✓ Donate to NGO(s)
 - ✓ Get NGO information
 - ✓ Post Awareness Content
 - ✓ Post Petitions
 - Post Polls
 - ✓ Post to Aware NGOs
 - ✓ Surf through Posts and React to them

Figure 91: Filled Post-Survey Form

SASAE

Select the features, you think the best in this project? *

Post Anonymously
 Poke NGO(s)
 In-App Notification(s)
 Filter NGOs by Field of Work
 Donation to NGO(s)
 Normal Post
 Poll Post
 Petition Post
 Report Post
 Click to Show Post Author

Where do you think the greatest benefit would be? *

Public
 NGO
 Other: _____

Please rate the overall project features? *

1 2 3 4 5
Pool Excellent

How satisfied are you with the project? *

Very Satisfied
 Satisfied
 Neutral
 Dissatisfied
 Very Dissatisfied

How likely are you to recommend this project to others? *

1 2 3 4 5
Not at All Extremely

Submitted 22/04/2022, 21:21

Figure 92: Filled Post-Survey Form Contd.

7.2.3. POST-SURVEY RESULT

What is your profession?

10 responses

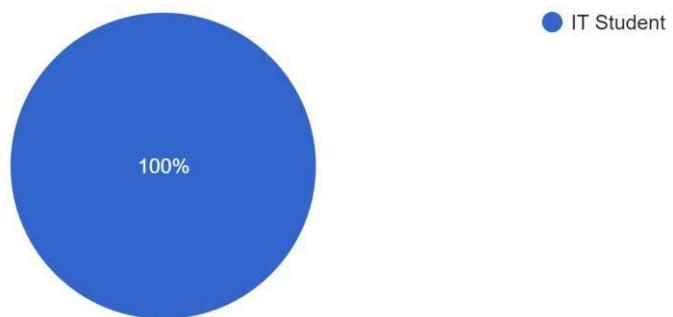


Figure 93: Post-Survey, Question-1 Result

Will the project impact social-work/welfare related activities?

10 responses



Figure 94: Post-Survey, Question-2 Result

How would you like to use this project in future?

10 responses

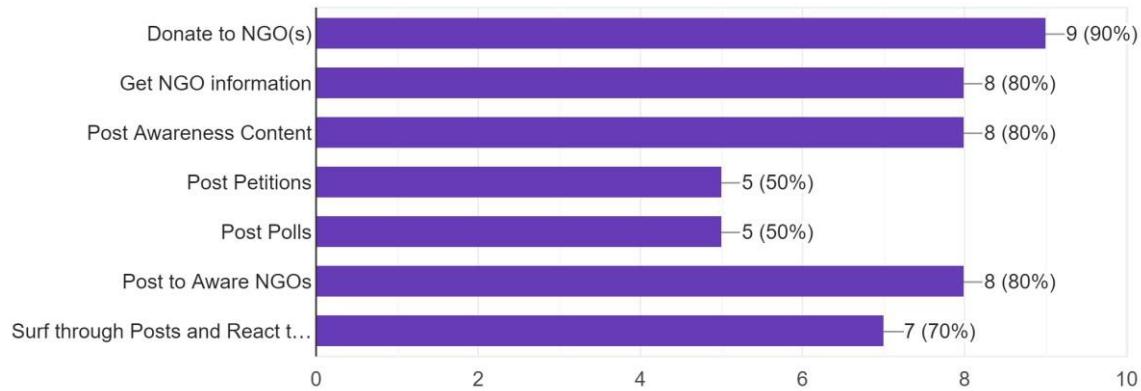


Figure 95: Post-Survey, Question-3 Result

Select the features, you think the best in this project?

10 responses

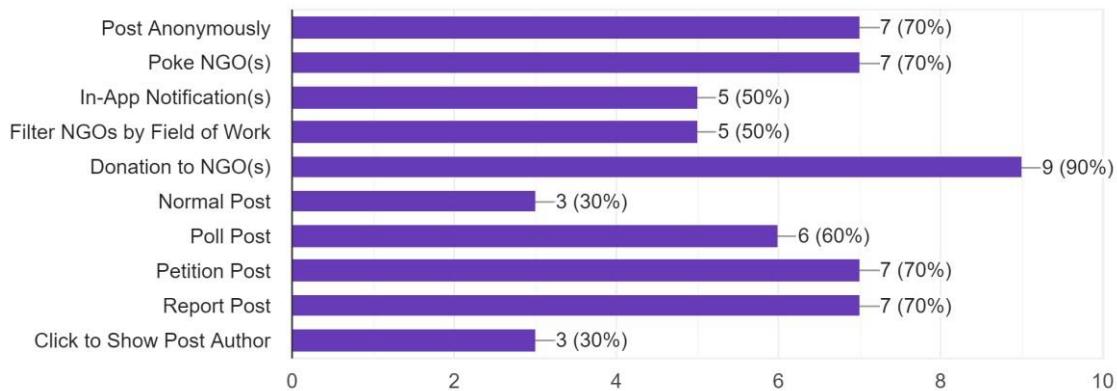


Figure 96: Post-Survey, Question-4 Result

SASAE

Where do you think the greatest benefit would be?

10 responses

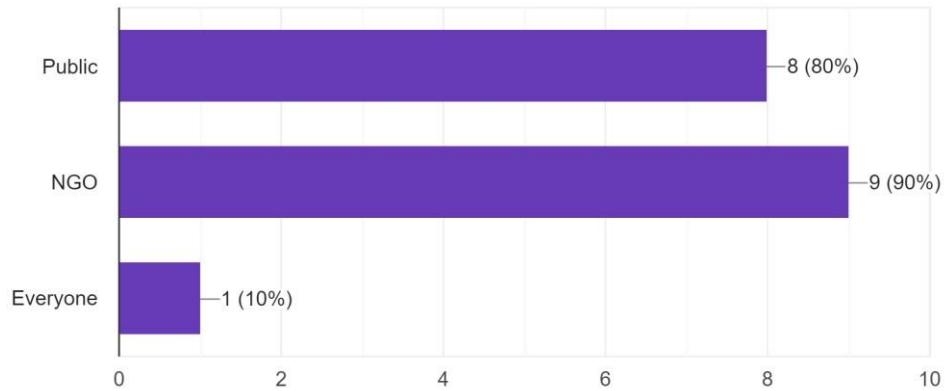


Figure 97: Post-Survey, Question-5 Result

Please rate the overall project features?

10 responses

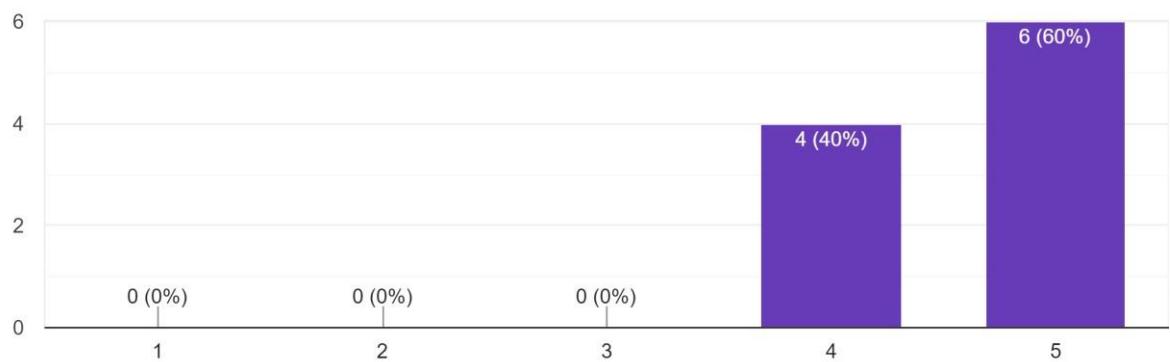


Figure 98: Post-Survey, Question-6 Result

How satisfied are you with the project?

10 responses

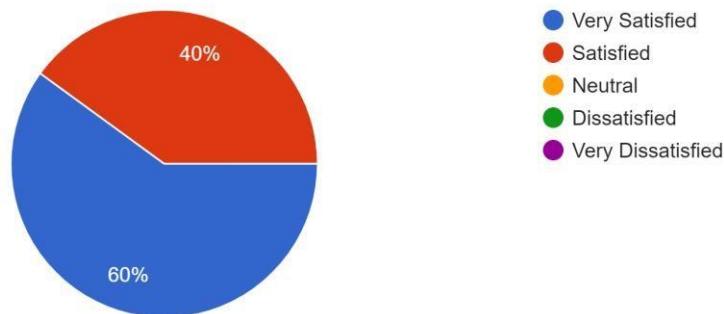


Figure 99: Post-Survey, Question-7 Result

How likely are you to recommend this project to others?

10 responses

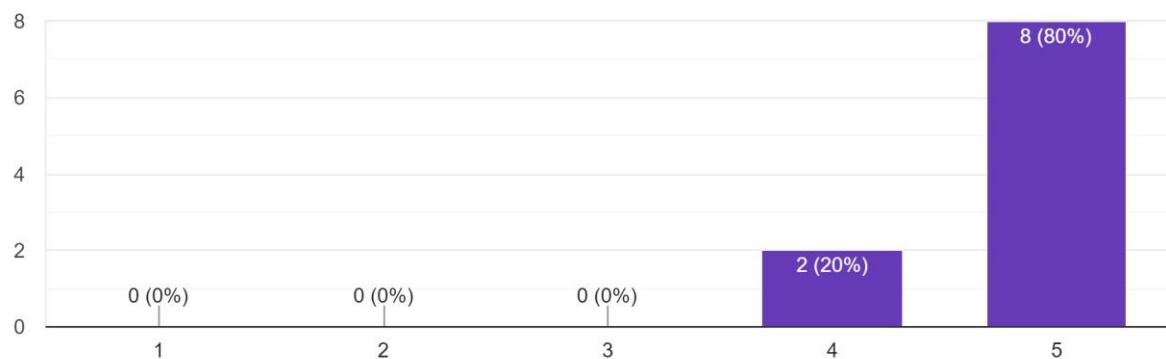


Figure 100: Post-Survey, Question-8 Result

7.3. APPENDIX C: REQUIREMENT ANALYSIS

7.3.1. SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

1. Introduction

The purpose, scope, abbreviations, acronyms, definitions, and overview of the Software Requirements (SRS) are all covered in the SRS introduction. The goal of this SRS document

is to collect, evaluate, and provide an in-depth understanding of the Social Service Application system. This document contains the Social Service Application's complete requirements.

1.1. Purpose

The purpose of this SRS document is to provide a comprehensive overview of the project outcome solution, including its specifications and objectives. The target audience for the project, as well as the user interface. Hardware and software requirements are described in this paper. It defines the end-users perceptions of the solution and its capabilities. It does, however, aid any designer or/and developer in the Software Development Lifecycle (SDLC) processes.

1.2. Scope

The scope mostly relates to the social media features required to bring the Social Service Application concept to life. It focuses on the applications that mostly enable NGO info retrieval, Donations and Post of awareness, attention, events, programmes, campaigns, polls, petitions etc. The SRS is used to establish requirements for software to be produced.

1.3. Definitions, Acronyms, and Abbreviations

CRUD	Create, Read, Update and/or delete a resource from the database
Khalti	A digital wallet for online payments in Nepal
Auth	Authentication by a server to exactly identify the individual who is accessing their information.
JS	JavaScript Programming Language for dynamic page Development
CSS	Cascade Style Sheet to beautify the page
HTML	HyperText Markup Language to structure page

1.4. Overview

The other parts of this document give a basic overview of the system's users, the system's hardware, and the system's functional and data needs. The project's general description is addressed in section 2 of this paper. The data requirements, functional requirements, assumptions, and limitations made when building the Social Service Application are detailed in Section 3. It also contains the product's particular needs. Section 3 also goes over the external interface requirements as well as the functional requirements in depth.

2. Overall description

The document summarizes and briefly describes the primary characteristics of the proposed system, as well as a brief overview of each component. It contains Social Service Application's product functions, as well as user-defined constraints, assumptions, dependencies, as well as requirements subsets.

3. Specific Requirements

The specific requirements are below:

3.1. Functionality

The requirements for the Social Service Application are listed in this section. The features are used to organize the requirements.

3.1.1. Manage General People

The system shall allow a user to read, update and delete a general people account

3.1.2. Manage NGO

The system shall allow a user to perform CRUD operations on NGO

3.1.3. Manage Staff

The system shall allow a user to perform CRUD operations on Staff

3.1.4. View Home/Dashboard

The system shall allow a user to view different homes/dashboards based on user type.

3.1.5. Perform Auth

The system shall allow a user to log in, log out, change the password, and reset the password.

3.1.6. Review Reported Post

The system shall allow a user to review the reported post by acting on it with proper reason.

3.1.7. Read Reviewed Reported Post

The system shall allow a user to read the reviewed reported post.

3.1.8. Manage Post

The system shall allow a user to perform CRUD operation on Post regardless of post type (i.e., Normal, Poll, Request)

3.1.9. View Post(s)

The system shall allow a user to view a list of posts and/or an individual post from the list.

3.1.10. View NGO(s)

The system shall allow a user to view a list of NGOs and/or an individual NGO from the list.

3.1.11. Register General People

The system shall allow a user to register as General People.

3.1.12. Donate to NGO

The system shall allow users to donate to NGOs using the Khalti payment Gateway.

3.1.13. View User's Posts

The system shall allow a user to view all his/her posted posts.

3.1.14. React Normal Post

The system shall allow a user to either upvote or downvote the Normal Post.

3.1.15. React Poll Post

The system shall allow a user to poll an option in Poll Post.

3.1.16. React Request Post

Depending on request type (i.e., Petition, Join), the system shall allow a user to engage in Request Post.

3.1.17. Report Post

The system shall allow the user to report any kind of Post.

3.1.18. Send Notification

The system shall send a notification to the user(s) to notify either of the following actions: react-post, remove-post, poke-NGO, account-verification, etc.

3.2. Reliability and Availability

3.2.1. Back-end Database

- The system shall provide MySQL database storage.
- The system shall provide the ability to clone, backup and recover databases using MySQL workbench.

3.2.2. Serverless backend

- The system shall have its backend process in a cloud computing service like AWS Lambda or Heroku for handling the traffic of the request.

3.3. Performance

The application shall be web-based, needing a webserver to run. The application will take time to load per request, depending on strength of the internet connection. The performance will be determined by the end user's hardware components.

3.4. Security

3.4.1. Data Transfer

- The system shall use an SSL certificate and CSRF token to encrypt server traffic and information between end-users and the server.
- The system shall need an authorization token in the request header for performing almost all the functions specified in the Functional requirement.

3.4.2. Data Storage

- A user's password must never be displayed in their web browser or mobile. The password is always echoed with dots or asterisk characters.
- Only authenticated administrators should have access to the system's back-end servers.
- The system's backend databases must be encrypted.
- The password is stored in hashes. Hence, a password can only be reset upon forgetting the password.

3.5. Supportability

- GitHub will be used to maintain, and version controls the source code and overall system repositories.

3.6. Design Constraints

3.6.1. Standard Development Tools

The system's backend is developed using the Django framework. The web frontend is developed using HTML, CSS, JS, and Bootstrap. The mobile-fronted is developed using the Flutter framework, following the Material You design principle.

3.6.2. Web-based Application

- A Web browser must be installed on the computer.
- The URL to access the application must be easy.
- The loading time for any request query should be no more than three minutes.
- The end-user must have basic computer skills.

3.6.3. Mobile-based Application

- The mobile must be running above or equal to android version 8.
- The connection time to the remote backend server must be less than or equal to 5 seconds.
- An active connection is required.

3.7. Purchased Components

Not Applicable

3.8. Interfaces

The Social Service Application system supports a variety of interfaces, including User Interface, Software Interface, and Hardware Interface.

3.8.1. User Interfaces

- The application's user interface must be interoperable with any browser and mobile phone that allow user accessibility.
- Any tool or software framework/libraries like HTML, CSS, JS, Flutter Dart, Bootstrap etc. can be used to implement the user interfaces.

3.8.2. Hardware Interfaces

Because the applications must run over the internet, all the hardware required to connect to the internet (i.e., Access points, routers, WAN-LANs, and ethernet cables) will serve as the system's hardware interface.

3.8.3. Software Interfaces

- The application shall communicate with the Khalti payment gateway to proceed with payment ad donation.
- The application shall communicate with the backend to perform any functional requirements.
- The application shall communicate with the database using an ODBC type connection to perform CRUD.

3.8.4. Communication Interfaces

- The HTTP protocol and RESTful API will be used by the Social Service Application system for communication over the internet to transfer data back and forth.

4. Supporting Information

Please refer to the followings:

- Structural Model
 - Class Diagram
- Behavioural Model
 - Use Case Diagram
 - Sequence Diagram
 - Communication/Collaboration Diagram

7.4. APPENDIX D: BACKGROUND

7.4.1. UNDERSTANDING THE SOLUTION

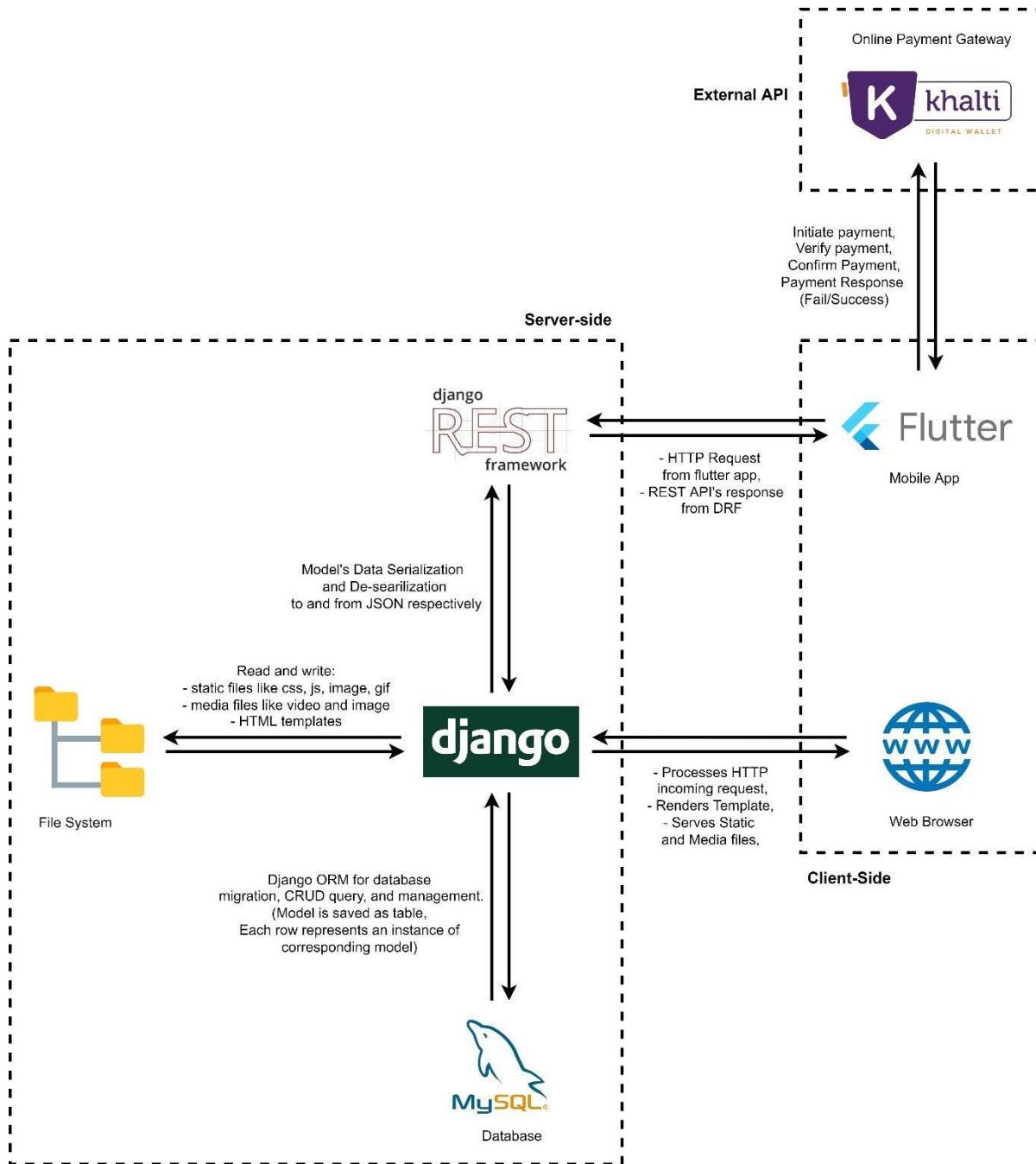


Figure 101: System Architecture

The front and backend are the buzzwords in the IT industries (Pastorino, 2021). The frontend application deals with client-side interaction that users can see, touch and experience, while the backend application deals with server-side processing to respond to user interaction (Mbaabu,

2021). The above system architecture illustrates the frontend and backend as client-side and serverside respectively involved in the Social Service Application development.

7.4.1.1. THREE-TIER ARCHITECTURE

One of the prevailing architectures for client-server applications, Three-Tier Architecture organizes the application into three independent logical and physical computing tiers that run on its infrastructure. The three computing tiers are the presentation tier, application tier, and data tier. The System follows three-tier architecture and involves full-stack development (frontend and backend) (Education, 2020). The project development includes a flutter mobile application and web applications for the client-side rendering of related data and information. The client-side rendering corresponds to the Presentation Tier. The data and information are stored in Data Tier: MySQL database. The user requests are processed by the Application Tier: Django Backed Server. Per request, the Application tier may need to query on Data tire for requested data and information.

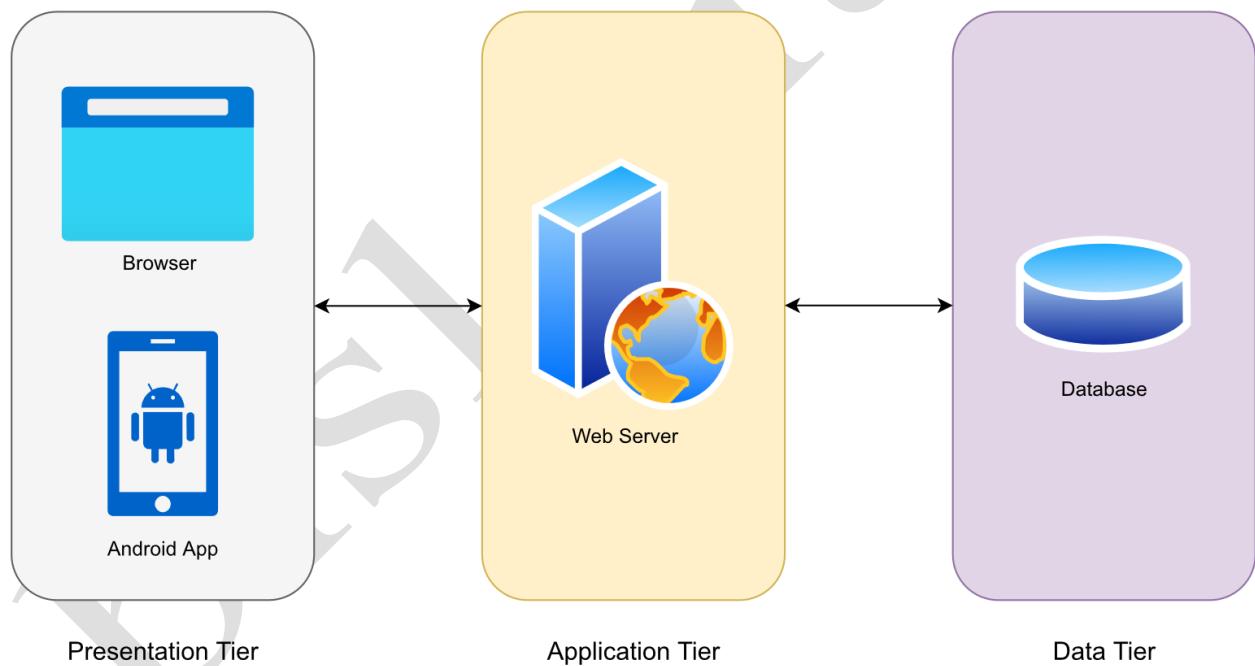


Figure 102: Three-Tier Architecture

7.4.2. SIMILAR PROJECTS

7.4.2.1. SYSTEM B: NEPAL RED CROSS SOCIETY (NRCS)

The Nepal Red Cross Society (NRCS) recognized by the international committee of the Red Cross (ICRC) is the largest humanitarian organization in Nepal with networks of 77 districts. IT is an

effective, self-sustaining, and autonomous humanitarian organization to alleviate or reduce human suffering without any discrimination regarding race, gender, caste, class, tribe, nationality etc. (NRCS, 2021). The NRCS website has multiple pages each for home, about, work, news and updates, resources, donation etc. the donation page has information on contact emails, bank details and multiple ways to donate including cheque, cash, bank transfer and draft.



To keep our ambulances running and relief operations prepared to take action in case of the disasters, we need your support!

If you're interested to donate to NRCS, you can do it by Draft / Cheque / Cash / Bank Transfer. Please contact Nepal Red Cross Society, National Headquarter via telephone 01-4671608 / 01-4270650 or send email to ashok.karki@nrcts.org if you need more detailed instructions.

You can also send "A/C payee" Cheque / Draft in favour of "Nepal Red Cross Society".

Deposit as per the following bank details:

Bank name: Standard Chartered Bank Nepal Limited
Bank address: New Baneswor, Kathmandu , Nepal
Account name: Nepal Red Cross Society
Account number (saving): 17-0002089-55
Swift code: SCBLNPKA

Thank you for your support.

We are currently developing different ways to donate money or goods to Nepal Red Cross Society. If you have some ideas, please feel free to contact us.

When fire took everything, Red Cross provided the help!

Ram Bahadur's house caught a fire, and eventually was burned down completely. He lost everything: home, food, clothes, and shelter. He and six other family members had to sleep under the open sky. Red Cross provided the family some cash and food to start their life again.



NRCS is the largest humanitarian organization in Nepal. Our team of volunteers

Figure 103: NRCS Website - Donate (NRCS, 2021)

7.4.2.2. SYSTEM C: FACEBOOK

Facebook is the world's largest social networking website in terms of global reach and total active users, with around 2.89 billion monthly active users as of the 2nd quarter of 2021 (Statista, 2021). It has simplified interaction with friends and family online by enabling to sharing and exchange of

thoughts, ideas, images, videos etc. (Forbes Media LLC, 2021). The feature to create free public pages has enabled users (I.e., businesses, brands, celebrities, individuals, and organizations) to increase advertisement and audience reachability (Meta, 2021). NGOs are one of many to transition to Facebook pages and take the benefits.



Figure 104: Facebook page - NGO Federation of Nepal (Meta, 2021)

7.5.

E: CONSIDERED METHODOLOGY

7.5.1. EXTREME PROGRAMMING

Extreme programming (XP) is one of the agile software development methodologies to develop quality software viable even in a small development team with changing requirements and technologies. It emphasizes collaborative and productive teamwork of developers, managers and client customers for simplicity, communication, feedback, respect, and courage to improve the project by reducing cost and software delivery time (*Wells, 2013*). Extreme programming methodology has six phases for project completion: Planning, Managing, Designing, Coding, Testing etc.

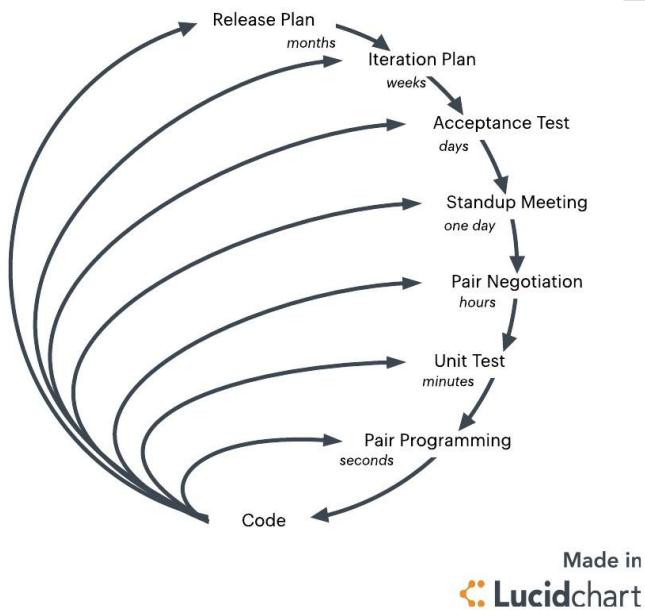


Figure 105: Extreme Programming Methodology (Team, 2021)

The reason to consider the XP methodology for managing this project is below:

- With a focus on timely delivery and regular testing, XP is time-saving.
- XP does not prioritize much documentation.
- XP allows extremely simple code writing that can be improved at any time
- The process/progress/accomplishment in XP is visible and explicable.
- XP is convenient in making mid-changes due to constant feedback and team discussions (*Kukhnavets, 2018*).

7.5.2. KANBAN

Kanban is a workflow management agile framework that mainly focuses on work visualization and Work-in-Progress (WIP) Limit of a project and enrolled people to deliver a quality product efficiently. The workflow activities to complete a task is incremental and evolutionary which helps to gradually improve the existing working method by highlighting the potential bottleneck. The work amount and work queues can be managed, observed, and measured using Kanban in different workflow phases and increase worker's productivity continuous development (Hyytiälä, 2011). The phases of software development using Kanban are Backlog, Impact Analysis, Build, User Acceptance Testing (UAT), Release, Documentation, Done etc.

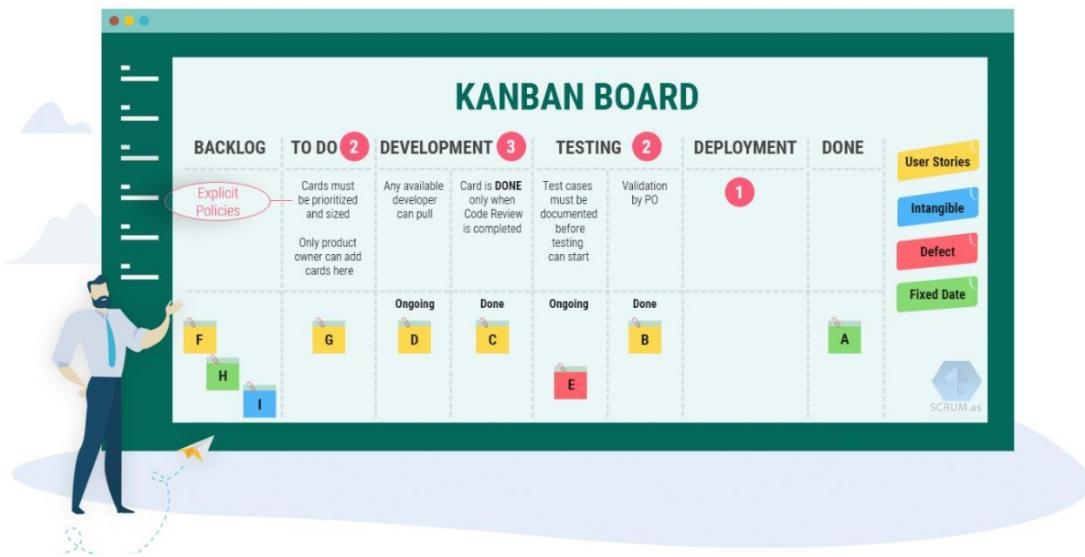


Figure 106: Kanban Methodology (Lerche-Jensen, 2019)

The reason to consider the Kanban methodology for managing this project is below:

- The work visualization practice enables to easily spot and solve inefficiency causing bottlenecks from forming.
- The Kanban board improves flow efficiency and increases workflow, resulting in increased productivity during project execution.
- Kanban Work in Progress (WIP) limit limits excessive tasks at the same time and decreases overburden and increases focus on upcoming tasks
- Kanban helps to reduce waiting time, idle tasks, and queuing states by focusing on required materials, time, and effort (Siderova, 2019).

7.6. F: PHASES OF METHODOLOGY

7.6.1. BACKLOG

A project backlog is a visual representation of items that will or will not be delivered. The backlog lives on the leftmost area of a Kanban board before the "To Do" stage (column). The Kanban backlog isn't simply for jotting down ideas or consumer feedback. Instead, It is to make ideas more actionable over time so that they may be brought to life. Everything in the backlog must be preserved in a semantic order, which should be accomplished with as little effort as possible (Kanbanize, 2022). The 'To Do' column section holds task cards pulled from the backlog that is well-understood to be carried out successfully. Task cards from the 'To Do' can be pulled to 'Doing', respecting the maximum WIP (Work In Progress) limits.

7.6.2. REQUIREMENTS

Requirements are the key part of a project regardless of any framework or methodologies. The requirements gathering proceeds after the feasibility study. The feasibility study can be done by conducting a pre-survey among multiple user candidates. After the analysis of the survey is done and if the project seems feasible, all the software requirements are gathered, defined, and documented in a Software Requirement Specification (SRS).

7.6.3. DESIGN

As the sub-heading name implies, the design phase involves designing of necessary logical components and architecture that the system will rely on. Before the development phase is executed, The database is modelled in an Entity relational Diagram (ERD) and Data Dictionary is created to define constraints and data types. Afterwards, DFDs and various UML diagrams like use case, communication/collaboration, sequence, and class diagram are made to map/model the structure of the system. The UML diagrams mostly model the system internals. The system external or User Interface (UI) that is perceived by human eyes, is modelled into Wireframes and prototypes.

7.6.4. DEVELOPMENT

The software is constructed during the development stage. This requires several steps, including development, infrastructure setup, and documenting the system's operation. Developers and designers may collaborate to ensure that their work is consistent with the

designs that were developed in the design phase. If a problem arises, the development team may collaborate with the design team to resolve it (Editor, 2021).

7.6.5. TESTING

The development is the bulkiest job in the overall software project. After the bulk of the job is finished, the software product is sent for testing purposes to ensure its quality before being handed or publish to the public or client. Various tools and frameworks are used to create a testing environment and automate the test to identify any software flaws (Editor, 2021).

7.6.6. DEPLOYMENT

The project phases end with the deployment phase, which puts the product into production. The software product is ready to go live in use after it is tested thoroughly with the passed test result. The test success indicates that the software product is read for all end-users to utilize in a real-world setting. The software deployment is carried out either by developing physical server(s) or using the cloud computing services with the server or serverless options.

7.7. G: SAMPLE CODES

7.7.1. SAMPLE CODE OF THE UI

7.7.1.1. WEB APP

```

{%
    extends 'core/base.html' %
}
{%
    load bootstrap5 %
} {%
    load static %
}

{%
    block title %}Review Report Post{%
    endblock %
}
{%
    block linkscript %
        <link rel="stylesheet" type="text/css" href="{%
            static
            'core/css/reported-posts.css'
        }"%
}{%
    endblock %
}

{%
    block body %
        {%
            include 'core/extensions/navbar.html' with report='active' %
        }
        <div class="container-fluid px-4 my-2">
            <div class="row">
                <section id="post-section" class="col-sm-7">
                    <div>
                        <div class="my-2"><i class="bi bi-person-linesfill"></i> Posted by:
                            <span class="fw-bold">
                                {%
                                    if object.people_posted_post_rn.exists %
                                        <a href="{%
                                            url 'read-people'
                                            object.people_posted_post_rn.first.id
                                        }" class="text-decoration-none">
                                            {%
                                                object.people_posted_post_rn.first.account.username
                                            } %
                                    {% else %}
                                        <a href="{%
                                            url 'read-ngo'
                                            object.ngo_posted_post_rn.first.id
                                        }" class="text-decoration-none">
                                            {%
                                                object.ngo_posted_post_rn.first.account.username
                                            } %
                                    {% endif %
                                }
                            </a>
                            </span>
                        </div>
                        <div class="my-2"><i class="bi bi-calendar2-weekfill"></i> Date Posted: {{ object.created_on.astimezone }}</div>
                    <div class="my-2"><i class="bi bi-signpost-2fill"></i> Post Type:
                        <span class="badge bg-primary">{{ object.post_type }}</span>
                        {%
                            if object.post_type == 'Request' %
                                <span class="badge bg-primary">{{ object.postrequest.request_type }}</span>
                            {% endif %
                        }
                    </div>
                    {#
                        Actual Post-----#
                    <div class="mt-4" style="background-color: white; padding:

```

```

2vh; border-radius: 1.5vh">
    {#
        Post related to-----
    #}
    <div class="mb-4">
        <div class="fw-bold mb-2" style="display: block;">
Post related to:</div>
        {%
            for field in object.related_to %
                <div class="pills">{{ field }}</div>
            {% endfor %}
        </div>
        {#
            post description-----#
        <div class="my-4" style="text-align: justify">
            <div class="fw-bold mb-2" style="display: block;">
Post Content:</div>
            {{ object.post_content }}
        </div>
        {#
            post type body -----
        --#}
        <div class="my-4">
            {%
                if object.post_type == 'Normal' %
            <div class="fw-bold mb-2" style="display: block;"> Post attached
Photo:</div>
                {%
                    if object.postnormal.post_image %
                        <a href="{{ object.postnormal.post_image.url
}}>
                            <span class="badge bg-secondary">Total Reactions: {{ object.poll_reactions }}</span> | Poll ends on: <span class="fw-bold">{{ object.postpoll.ends_on }}</span></div>
                            {%
                                for data in object.poll_data %
                                    <div class="row">
                                        <div class="col my-2"><strong>{{ data.0
}}</strong></div>
                                            <div class="col my-2" style="textalign:
right"><span class="badge bg-secondary">Reaction: {{ data.1
}}</span> | {{ data.2 }}%</div>
                                                </div>
                                                <div class="progress">
                                                    <div class="progress-bar progress-
barstriped progress-bar-animated" role="progressbar" aria-valuenow="{{ data.2
}}" aria-valuemin="0" aria-valuemax="100" style="width: {{ data.2
}}%"></div>

```

```
        </div>
    {%
      % endfor %
    }
    {%
      % if object.post_type == 'Request' %
    <div class="d-block" style="text-align: right"><span class="badge
bg-secondary">Total Signs: {{ object.postrequest.reacted_by.count }}</span> | Request ends on: <span
class="fw-bold">{{ object.postrequest.ends_on }}</span></div>
      <div class="progress my-2">
        <div class="progress-bar bg-warning"
role="progressbar" style="width: {{ object.request_data.sign_percentage
}}%" aria-valuenow="{{ object.request_data.sign_percentage }}" aria-
```

```

valuemin="0" aria-valuemax="100" data-bs-toggle="tooltip" data-
bsplacement="top" title="People who signed: {{ object.request_data.sign }}}
| {{ object.request_data.sign_percentage }}%"></div>


BISHAL GHARTI CHHETRI



120


```

```

        <li class="list-group-item d-flex justify-content-between align-items-center">Down-Votes
            <span class="badge bg-warning text-dark">{{ object.postnormal.down_vote.count }}</span>
        </li>
        <li class="list-group-item d-flex justify-content-between align-items-center">Reports
            <span class="badge bg-danger">{{ object.postnormal.reported_by.count }}</span>
        </li>
        { % endif %}
        { % if post.post_type == 'Request' %}
<li class="list-group-item d-flex justify-content-between align-items-center">Reports
            <span class="badge bg-danger">{{ object.postrequest.reported_by.count }}</span>
        </li>
        { % endif %}
        { % if post.post_type == 'Poll' %}
            <li class="list-group-item d-flex justify-content-between align-items-center">Reports
                <span class="badge bg-danger">{{ object.postpoll.reported_by.count }}</span>
            </li>
            { % endif %}
        </ul>
    </div>
    <div id="report-post-in-div">
        <form action="{% url 'review-report' object.report.id %}" method="post">
            { % csrf_token %}
            { % bootstrap_form object.report_form %}
            { % if user.is_superuser %}
                <button type="submit" formaction="{% url 'read-staff' object.report_form.instance.report_reviewed_by.first.id %}" formmethod="get" class="btn btn-secondary w-100">Reviewed by: {{ object.report_form.instance.report_reviewed_by.first.full_name }}</button>
            { % else %}
                <button type="submit" class="btn btn-primary w-100">Take Action</button>
            { % endif %}
        </form>
    </div>
</div>
{ % endblock %}

```

7.7.1.2. MOBILE APP

```

import 'package:flutter/material.dart'; import
'package:provider/provider.dart'; import
'package:sasae_flutter_app/providers/post_provider.dart'; import

```

```
'package:sasae_flutter_app/widgets/misc/custom_appbar.dart'; import
'package:sasae_flutter_app/widgets/misc/custom_loading.dart'; import
'package:sasae_flutter_app/widgets/misc/fetch_error.dart'; import
'package:sasae_flutter_app/ui/post/post_type/post_dependent_widgets/normal_im
age_attachment_card.dart'; import
'package:sasae_flutter_app/ui/post/post_type/post_dependent_widgets/post_poke
d_ngo_card.dart'; import
'package:sasae_flutter_app/ui/post/post_type/post_dependent_widgets/post_auth
or_card.dart'; import
'package:sasae_flutter_app/ui/post/post_type/post_dependent_widgets/post_desc
ription_card.dart'; import
'package:sasae_flutter_app/ui/post/post_type/post_dependent_widgets/post_rela
ted_card.dart'; import
'package:sasae_flutter_app/ui/post/post_type/post_dependent_widgets/post_tail
_card.dart'; import
'package:sasae_flutter_app/ui/post/post_type/post_dependent_widgets/normal_vo
ting_bar.dart';
class NormalPostScreen extends StatefulWidget
{
  static const routeName = '/post/normal/';
  final int postID;

  const NormalPostScreen({
    Key? key,
    required
    this.postID,
  }) : super(key: key);

  @override
  _NormalPostScreenState createState() => _NormalPostScreenState();
}
class _NormalPostScreenState extends
State<NormalPostScreen> {
  final ScrollController
  _scrollController;
  late NormalPostProvider _provider;
  late Future<void> _fetchNormalPostFUTURE;

  _NormalPostScreenState() : _scrollController = ScrollController();

  @override
  void
  initState() {
  super.initState();
  _provider = Provider.of<NormalPostProvider>(context, listen: false);
  _fetchNormalPostFUTURE = _fetchNormalPost();
}

  Future<void> _fetchNormalPost() async {
    await
    _provider.initFetchNormalPost(postID: widget.postID);
  }
}
```

```
    @override void
dispose() {
    _scrollController.dispose();
_provider.nullifyNormalPost();
super.dispose();
}

@Override
Widget build(BuildContext context) {
return Scaffold(      appBar: const
CustomAppBar(          title: 'View
Normal Post',
),      body: FutureBuilder(
future: _fetchNormalPostFUTURE,
builder: (context, snapshot) => snapshot.connectionState ==
ConnectionState.waiting
? const CustomLoading()
: Consumer<NormalPostProvider>(
        builder: (context, postP, child) => RefreshIndicator(
onRefresh: () =>
        postP.refreshNormalPost(postID: widget.postID),
child: postP.getNormalpostData == null
? const ErrorView()
: ListView(
            controller: _scrollController,
physics: const AlwaysScrollableScrollPhysics(),
padding: const EdgeInsets.fromLTRB(15, 10, 15, 0),
children: [
        const SizedBox(
height: 15,
),
if (postP
        .getNormalpostData!.pokedNGO.isNotEmpty) ...[
PokedNGOCARD(
postP.getNormalpostData!.pokedNGO,
),
const SizedBox(
height: 15,
),
if (postP.getNormalpostData!.attachedImage !=
null) ...[
NormalImageAttachmentCard(
 imageURL:
postP.getNormalpostData!.attachedImage!,
```



```

    }
}

```

7.7.2. SAMPLE CODE FOR THE TEST AUTOMATION SCRIPT

7.7.2.1. DJANGO

```

from django.contrib.auth.models import User from
django.urls import reverse
from core.models import PeopleUser, Post, PostNormal, Report from
core.tests.base import BaseTest
class
PostReportReviewTest(BaseTest):
    def setUp(self):
super().setUp()

        self.read_reports = reverse('read-reports')

        self.payload = {
            'reason': 'Lorem ipsum dolor sit amet, consectetur adipiscing
elit. Etiam aliquam, ipsum efficitur '
                    'ultricies dapibus, tortor mi rhoncus nisi, sit amet
vehicula justo arcu eget massa. Proin '
                    'libero sapien, dignissim ut malesuada at, lobortis
quis lacus. Sed ornare mauris ac leo '
                    'rhoncus vestibulum. Praesent lacus purus,
sollicitudin vel velit sagittis, volutpat varius '
                    'tortor. Nulla non fermentum sapien. Nam sed
fermentum neque. Nunc aliquet lacinia ornare.',
            'action': 'Account Ban',
        }
        user = User.objects.create_user(username='people1',
password='mango321')
        self.people = PeopleUser.objects.create(account=user,
full_name='mr xyz', phone='9779800730959')
        post = Post.objects.create(related_to=['Communication', 'Labor'],
post_content='lorem ipsum',
                           post_type='Normal')
normal_post = PostNormal.objects.create(post=post,)
normal_post.reported_by.add(self.people)
self.people.posted_post.add(post)           self.report =
Report.objects.create(post=post)
self.new_staff.staff.report_review.add(self.report)

#
def
test_admin_access_reports(self):
    self.login_as_admin()
    response = self.client.get(self.read_reports)
self.assertEqual(response.status_code, 200)
    self.assertTemplateUsed(response, 'core/report/reports-read.html')
def
test_staff_access_reports(self):

```

```

        self.login_as_staff()
        response = self.client.get(self.read_reports)
    self.assertEqual(response.status_code, 200)
        self.assertTemplateUsed(response, 'core/report/reports-read.html')
def test_admin_read_report(self):
        self.login_as_admin()           response =
    self.client.get(reverse('read-report', kwargs={'pk':
self.report.id}))
        self.assertEqual(response.status_code, 200)
        self.assertTemplateUsed(response, 'core/report/report-
reviewread.html')
    def
test_staff_read_report(self):
        self.login_as_staff()           response =
    self.client.get(reverse('read-report', kwargs={'pk':
self.report.id}))
        self.assertEqual(response.status_code, 200)
        self.assertTemplateUsed(response, 'core/report/report-
reviewread.html')
    def
test_admin_review_report(self):
        self.login_as_admin()           response =
    self.client.post(reverse('review-report', kwargs={'pk':
self.report.id}), data=self.payload)
        self.assertEqual(response.status_code, 200)
        self.assertTemplateUsed(response, 'core/extensions/403-page.html')
    def
test_staff_review_report(self):
        self.login_as_staff()           response =
    self.client.post(reverse('review-report', kwargs={'pk':
self.report.id}), data=self.payload,
follow=True)           self.assertEqual(response.status_code,
200)           new = Report.objects.get(pk=self.report.id)
    self.assertEqual(new.is_reviewed, True)
    self.assertEqual(new.reason, self.payload['reason'])
    self.assertEqual(new.action, self.payload['action'])           if
    self.payload['action'] == Report.ACTION[0][0]:
        self.assertEqual(new.post.is_removed, True)           if
    self.payload['action'] == Report.ACTION[1][0]:
        self.assertEqual((new.post.people_posted_post_rn or
new.post.ngo_posted_post_rn).first().account.is_active,
False)
    self.assertTemplateUsed(response, 'core/report/reports-read.html')

```

7.7.2.2. FLUTTER

```

import 'dart:math'; import 'package:faker/faker.dart'; import
'package:jiffy/jiffy.dart'; import
'package:sasae_flutter_app/models/post/ngo_.dart'; import
'package:sasae_flutter_app/models/post/request_post.dart'; import
'package:test/test.dart';

```

```

void main() {    late Map<String,
dynamic> json;    late RequestPostModel
requestPostModel;
    setUpAll(() {        int min =
faker.randomGenerator.integer(1500);        int target =
faker.randomGenerator.integer(2000, min: min);        int? max =
faker.randomGenerator.boolean()
            ? faker.randomGenerator.integer(3000, min: target)
            : null;
        Random rand = Random();
        json = {
            "id": faker.randomGenerator.integer(1000),
            "related_to": List.generate(
rand.nextInt(8 - 1) + 1,
                (index) => faker.lorem.word(),
            ),
            "post_content": faker.lorem.sentences(rand.nextInt(20 - 3) + 3).join('
'),
            "created_on": Jiffy(faker.date.dateTime(maxYear: 2010, minYear: 1900))
                .format("yyyy-MM-dd'T'HH:mm:ss"),
            "modified_on": Jiffy(faker.date.dateTime(maxYear: 2010, minYear: 1900))
                .format("yyyy-MM-dd'T'HH:mm:ss"),
            "is_anonymous": faker.randomGenerator.boolean(),
            "post_type": "Normal",            "poked_ngo":
List.generate(
faker.randomGenerator.integer(8, min: 0),
                (index) => {
                    "id": faker.randomGenerator.integer(1000),
                    "full_name": faker.company.name(),
                    "display_picture": faker.image.image(random: true),
                }),
            "author": faker.person.name(),
            "author_id": faker.randomGenerator.integer(1000),
            "post_request": {
                "id": faker.randomGenerator.integer(1000),
                "min": min,
                "max": max,
                "target": target,
                "ends_on": Jiffy(faker.date.dateTime(maxYear: 2010, minYear: 1900))
                    .format("yyyy-MM-dd'T'HH:mm:ss"),
                "request_type": faker.randomGenerator.element(['Join', 'Petition']),
                "reacted_by": faker.randomGenerator
                    .numbers(1500, faker.randomGenerator.integer(1500)),
                "is_participated": faker.randomGenerator.boolean()
            }
        };
}

```

```

});    test('JSON deserialized to RequestModel Instance', () {
{      requestPostModel =
RequestPostModel.fromAPIResponse(json);
expect(requestPostModel, isA<RequestPostModel>());
});    test('Normal Post ID casted', () {
expect(requestPostModel.id, json['id']);
});    test('Related-to casted', () {
expect(requestPostModel.relatedTo, json['related_to']);
});    test('Post Content casted', () {
expect(requestPostModel.postContent, json['post_content']);
});    test('Created-on casted', () {
{
expect(requestPostModel.createdOn,
      Jiffy(json['created_on'], "yyyy-MM-dd'T'HH:mm:ss").dateTime);
});
test('Modified-on casted', () {
expect(requestPostModel.modifiedOn,
      Jiffy(json['modified_on'], "yyyy-MM-dd'T'HH:mm:ss").dateTime);
});    test('Is-Anonymous
casted', () {
expect(requestPostModel.isAnonymous, json['is_anonymous']);
});    test('Post Type casted', () {
expect(requestPostModel.postType, json['post_type']);
});    test('Poked NGO casted', () {
expect(requestPostModel.pokedNGO,
      (json['poked_ngo'] as List).map((e) =>
NGO__Model.fromAPIResponse(e)));
});    test('Author name casted', () {
expect(requestPostModel.author, json['author']);
});    test('Author ID casted', () {
expect(requestPostModel.authorID, json['author_id']);
});    test('Minimum participation casted', () {
expect(requestPostModel.min, json['post_request']['min']);
});    test('Target participation casted', () {
expect(requestPostModel.target, json['post_request']['target']);
});    test('Maximum participation casted', () {
expect(requestPostModel.max, json['post_request']['max']);
});    test('Request-post ends-on
casted', () {      expect(
requestPostModel.endsOn,
      Jiffy(json['post_request']['ends_on'], "yyyy-MM-dd'T'HH:mm:ss")
      .dateTime);
});
test('Is-Participated casted', () {
expect(requestPostModel.isParticipated,
json['post_request']['is_participated']);
}

```

```

});  

test('Participation casted', () {  

    expect(requestPostModel.reaction, json['post_request']['reacted_by']);  

});  

test('Request-type casted', () {  

expect(requestPostModel.requestType,  

json['post_request']['request_type']);  

});  

}

```

7.8. APPENDIX H: DESIGN

7.8.1. GANTT CHART

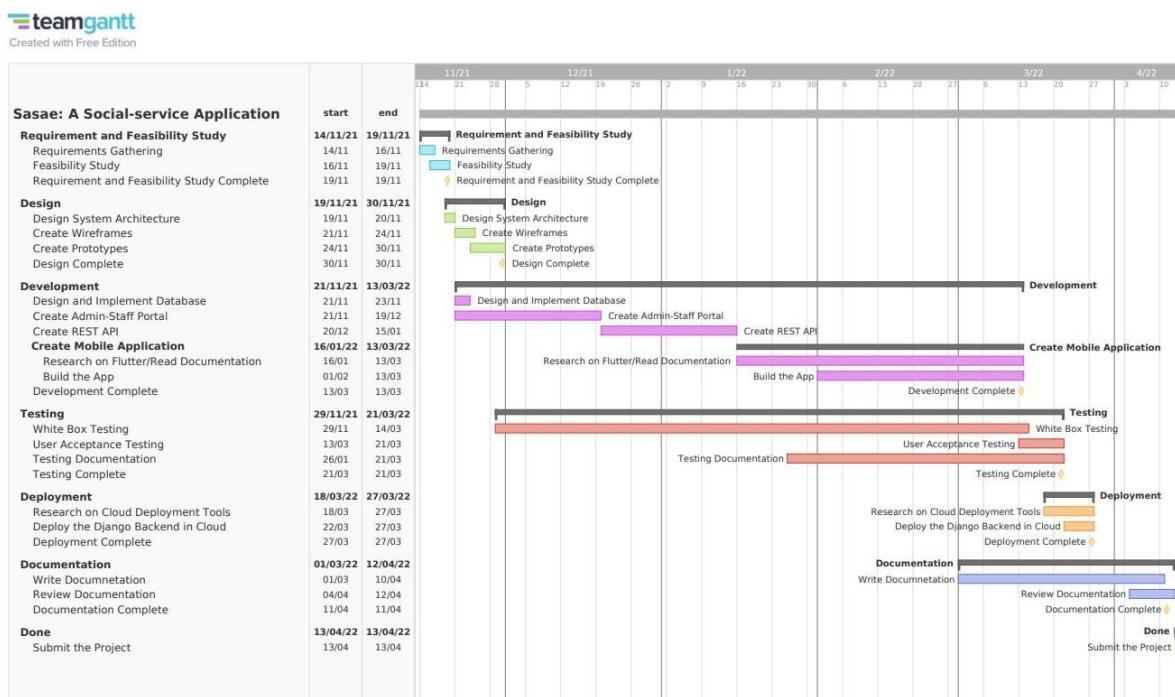


Figure 107: Old Gantt Chart

SASAE

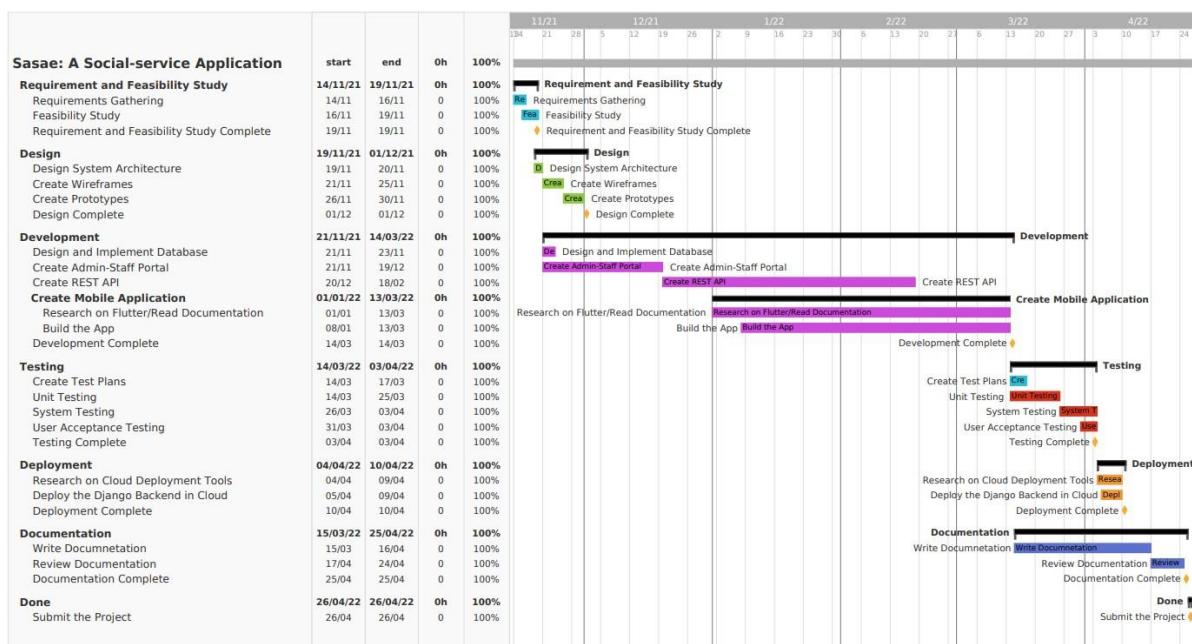


Figure 108: New Gantt Chart (Because of Deadline Extension)

7.8.2. WORK BREAKDOWN STRUCTURE

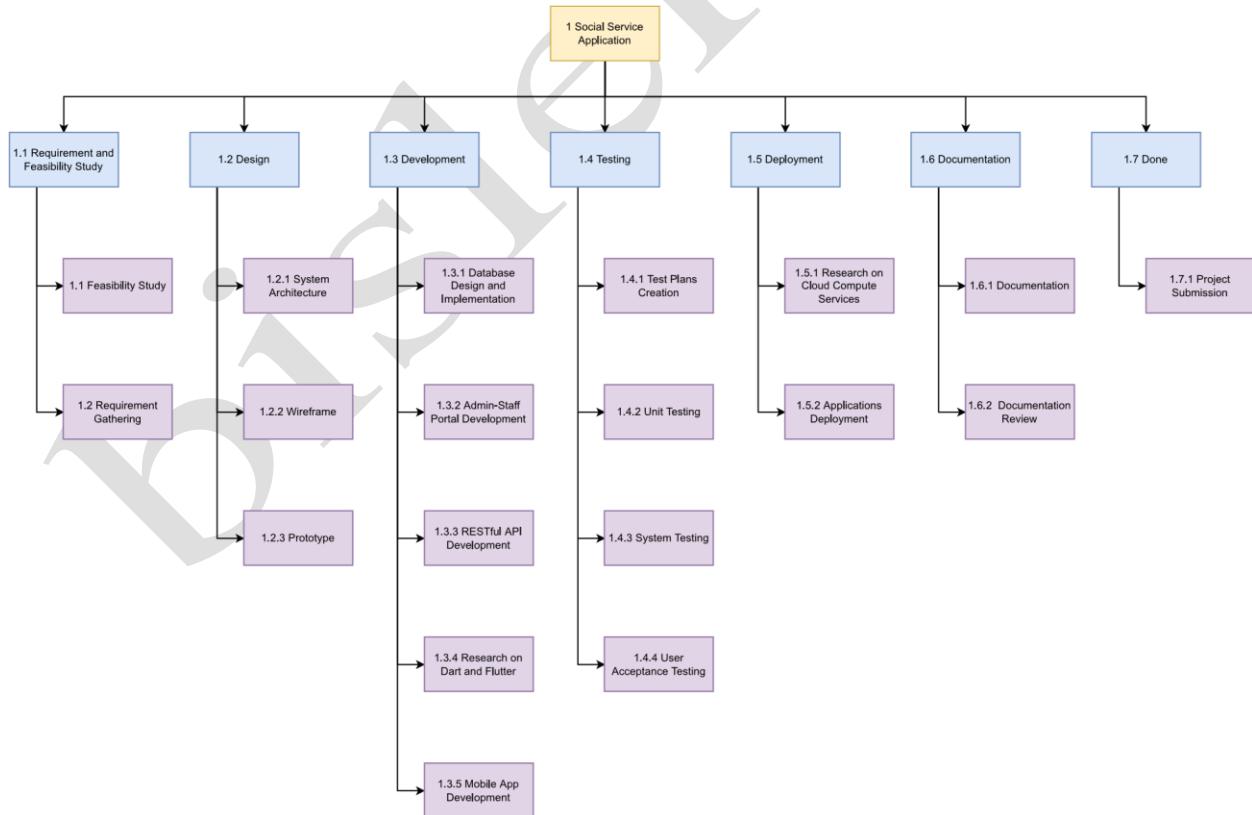


Figure 109: Work Breakdown Structure (WBS)

7.8.3. ALGORITHMS & FLOWCHARTS

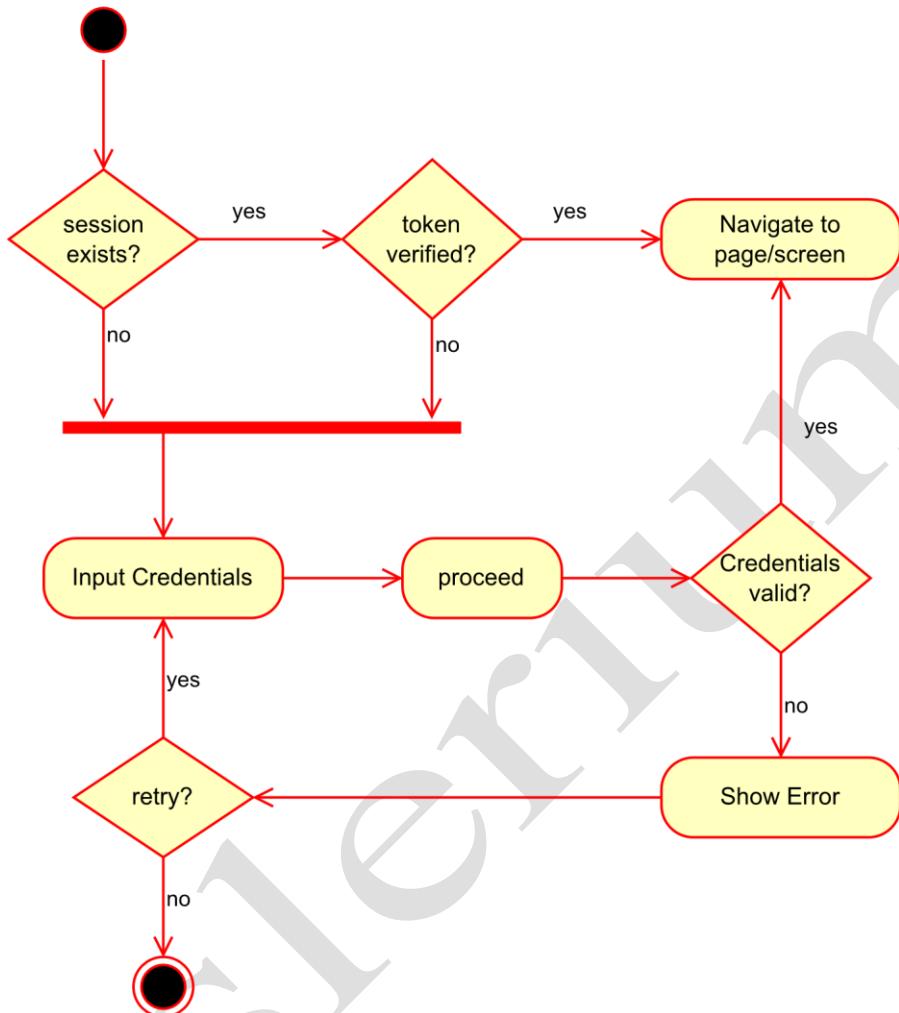


Figure 110: Auto Login Activity (Session-based)

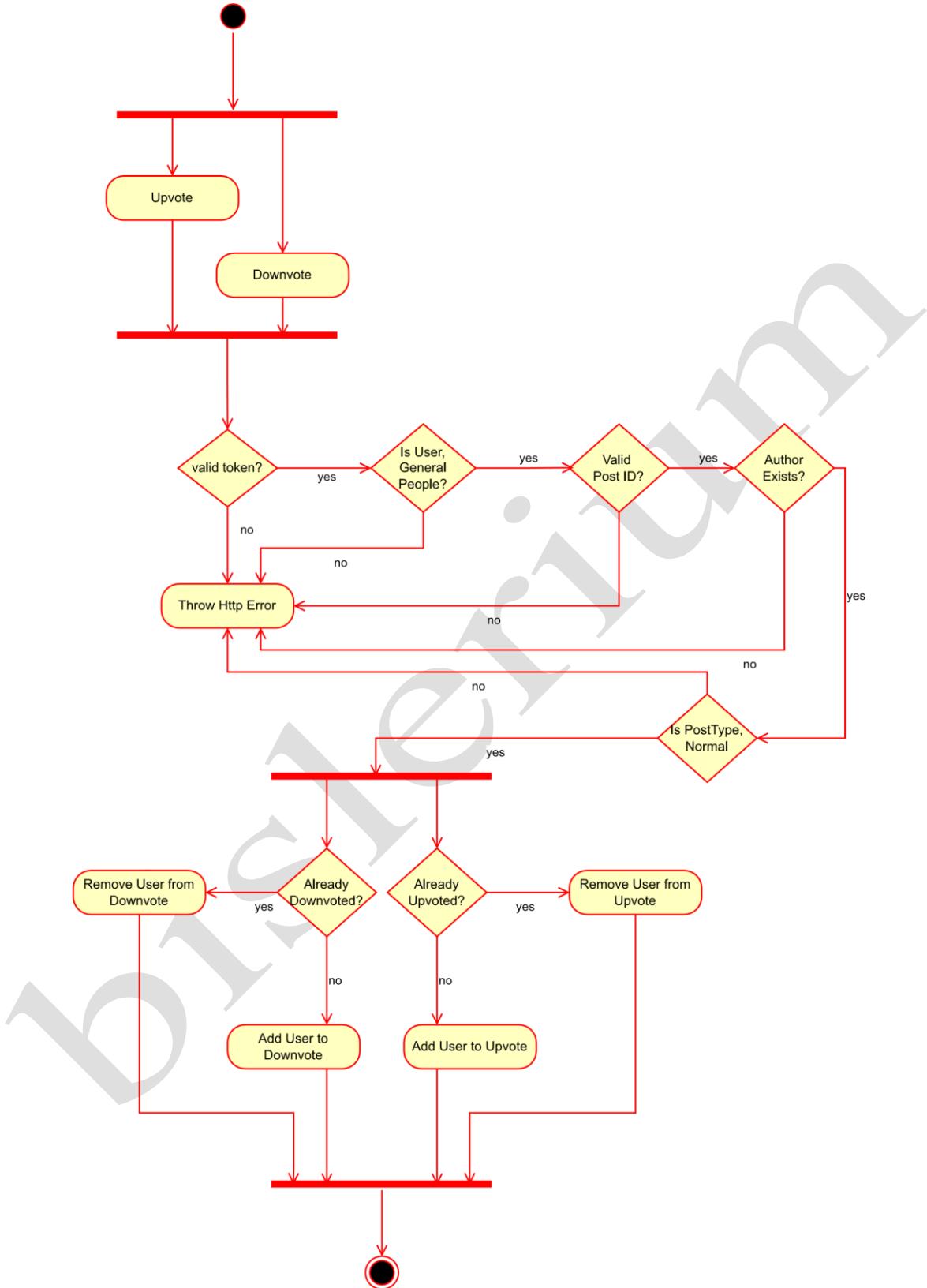


Figure 111: Upvote/Downvote Activity

7.8.4. DATABASE DESIGN

7.8.4.1. DATA DICTIONARY

7.8.4.1.1. PEOPLEUSER

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		Primary Key			To store an id that uniquely identifies General People	9328
full_name	VARCHAR	150	NOT NULL			To store the full name of General People	Ramina Shah
date_of_birth	DATE					To store the birth date of General People	2001-12-05
gender	VARCHAR	6	NOT NULL			To store Gender	Female
phone	VARCHAR	128	NOT NULL			To phone number	9779800465734
address	VARCHAR	150	NOT NULL			To store address	Kalikanagar, Butwal-12
display_picture	VARCHAR	100				To store a display picture of General People	/media/32422.jpg
citizenship_photo	VARCHAR	100				To store Citizenship Photo	/media/83482.jpg

is_verified	BOOL		NOT NULL			To store the verified status	TRUE
account	INTEGER		FOREIGN KEY	User	id	To associate the General People to a User record	293
post_id	BIGINT		FOREIGN KEY	Post	id	To associate all the posts that General People has posted	123,324,122

Table 30: Data Dictionary, PeopleUser Table

7.8.4.1.2. STAFF

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
Id	INTEGER		Primary Key			To store an id that uniquely identifies the Staff	324
full_name	VARCHAR	150	NOT NULL			To store the full name of General People	Priyanka Chopra
date_of_birth	DATE					To store the birth date of General People	1999-07-10
gender	VARCHAR	6	NOT NULL			To store Gender	Female
phone	VARCHAR	128	NOT NULL			To phone number	977980849384
address	VARCHAR	150	NOT NULL			To store address	Kailali -12

display_picture	VARCHAR	100				To store a display picture of General People	/media/3423.jpg
citizenship_photo	VARCHAR	100				To store Citizenship Photo	
is_married	BOOL		NOT NULL			To store the married status	TRUE
account	INTEGER		FOREIGN KEY	User	id	To associate the General People to a User record	1272
report_review	BIGINT		FOREIGN KEY	Report	id	To store the allotted reports that need to be reviewed	421

Table 31: Data Dictionary, Staff Table

7.8.4.1.3. NGOUSER

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		Primary Key			To store a unique ID that identifies NGO	9328
latitude	DECIMAL		NOT NULL			To store the latitude of a location	65.434343
longitude	DECIMAL		NOT NULL			To store the Longitude of the Location	-102.3433232
full_name	VARCHAR	150	NOT NULL			To store the organization name of General People	INSEC Nepal

SASAE

establishment_date	DATE		NOT NULL			To store the birth date of General People	2008-08-25
--------------------	------	--	----------	--	--	-------------------------------------------	------------

field_of_work	VARCHAR	546	NOT NULL			To store Gender	Education, Awareness
phone	VARCHAR	128	NOT NULL			To phone number	9779800475638
address	VARCHAR	150	NOT NULL			To store address	Kathmandu, Maitidevi
display_picture	VARCHAR	100				To store a display picture of General People	/media/23212.jpg
epay_account	VARCHAR	20	NOT NULL			To store	9833746573
post_id	BIGINT		FOREIGN KEY	Post	id	To associate all the posts that the NGO has posted	423,221,12
swc_affl_cert	VARCHAR	100				To store Social Welfare Council affiliation certificate	/media/03922.jpg
pan_cert	VARCHAR	100				To store PAN certificate	/media/03421.jpg
bank_id	BIGINT		FOREIGN KEY	Bank	id	To store the associate the NGOUser to its Bank account	234

Is_verified	BOOL		NOT NULL			To store the status of whether the user is verified or not	TRUE
account	INTEGER		FOREIGN KEY	User	id	To associate the General People to a User record	293
poked_on	BIGINT		FOREIGN KEY	Post	id	To store the ids of the post where the NGO was poked	232,1234

Table 32: Data Dictionary, NGOUser Table

7.8.4.1.4. BANK

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		Primary Key			To store an id that uniquely identifies the Bank	324
bank_name	VARCHAR	100	NOT NULL			To store the name of the bank	NIC Asia
bank_branch	VARCHAR	100	NOT NULL			To store the branch of the bank	Kamaladi

bank_BSB	VARCHAR	10	NOT NULL			To store Bank State Branch Number	2323432
bank_account_name	VARCHAR	100	NOT NULL			To store the account name of the holder	Hari Lal Koirala
bank_account_number	VARCHAR	20	NOT NULL			To store the account number of the holder	123-22112-3221-3221

Table 33: Data Dictionary, Bank Table

7.8.4.1.5. POST

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		Primary Key			To store an id that uniquely identifies the post	8392
related_to	VARCHAR	546	NOT NULL			To store what post is related to	Labour, Awareness
post_content	TEXT		NOT NULL			To store the content of the post	This is post content
modified_on	DATETIME					To store post modified data time	2012-12-08 12:23:60
is_anonymous	BOOL		NOT NULL			To store anonymous post status	TRUE

is_removed	BOOL		NOT NULL			To store the removed status of the post	FALSE
post_type	VARCHAR	20	NOT NULL			To store the type of current post	Normal
created_on	DATETIME		NOT NULL			To store the created date time of post	2012-10-12 21:12:09

Table 34: Data Dictionary, Post Table

7.8.4.1.6. POSTNORMAL

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		PRIMARY KEY			To store an id that uniquely identifies the post	1232
post_id	BIGINT		FOREIGN KEY	Post	id	To associate the normal post to post.	2332
post_image	VARCHAR					To store an image attached to the normal post	/media/232.jpg
up_vote	BIGINT		FOREIGN KEY	PeopleUser	Id	To store the General peoples who have upvoted this normal post	12,32

down_vote	BIGINT		FOREIGN KEY	PeopleUser	id	To store the General peoples who have downvoted this normal post	32,123
reported_by	BIGINT		FOREIGN KEY	PeopleUser	Id	To store the General peoples who have reported this normal post	43

Table 35: Data Dictionary, PostNormal Table

7.8.4.1.7. POSTPOLL

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		PRIMARY KEY			To store an id that uniquely identifies the post-poll	2139
post_id	BIGINT		FOREIGN KEY	Post	id	To associate the poll post to post.	4231
ends_on	DATETIME					To store the duration of the poll post	2022-04-12 12:30:23
reported_by	BIGINT		FOREIGN KEY	PeopleUser	Id	To store the General peoples who have reported this normal post	43

Table 36: Data Dictionary, PostPoll Table

7.8.4.1.8. POSTREQUEST

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
Id	INTEGER		PRIMARY KEY			To store an id that uniquely identifies the post request	8362
post_id	BIGINT		FOREIGN KEY	Post	id	To associate the poll post to post.	9328
min	INTEGER	NOT NULL				To store the number of minimum participants	87
target	INTEGER	NOT NULL				To store the number of targeted participants	123
max	INTEGER					To store the number of maximum participants	212
ends_on	DATETIME	NOT NULL				To store the duration of the poll post	2021-04-12 12:20:13
signed_by	BIGINT		FOREIGN KEY	PeopleUser	id	To store the General People who has involved in this post	32,31
reported_by	BIGINT		FOREIGN KEY	PeopleUser	id	To store the General peoples who have reported this normal post	43

Table 37: Data Dictionary, PostRequest Table

7.8.4.1.9. POLLOPTION

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		PRIMARY KEY			To store an id that uniquely identifies the poll option	321
option	VARCHAR	50				To associate the poll post to post.	123
reacted_by	BIGINT		FOREIGN KEY	PeopleUser	id	To store the general people user who has reacted to this poll option	122, 233

Table 38: Data Dictionary, PollOption Table

7.8.4.1.10. REPORT

Column Name	Data Type	Size	Constraint	Reference Table	Reference Column	Description	Example Data
id	INTEGER		PRIMARY KEY			To store an id that uniquely identifies the report	321
reason	TEXT					To store the reason for acting.	123
action	BIGINT					To store action applied on the reported post	Account Ban
is_reviewed	BOOL					To store the reviewed status	True
post_id	BIGINT		FOREIGN KEY	Post	id	To associate the report to post.	2231

Table 39: Data Dictionary, Report Table

bisleriump

7.8.4.2. CONCRETE ERD

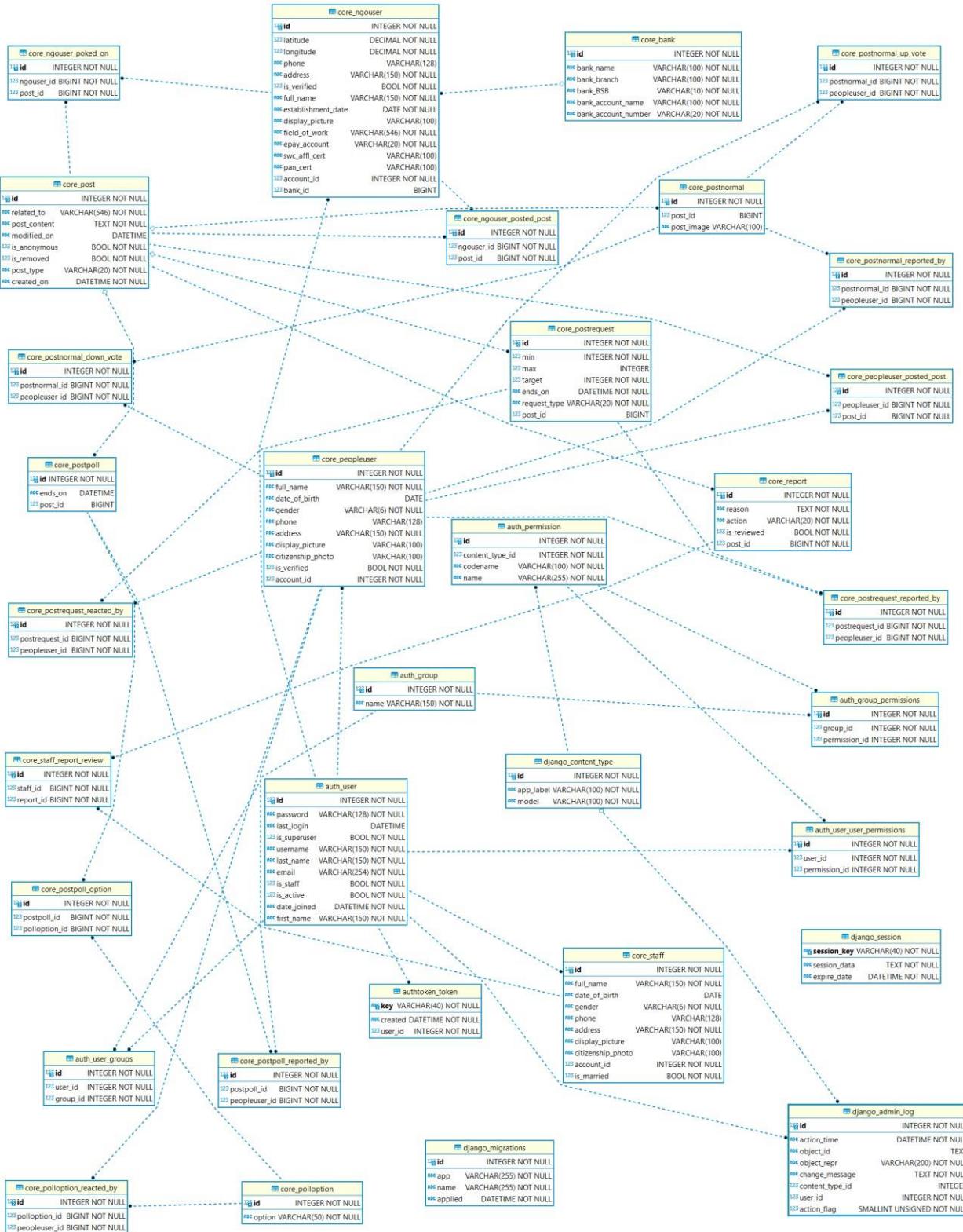


Figure 112: Database Design, Generated Concrete ERD

7.8.5. LEVEL-1 DFDS

7.8.5.1. UC2

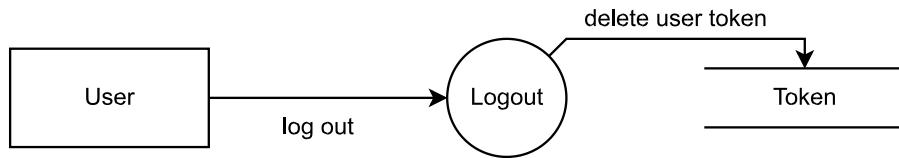


Figure 113: Level-1 DFD, UC2

7.8.5.2. UC3

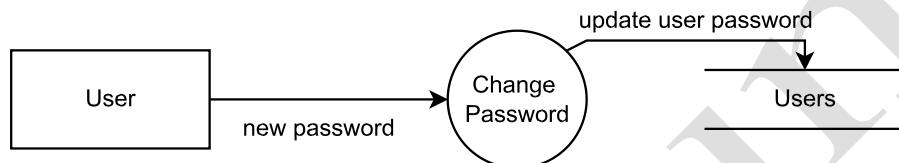


Figure 114: Level-1 DFD, UC3

7.8.5.3. UC4

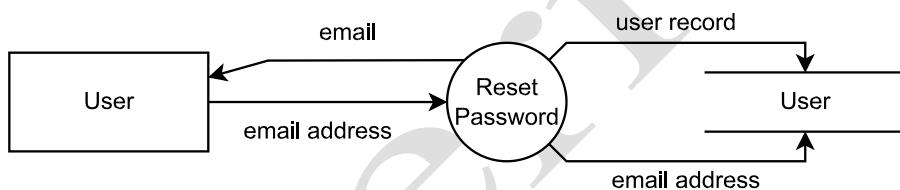


Figure 115: Level-1 DFD, UC4

7.8.5.4. UC5

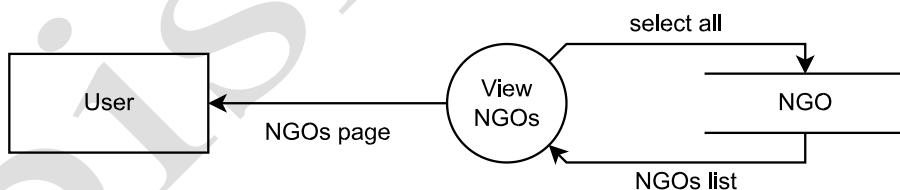


Figure 116: Level-1 DFD, UC5

7.8.5.5. UC6

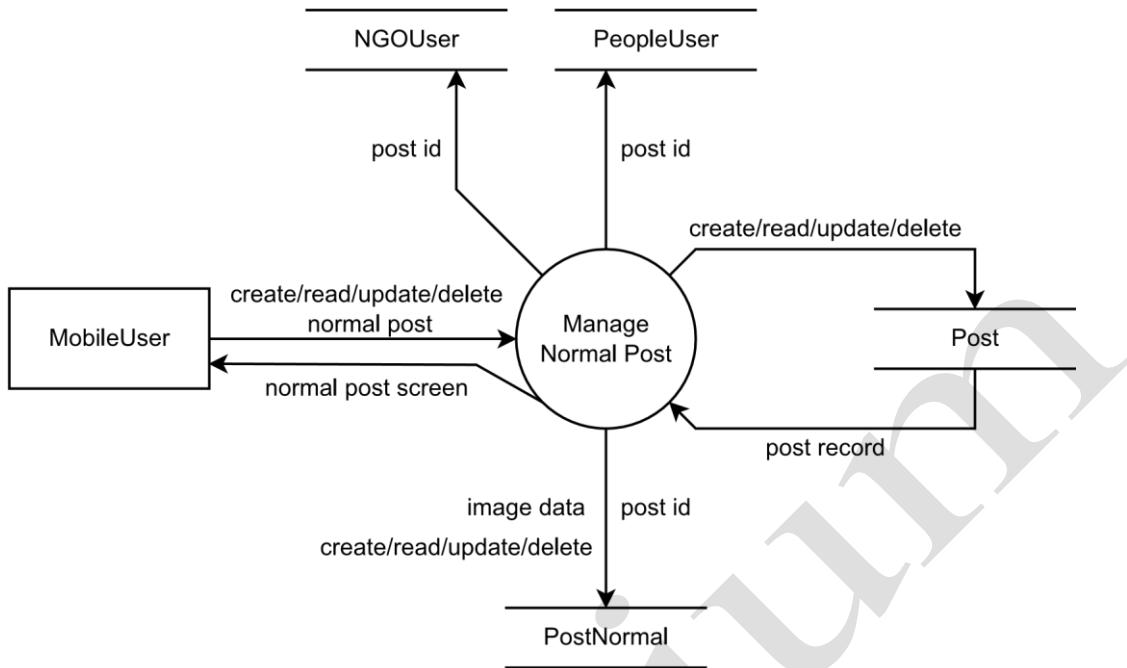


Figure 117: Level-1 DFD, UC6

7.8.5.6. UC7

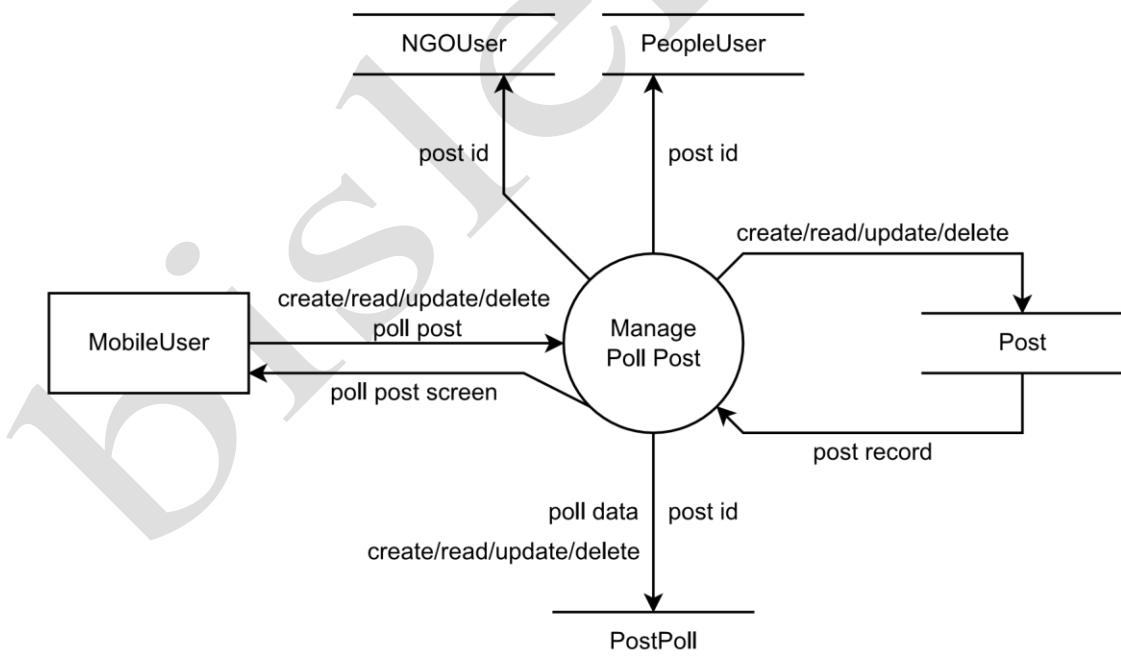


Figure 118: Level-1 DFD, UC7

7.8.5.7. UC8

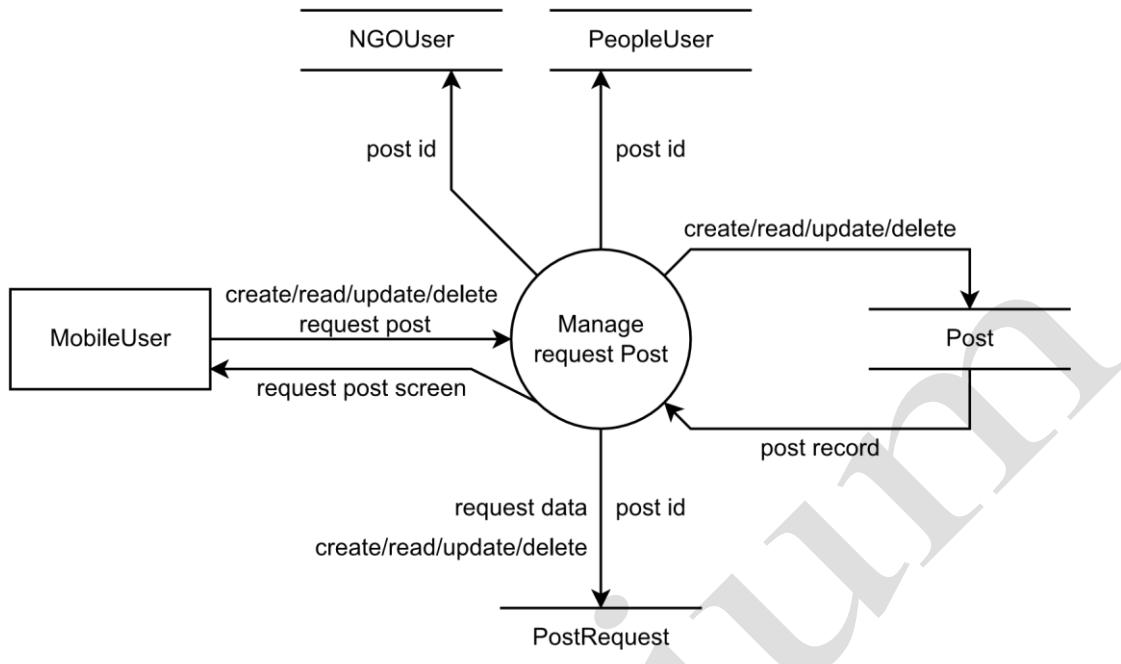


Figure 119: Level-1 DFD, UC8

7.8.5.8. UC9

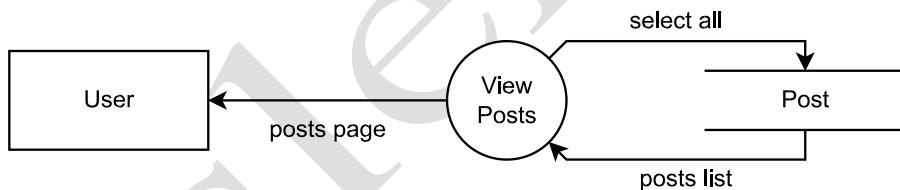


Figure 120: Level-1 DFD, UC9

7.8.5.9. UC10

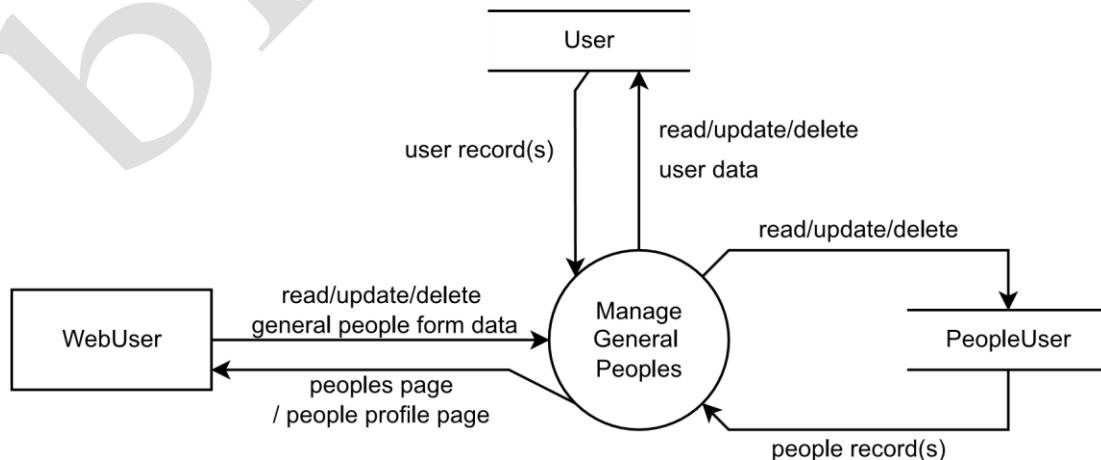


Figure 121: Level-1 DFD, UC10

7.8.5.10. UC11

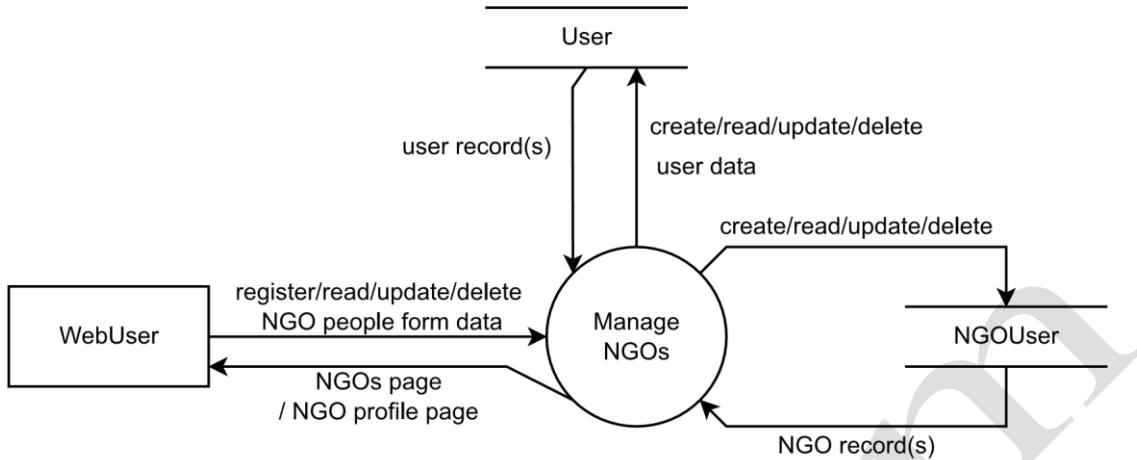


Figure 122: Level-1 DFD, UC11

7.8.5.11. UC12

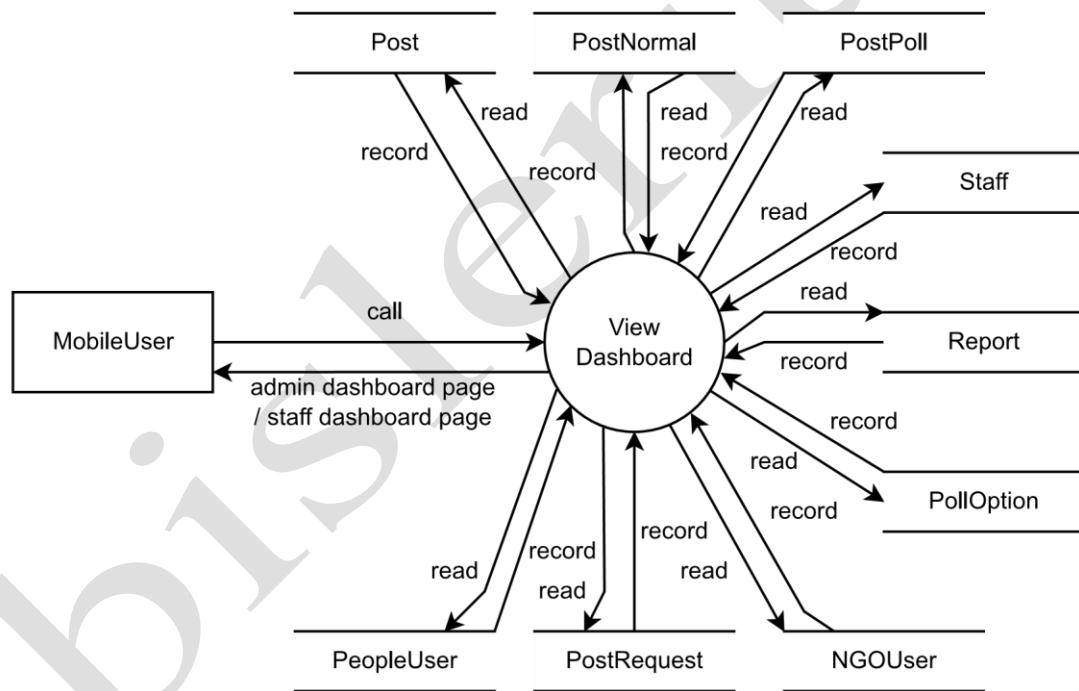


Figure 123: Level-1 DFD, UC12

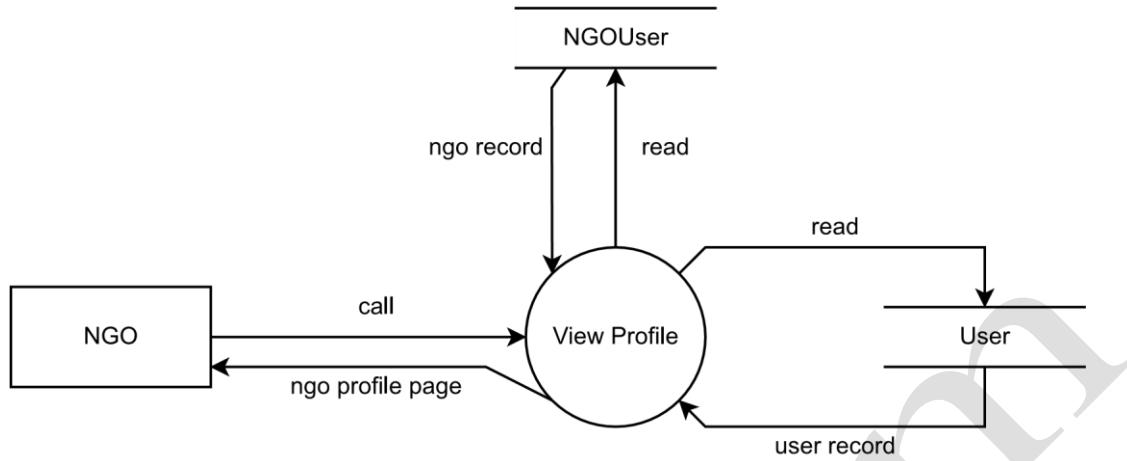
7.8.5.12. UC13

Figure 124: Level-1 DFD, UC13

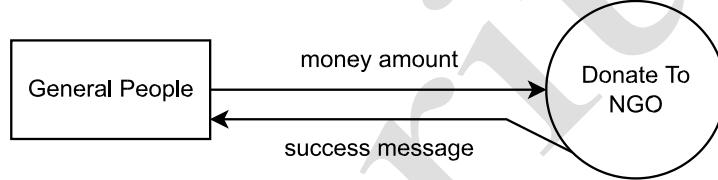
7.8.5.13. UC14

Figure 125: Level-1 DFD, UC14

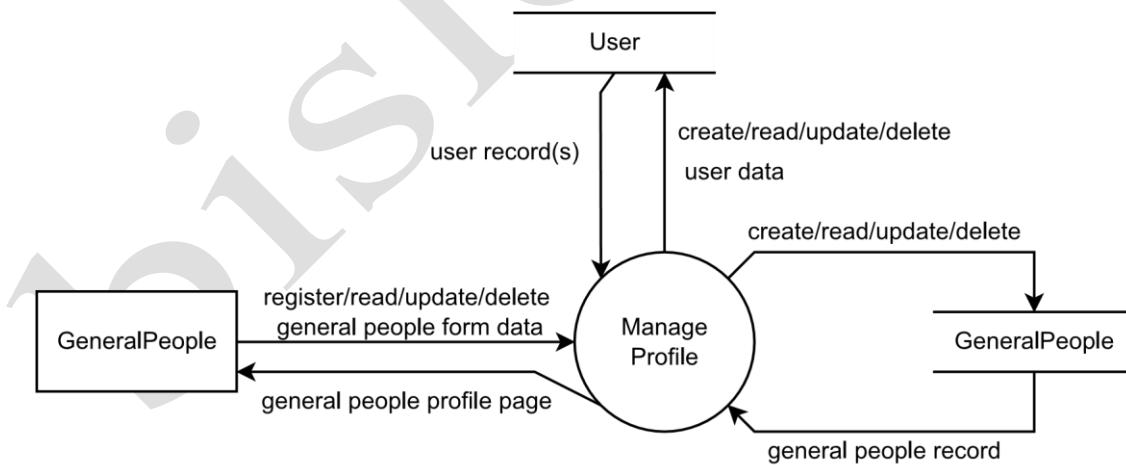
7.8.5.14. UC15

Figure 126: Level-1 DFD, UC15

7.8.5.15. UC16

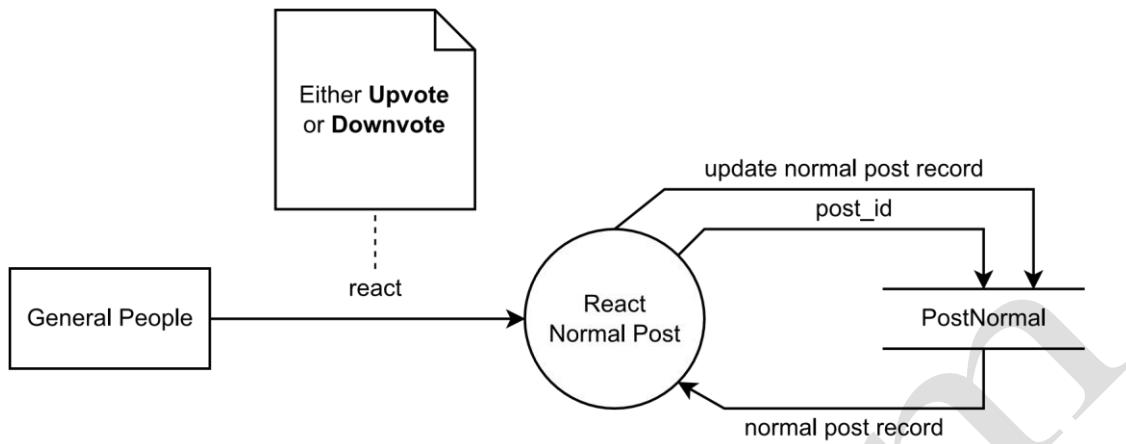


Figure 127: Level-1 DFD, UC16

7.8.5.16. UC17

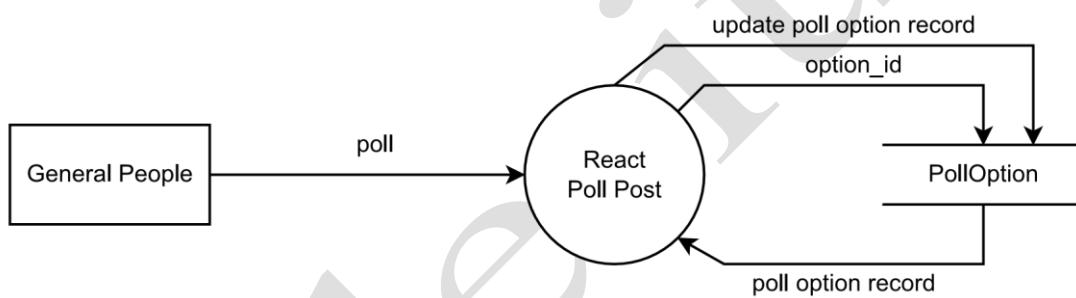


Figure 128: Level-1 DFD, UC17

7.8.5.17. UC18

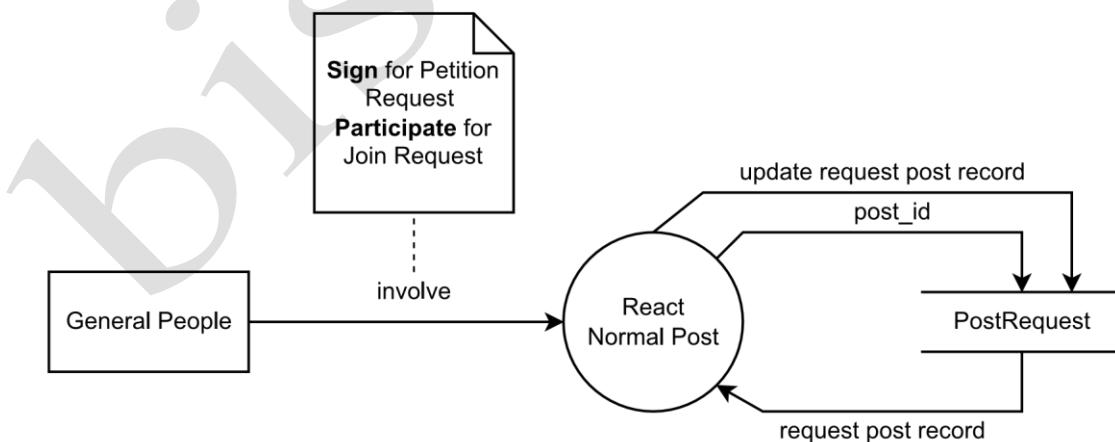


Figure 129: Level-1 DFD, UC18

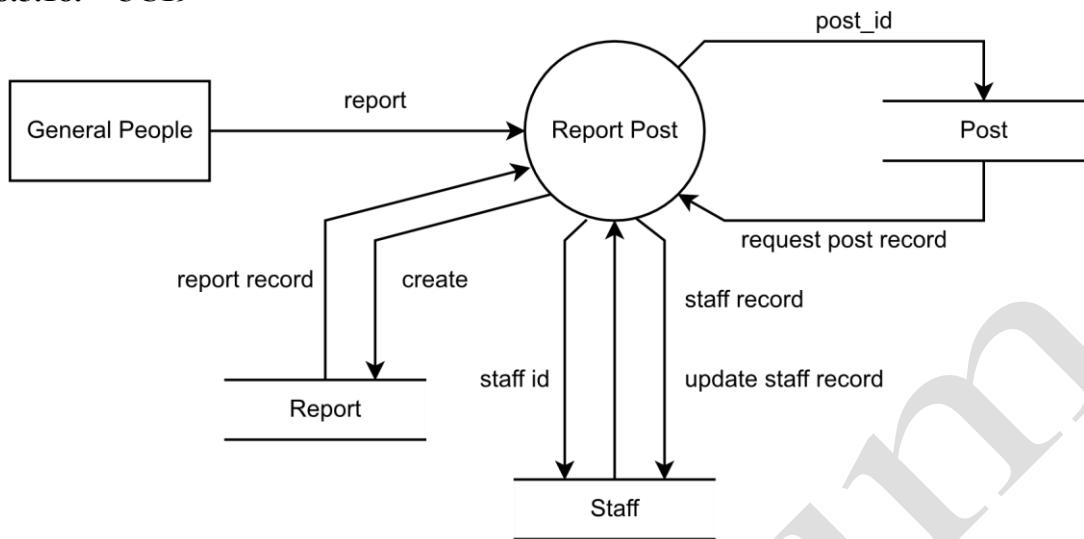
7.8.5.18. UC19

Figure 130: Level-1 DFD, UC19

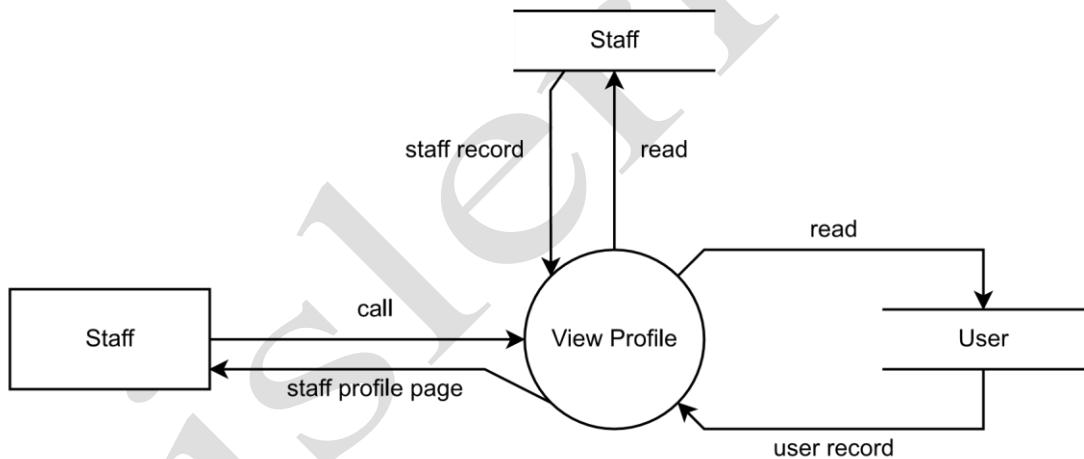
7.8.5.19. UC20

Figure 131: Level-1 DFD, UC20

7.8.5.20. UC21

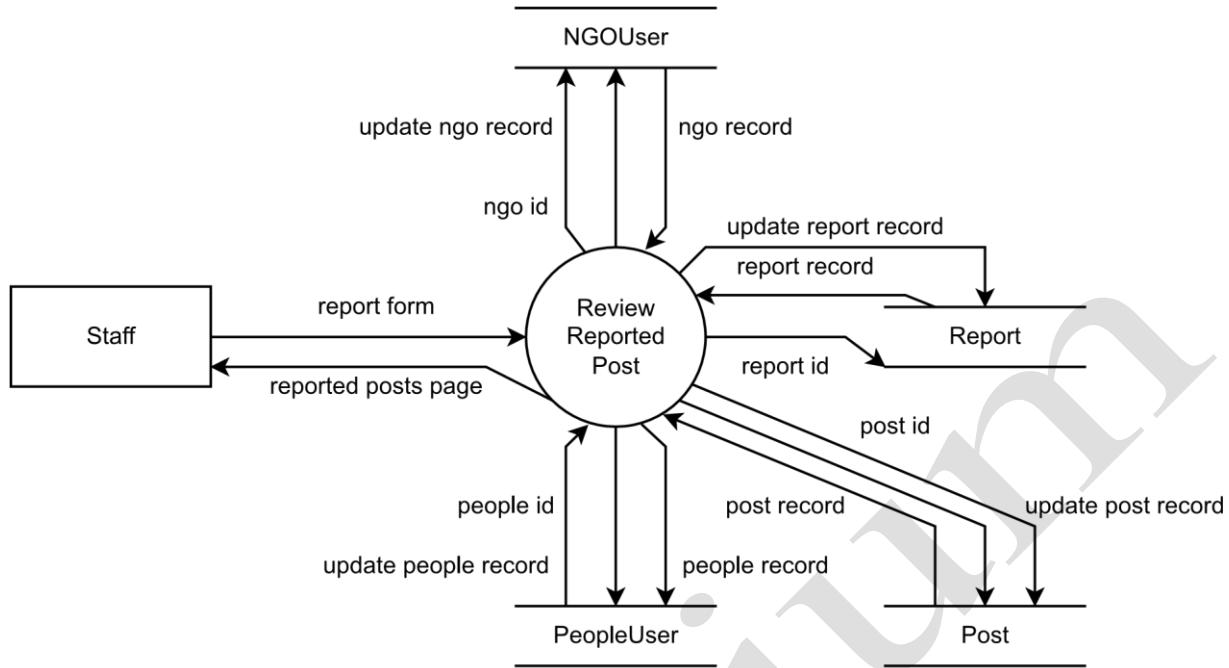


Figure 132: Level-1 DFD, UC21

7.8.5.21. UC22

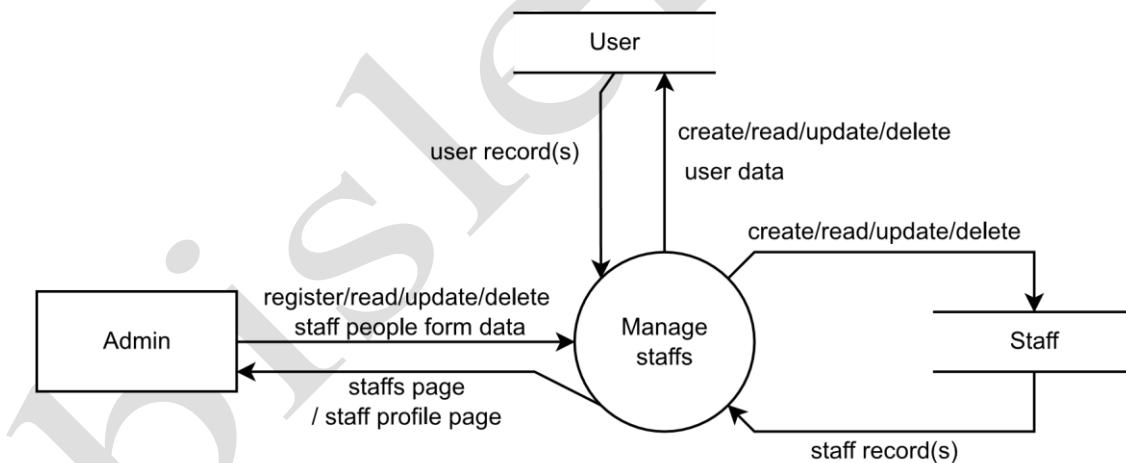


Figure 133: Level-1 DFD, UC22

7.8.5.22. UC23

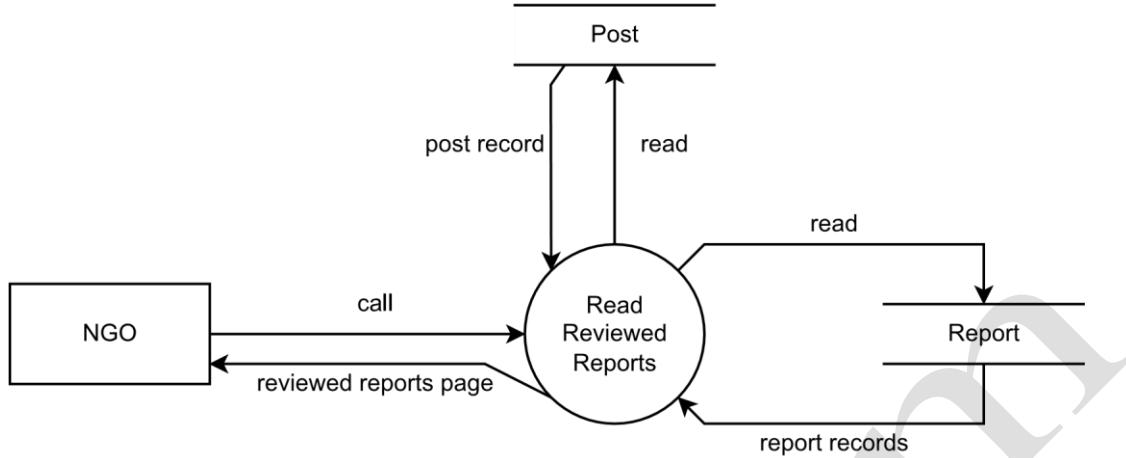


Figure 134: Level-1 DFD, UC23

7.8.6. USE CASES

7.8.6.1. HIGH-LEVEL USE CASE DESCRIPTION

7.8.6.1.1. UC1

Use Case	Log in
Actor(s)	User
Description	A registered user (i.e., Mobile User, Web User) can log in to access the system (i.e., mobile app and Admin-staff portal) and takes benefit of its services and features.

Table 40: High-Level UCI Description

7.8.6.1.2. UC2

Use Case	Log out
Actor(s)	User
Description	A logged-in user can log out from the system and needs to re-login next time if system accessibility is needed.

Table 41: High-Level UC2 Description

7.8.6.1.3. UC3

Use Case	Change Password
Actor(s)	User
Description	A logged-in user can change their password by entering an old password and a new password.

*Table 42: High-Level UC3 Description***7.8.6.1.4. UC4**

Use Case	Reset Password
Actor(s)	User
Description	A user who has forgotten their password can reset the password via an email sent after entering the email address associated with the account.

*Table 43: High-Level UC4 Description***7.8.6.1.5. UC5**

Use Case	View NGOs
Actor(s)	Mobile User
Description	A logged-in mobile user (i.e., NGO, General People) can view a list of NGOs and accesses their information.

*Table 44: High-Level UC5 Description***7.8.6.1.6. UC6**

Use Case	Manage Normal Post
Actor(s)	Mobile User
Description	A logged-in mobile user can create a normal post and then view, update and deletes it.

Table 45: High-Level UC6 Description

7.8.6.1.7. UC7

Use Case	Manage Poll Post
Actor(s)	Mobile User
Description	A logged-in mobile user can create a poll post and then view, update and deletes it.

*Table 46: High-Level UC7 Description***7.8.6.1.8. UC8**

Use Case	Manage Request Post
Actor(s)	Mobile User
Description	A logged-in mobile user can create a request post and then view, update and deletes it.

*Table 47: High-Level UC8 Description***7.8.6.1.9. UC9**

Use Case	View Posts
Actor(s)	Mobile User
Description	A logged-in mobile user can view a list of posts and accesses the provided information per post.

*Table 48: High-Level UC9 Description***7.8.6.1.10. UC10**

Use Case	Manage General Peoples
Actor(s)	Web User
Description	A logged-in web user can view, update, or/and delete registered General People from the General People user list

*Table 49: High-Level UC10 Description***7.8.6.1.11. UC11**

Use Case	Manage NGOs
Actor(s)	Web User

Description	A logged-in web user can register an NGO and view, update, or/and delete it or others from the NGO user list.
--------------------	---------------------------------------------------------------------------------------------------------------

*Table 50: High-Level UC11 Description***7.8.6.1.12. UC12**

Use Case	View Dashboard
Actor(s)	Web User
Description	A logged-in web user can view the dashboard that contains key information. The admin dashboard contains analytical information on the overall post and mobile user data. The staff dashboard contains the post report information.

*Table 51: High-Level UC12 Description***7.8.6.1.13. UC13**

Use Case	View Profile
Actor(s)	NGO
Description	A logged-in NGO user can view own profile.

*Table 52: High-Level UC13 Description***7.8.6.1.14. UC14**

Use Case	Donate to NGO
Actor(s)	General People
Description	A logged-in general people user can donate money to the registered NGO

*Table 53: High-Level UC14 Description***7.8.6.1.15. UC15**

Use Case	Manage Profile
Actor(s)	General People

Description	A general people user can register himself/herself and then view, update, and delete the profile.
--------------------	---------------------------------------------------------------------------------------------------

*Table 54: High-Level UC15 Description***7.8.6.1.16. UC16**

Use Case	React Normal Post
Actor(s)	General People
Description	A logged-in general people user can either upvote, downvote or undo react to a Normal Post.

*Table 55: High-Level UC16 Description***7.8.6.1.17. UC17**

Use Case	React Poll Post
Actor(s)	General People
Description	A logged-in general people user can poll the poll options in a Poll Post.

*Table 56: High-Level UC17 Description***7.8.6.1.18. UC18**

Use Case	React Request Post
Actor(s)	General People
Description	A logged-in general people user can sign the Petition Request Post or participate in Join Request Post.

*Table 57: High-Level UC18 Description***7.8.6.1.19. UC19**

Use Case	Report Post
Actor(s)	General People
Description	A logged-in general people can report post.

Table 58: High-Level UC19 Description

7.8.6.1.20. UC20

Use Case	View Profile
Actor(s)	Staff
Description	A logged-in staff user can view own profile.

*Table 59: High-Level UC20 Description***7.8.6.1.21. UC21**

Use Case	Review Reported Post
Actor(s)	Staff
Description	A logged-in staff user can review reported posts.

*Table 60: High-Level UC21 Description***7.8.6.1.22. UC22**

Use Case	Manage Staffs
Actor(s)	Admin
Description	A logged-in admin user can register and view, update, or/and delete it or others from the Staff user list.

*Table 61: High-Level UC22 Description***7.8.6.1.23. UC23**

Use Case	Read Reviewed Posts
Actor(s)	Admin
Description	A logged-in admin user can read reviewed posts from a list.

*Table 62: High-Level UC23 Description***7.8.7. COMMUNICATION/COLLABORATION DIAGRAMS****7.8.7.1. UC6**

SASAE

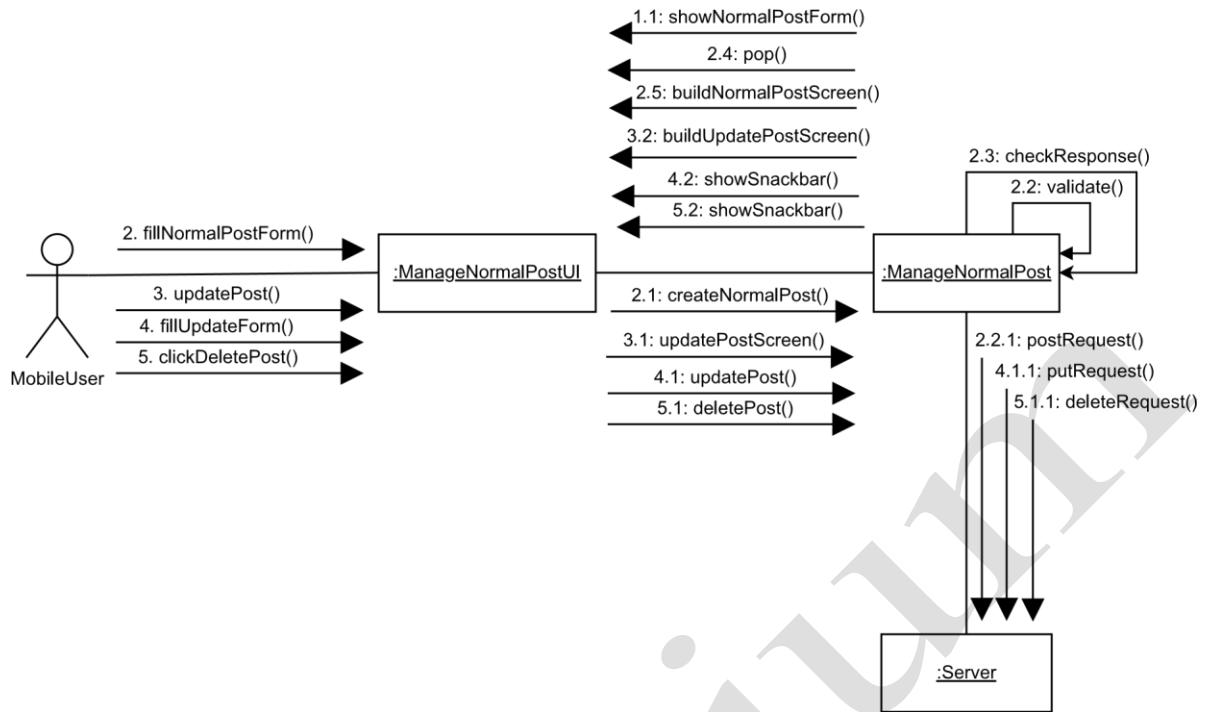


Figure 135: Communication/Collaboration Diagram – UC6

7.8.7.2. UC7

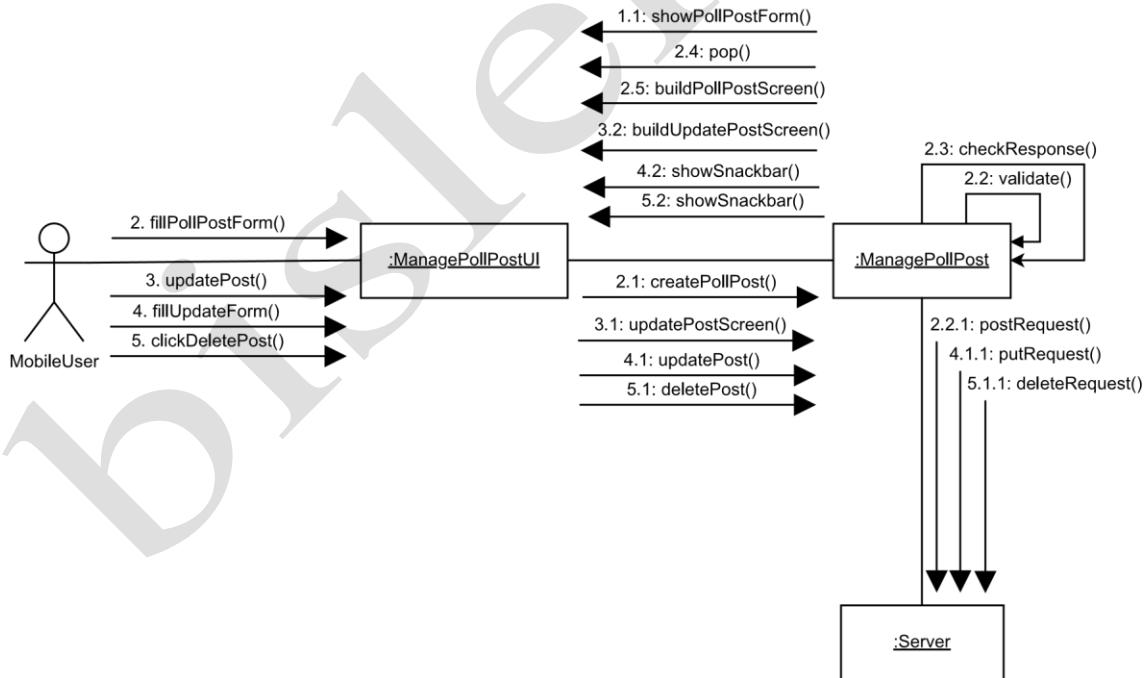


Figure 136: Communication/Collaboration Diagram – UC7

7.8.7.3. UC8

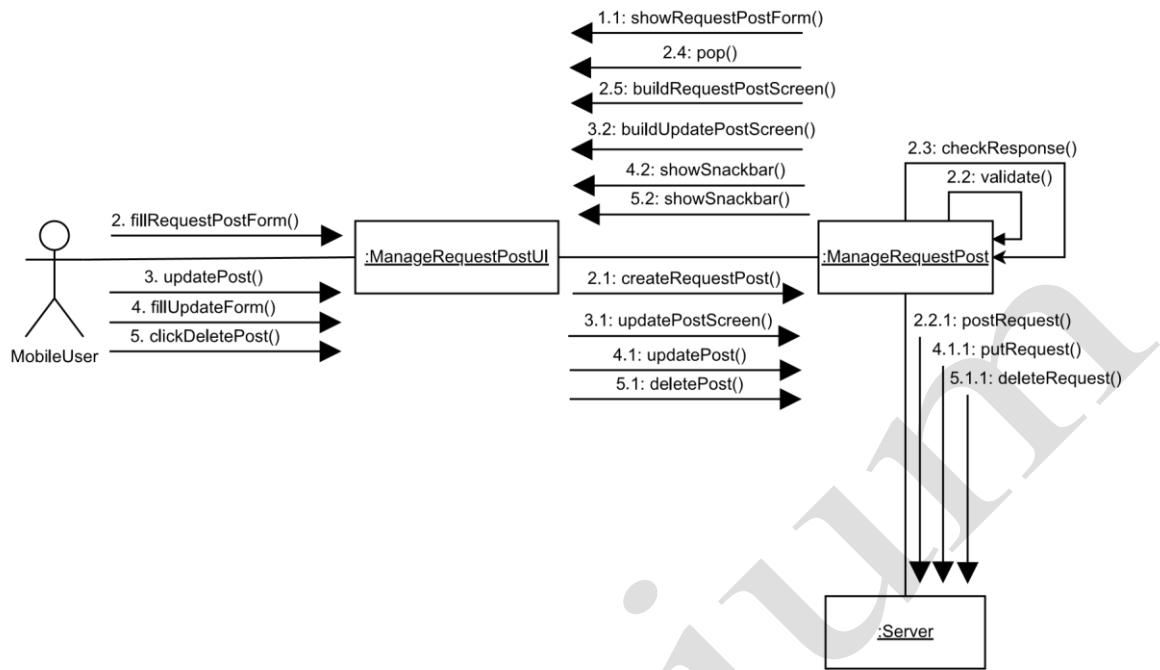


Figure 137: Communication/Collaboration Diagram – UC8

7.8.7.4. UC9

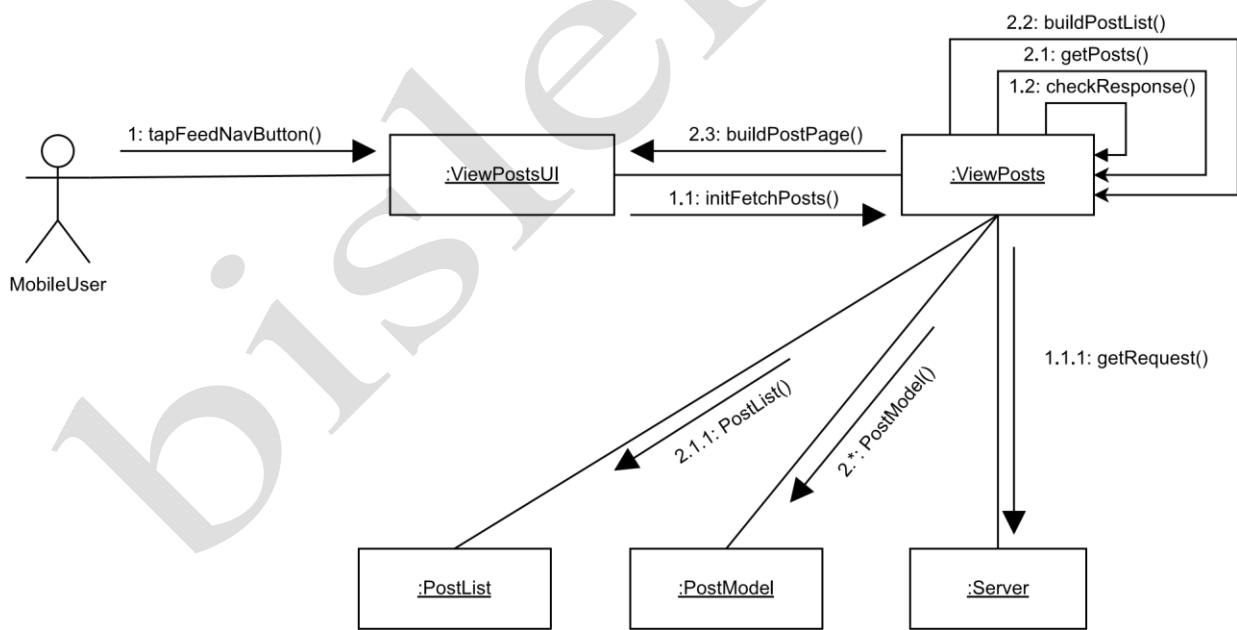


Figure 138: Communication/Collaboration Diagram – UC8

7.8.7.5. UC10

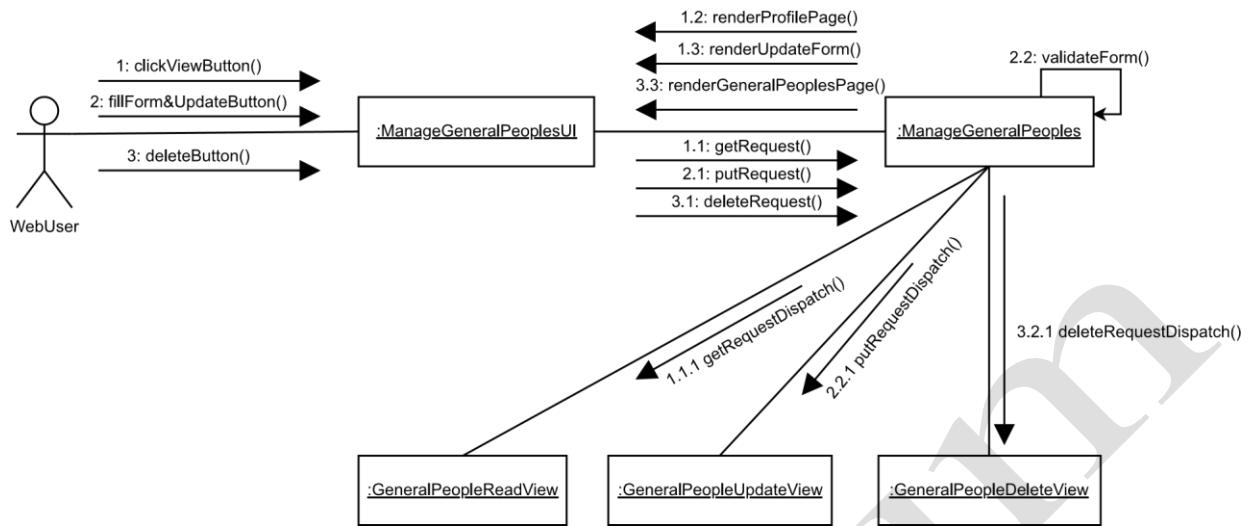


Figure 139: Communication/Collaboration Diagram – UC10

7.8.7.6. UC11

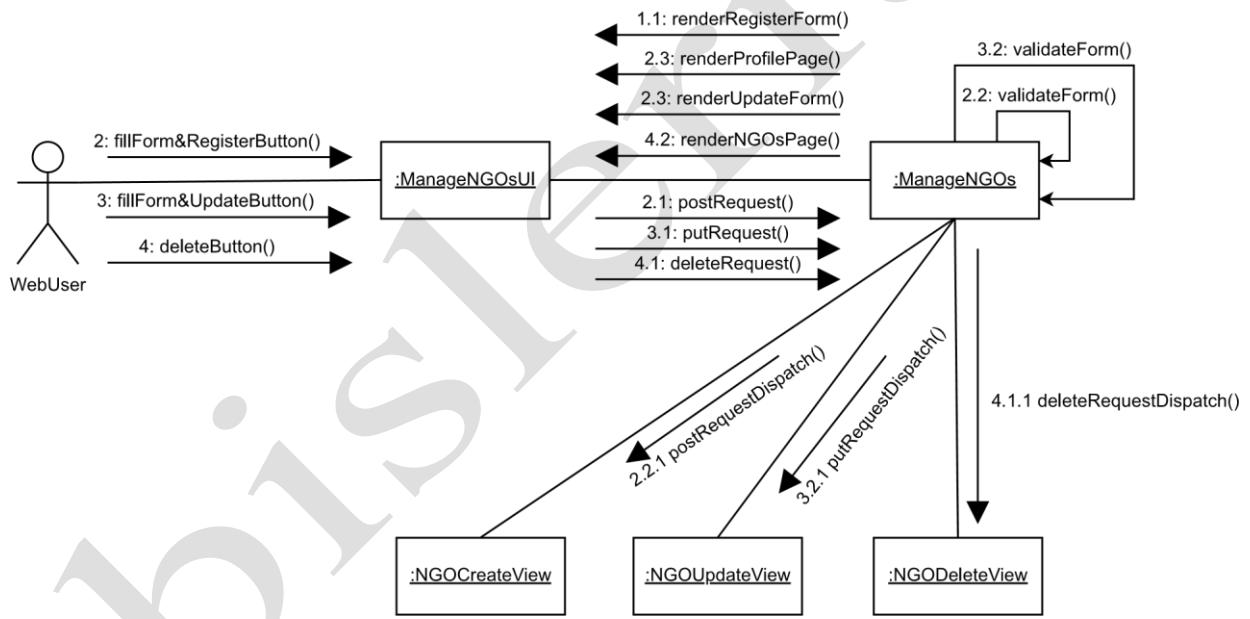


Figure 140: Communication/Collaboration Diagram – UC11

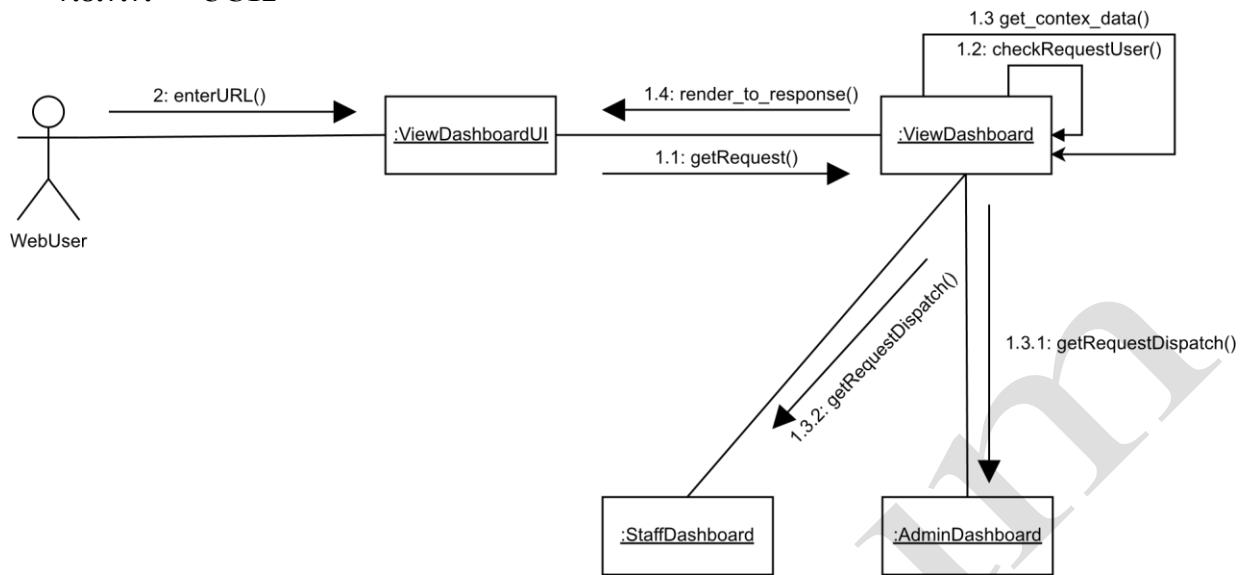
7.8.7.7. UC12

Figure 141: Communication/Collaboration Diagram – UC12

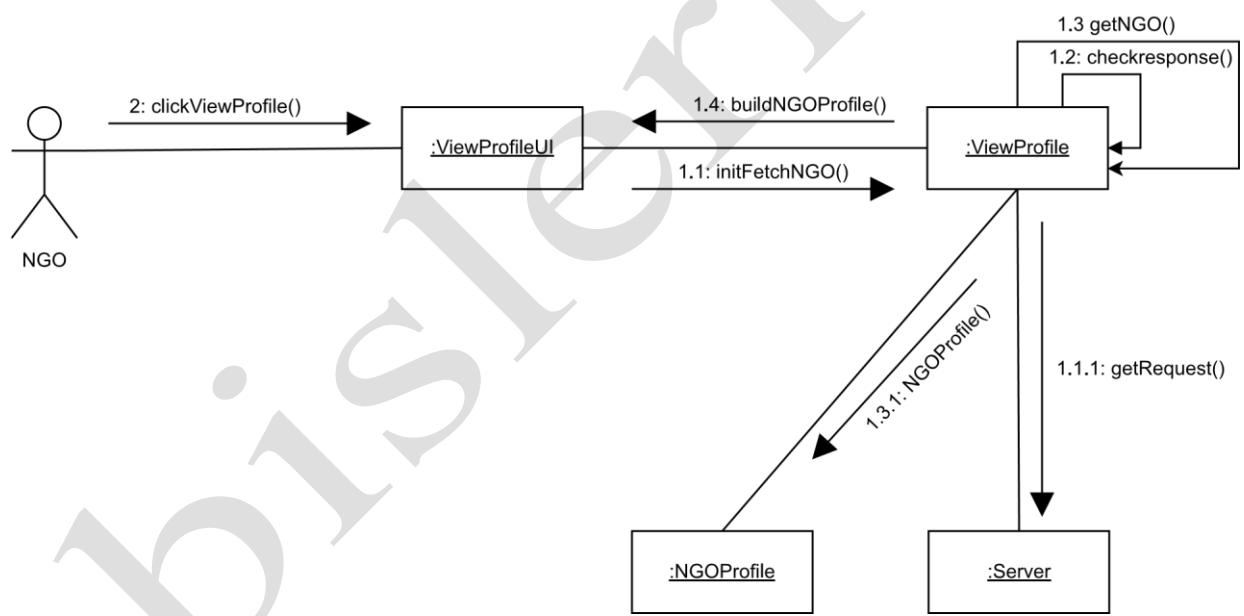
7.8.7.8. UC13

Figure 142: Communication/Collaboration Diagram – UC13

7.8.7.9. UC14

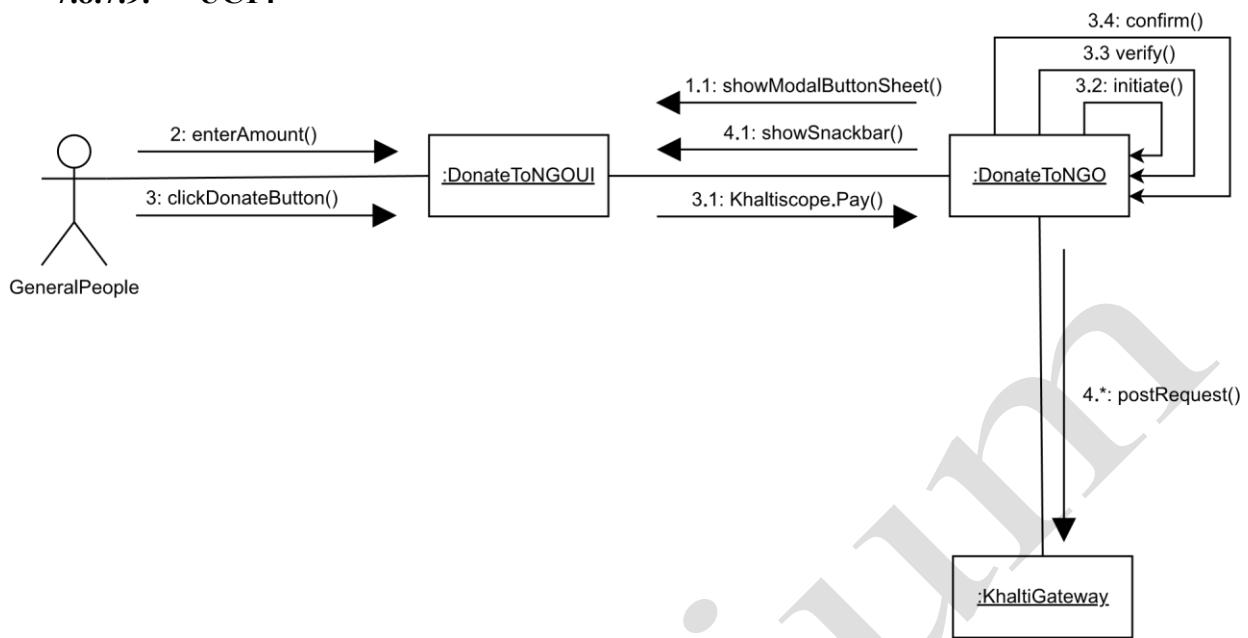


Figure 143: Communication/Collaboration Diagram – UC14

7.8.7.10. UC15

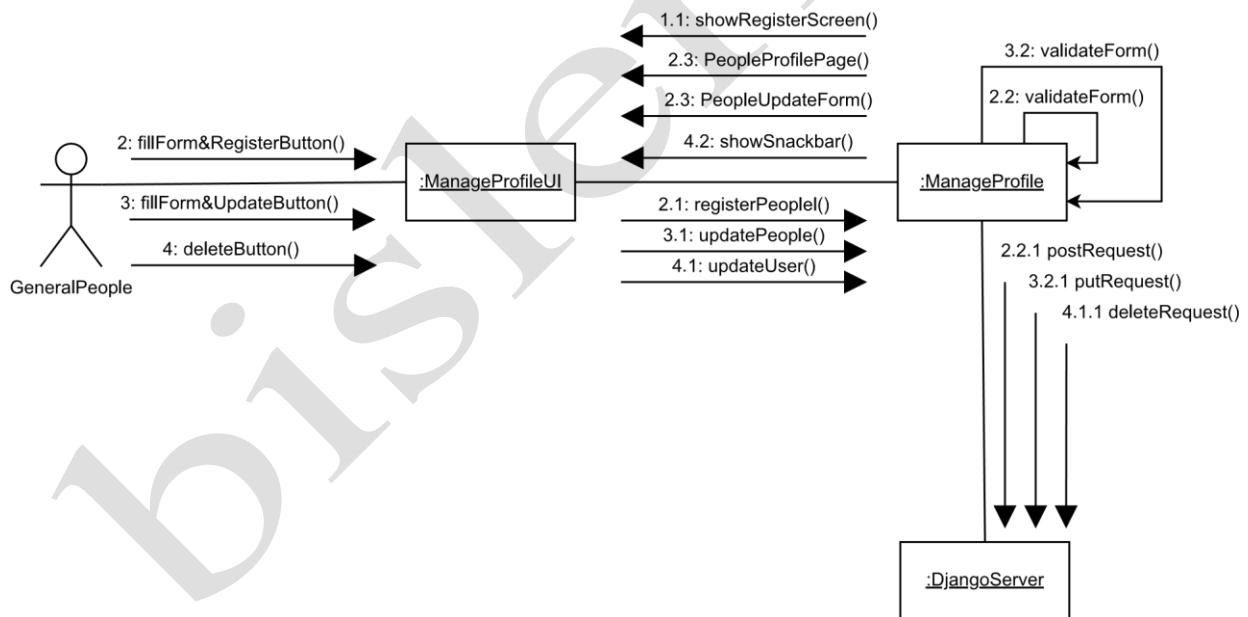


Figure 144: Communication/Collaboration Diagram – UC15

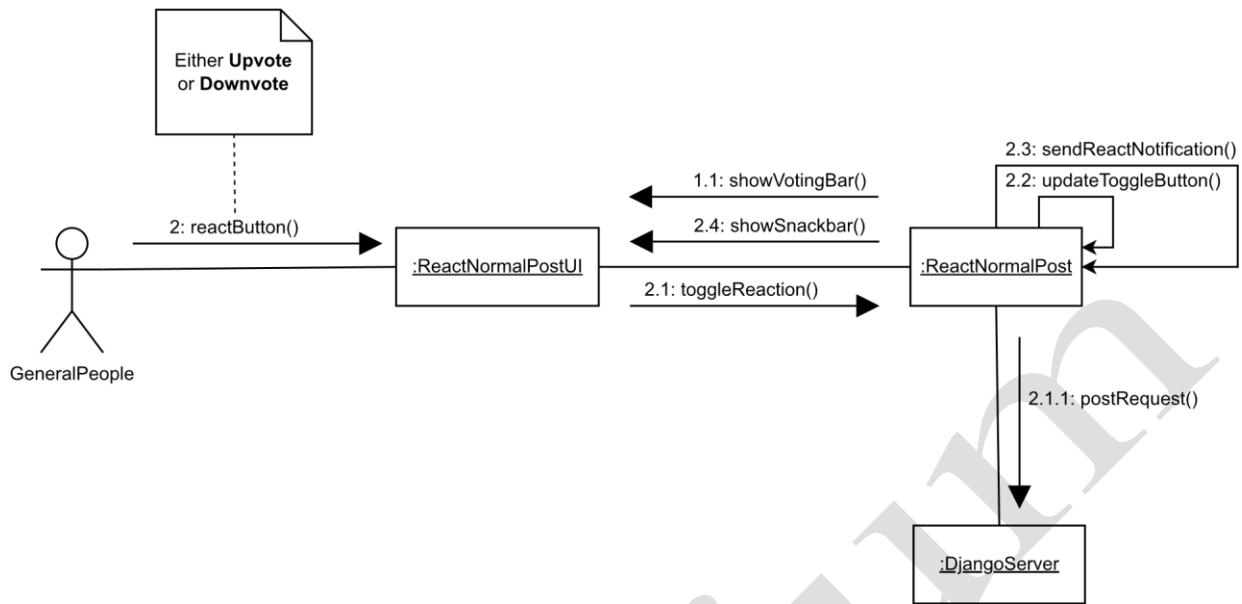
7.8.7.11. UC16

Figure 145: Communication/Collaboration Diagram – UC16

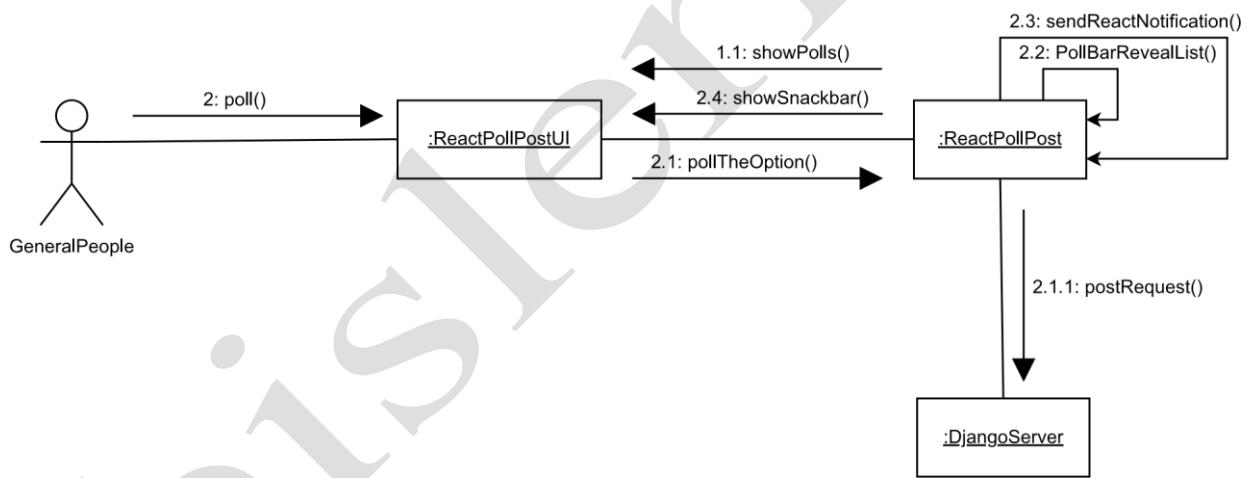
7.8.7.12. UC17

Figure 146: Communication/Collaboration Diagram – UC17

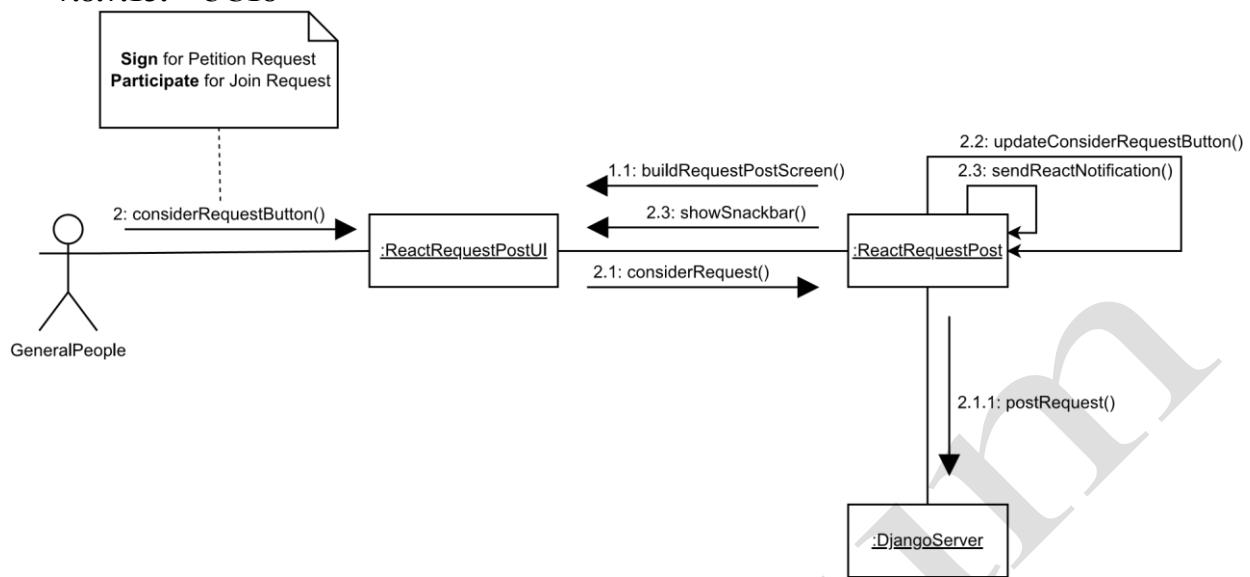
7.8.7.13. UC18

Figure 147: Communication/Collaboration Diagram – UC18

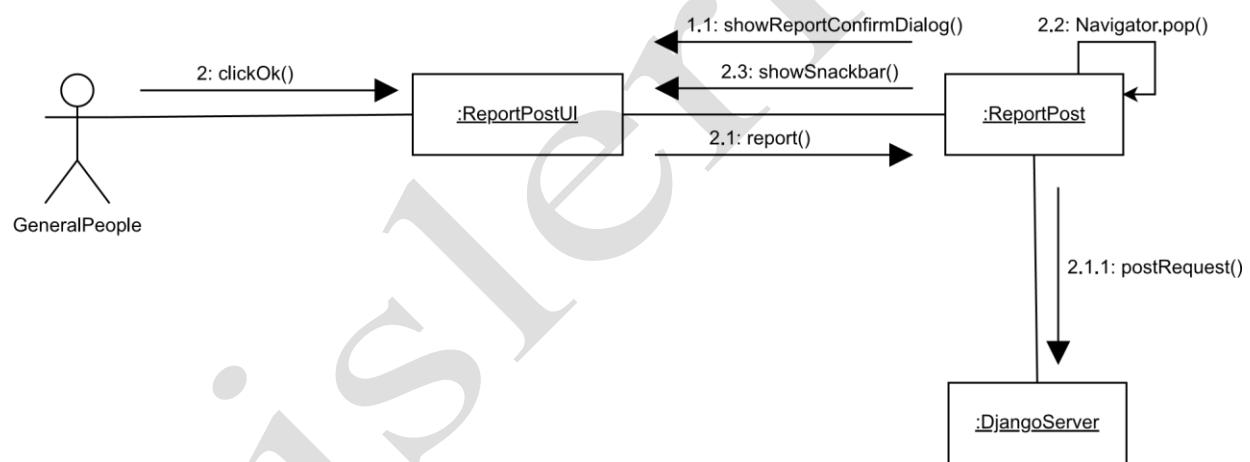
7.8.7.14. UC19

Figure 148: Communication/Collaboration Diagram – UC19

7.8.7.15. UC20

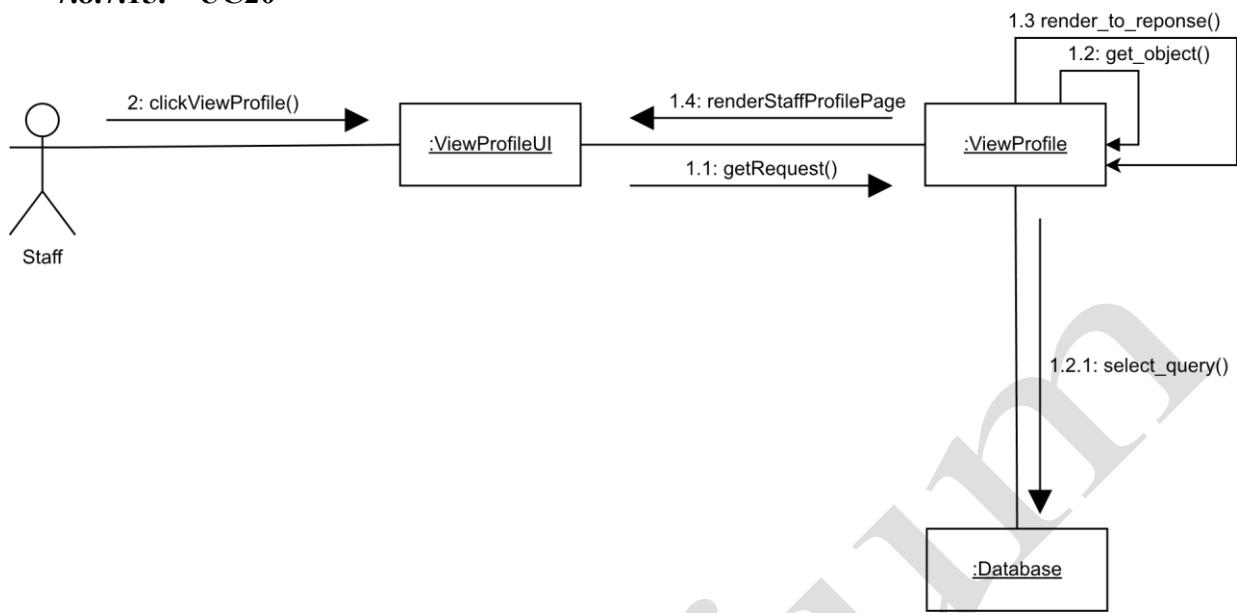


Figure 149: Communication/Collaboration Diagram – UC20

7.8.7.16. UC21

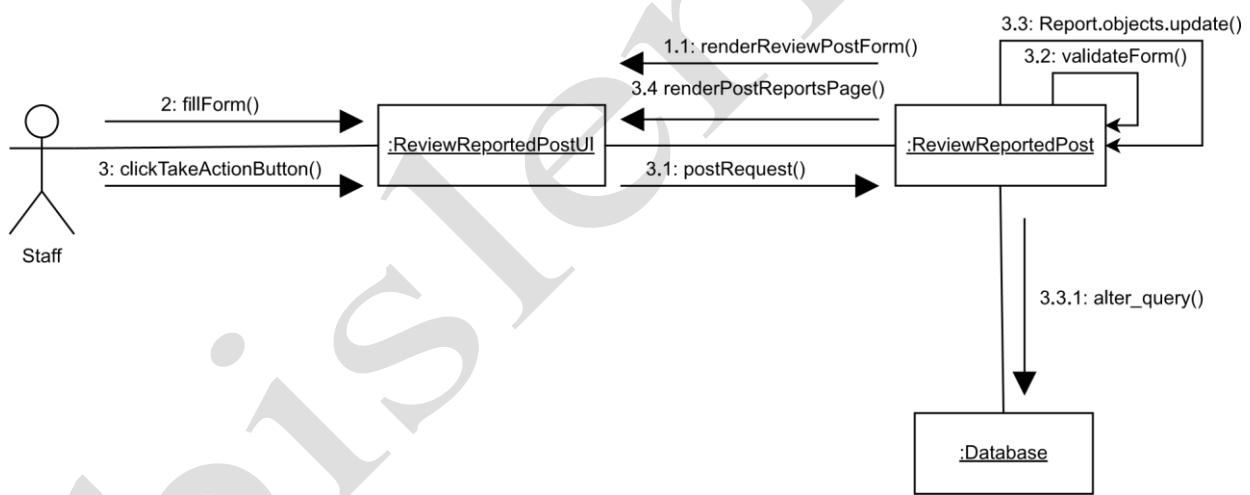


Figure 150: Communication/Collaboration Diagram – UC21

7.8.7.17. UC22

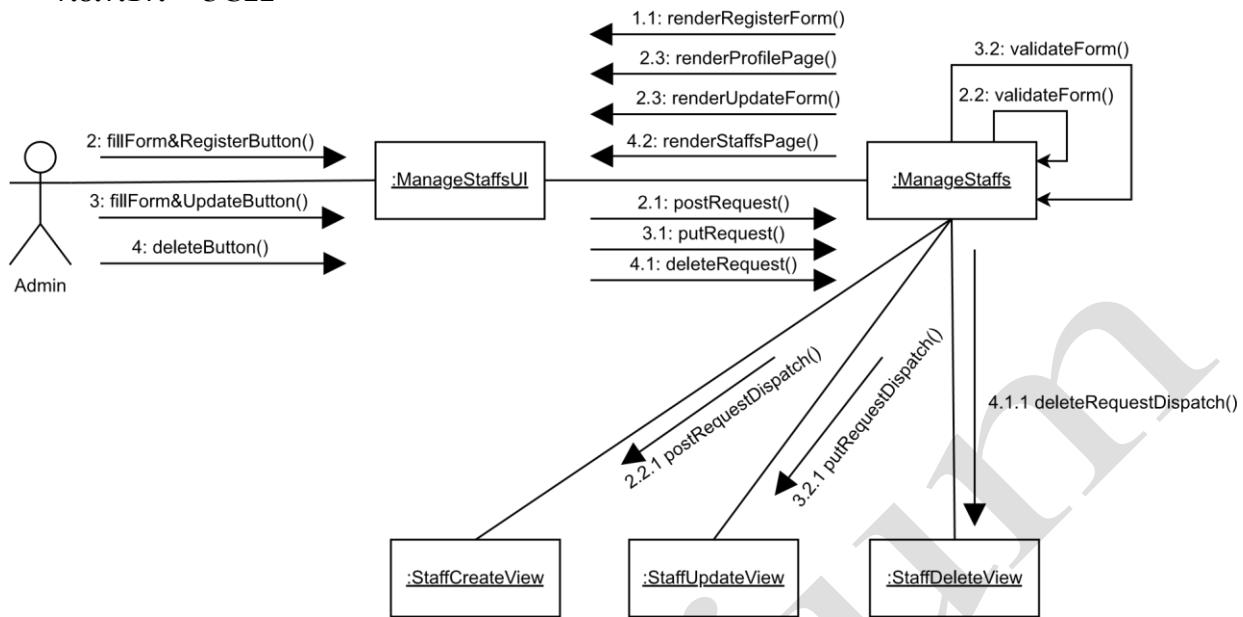


Figure 151: Communication/Collaboration Diagram – UC22

7.8.7.18. UC23

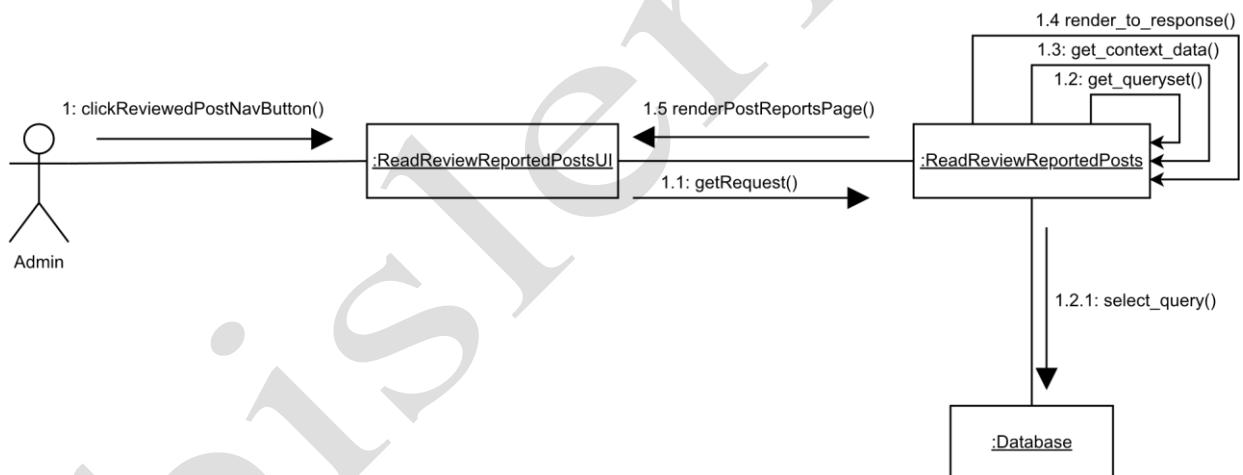


Figure 152: Communication/Collaboration Diagram – UC23

7.8.8. SEQUENCE DIAGRAMS

7.8.8.1. UC6

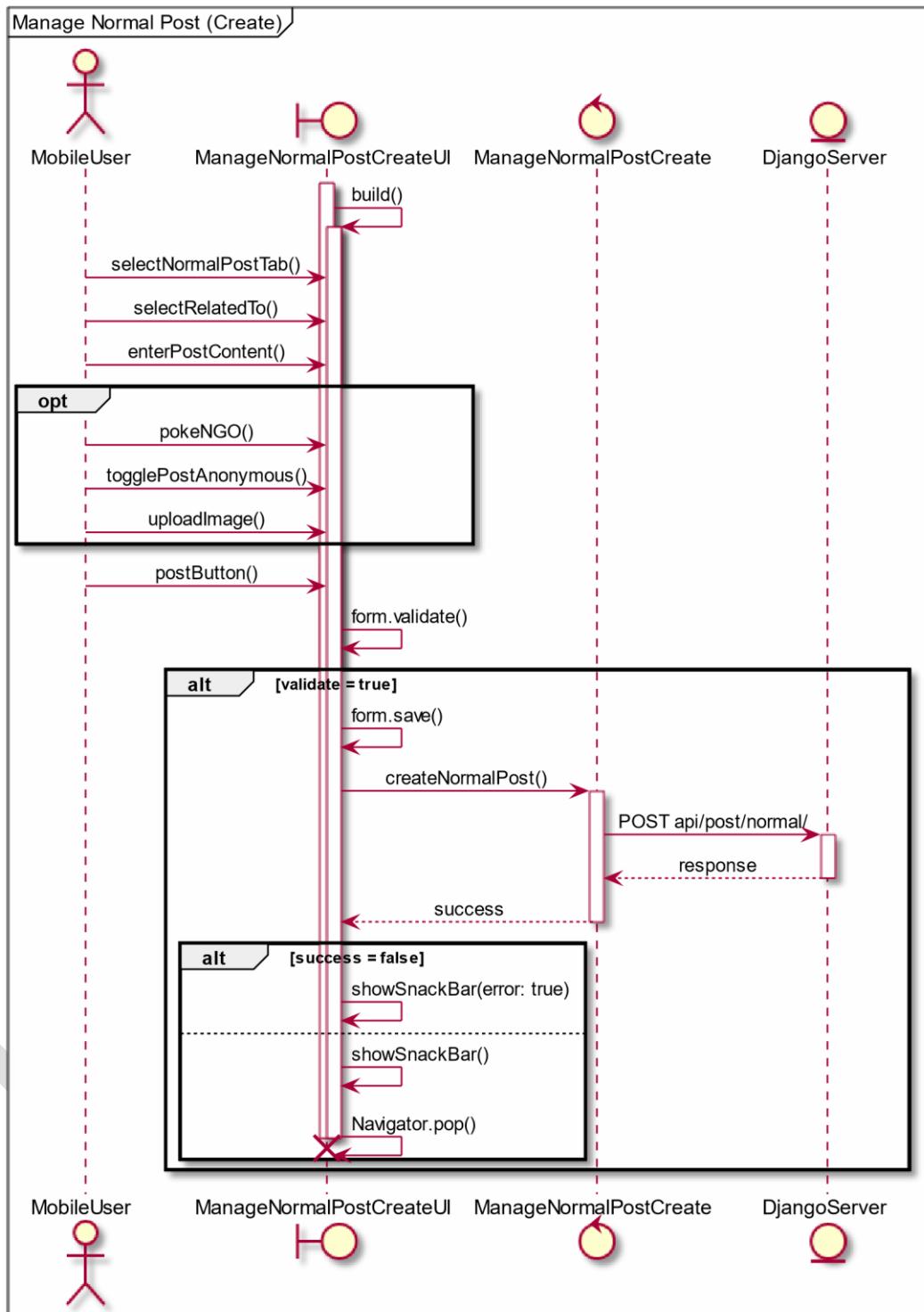


Figure 153: Sequence Diagram - UC6 (Create)

SASAE

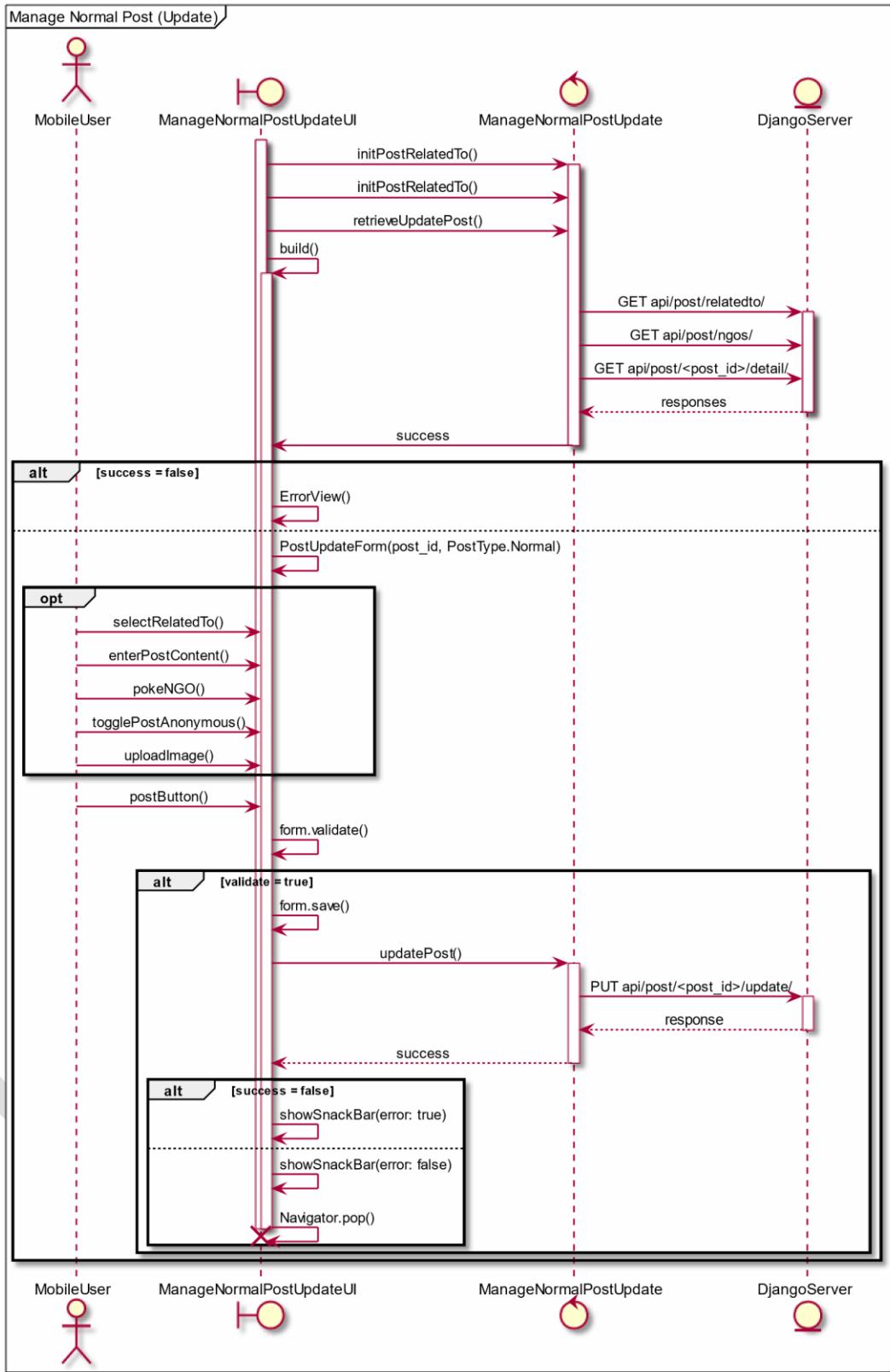


Figure 154: Sequence Diagram - UC6 (Update)

SASAE

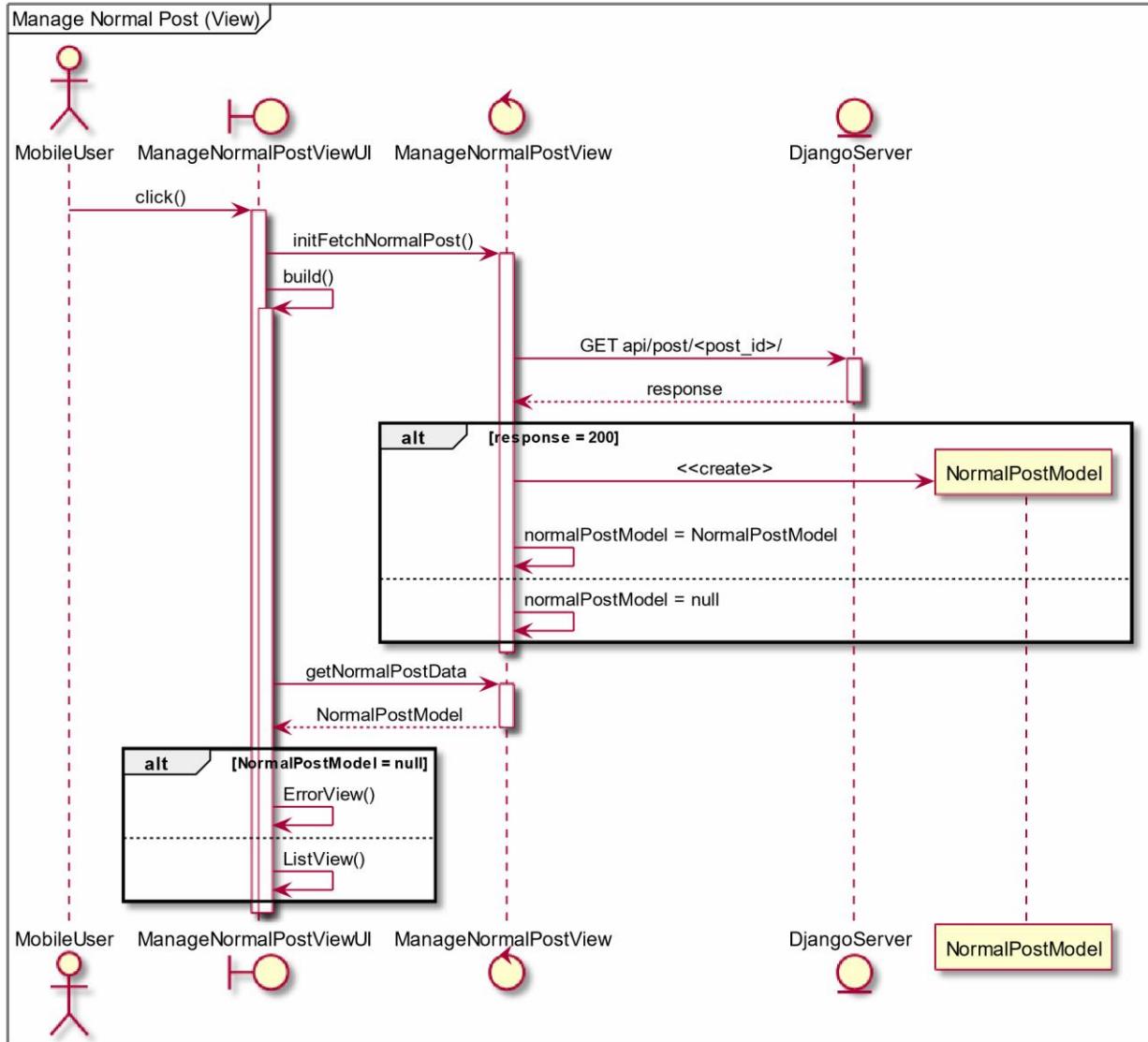
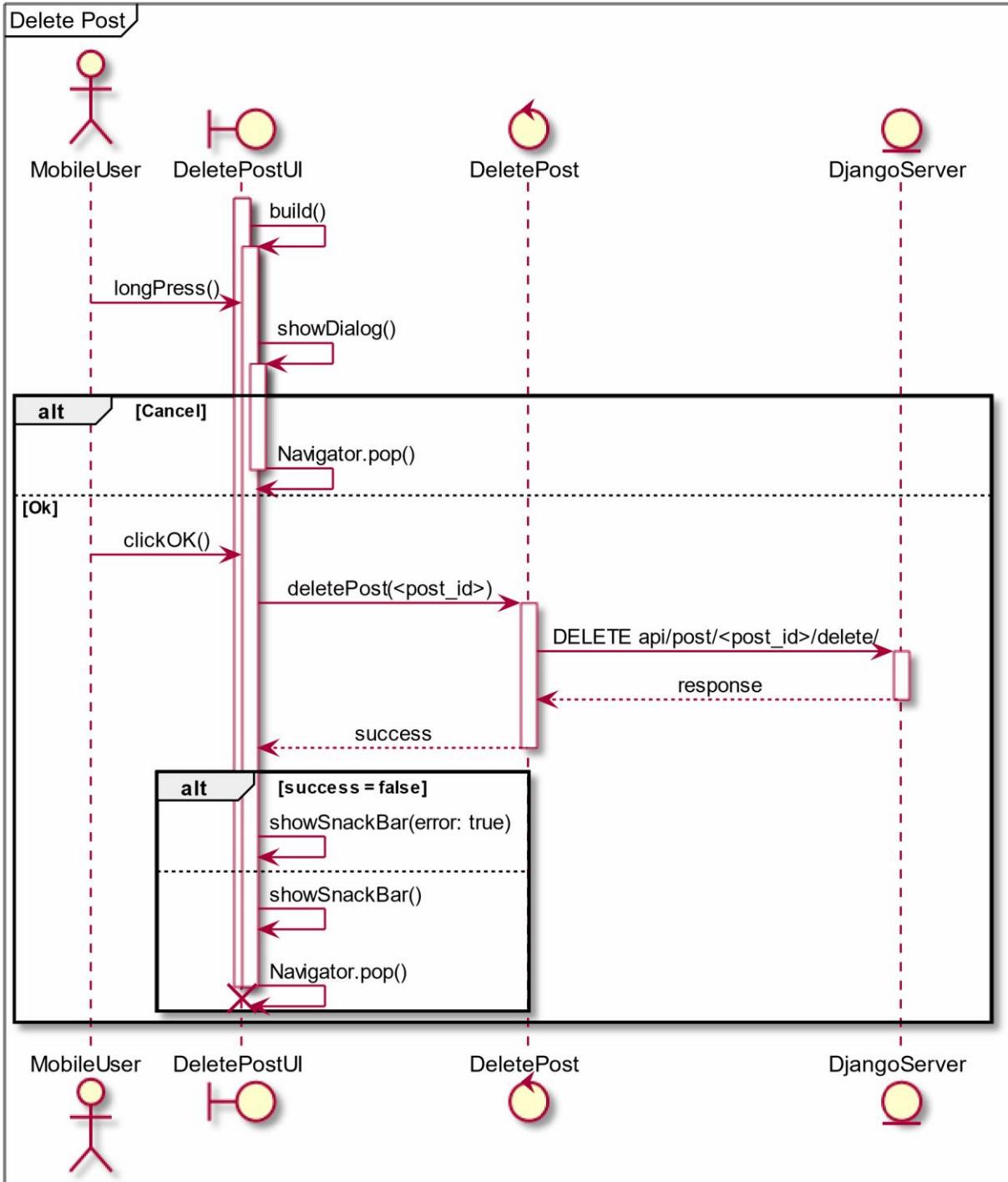


Figure 155: Sequence Diagram – UC6 (View)

Figure 156: Sequence Diagram – UC6 (Delete) *applies to all kinds of post types.*

7.8.8.2. UC7

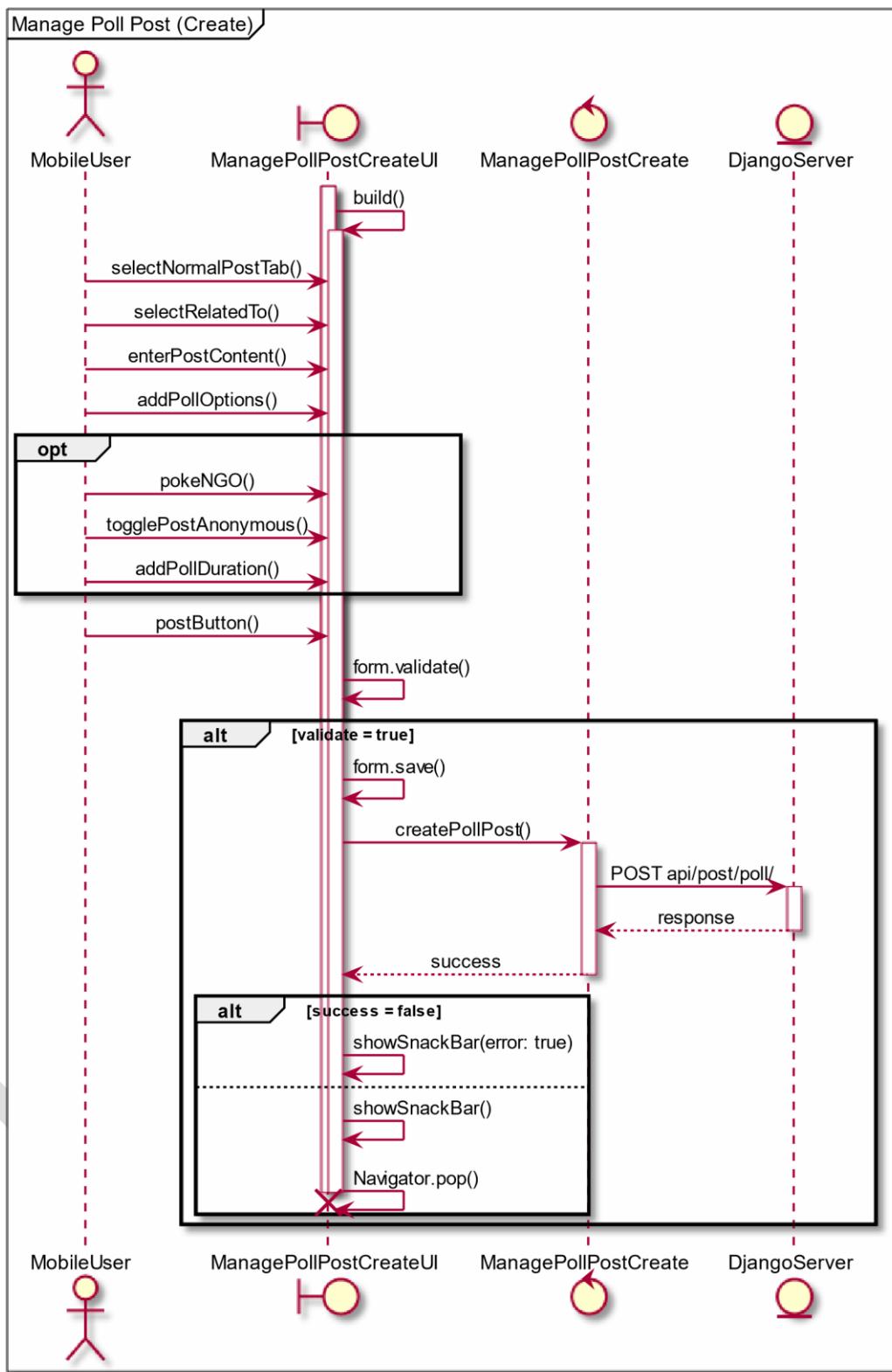


Figure 157: Sequence Diagram – UC7 (Create)

SASAE

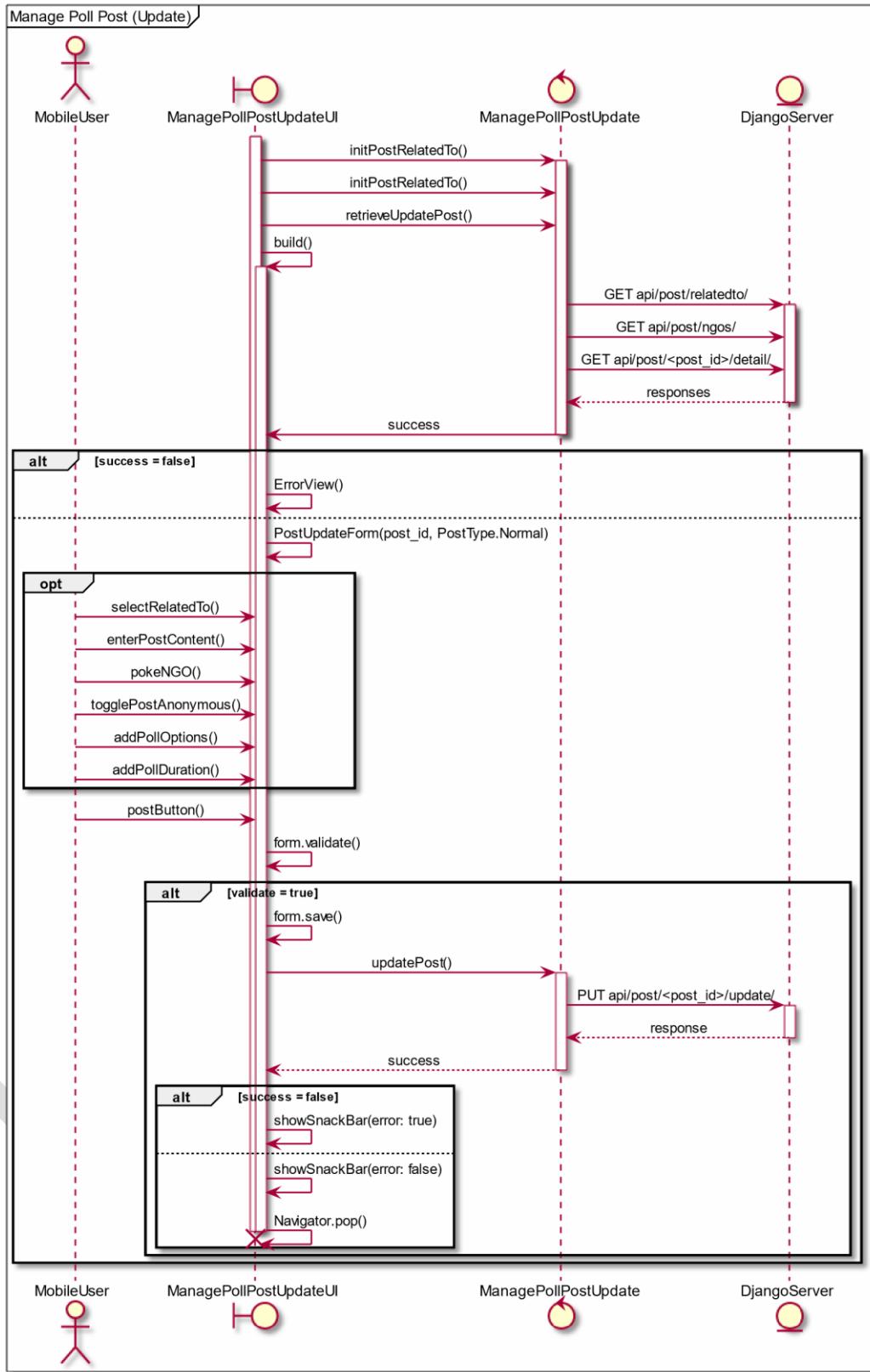


Figure 158: Sequence Diagram – UC7 (Update)

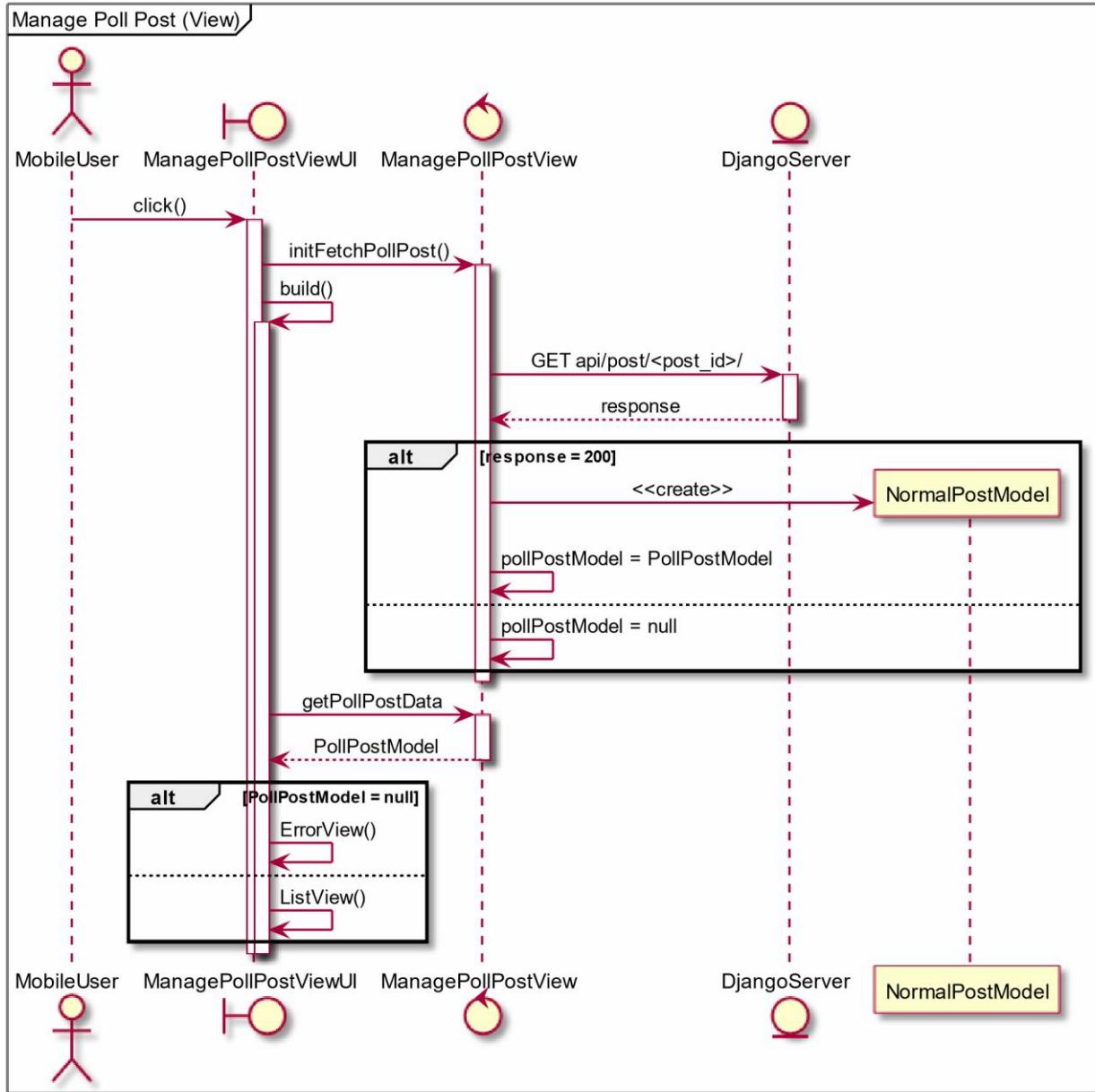


Figure 159: Sequence Diagram – UC7 (View)

7.8.8.3. UC8

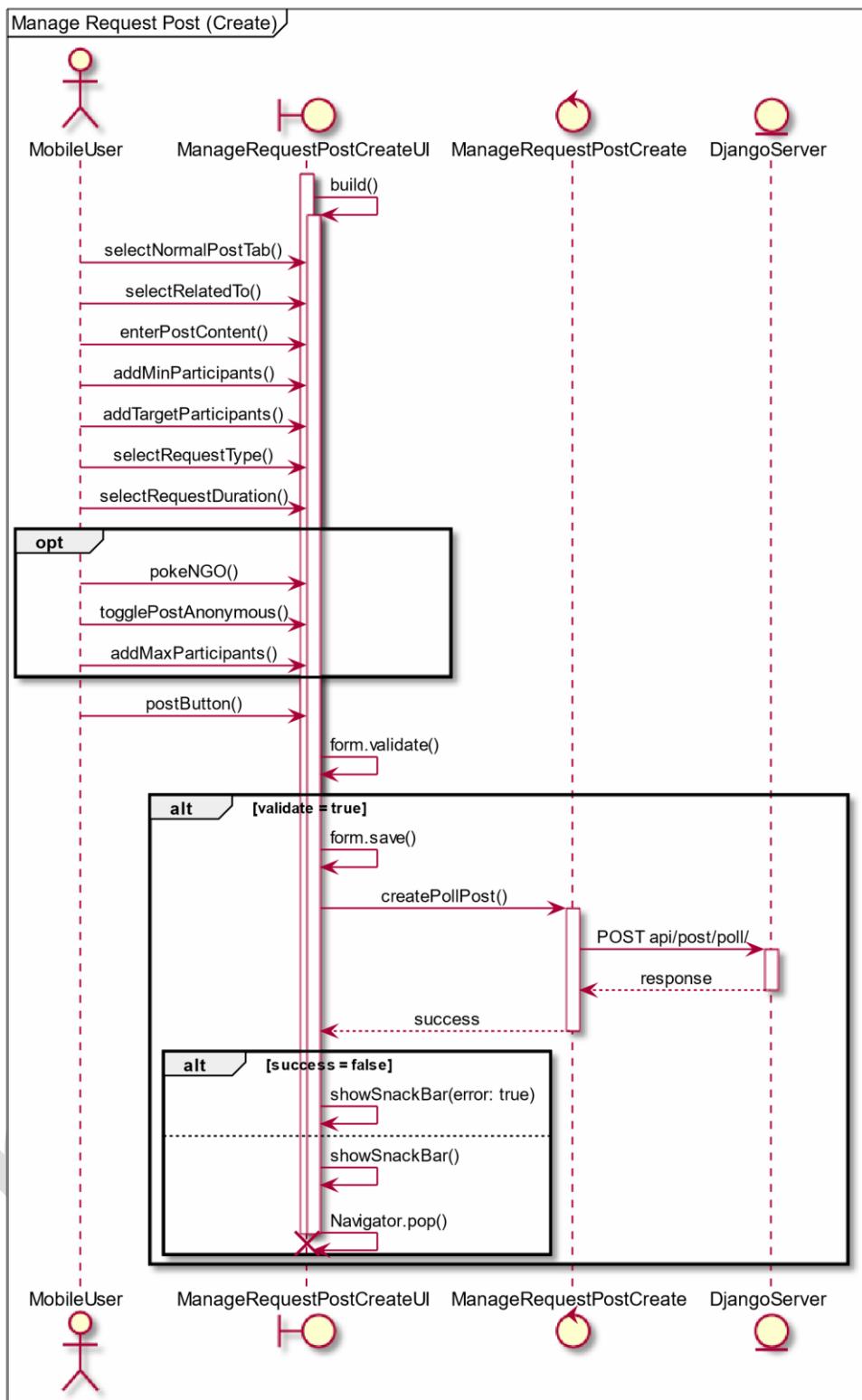


Figure 160: Sequence Diagram – UC8 (Create)

SASAE

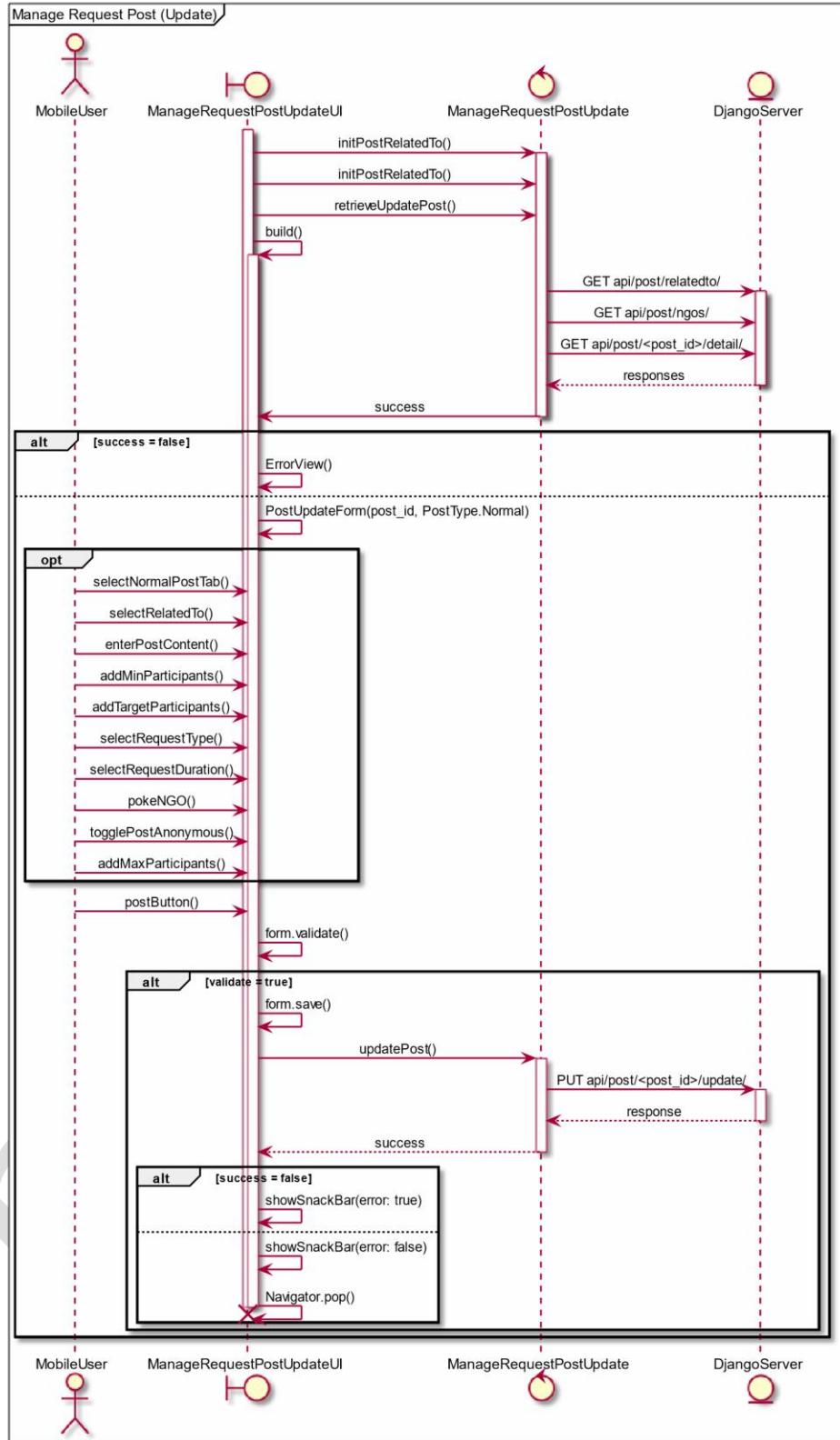


Figure 161: Sequence Diagram – UC8 (Update)

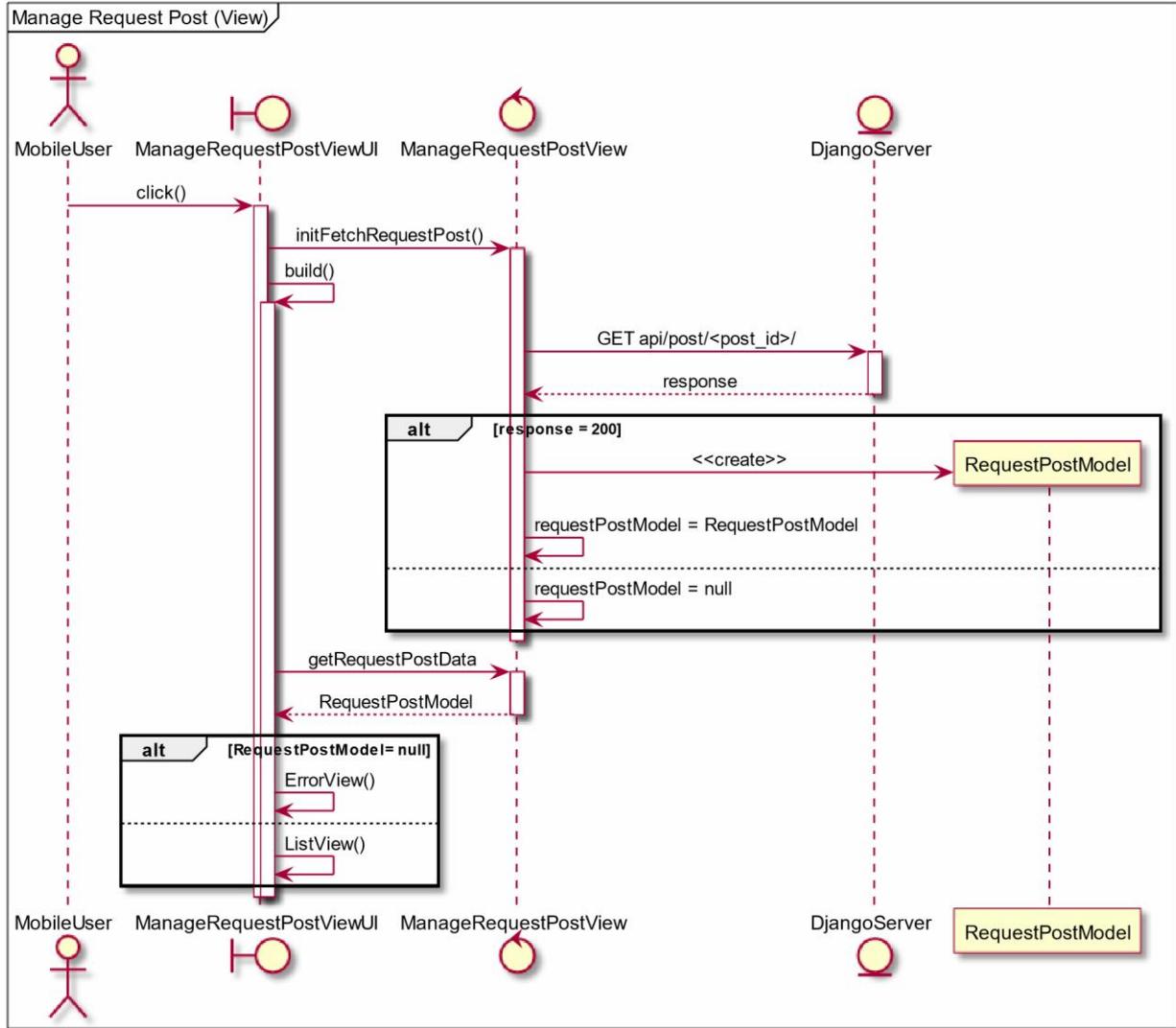


Figure 162: Sequence Diagram – UC8 (View)

7.8.8.4. UC9

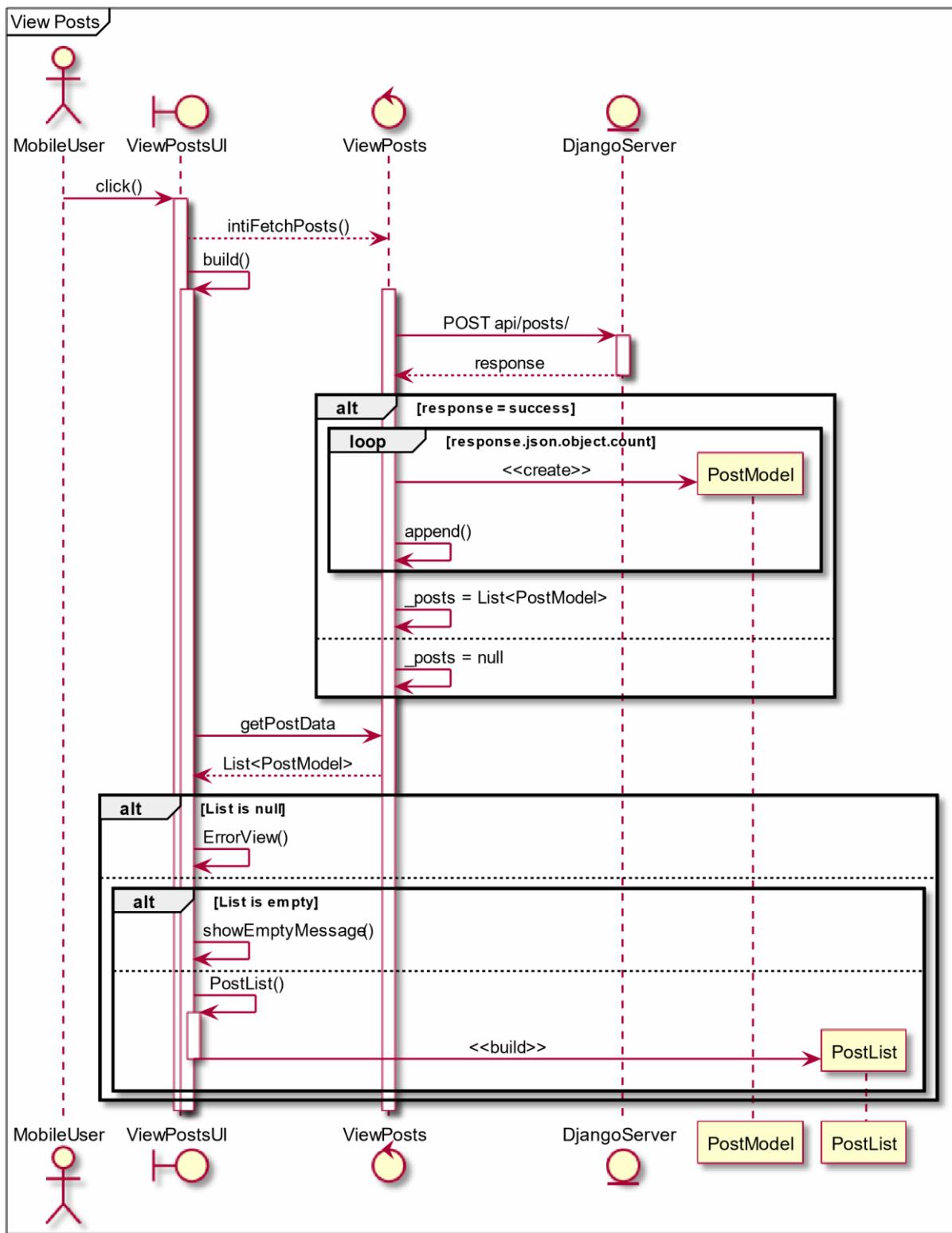


Figure 163: Sequence Diagram – UC9

7.8.8.5. UC10

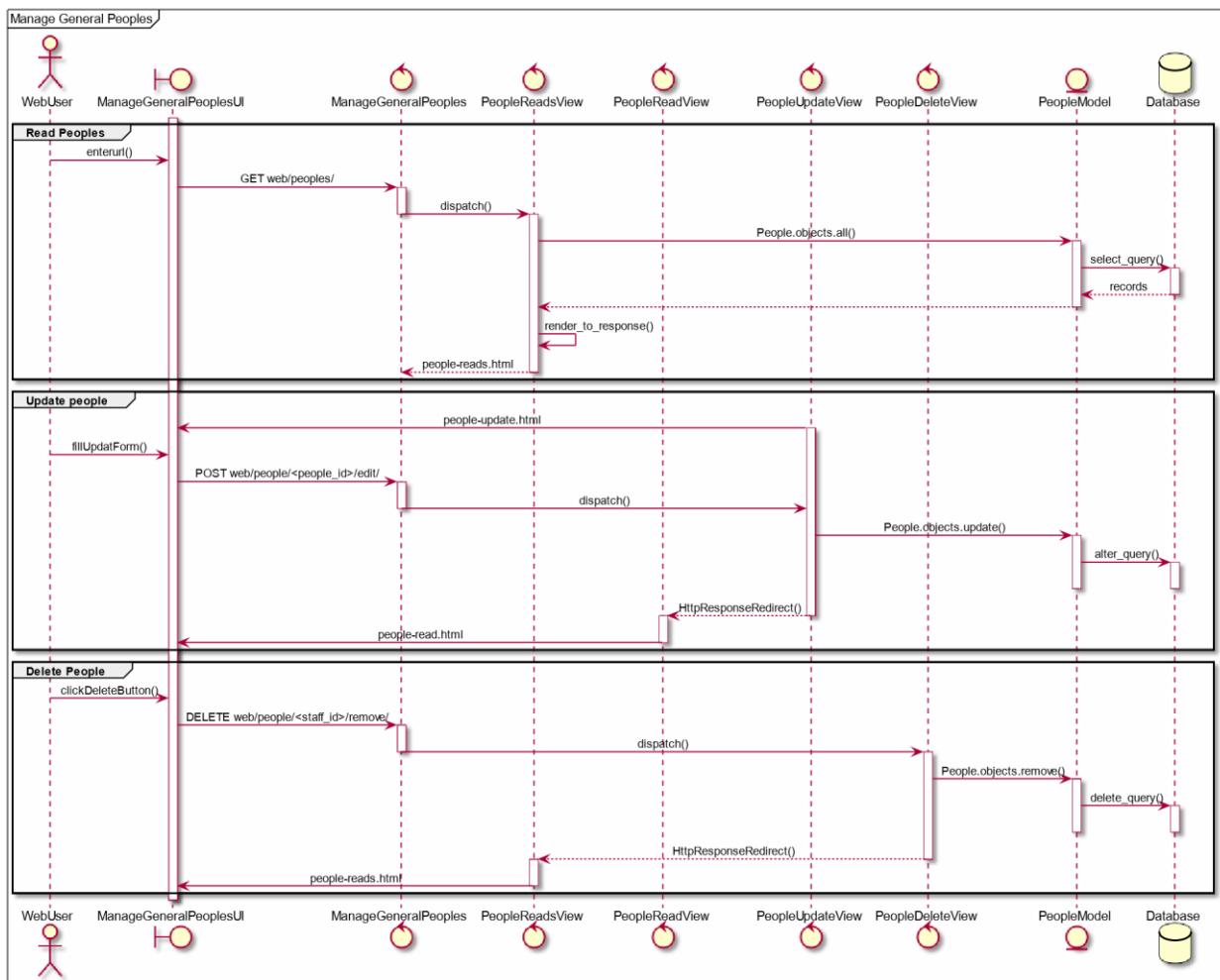


Figure 164: Sequence Diagram – UC10

7.8.8.6. UC11

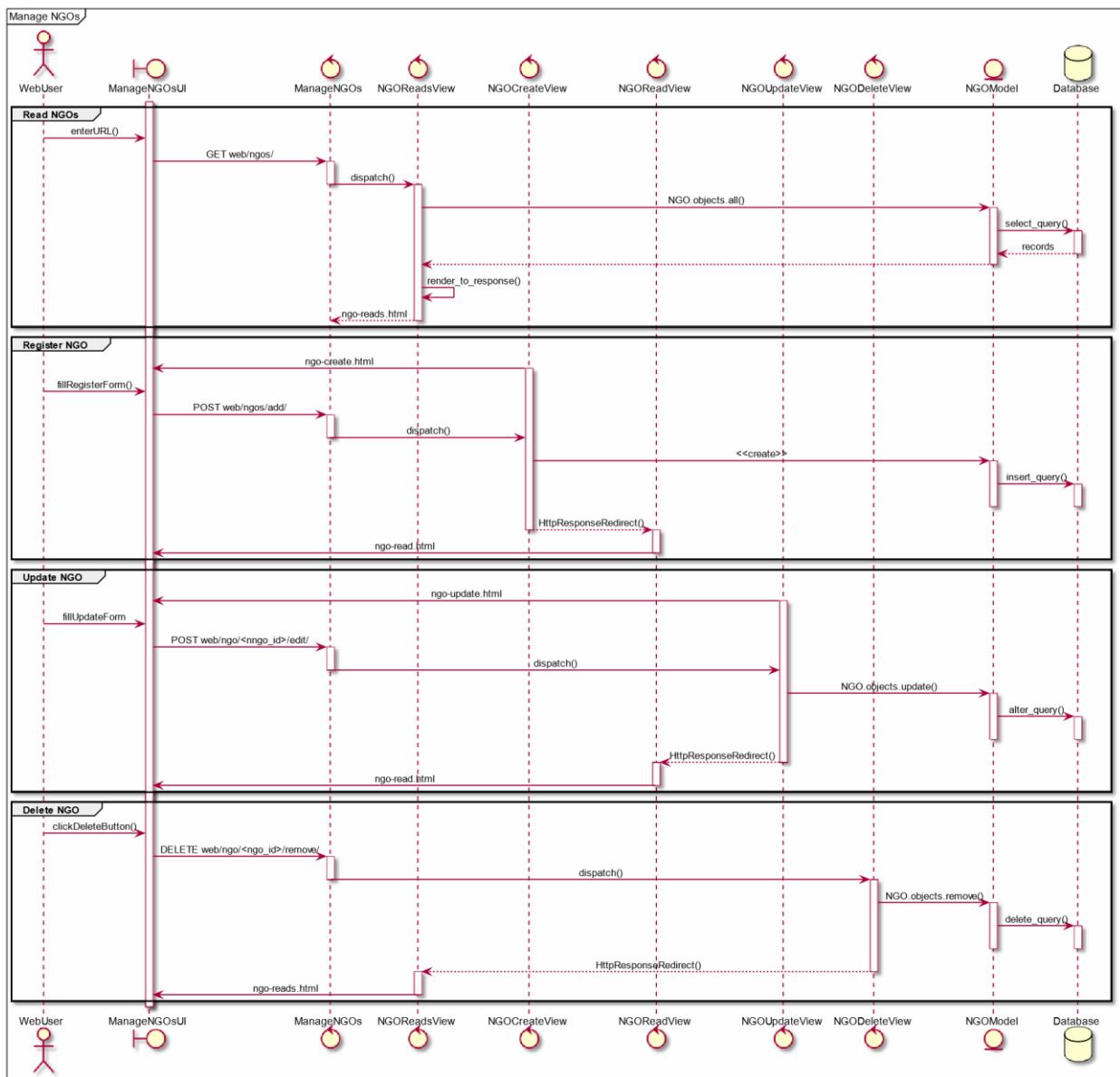


Figure 165: Sequence Diagram – UC11

7.8.8.7. UC12

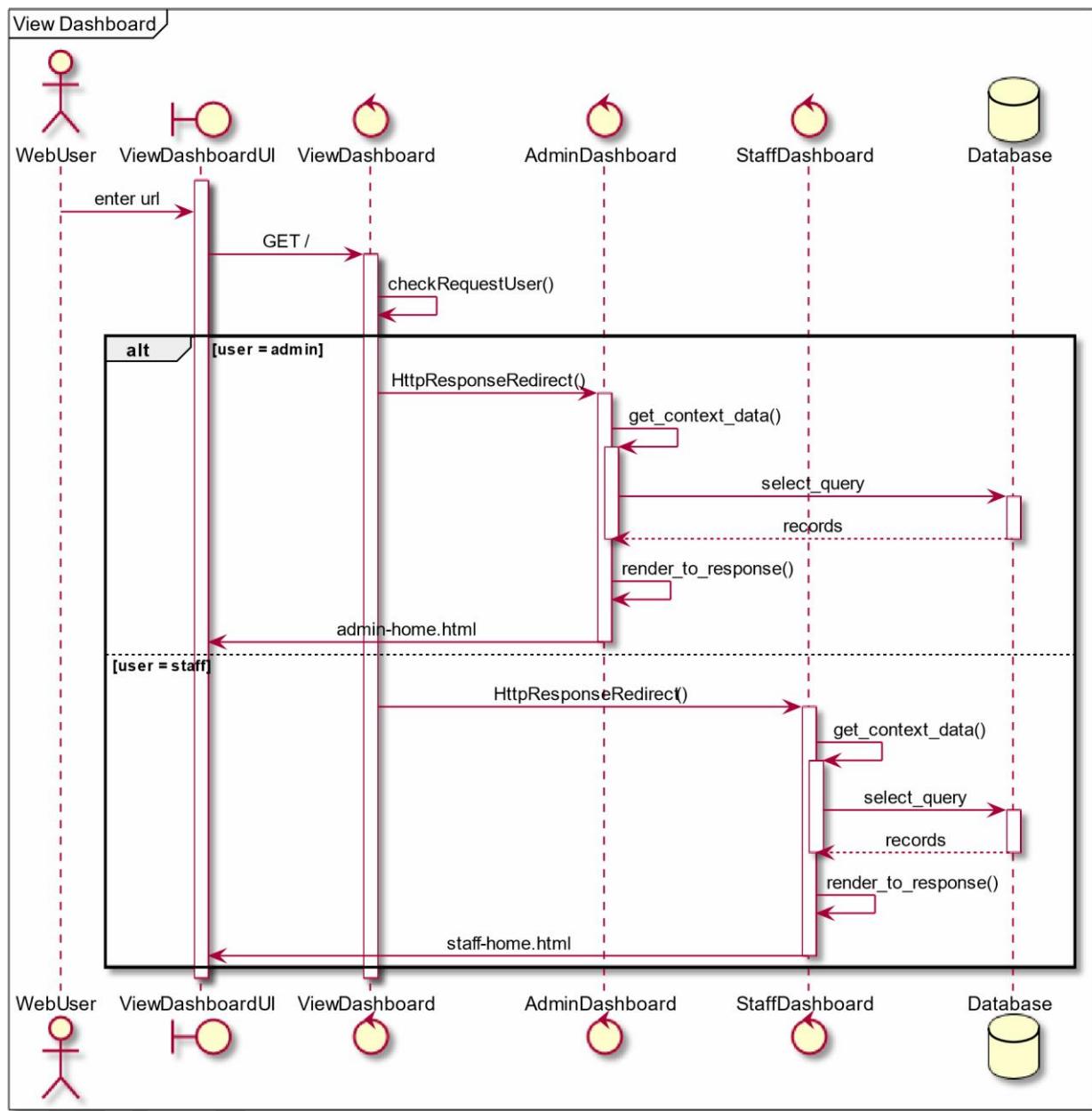


Figure 166: Sequence Diagram – UC12

7.8.8. UC13

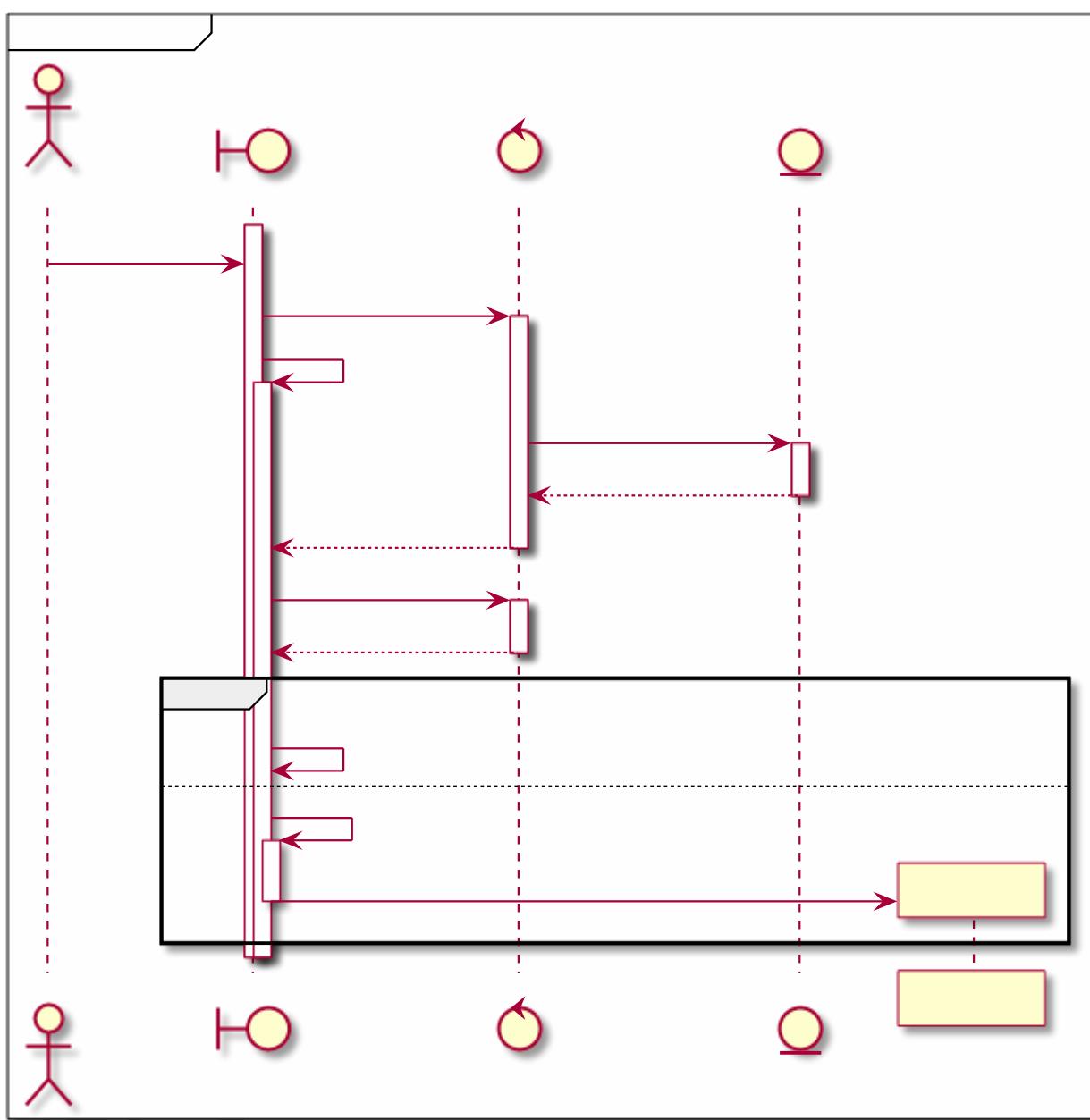


Figure 167: Sequence Diagram – UC13

7.8.8.9. UC14

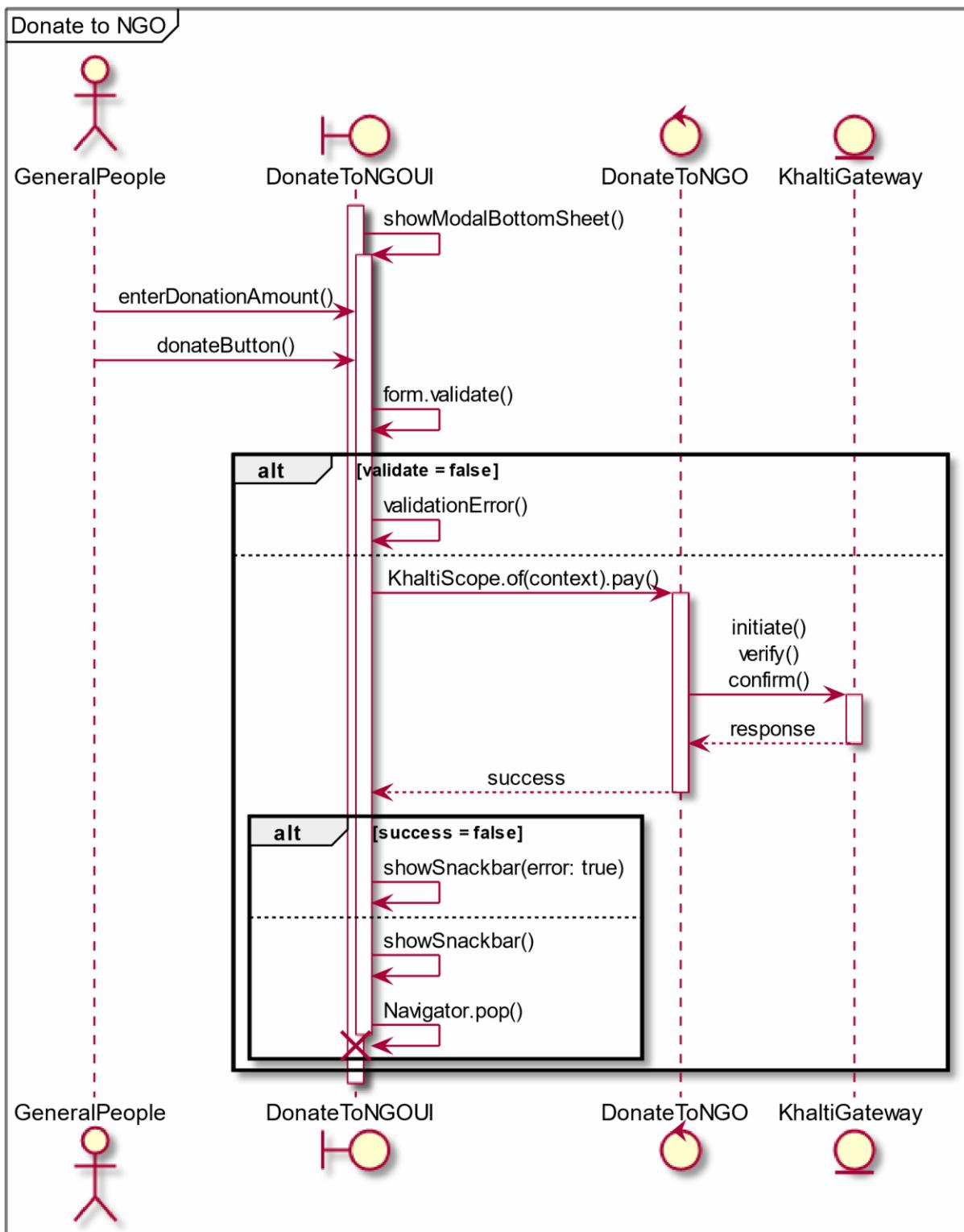


Figure 168: Sequence Diagram – UC14

7.8.8.10. UC15

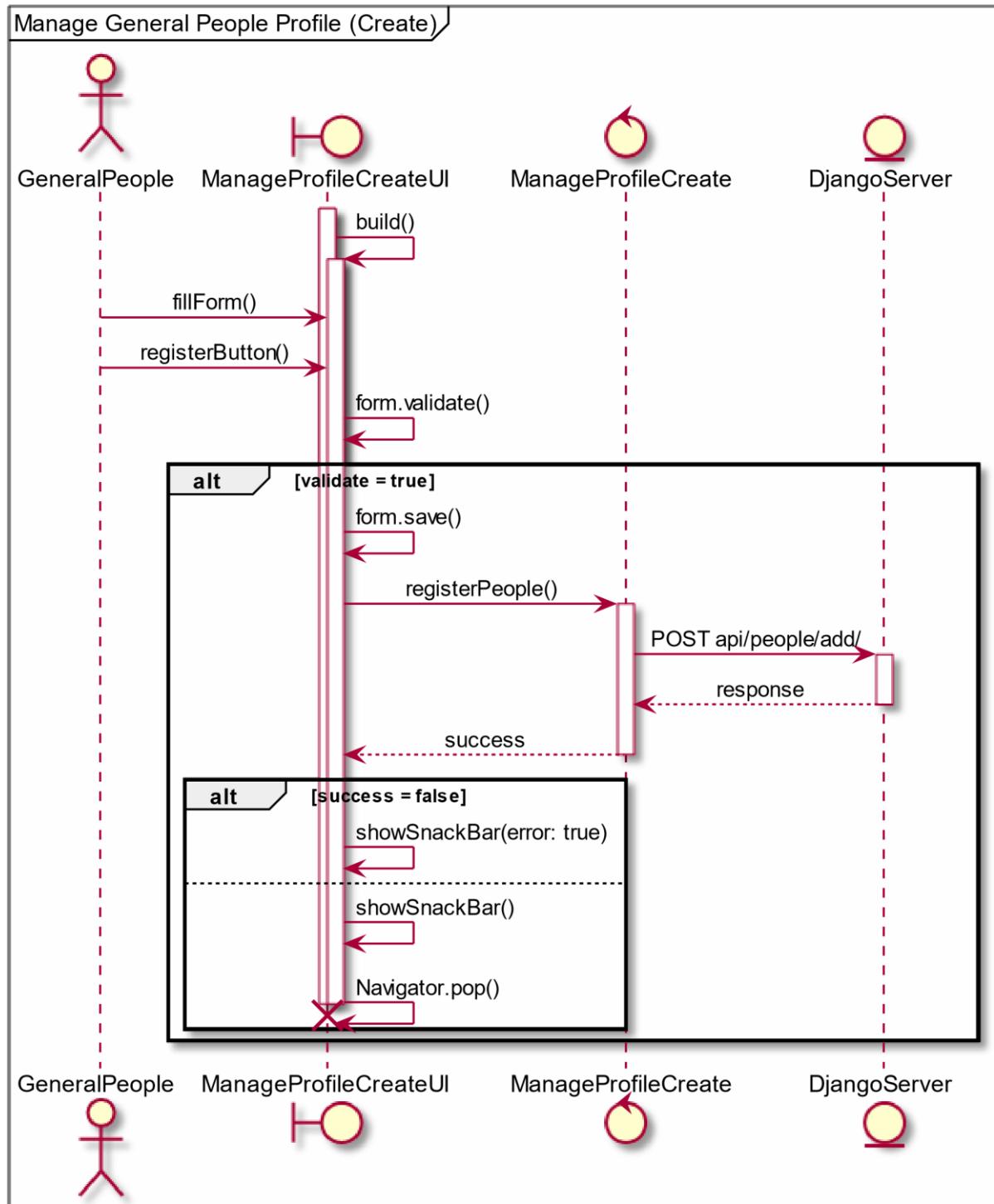


Figure 169: Sequence Diagram – UC15 (Create)

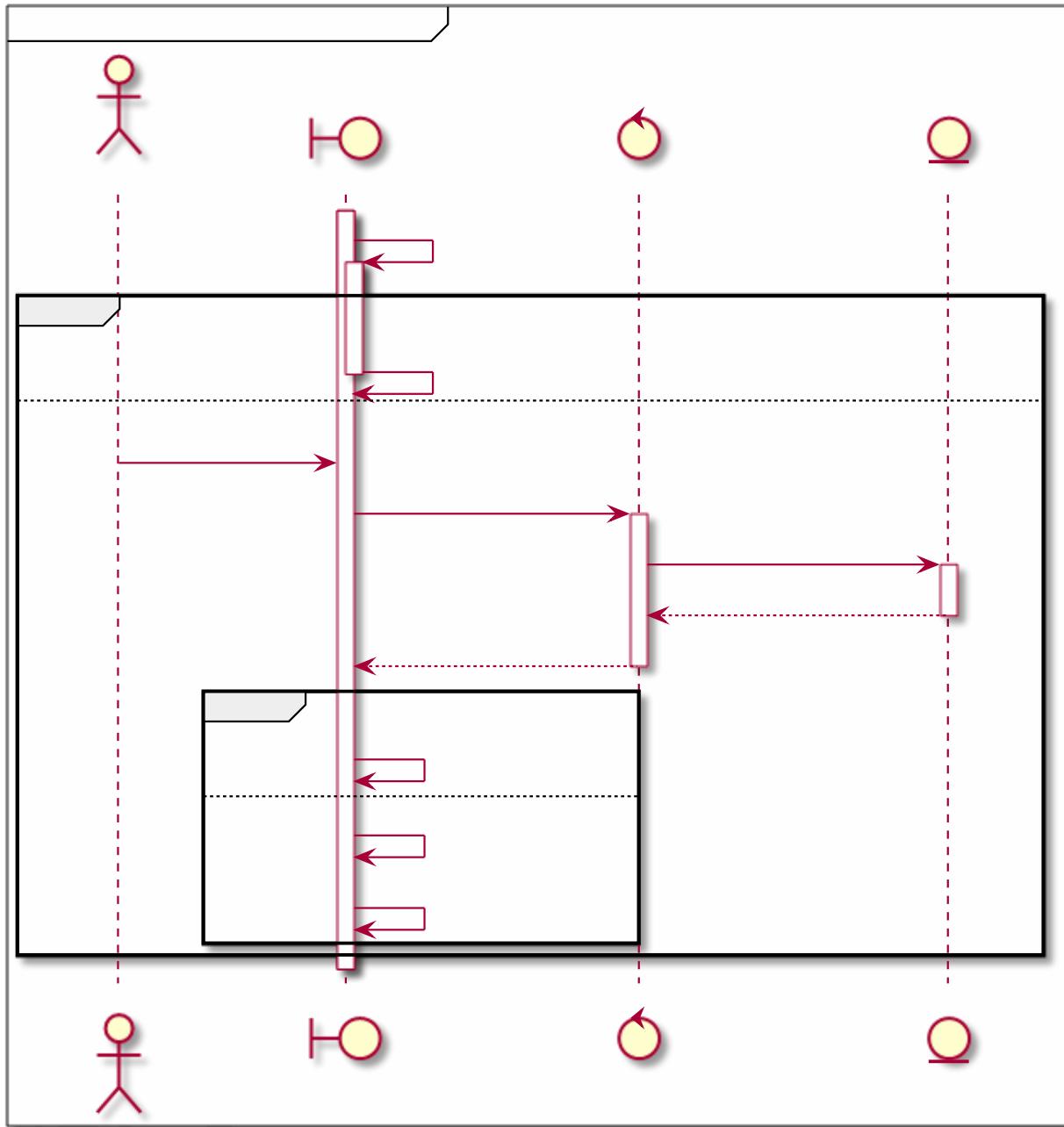


Figure 170: Sequence Diagram – UC15 (Delete)

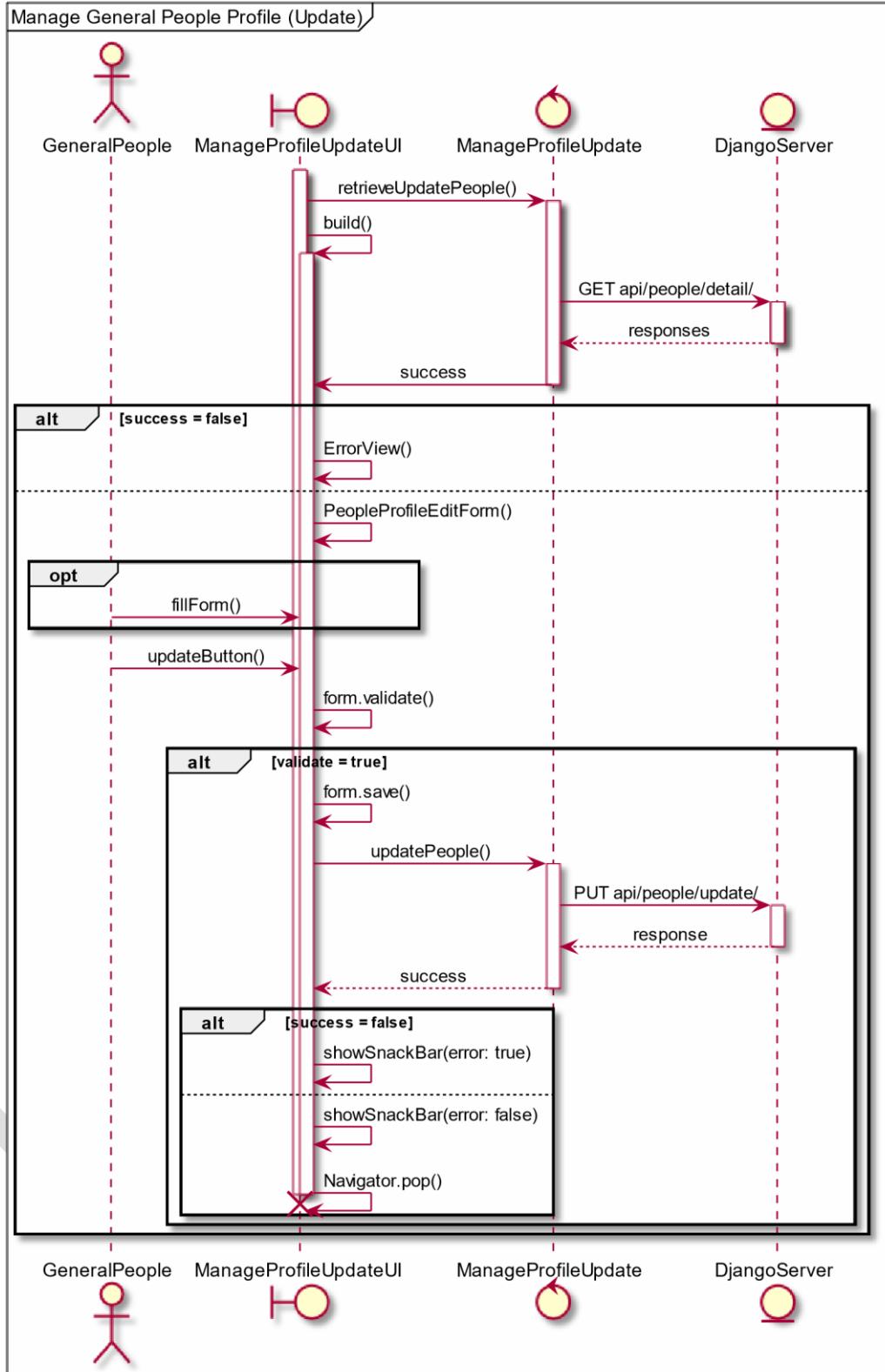


Figure 171: Sequence Diagram – UC15 (Update)

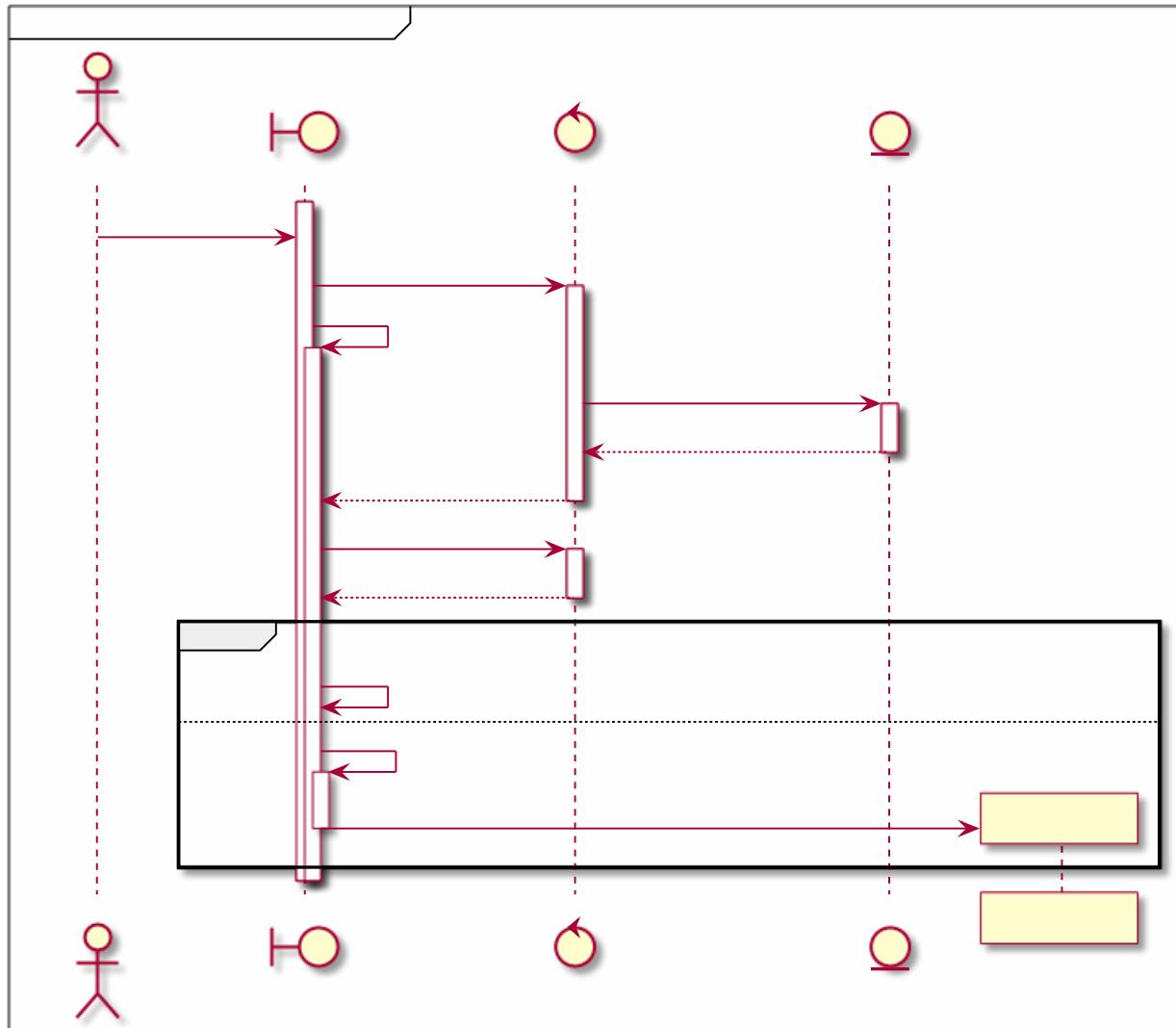


Figure 172: Sequence Diagram – UC15 (View)

7.8.8.11. UC16

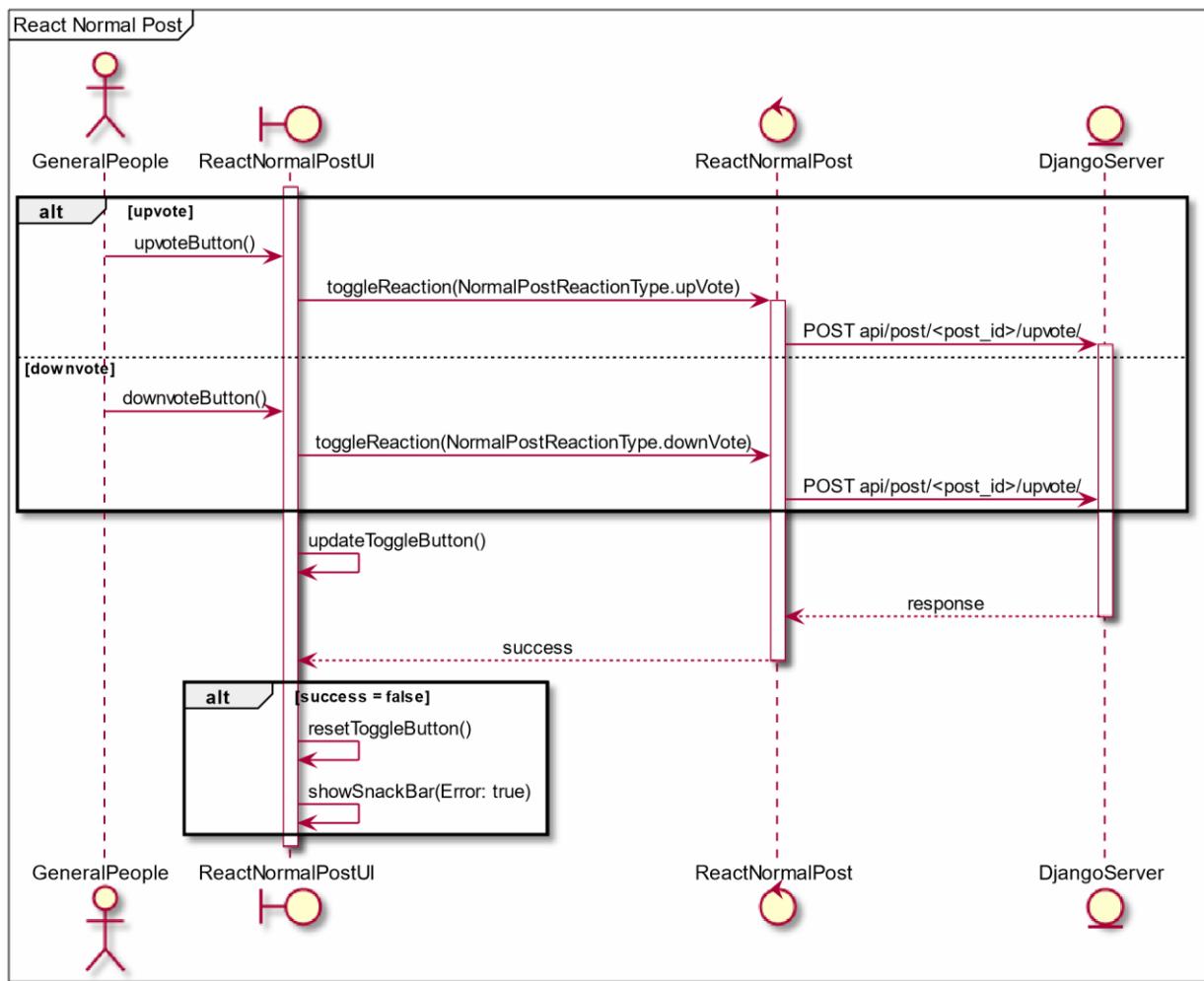


Figure 173: Sequence Diagram – UC16

7.8.8.12. UC17

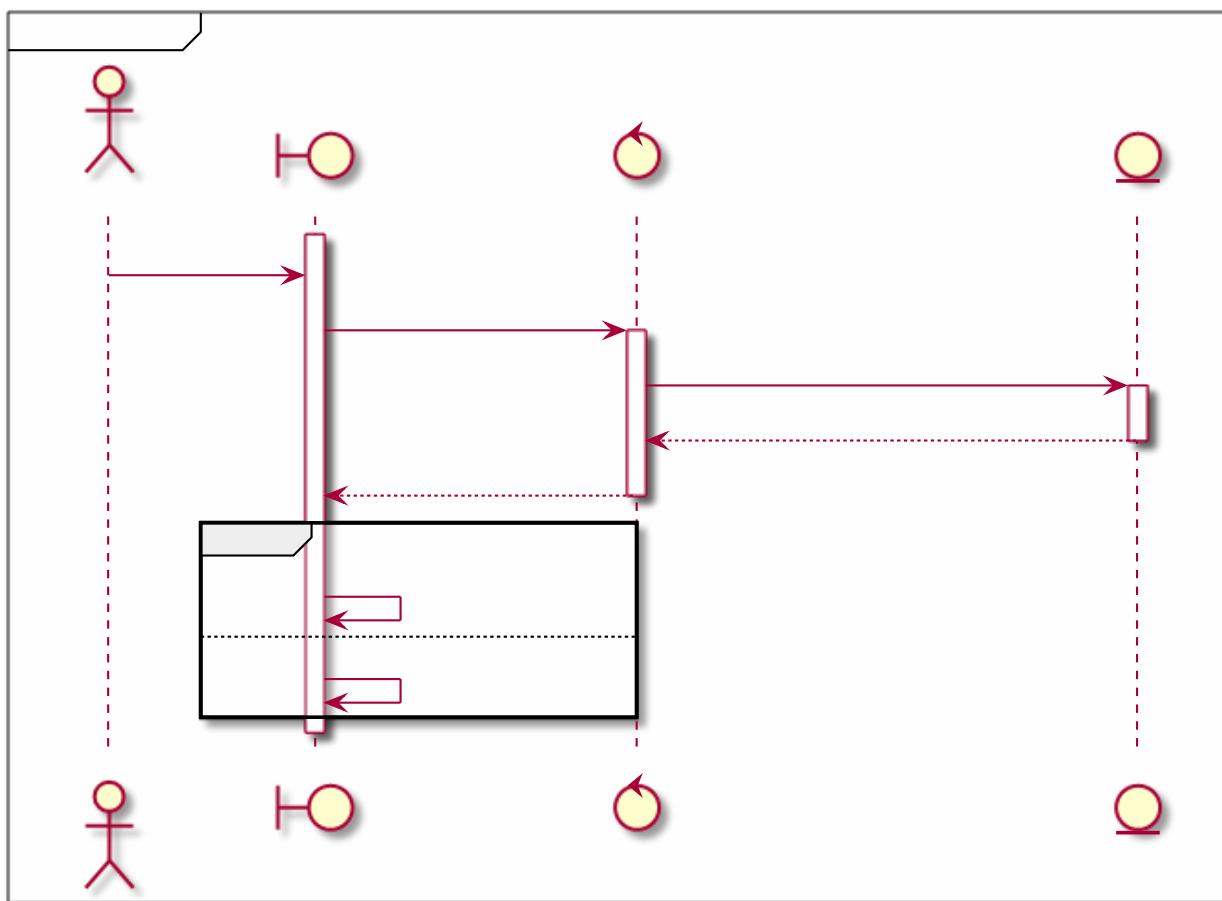


Figure 174: Sequence Diagram – UC17

7.8.8.13. UC18

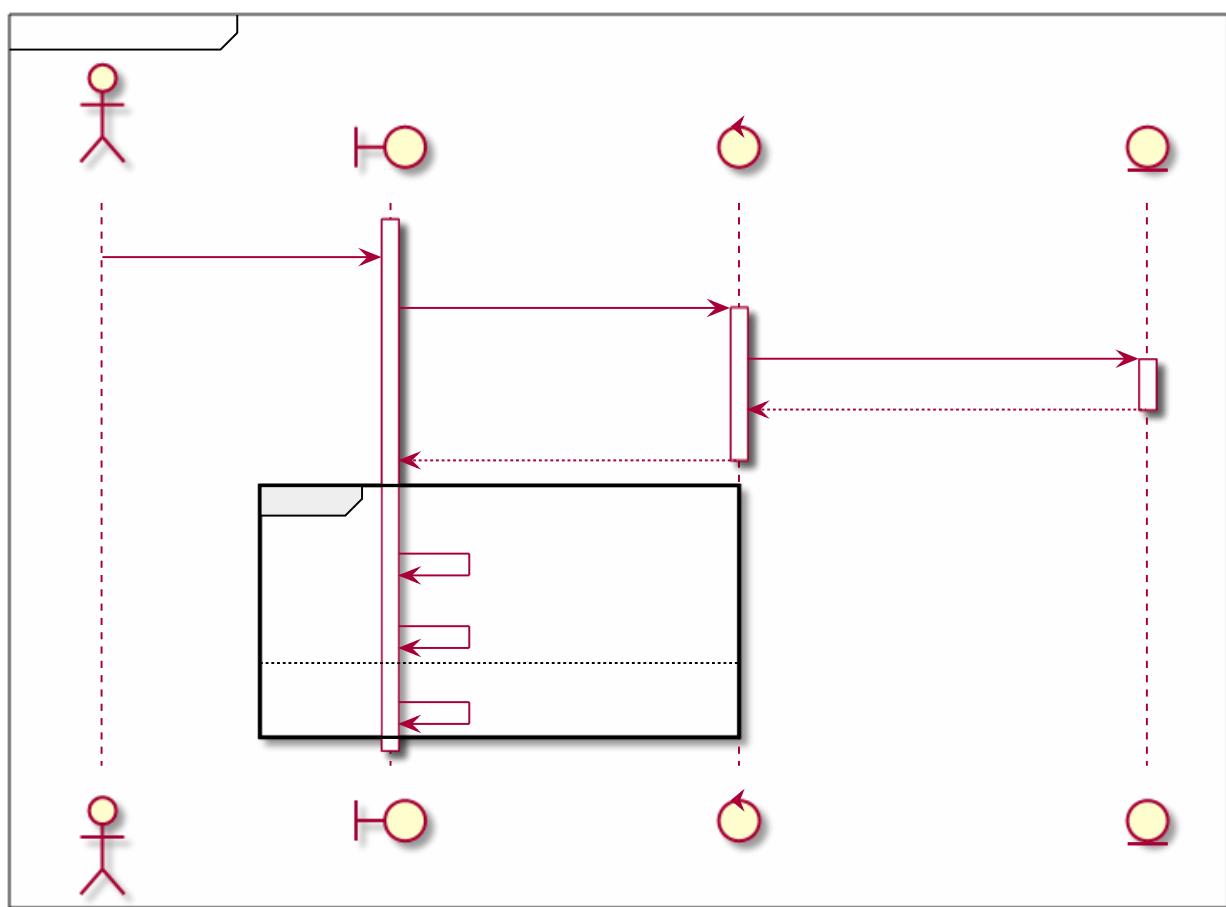


Figure 175: Sequence Diagram – UC18

7.8.8.14. UC19

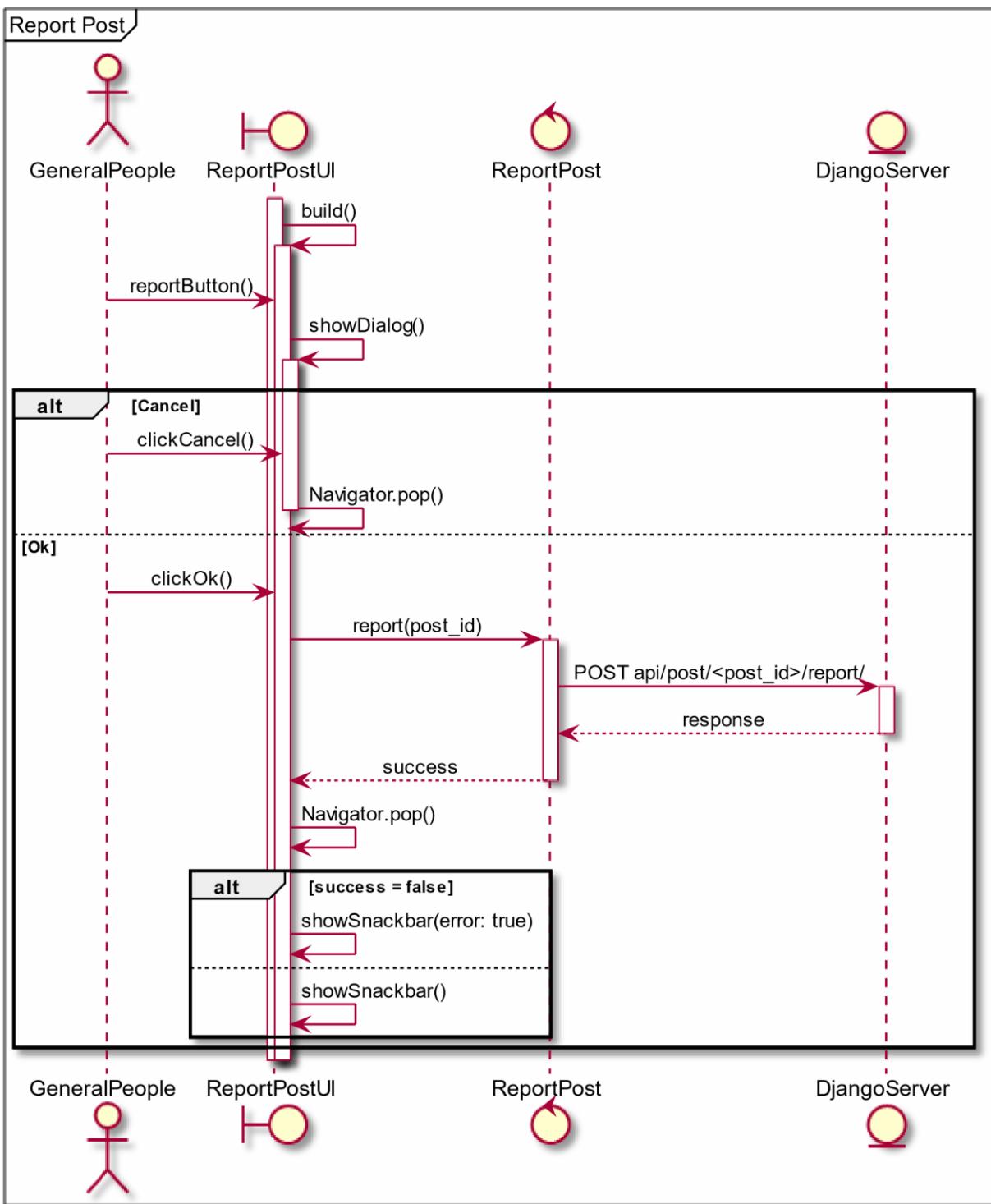


Figure 176: Sequence Diagram – UC19

7.8.8.15. UC20

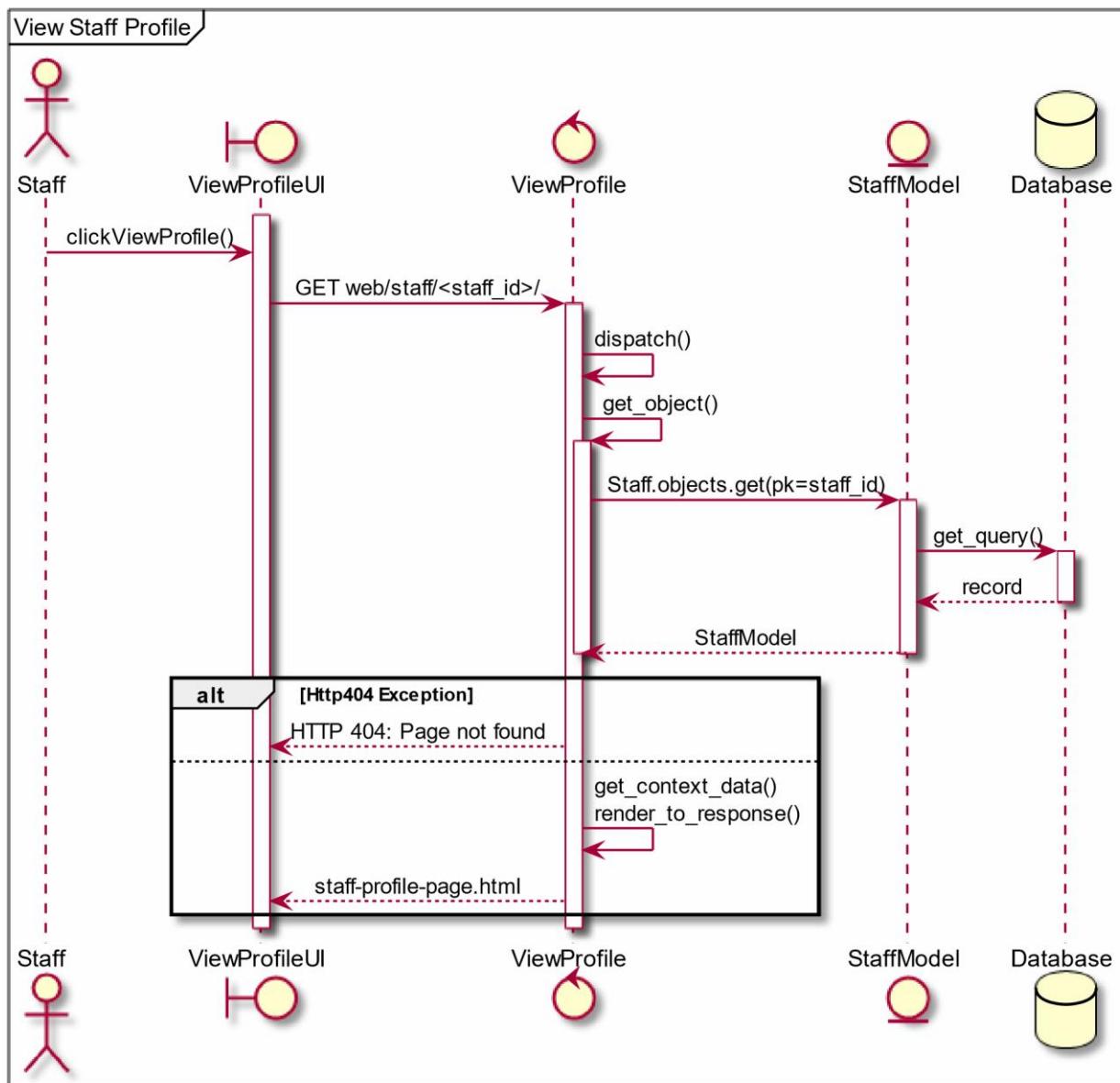


Figure 177: Sequence Diagram – UC20

7.8.8.16. UC21

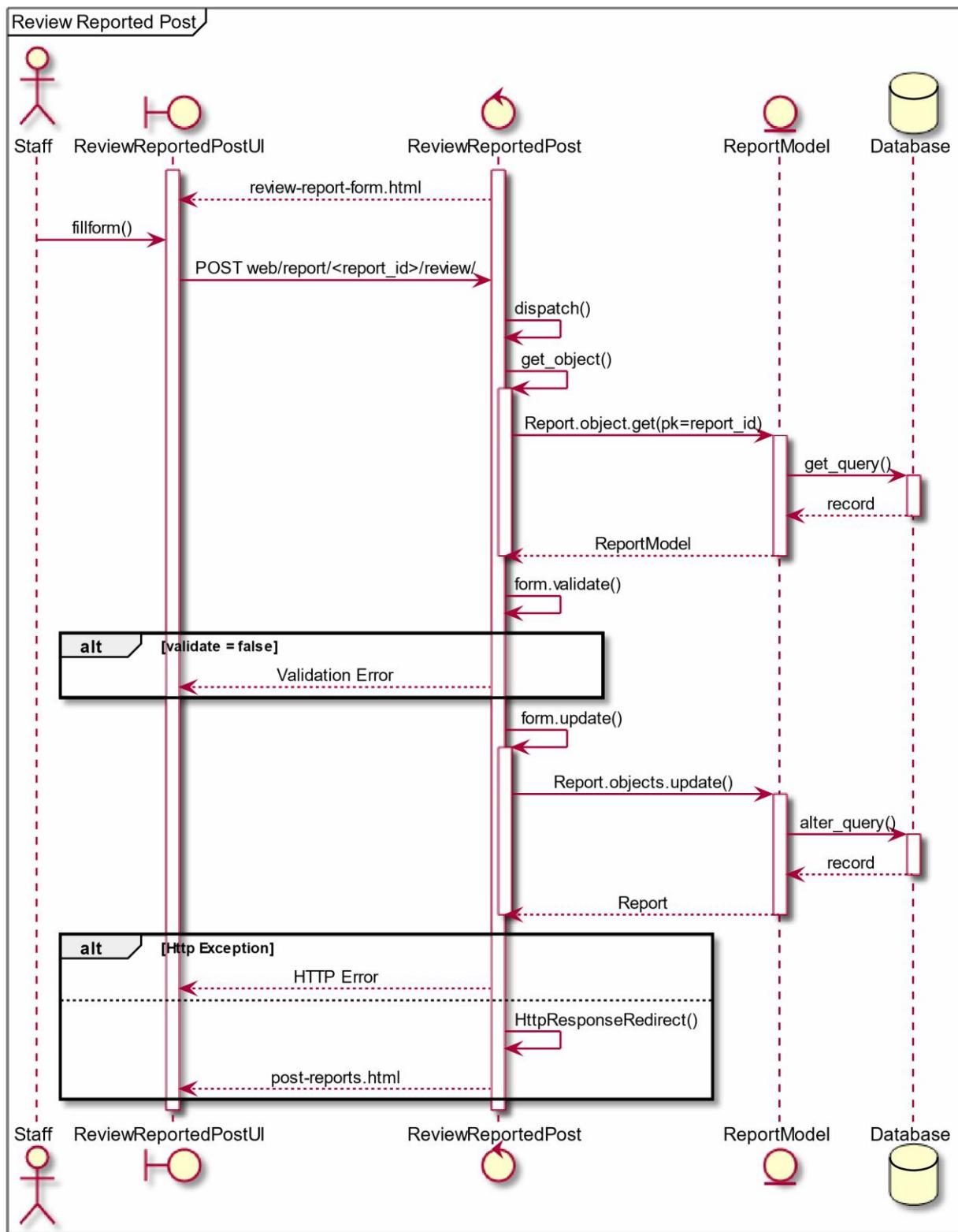


Figure 178: Sequence Diagram – UC21

7.8.8.17. UC22

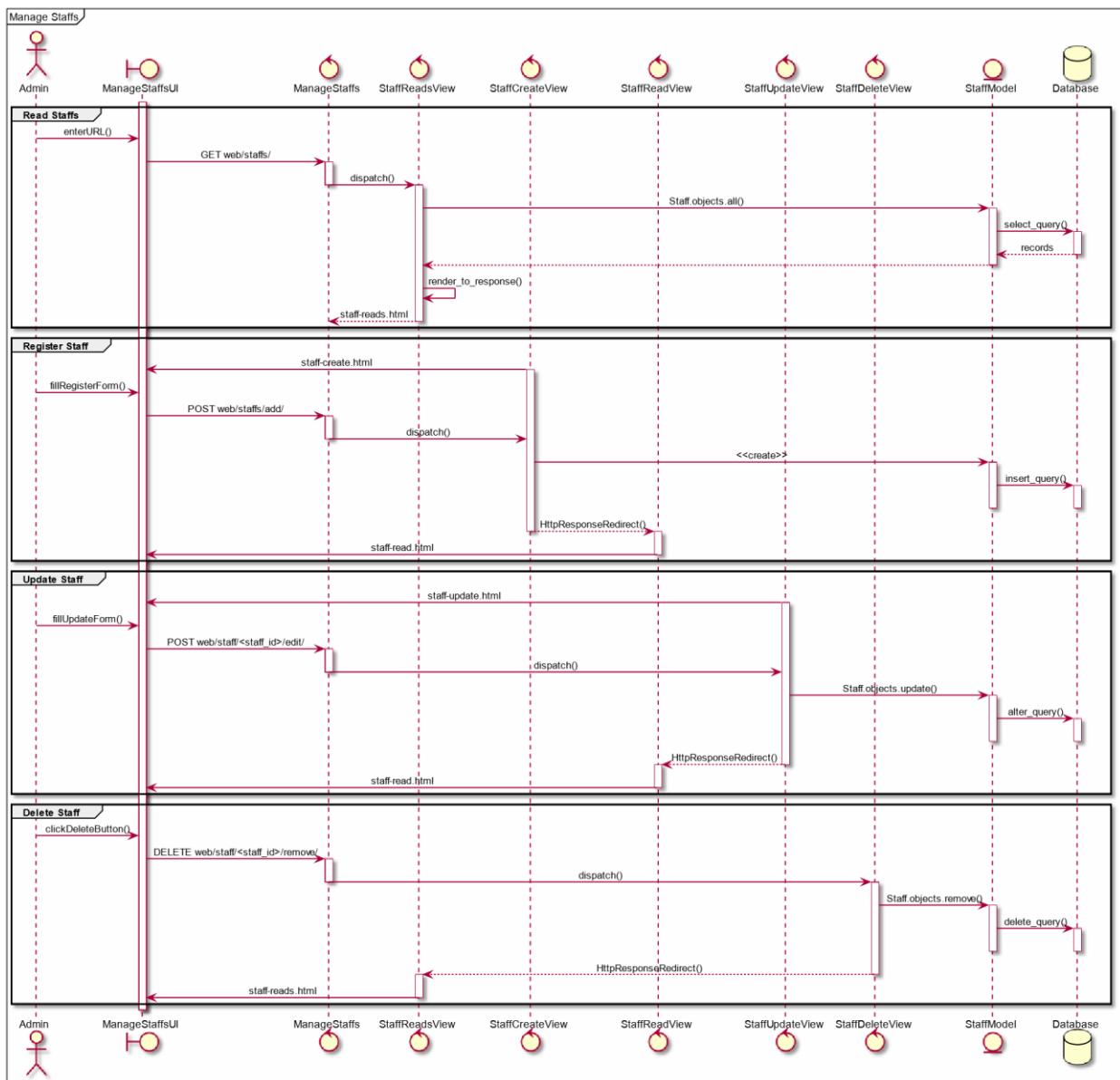


Figure 179: Sequence Diagram – UC22

7.8.8.18. UC23

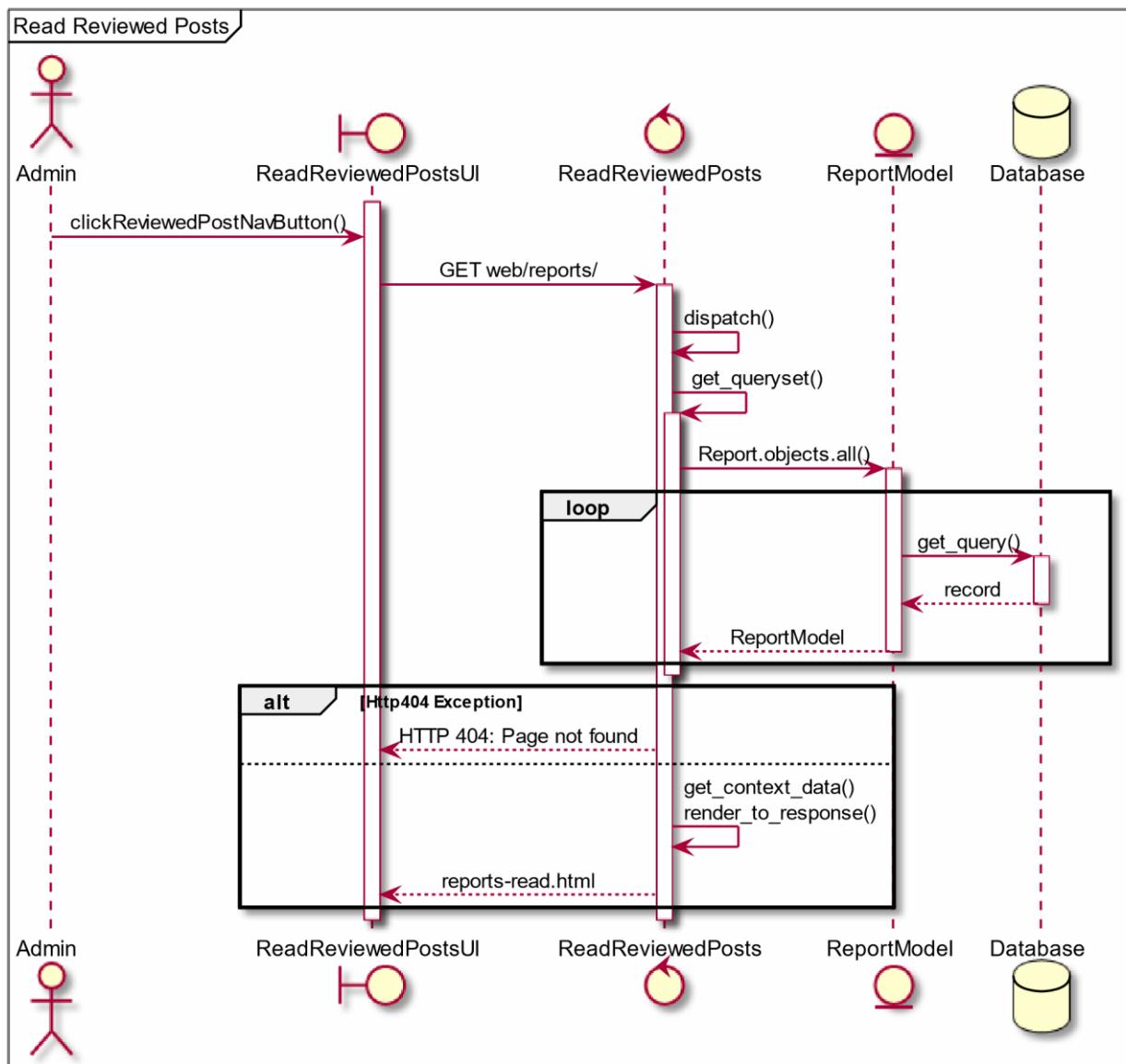


Figure 180: Sequence Diagram – UC23

7.8.9. WIREFRAMES

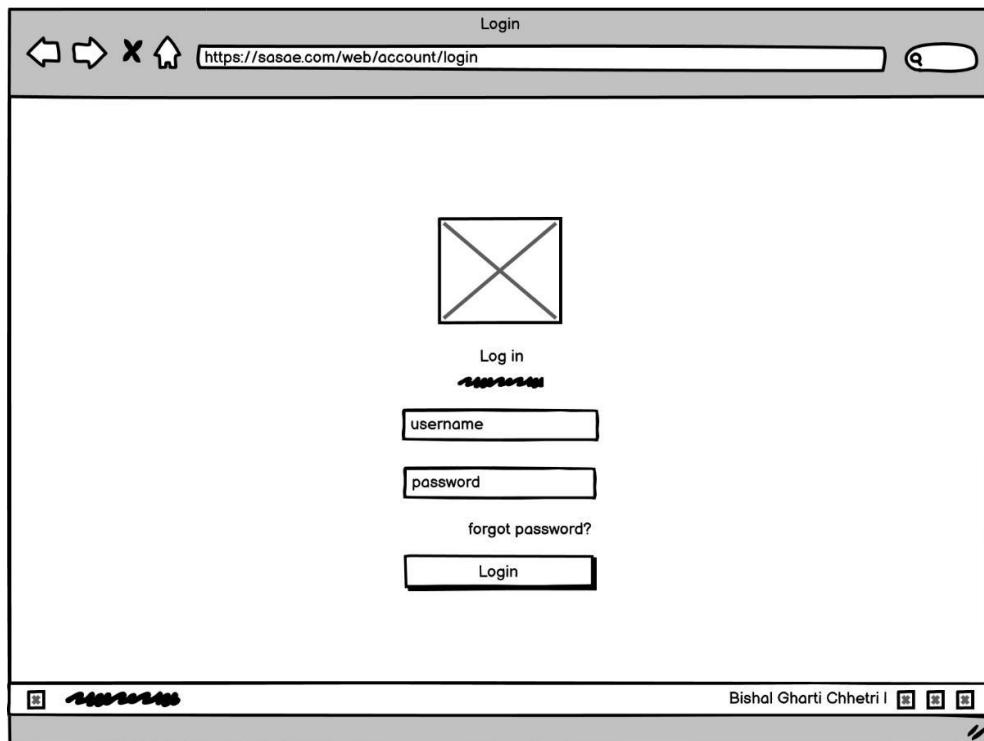


Figure 181: Web Login

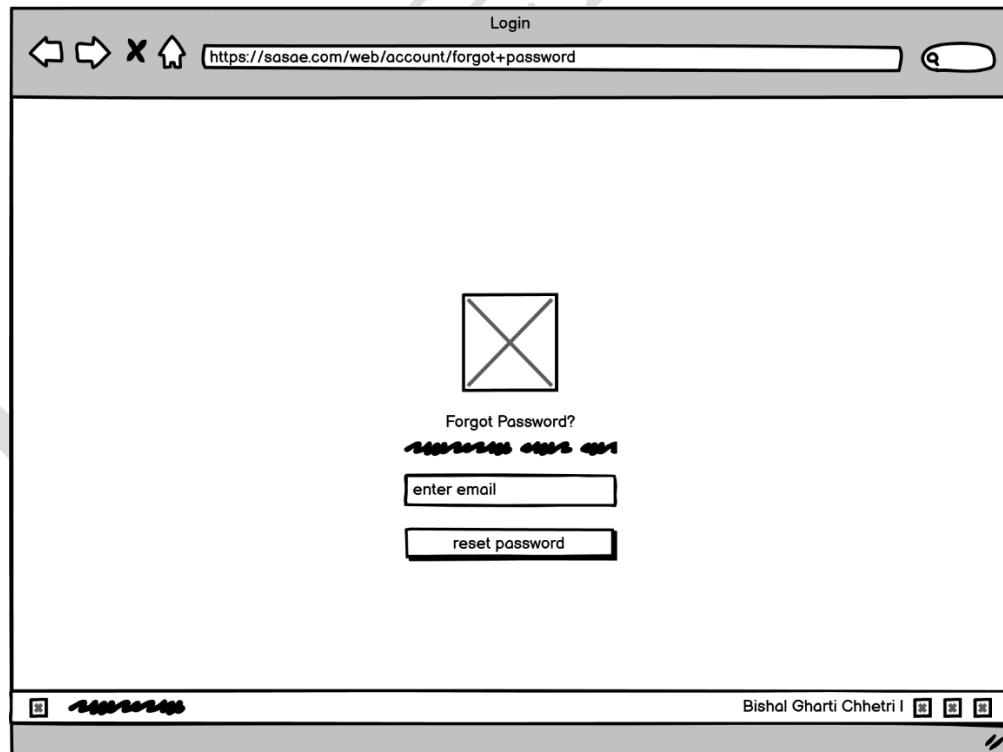


Figure 182: Web Forgot Password

The screenshot shows a web browser window with the title 'Login' at the top. The URL bar contains 'https://sasae.com/web/account/change+password'. The main content area is titled 'Change Password' and contains three input fields: 'Previous Password', 'New Password', and 'Confirm Password'. Below these fields is a large, bold, black-bordered button labeled 'Change Password'.

Figure 183: Web Change Password

The screenshot shows a web browser window with the title 'Hi admin' at the top. The navigation menu includes links for 'Home', 'Staff', 'NGO', 'People', 'Change Password', and 'Logout'. The main content area features a grid of 20 small, square icons arranged in five rows and four columns. To the right of the grid is a table with two columns: 'Field of Work' and 'NGOs Count'. The table has 5 rows, each corresponding to one of the grid rows. The 'Field of Work' column contains empty text boxes, and the 'NGOs Count' column contains empty text boxes.

Figure 184: Web Admin Home

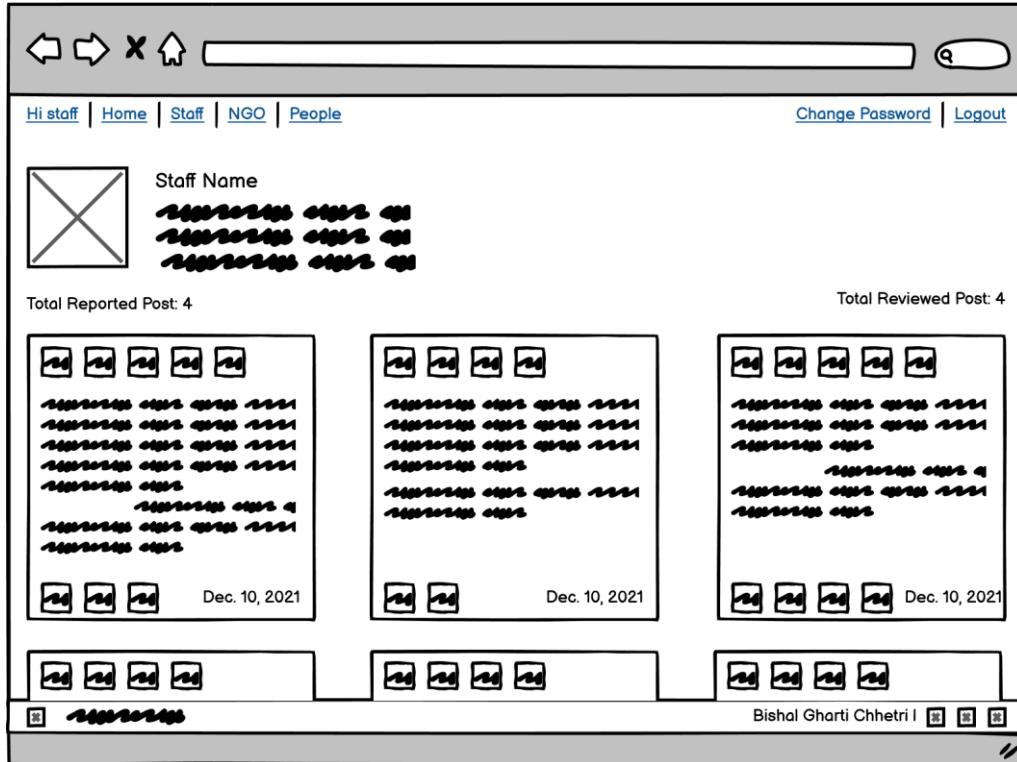


Figure 185: Web Staff Home

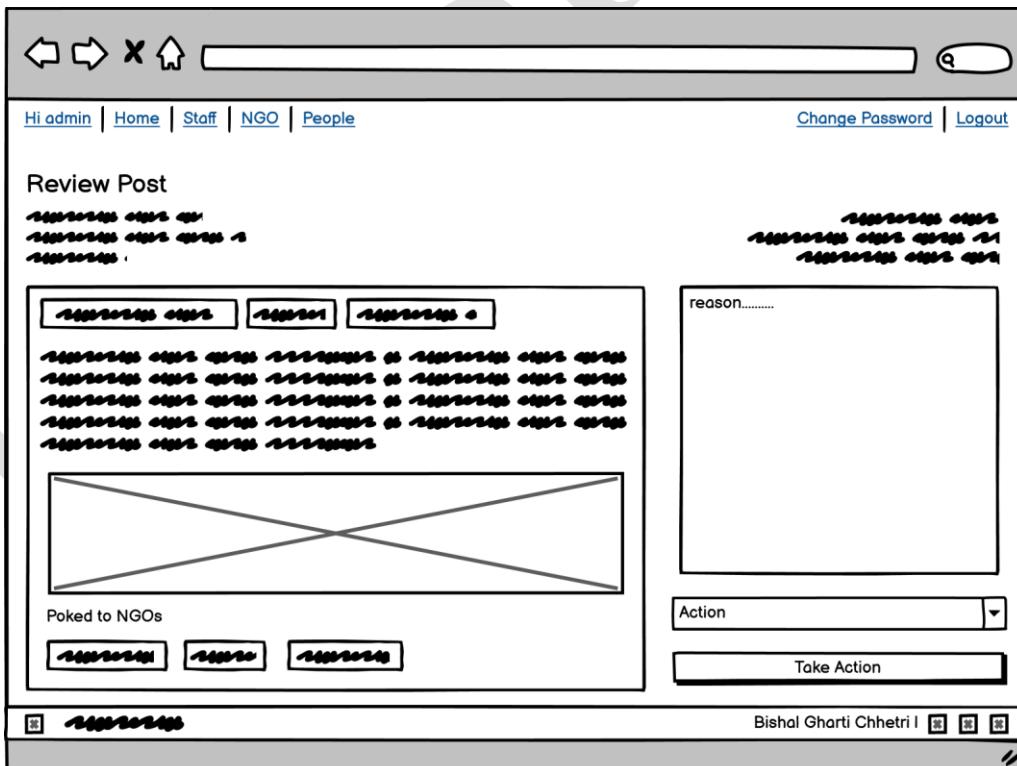


Figure 186: Web Reported Post Review

SASAE

DP	Name	Gender	DOB	Married	Phone	Email	Address	Date Joined	Active	Last Login	Action

Figure 187: Web Staff Tab

DP	Name	Est. Date	Phone	Email	Address	Date Joined	Active	Last Login	Verified	Action

Figure 188: Web NGO Tab

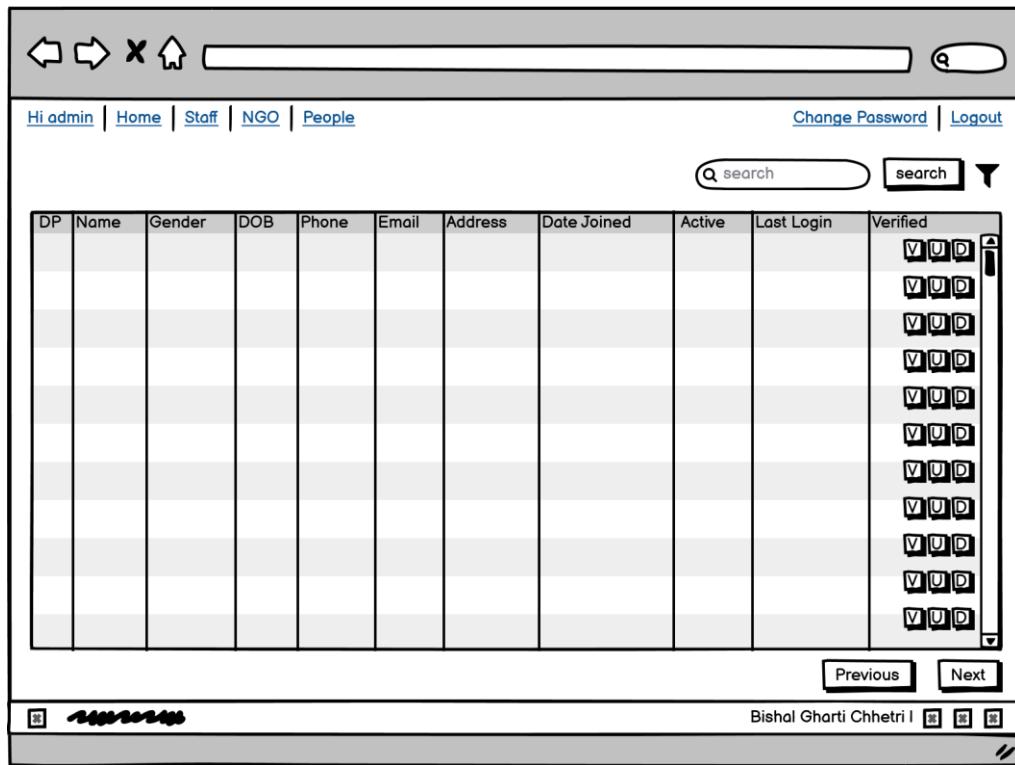


Figure 189: Web People Tab

User Name *

Password *

Confirm Password *

Name *

DOB *

Gender *

Phone *

Email *

Address *

Display Picture *

Citizenship Picture *

Marital Status *

Register

Fields to be rendered,
While creating an account

Figure 190: Web Create Staff

SASAE

The screenshot displays a web-based application for registering NGOs. On the left, a browser window titled 'Login' shows the URL <https://sasae.com/web/account/login>. The main content area is titled 'Register NGO'. It contains several input fields:

- User Name
- Password
- Confirm Password
- Organization Name
- Establishment Date

Below these are dropdown menus for 'Field Of Work' containing options like Advocacy & Awareness, Youth Empowerment, Charity/Philanthropy, Transparency, Displaced Population & Refugees, and Rehabilitation. There are also fields for Phone, Email, Address, and a 'Display Picture' file upload. At the bottom, there are buttons for 'Choose File' and 'Social Welfare Council Affl Certificate', 'PAN Certificate', and a checked checkbox for 'Verified'. A large 'Register' button is at the bottom right. A note at the bottom right states: 'Fields to be rendered, While creating an account'.

Figure 191: Web Create NGO

SASAE

Organization Name
xxxxxxxxxxxxx xxxxxxxx

Establishment Date
xx/xx/xxxx

Field Of Work

Advocacy & Awareness
Youth Empowerment
Charity/Philanthropy
Transparency
Displaced Population & Refugees
Rehabilitation

Phone
+xxx-xxxxxxxx

Address
xxxxxxxxx-xxx, xxxxxxx

Display Picture
 32764328342.jpg

Social Welfare Council Affl Certificate
 83478343.jpg

PAN Certificate
 34563884329.jpg

Verified

Fields to be rendered,
While updating on Account

Figure 192: Web Update NGO

name
xxxxxxxxxxxxx xxxxxxxx

DOB
xx/xx/xxxx

Gender
xxxxx

Phone
+xxx-xxxxxxxx

Address
xxxxxxxxx-xxx, xxxxxxx

Display Picture
 5435463453.jpg

Citizenship Picture

Verified Marital Status

Fields to be rendered,
While updating on Account

Figure 193: Web Update Staff and People

SASAE

Full Name	xyz	xy
Gender		xy
Date of Birth		xxxx-xx-xx
Phone	+xxxx-xxxxxxx	
Email	xxx@email.com	
Address	xxxxxxxx-x, xxxxx, xx	
Verified	xxxx	xxx
Marital Status	xxxx	xxxx
Display Picture		
Citizenship Photo		
Posted		xx
Last Login	xxxx-xx-xx, xxxxxxxx xx	
Date Joined	xxx, xx, xxxx, xxxx xx	
Active	xxxx	
<input type="button" value="Update"/> <input type="button" value="Delete"/>		

Fields to be rendered,
While Viewing an Account

Figure 194: Web View Staff and People

SASAE

Hi admin | Home | Staff | NGO | People Change Password | Logout

View NGO

Organization Name	xxxxxxxx xxxx xxxx
Establishment Date	xxxx-xx-xx
Phone	+xxx-xxxxxxxx
Email	xxx@email.com
Address	xxxx-xx, xxxxxxxx
Fields of Work	xxxxxx xx xxxx, xxxxxxxxx xxxx, xxxxxxx
Epay Account	xxxxxxxxxxxx
Bank Name	xxxxxxxxxxxx xxxx
Bank Account Name	xxx-xxxxxx-xxxxxx-xxxx
Account Number	xxxxxxxx-xx, xxxx
Branch	xxxxxx
BSB (Bank State Branch)	xxxx
Verified	

Bishal Gharti Chhetri [Edit] [Delete] [Print] [View]

Organization Name	xxxxxxxx xxxx xxxx
Establishment Date	xxxx-xx-xx
Phone	+xxx-xxxxxxxx
Email	xxx@email.com
Address	xxxx-xx, xxxxxxxx
Fields of Work	xxxxxx xx xxxx, xxxxxxxxx xxxx, xxxxxxx
Epay Account	xxxxxxxxxxxx
Bank Name	xxxxxxxxxxxx xxxx
Bank Account Name	xxx-xxxxxx-xxxxxx-xxxx
Account Number	xxxxxxxx-xx, xxxx
Branch	xxxxxx
BSB (Bank State Branch)	xxxx
Verified	

Display Picture

Social Welfare Council Affil Certificate

PAN Certificate

Posted	xx
Poked on	xxxx, xxxx, xxxx
Last Login	xxxx-xx-xx, xxxx-xx
Date Joined	xx. xx, xxxx, xxxx xx
Active	xxxx

Update Delete

Fields to be rendered,
While Viewing on Account

Figure 195: Web View NGO

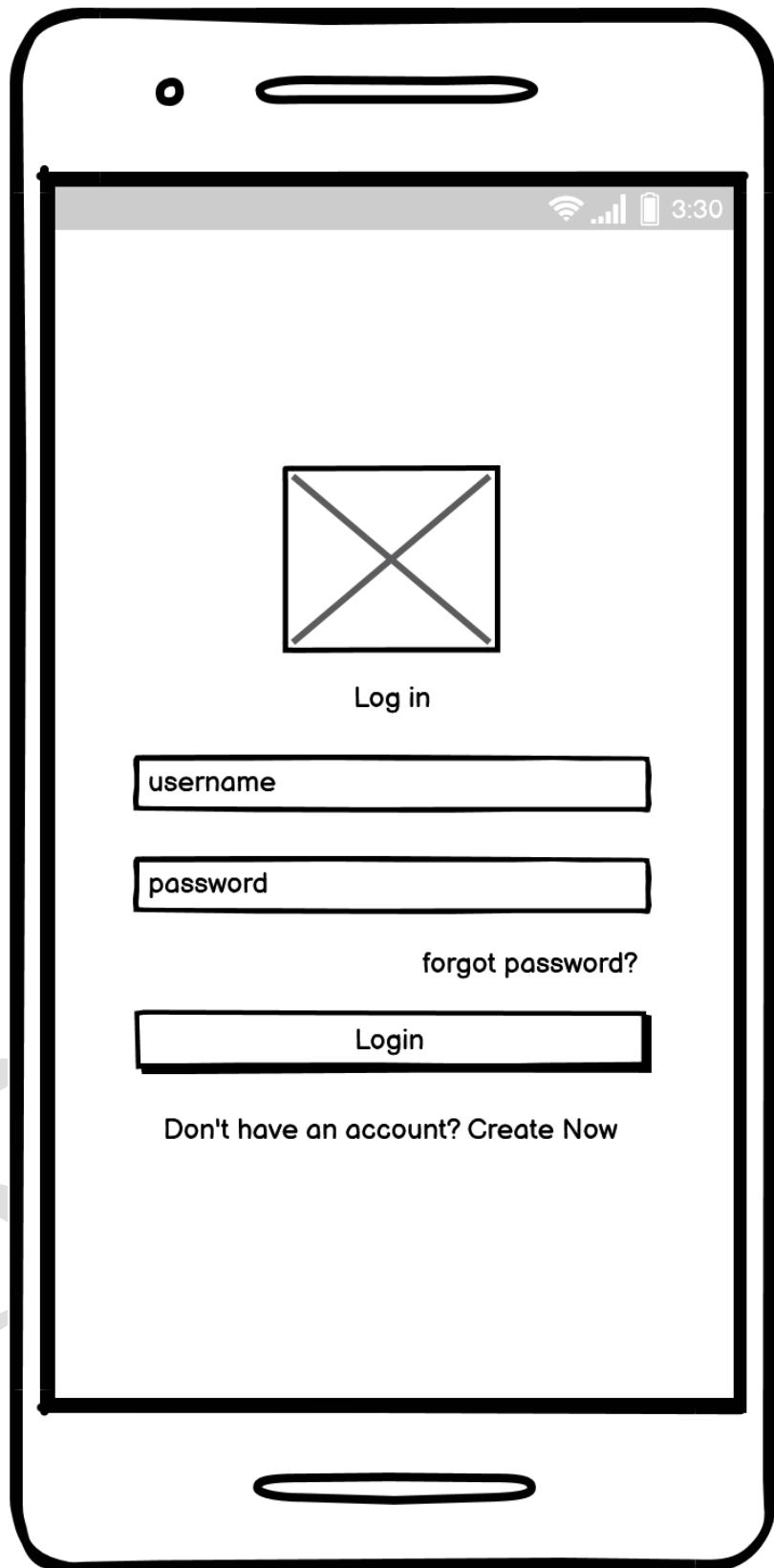


Figure 196: Mobile Login



Figure 197: Mobile Forgot Password

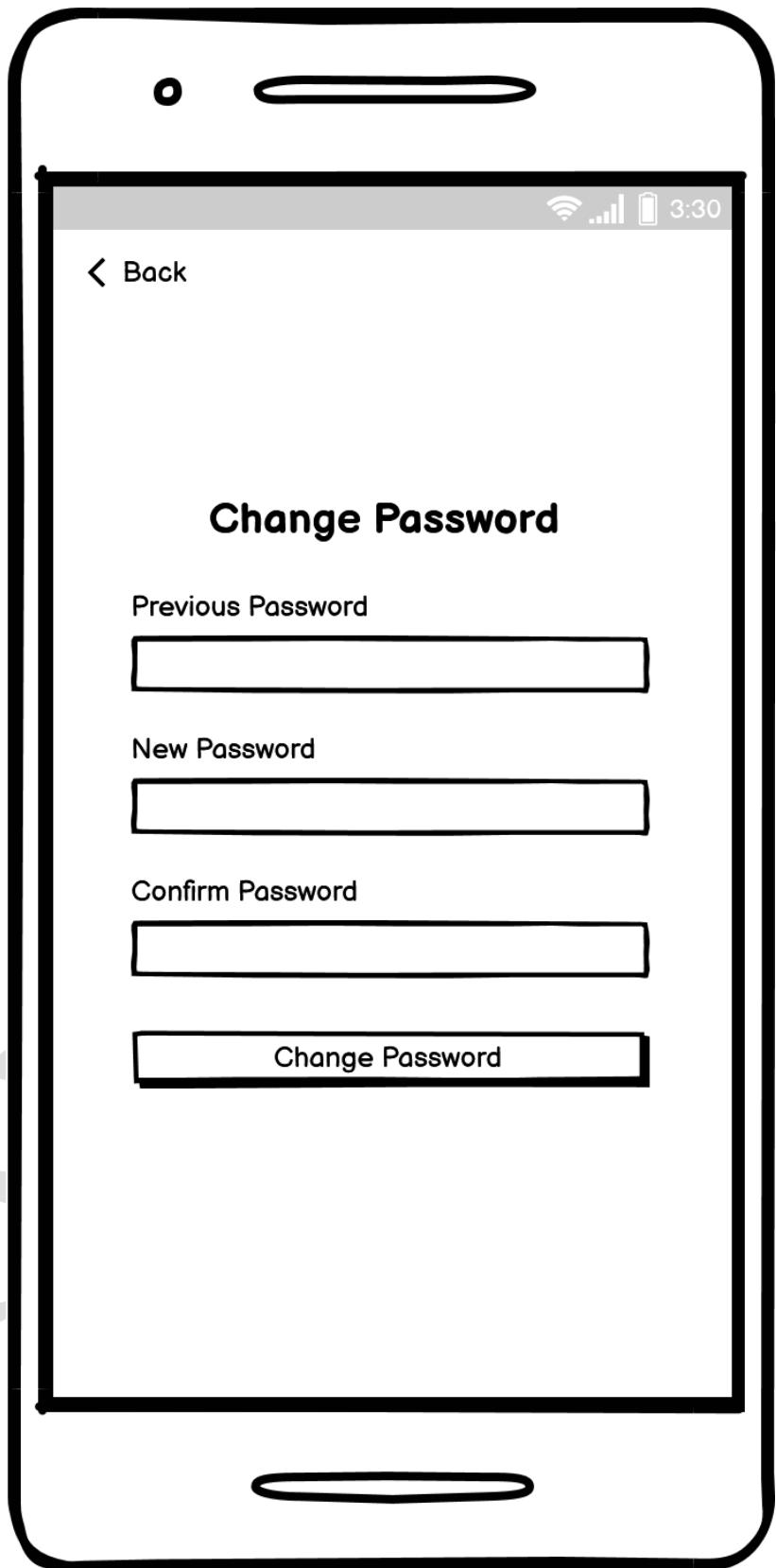


Figure 198: Mobile Password Change

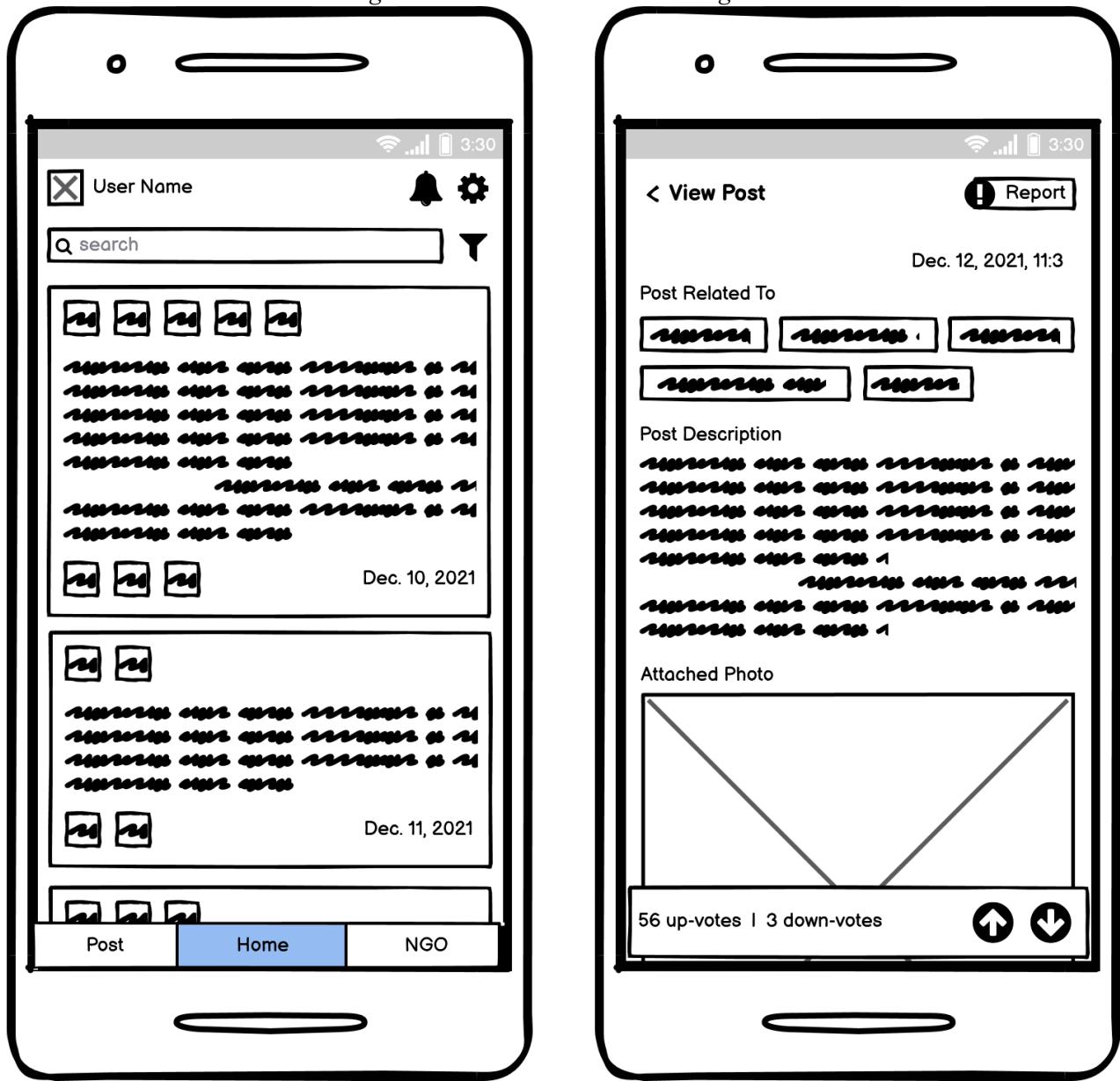


Figure 199: Mobile Home Tab

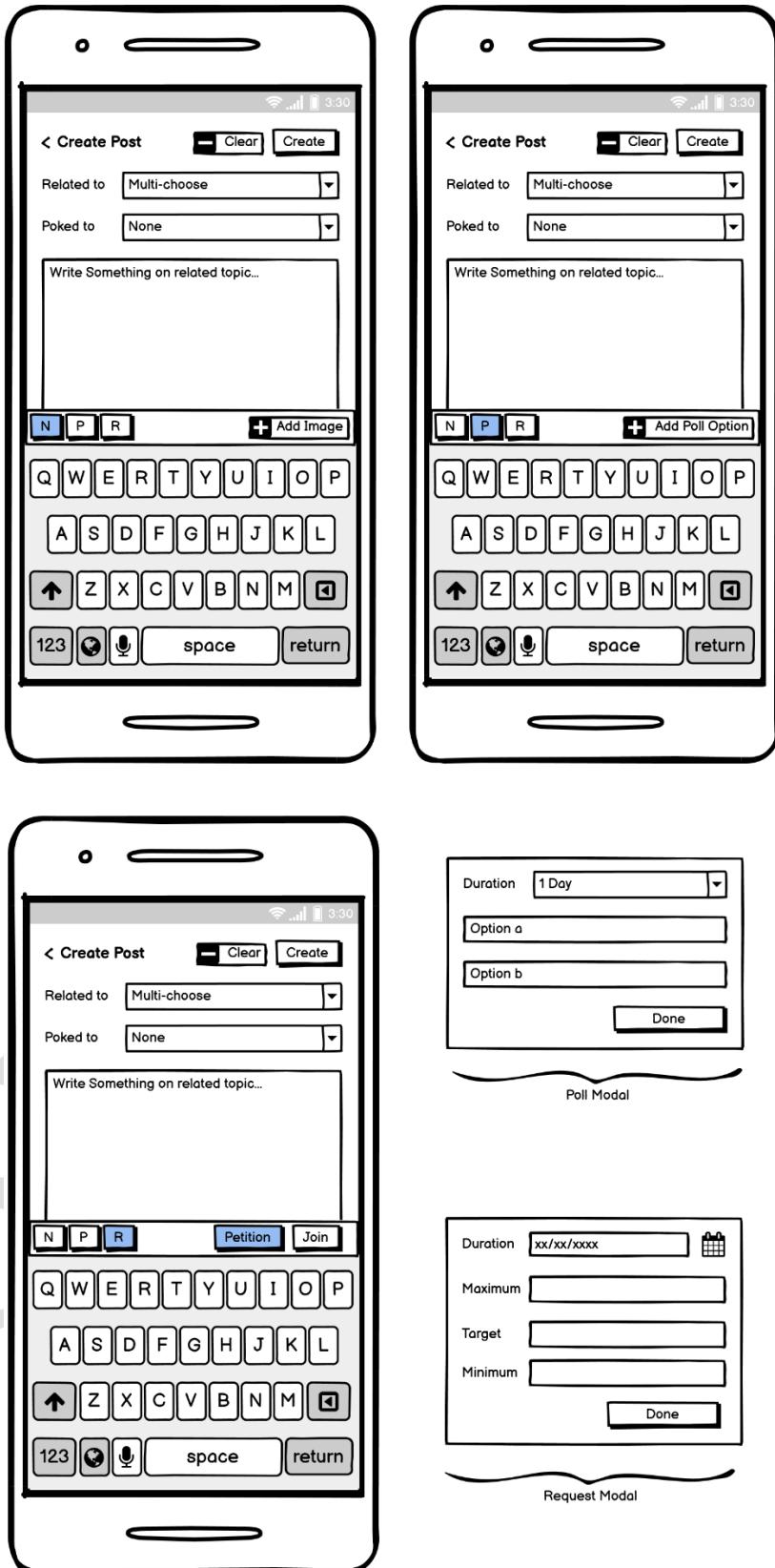


Figure 200: Mobile Post Tab

SASAE

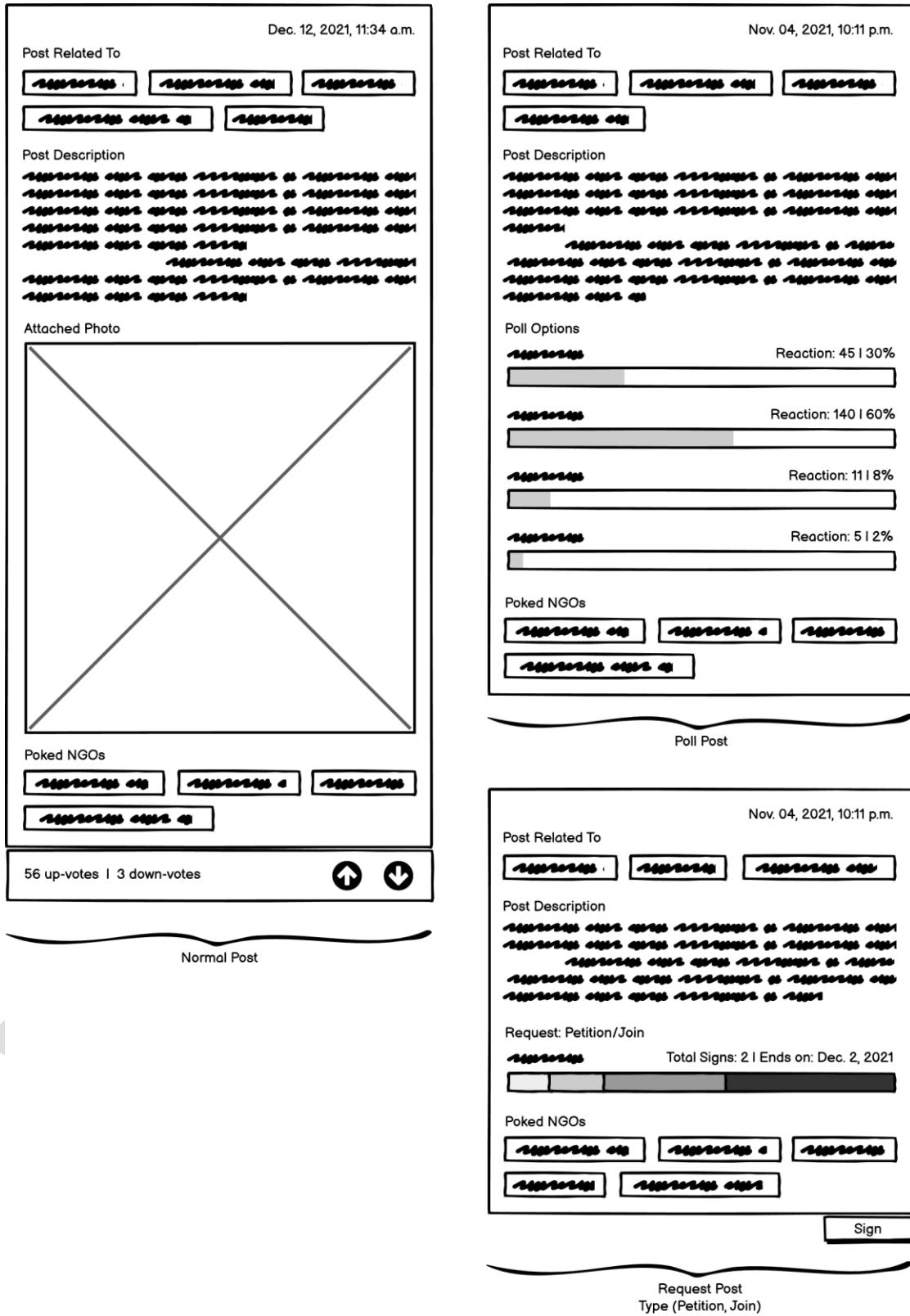


Figure 201: Mobile Post Types

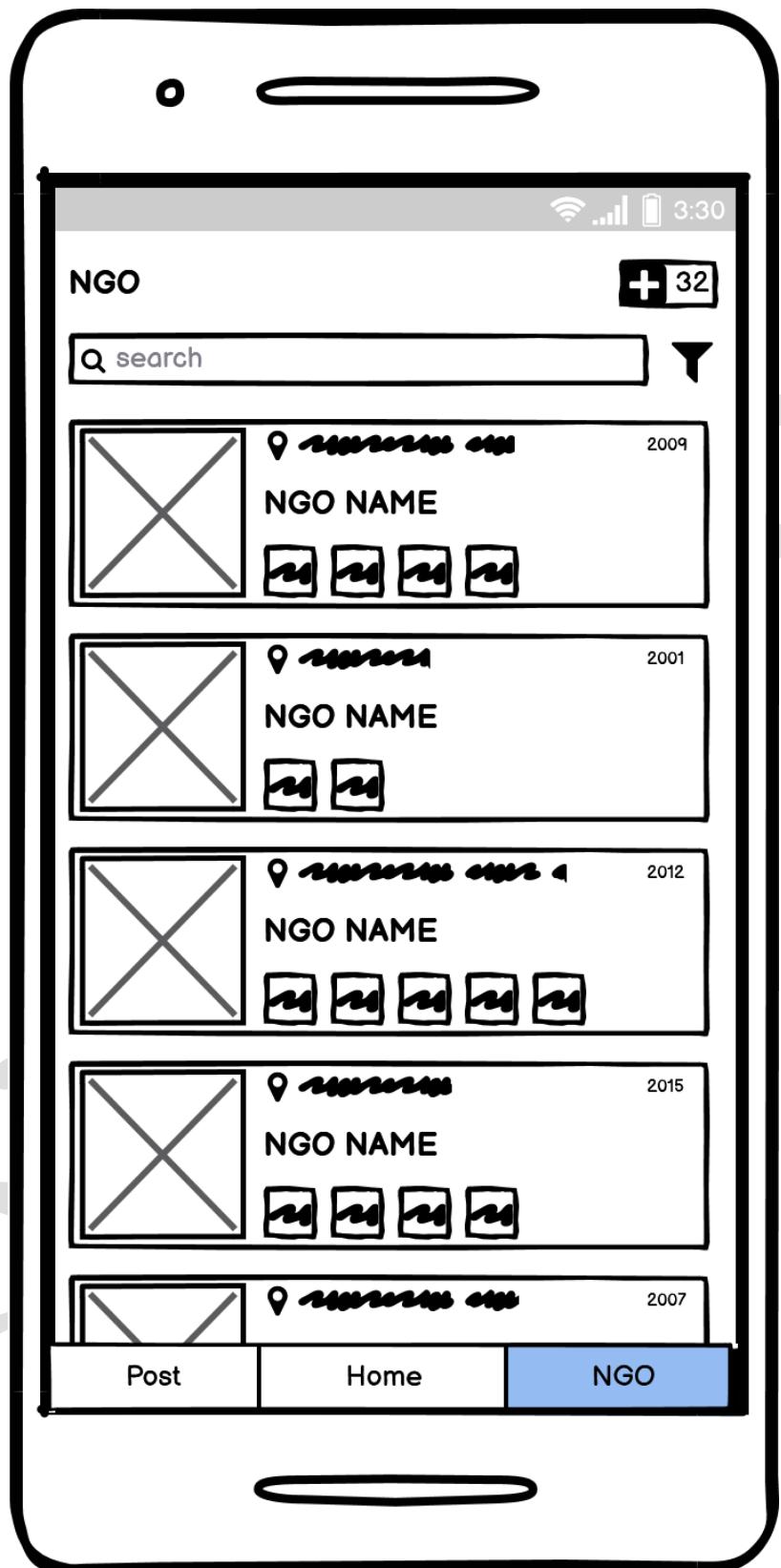
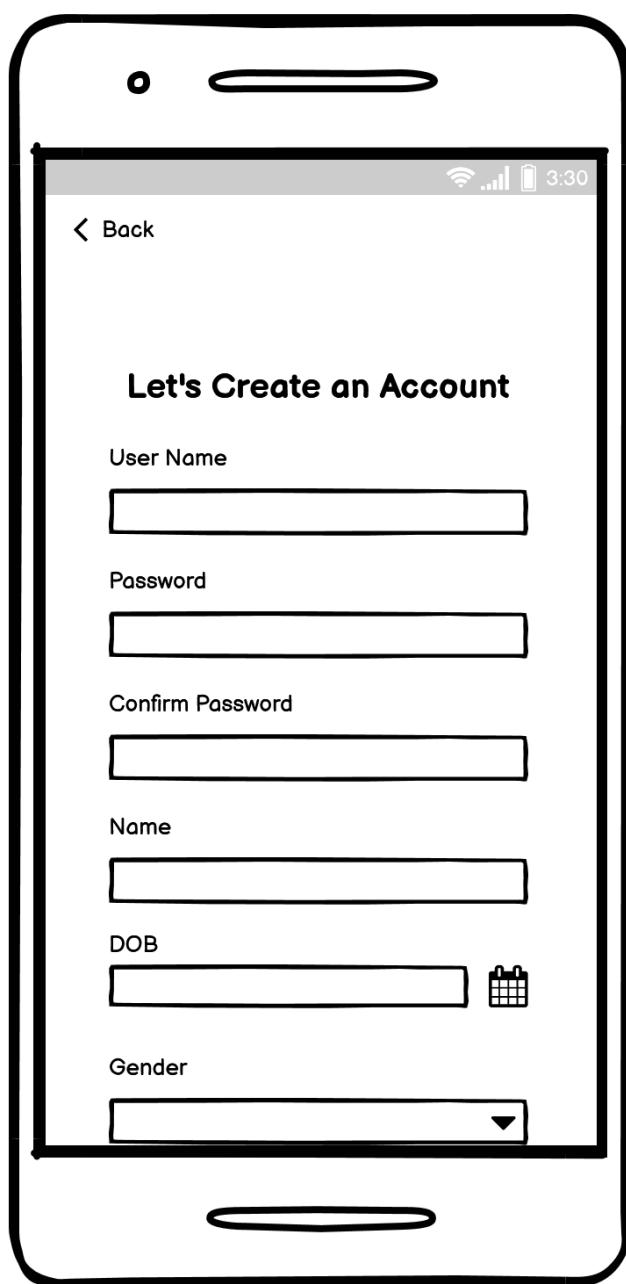


Figure 202: Mobile NGO Tab



Figure 203: Mobile NGO Details



User Name	*
<input type="text"/>	
Password	*
<input type="text"/>	
Confirm Password	*
<input type="text"/>	
Name	*
<input type="text"/>	
DOB	*
<input type="text"/>	
Gender	*
<input type="button" value="▼"/>	
Phone	
<input type="text"/>	
Email	*
<input type="text"/>	
Address	
<input type="text"/>	
Display Picture	
<input type="button" value="Choose File"/>	<input type="text"/>
Citizenship Picture	
<input type="button" value="Choose File"/>	<input type="text"/>
<input type="button" value="Register"/>	

Fields to be rendered,
While creating an account

Figure 204: Mobile People User Register

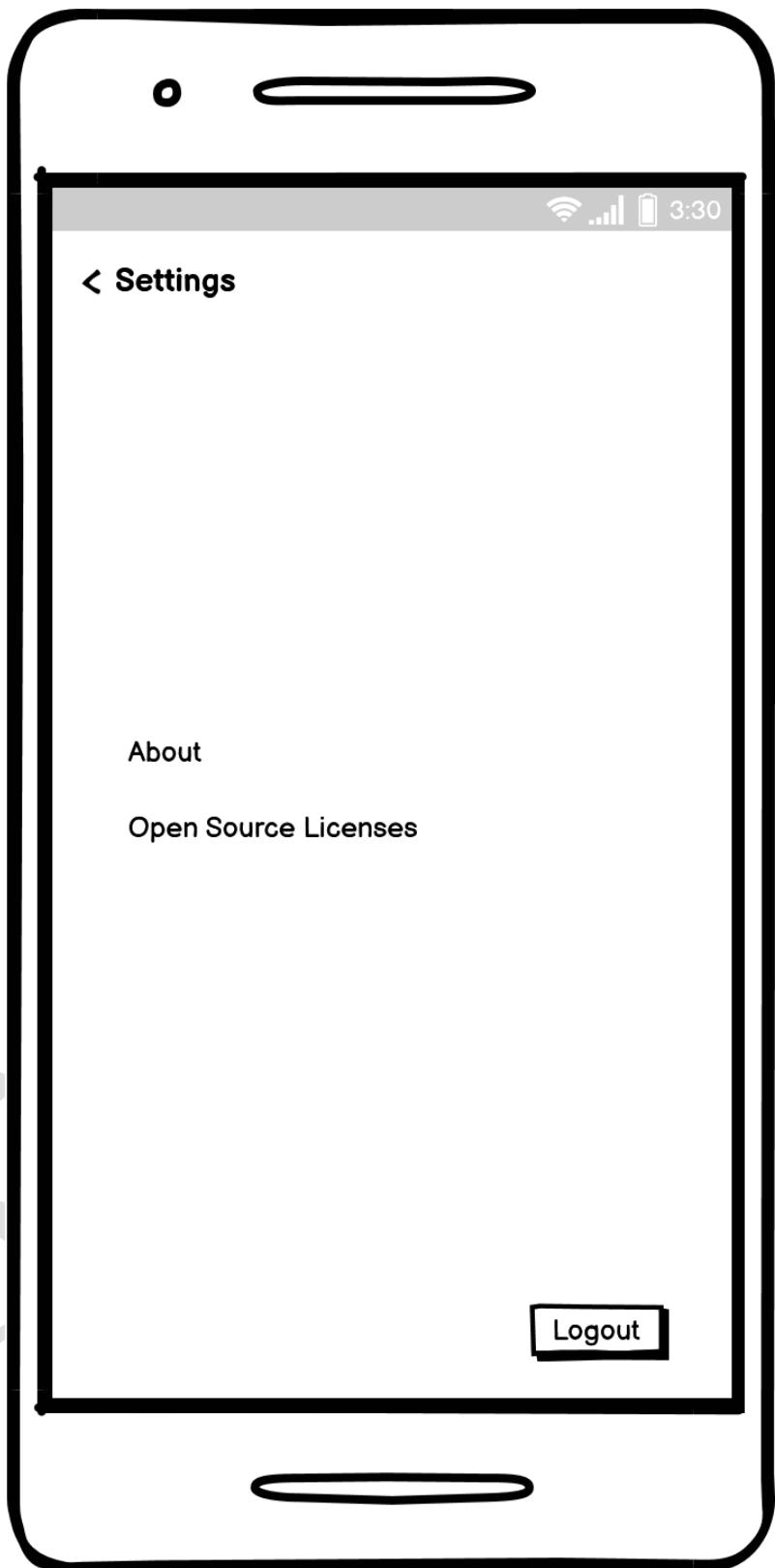


Figure 205: Mobile Settings

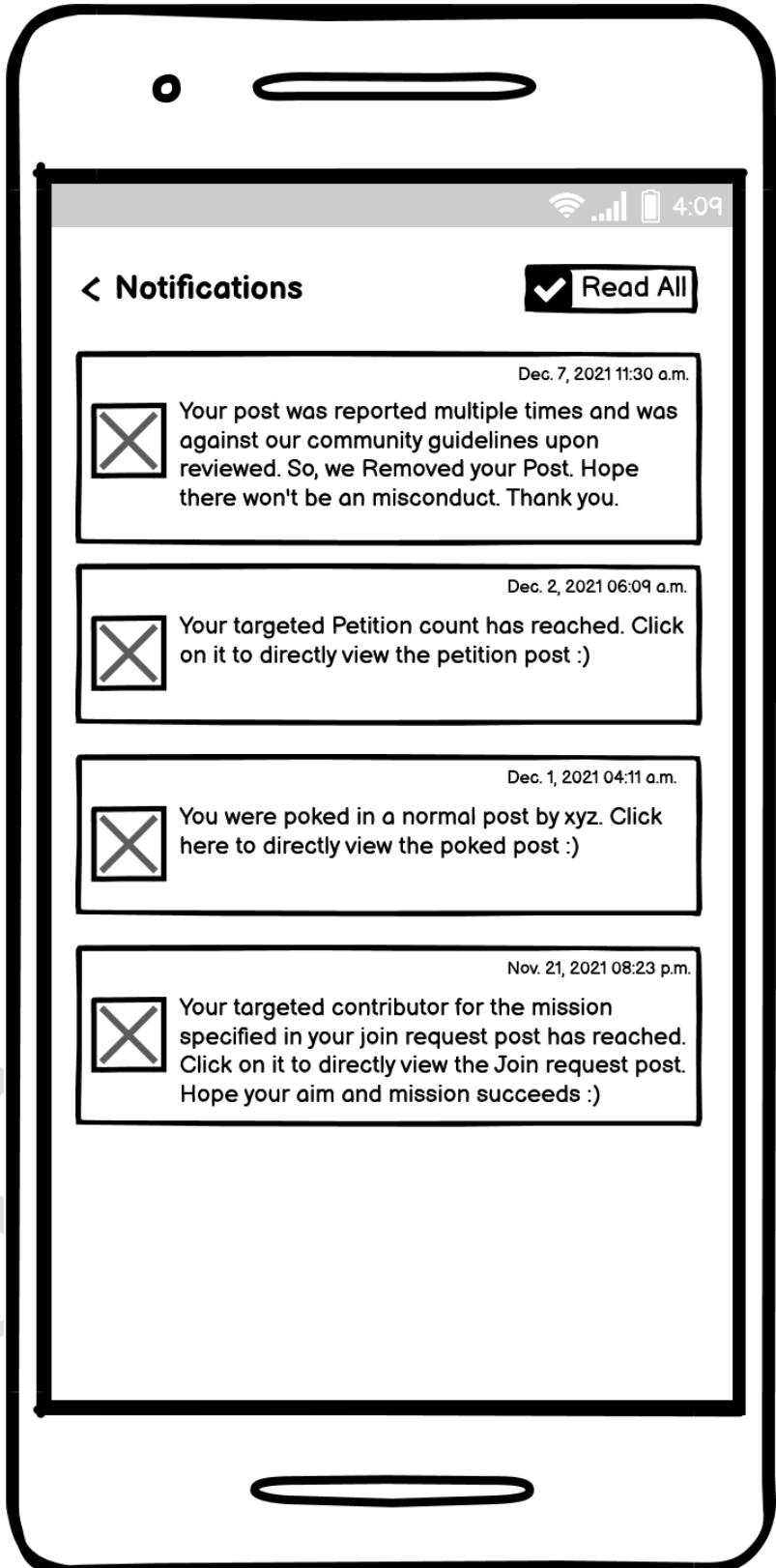


Figure 206: Mobile Notifications

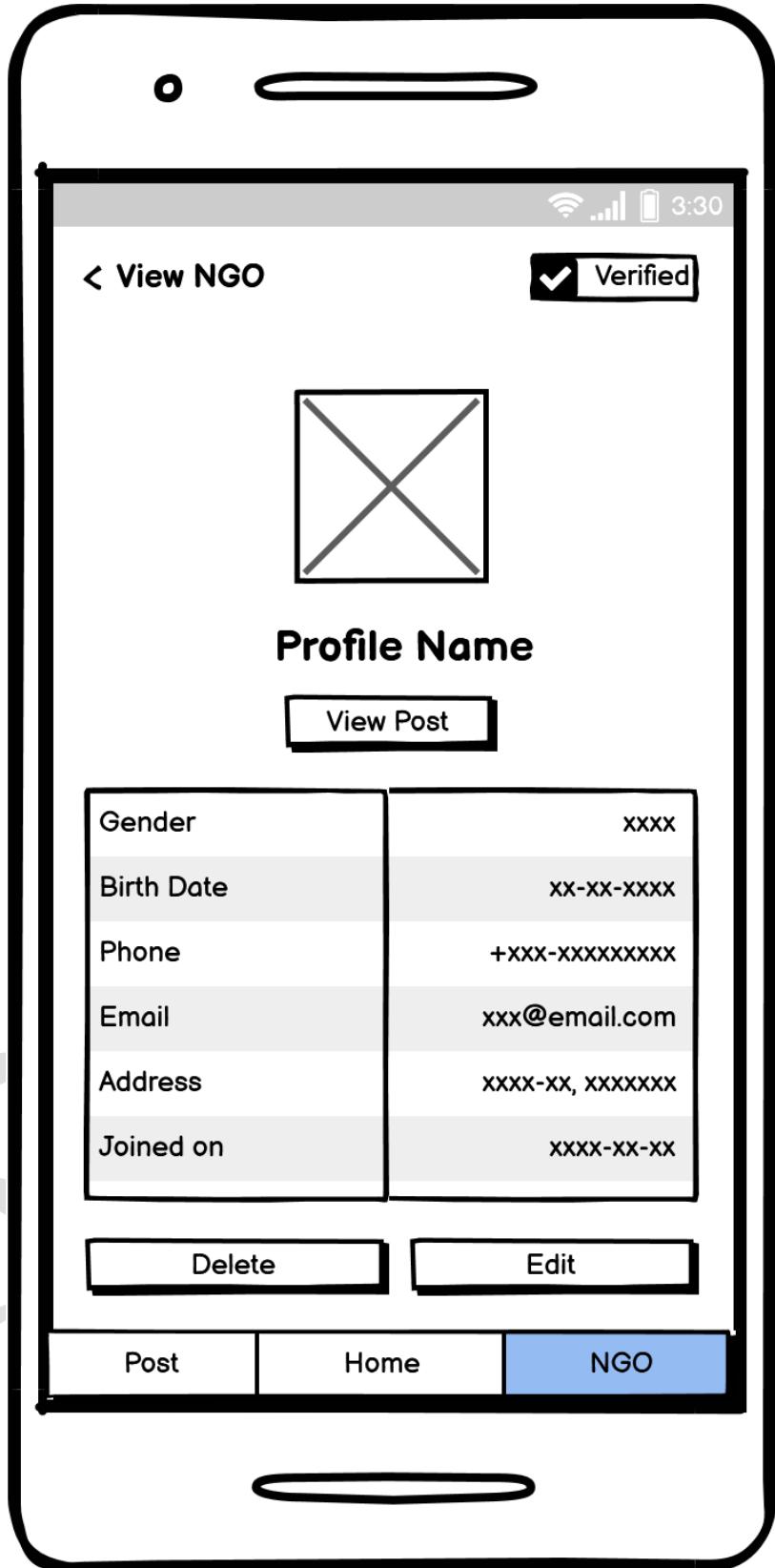


Figure 207: Mobile User Profile

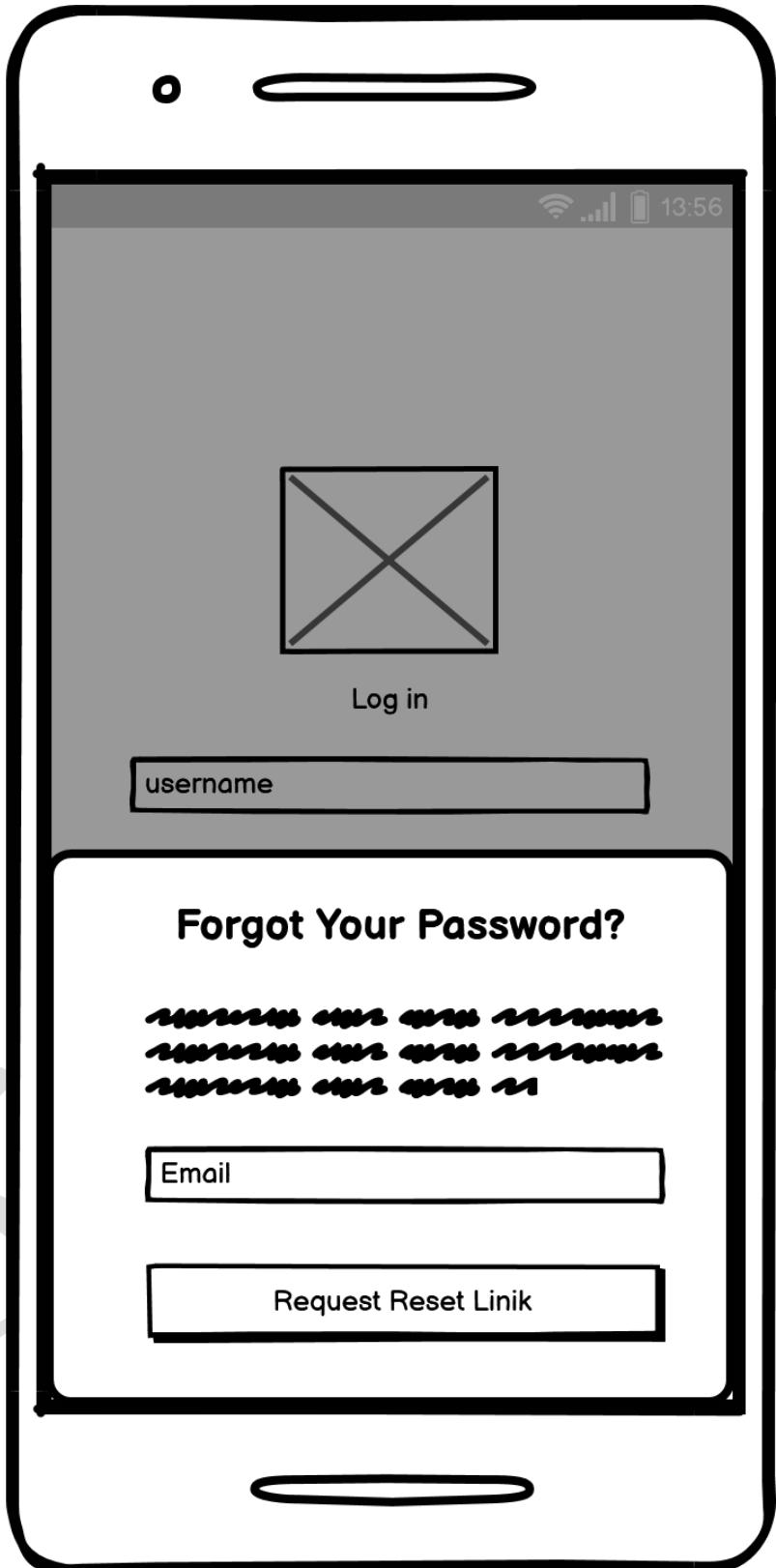


Figure 208: Wireframe – Reset Password Screen (Updated)

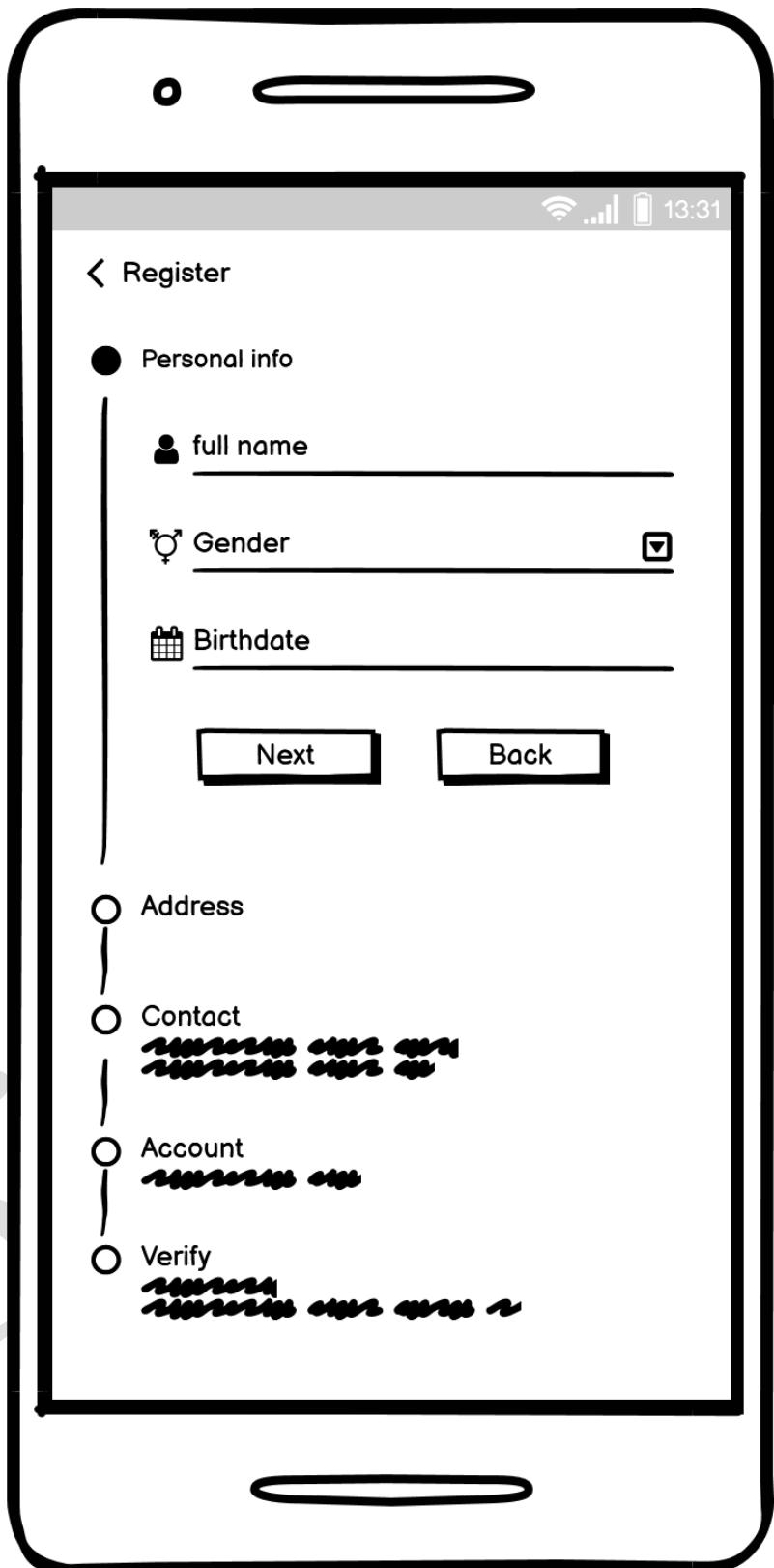


Figure 209: Wireframe – General People Register Screen (Updated)

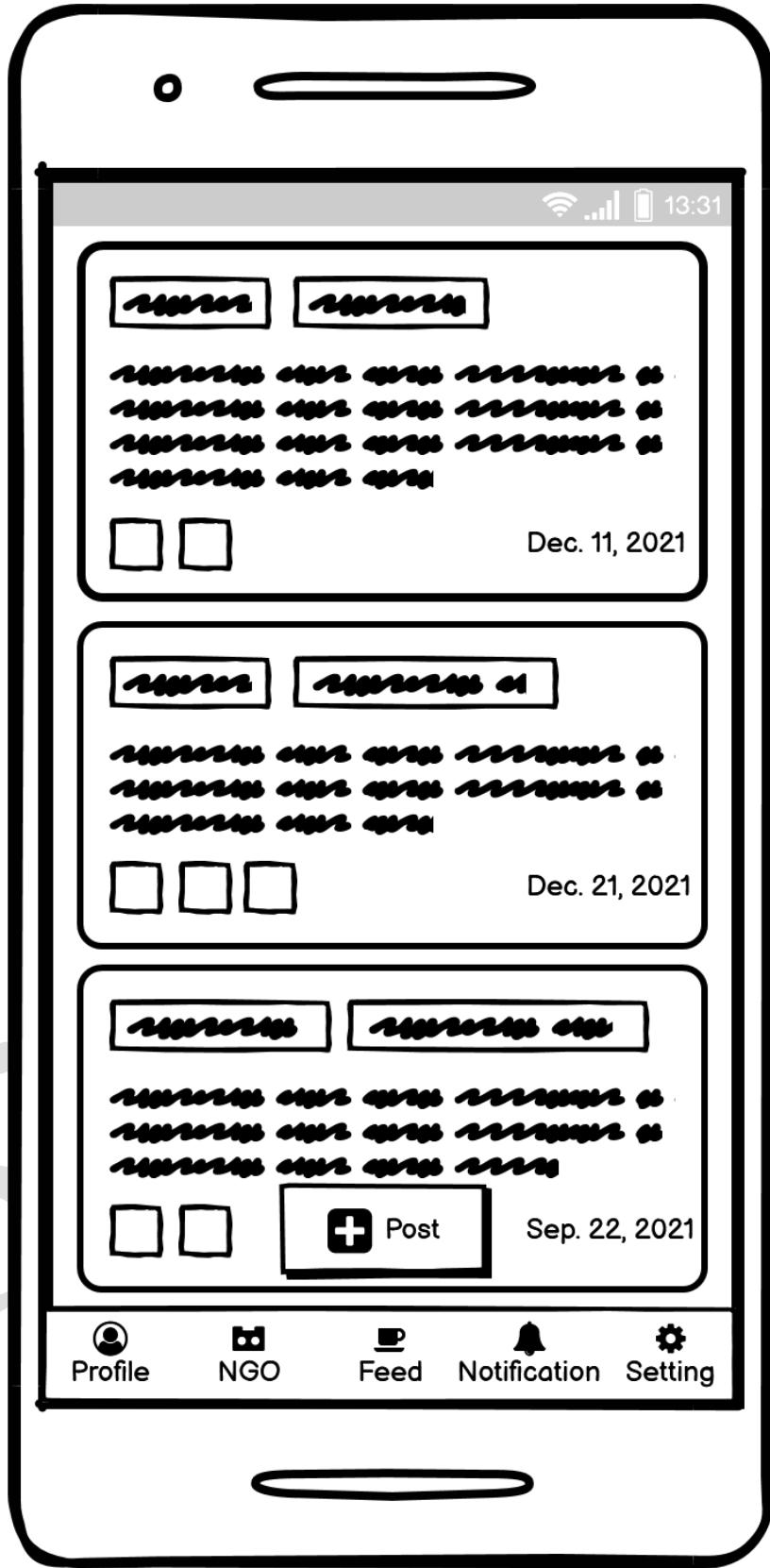


Figure 210: Wireframe - Posts Screen (Updated)

SASAE

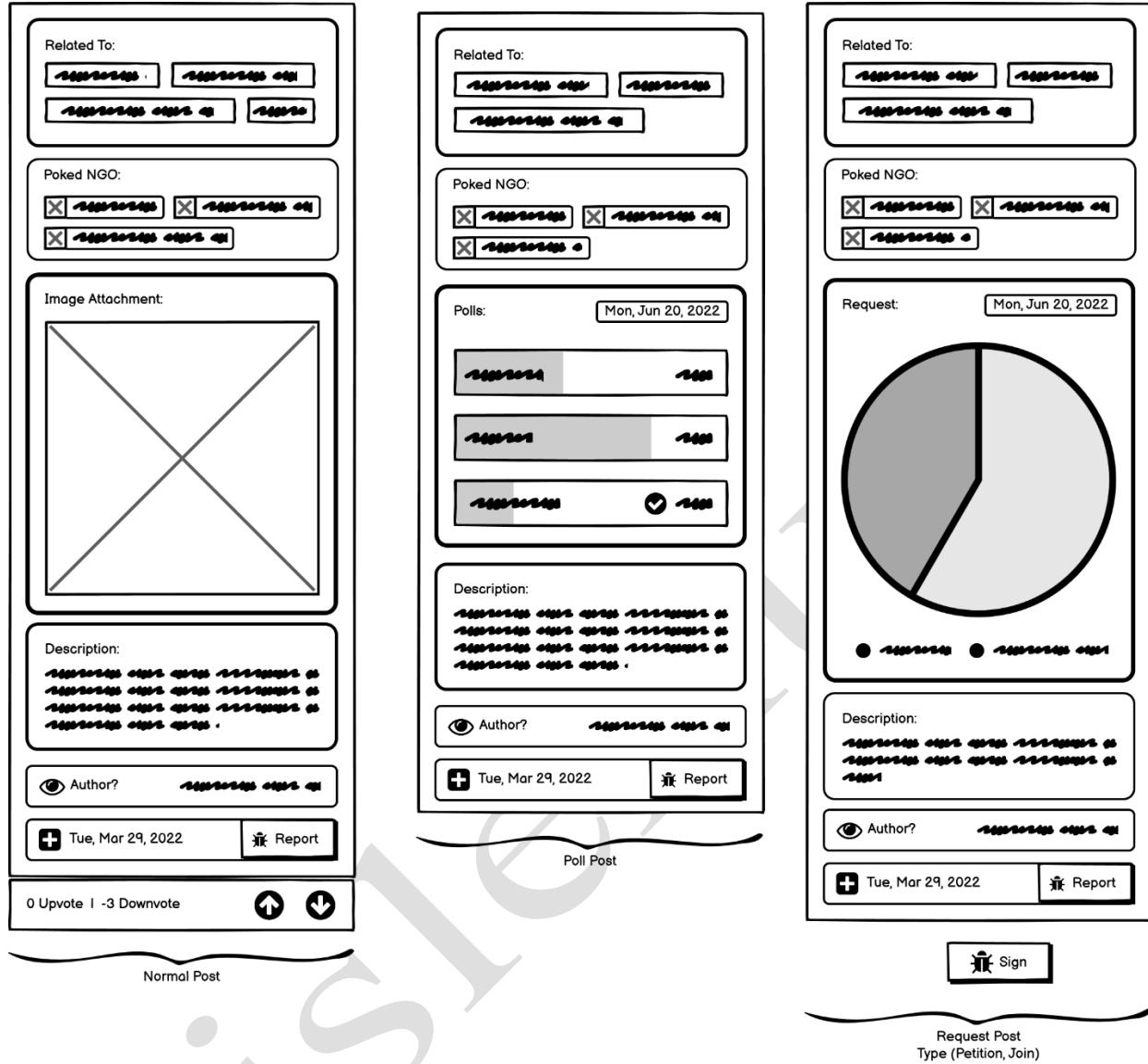


Figure 211: Wireframe – Different Types of Post Screen (Updated)

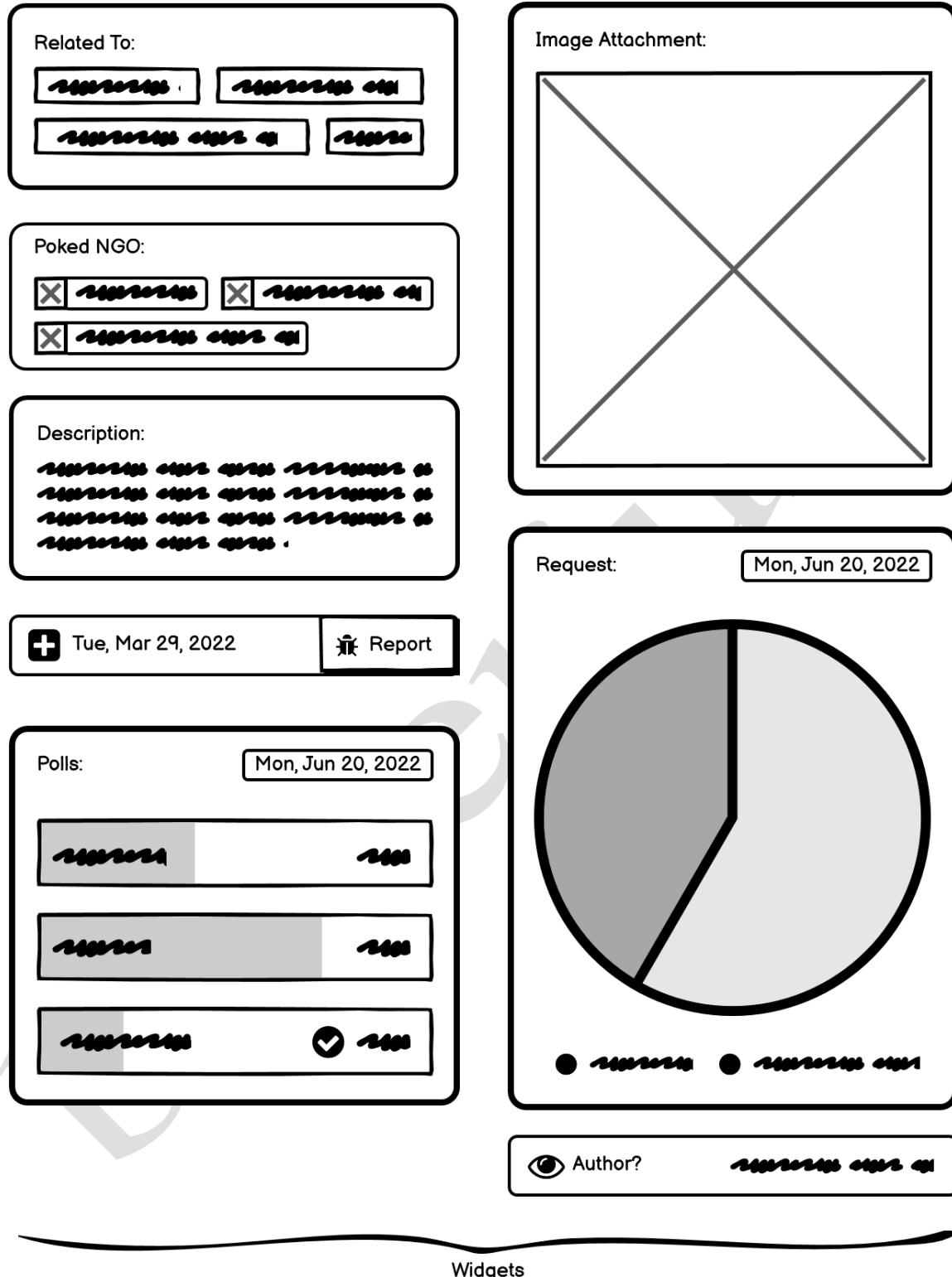


Figure 212: Wireframe – Post Type Widgets (Updated)

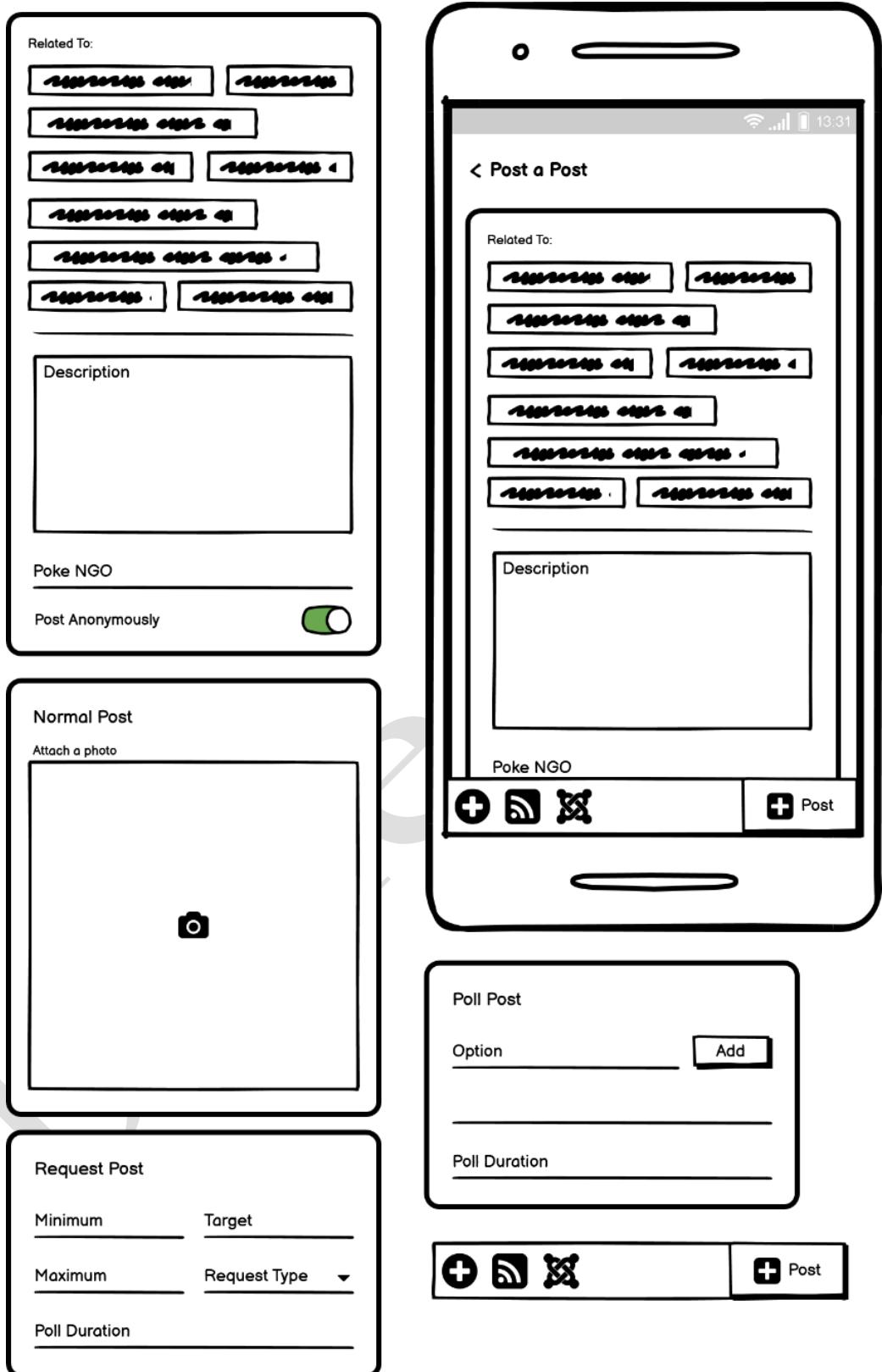


Figure 213: Wireframe – Post-Create Screen (Updated)

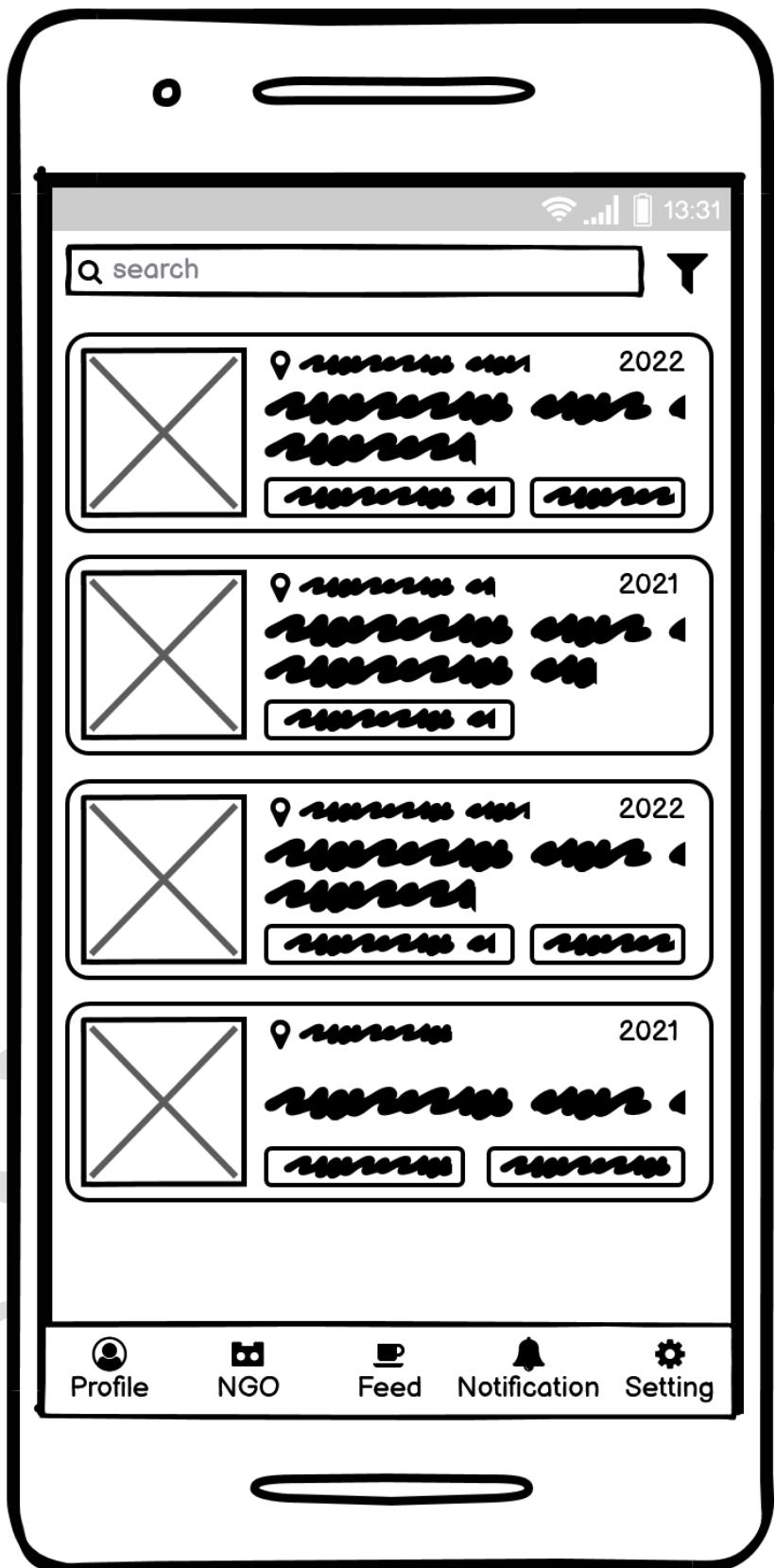


Figure 214: Wireframe – NGOs Screen (Updated)

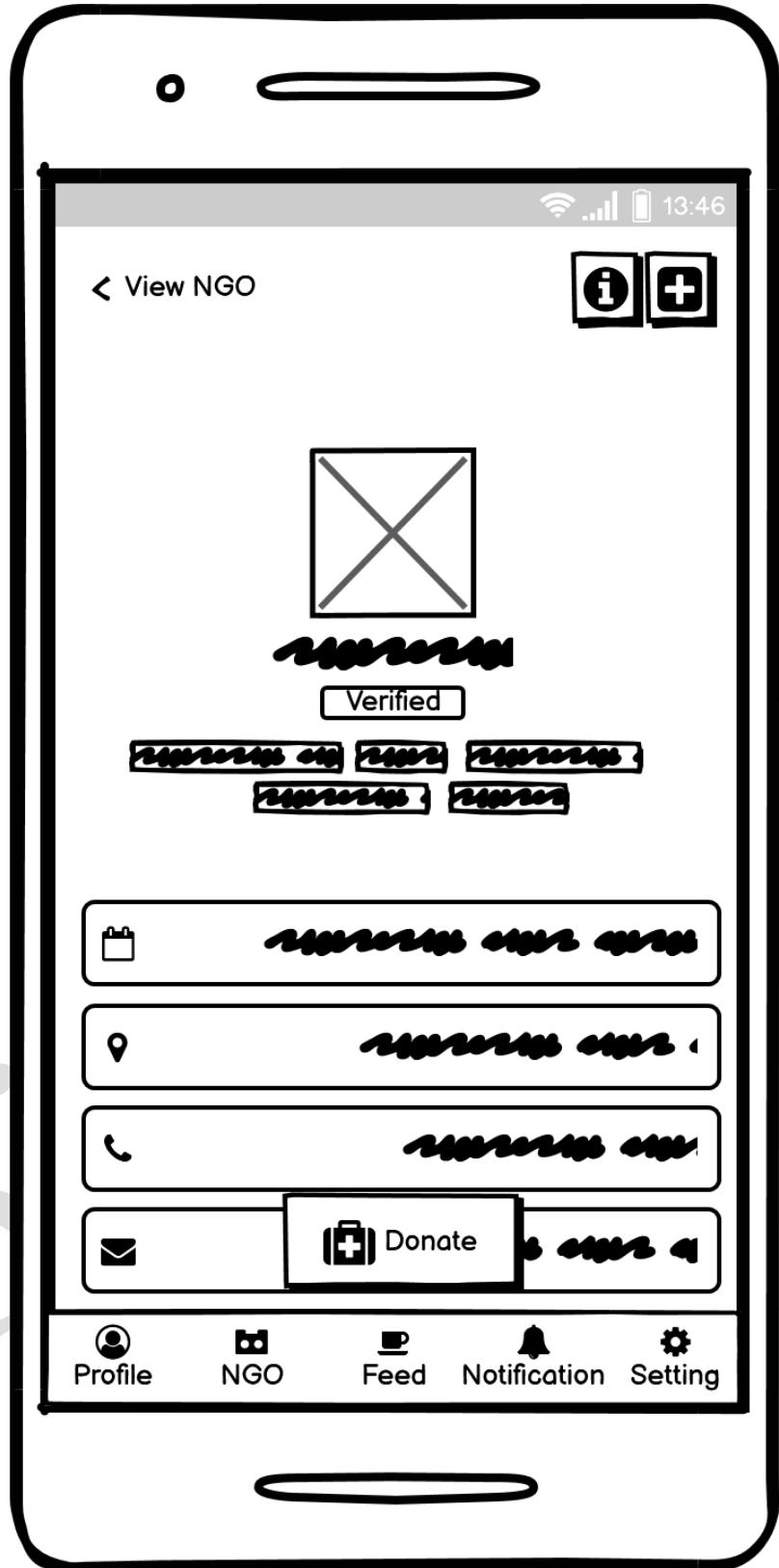


Figure 215: Wireframe – NGO Detail Screen (Updated)

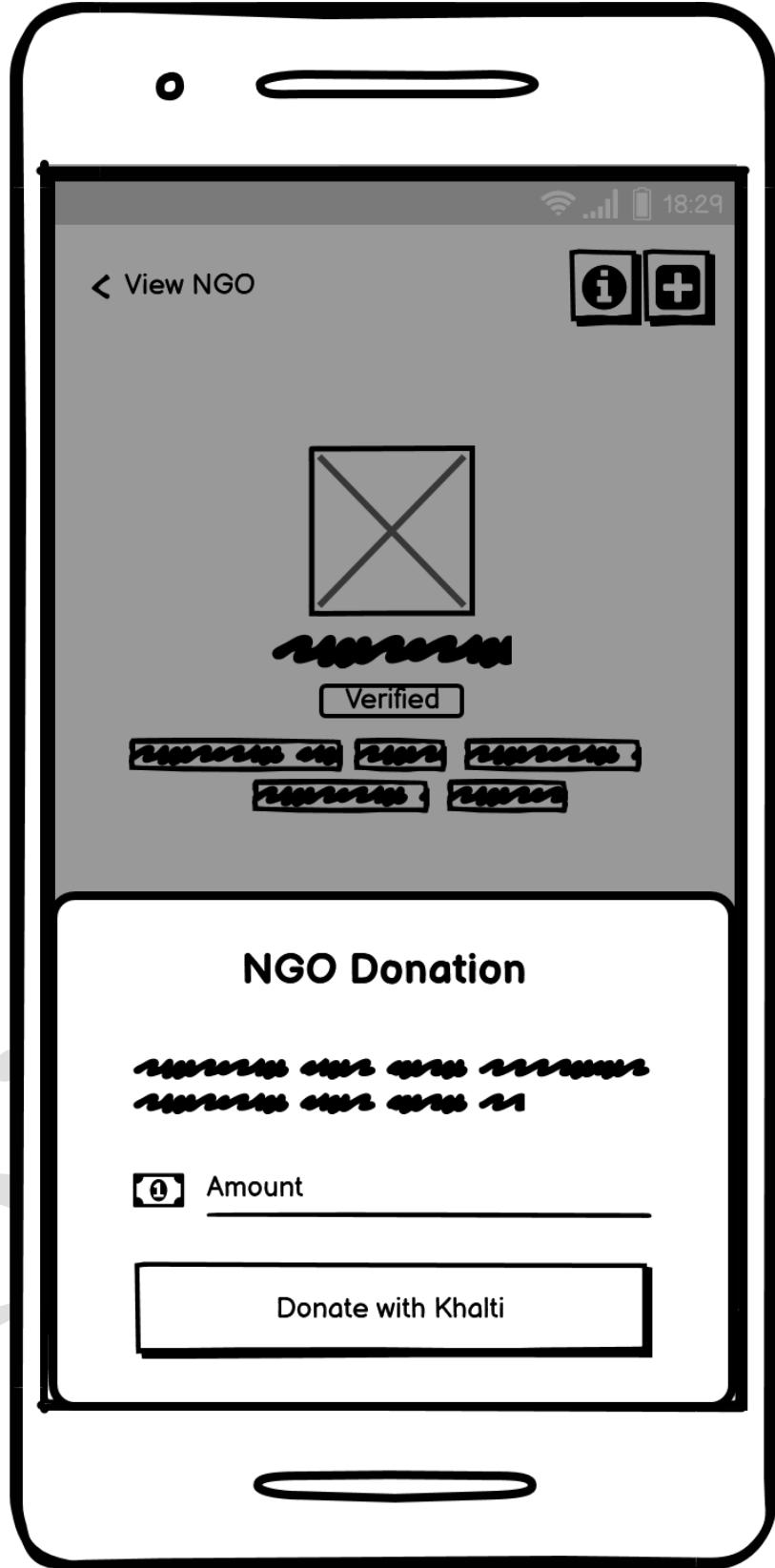


Figure 216: Wireframe – NGO Donation Screen (Updated)

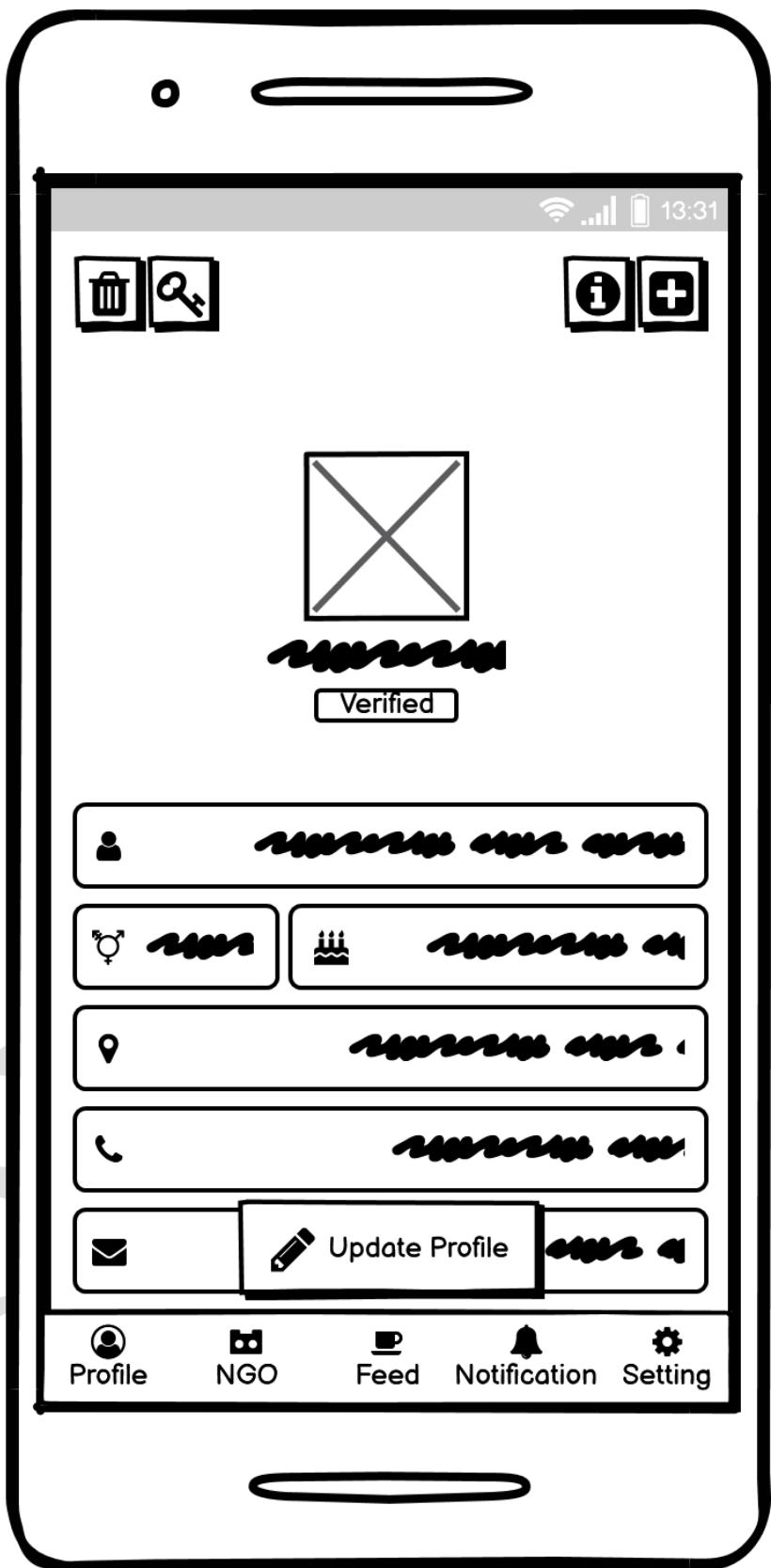


Figure 217: Wireframe –General People Profile Screen (Updated)



Figure 218: Wireframe – Change Password Screen (Updated)

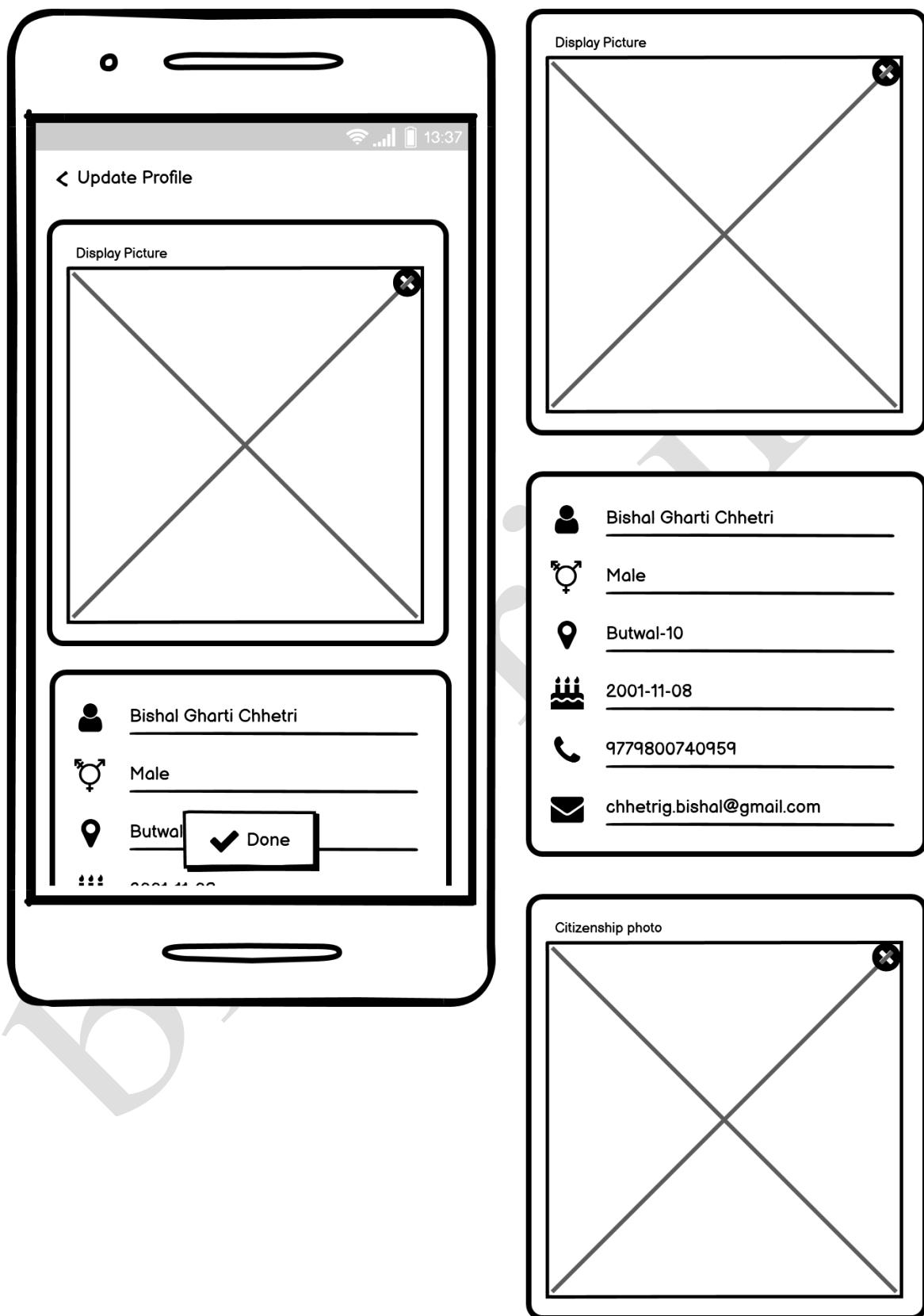


Figure 219: Wireframe – Update General People Screen (Updated)

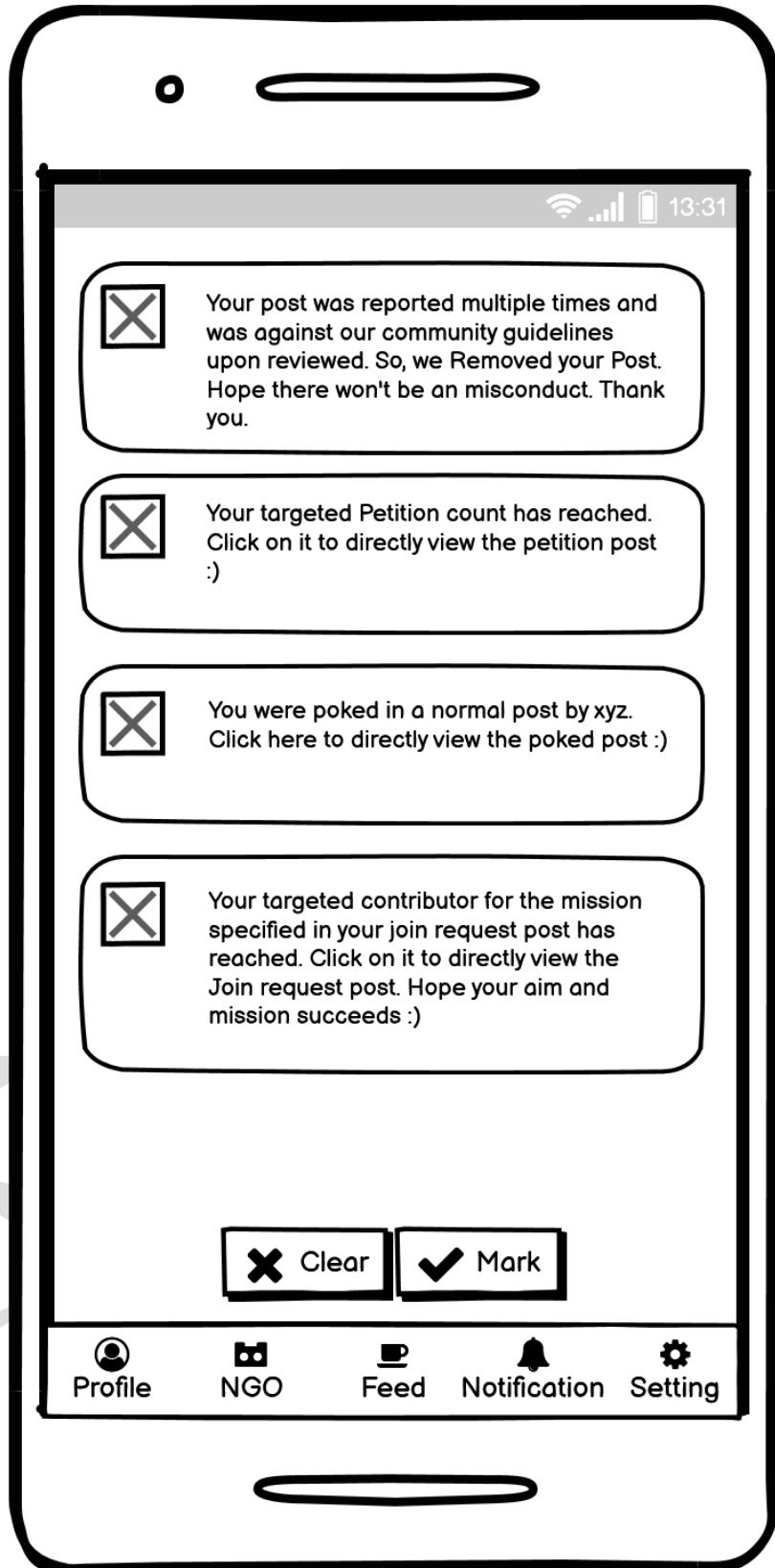


Figure 220: Wireframe - Notification Screen (Updated)

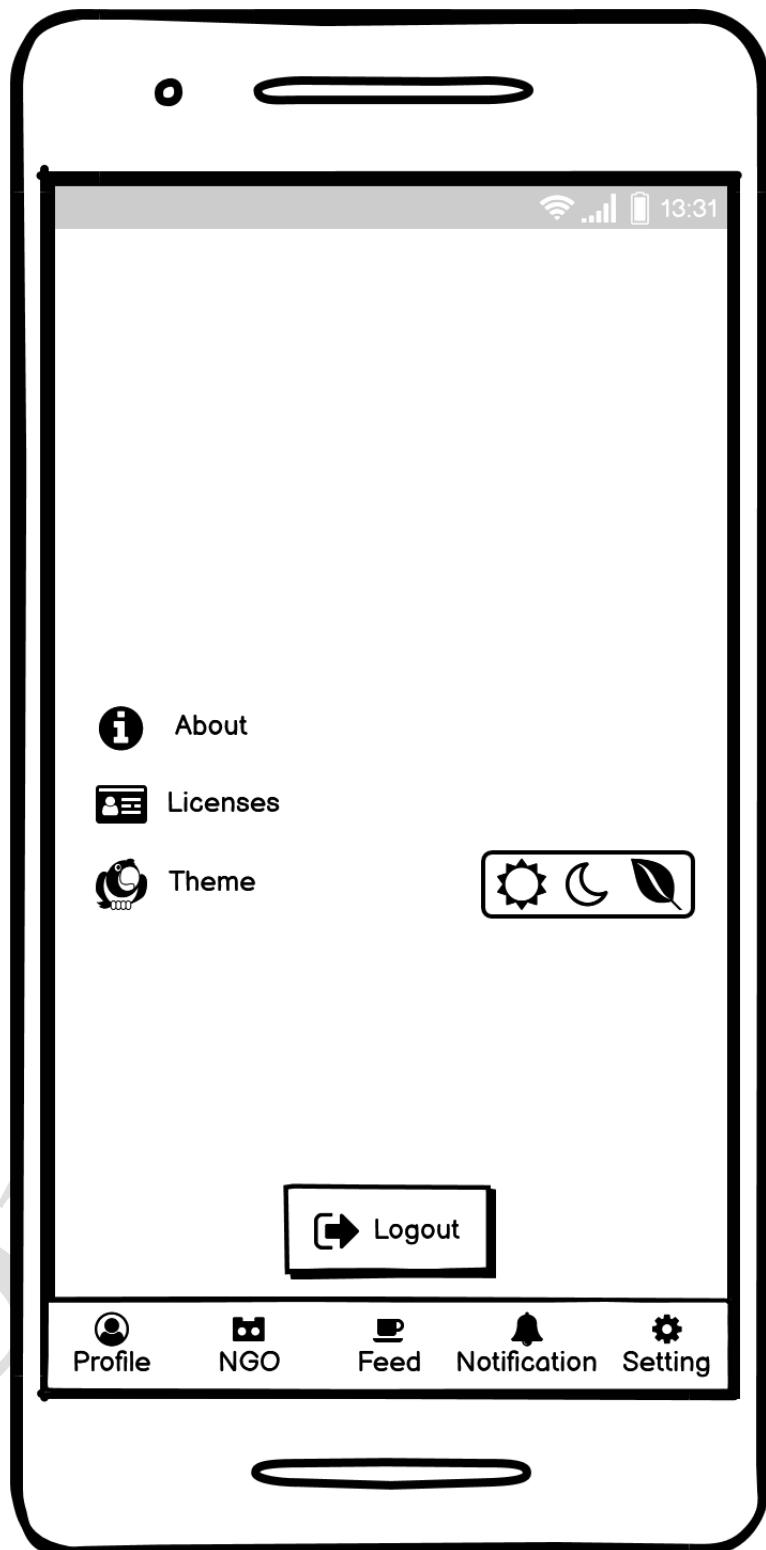


Figure 221: Wireframe - Logout Screen (Updated)

7.8.10. PROTOTYPES



Figure 222: Prototype, Posts

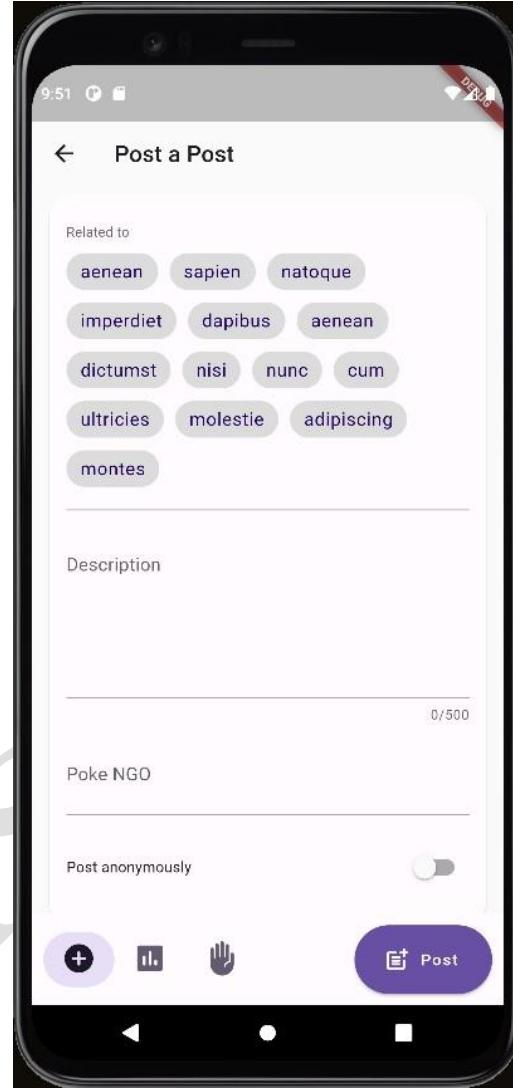


Figure 223: Prototype, Post a Post

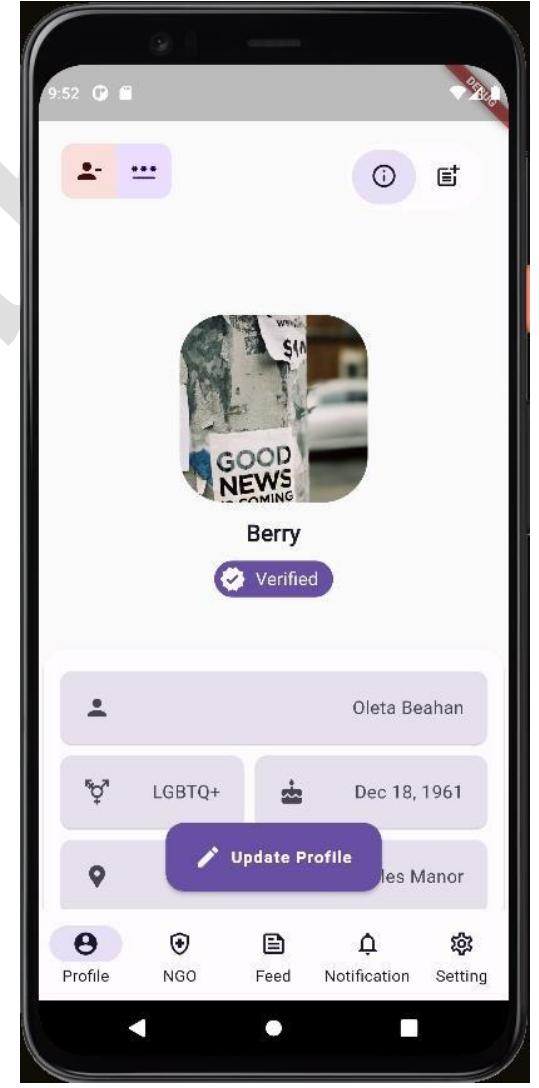
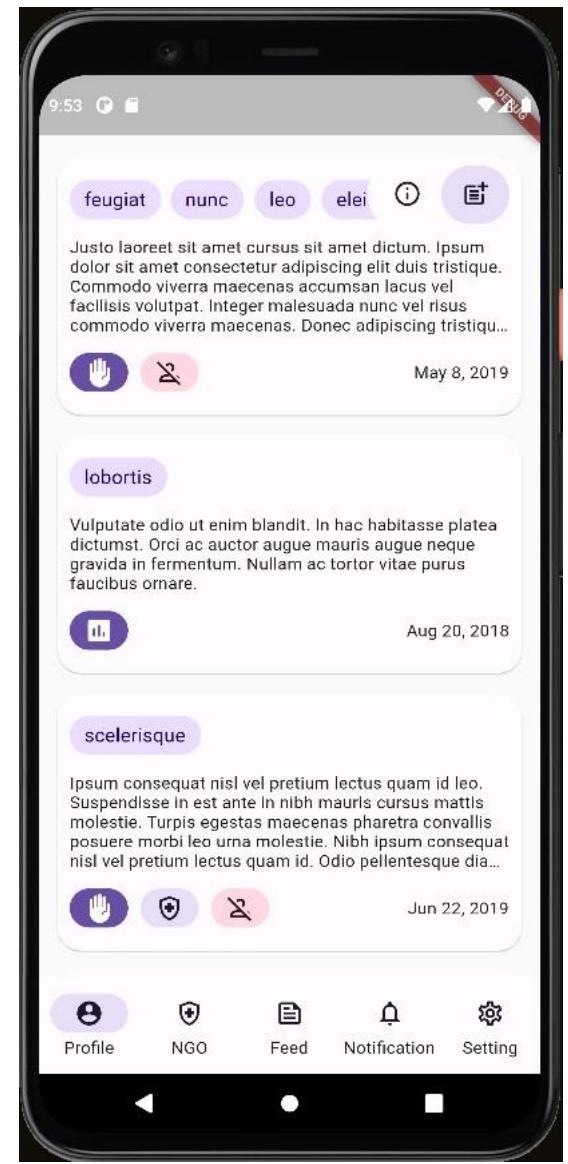
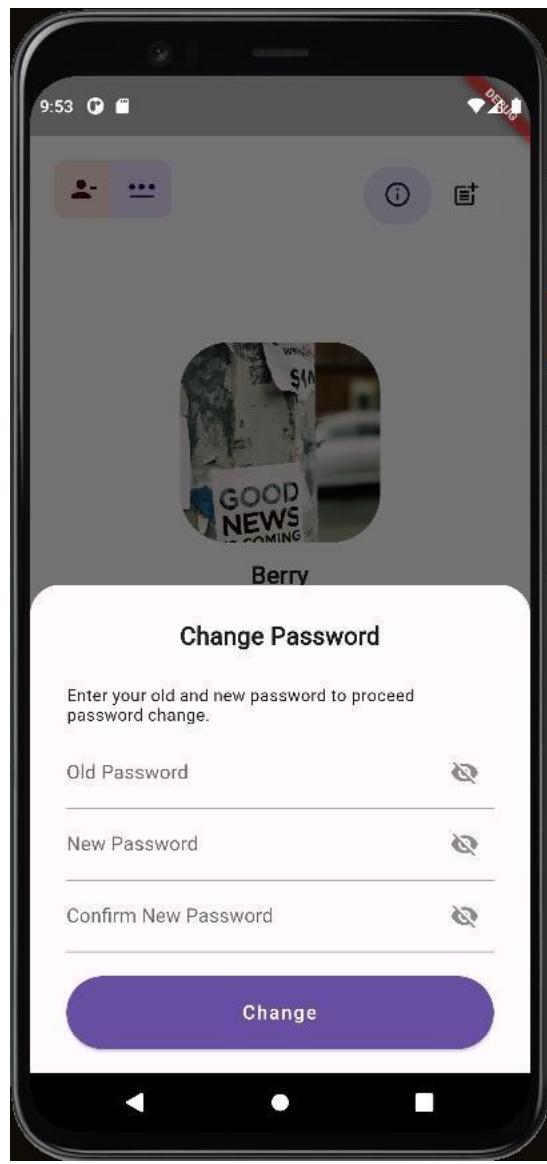
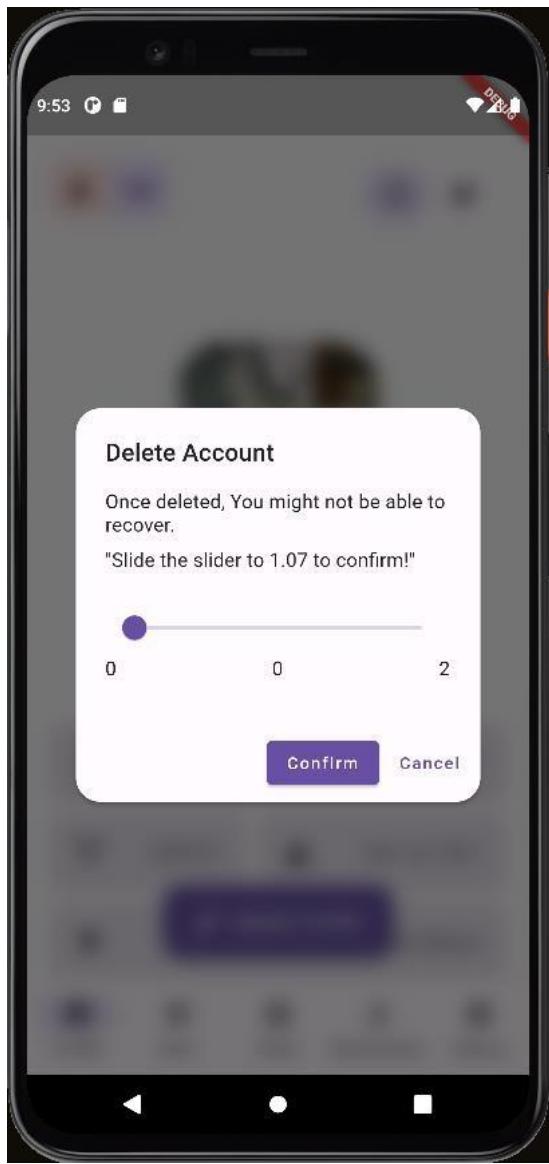


Figure 224: Prototype, Profile Screen



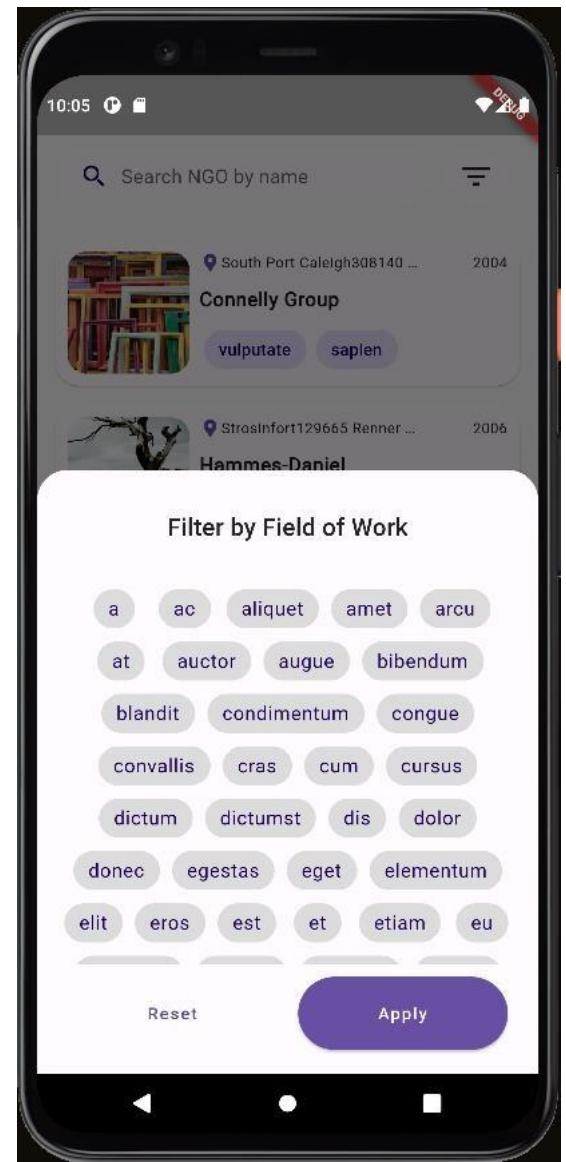
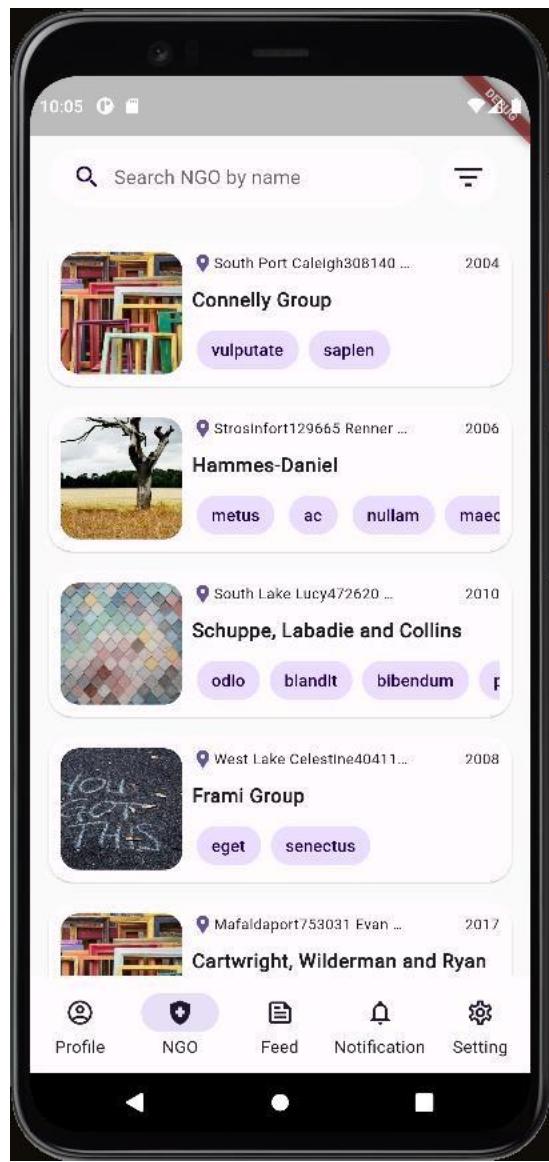
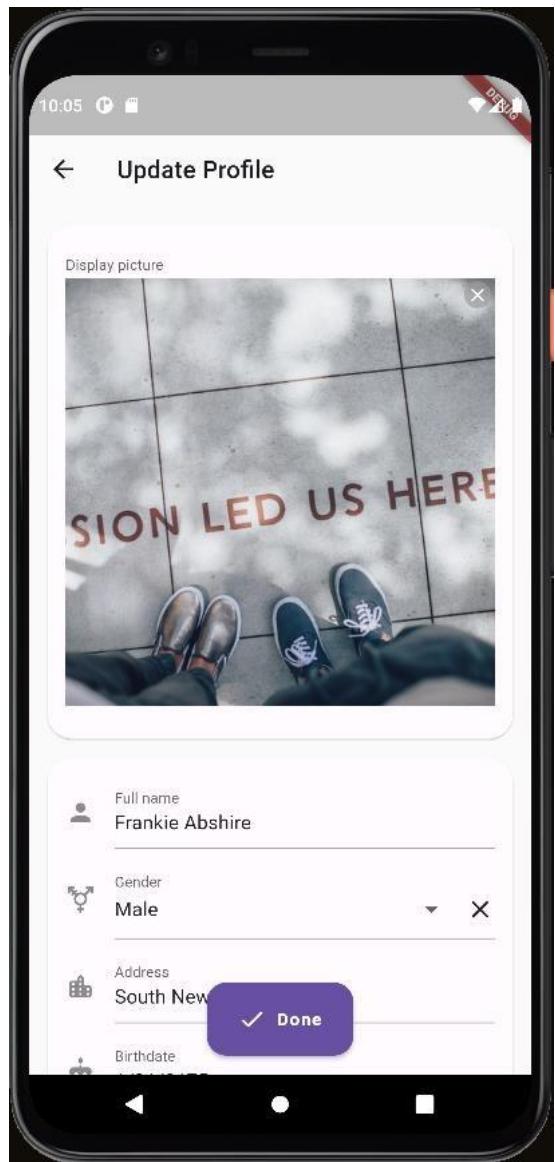
SASAE

Figure 225: Prototype, Account Delete

Figure 226: Prototype, Change Password

Figure 227: Prototype, User's Posts

SASAE



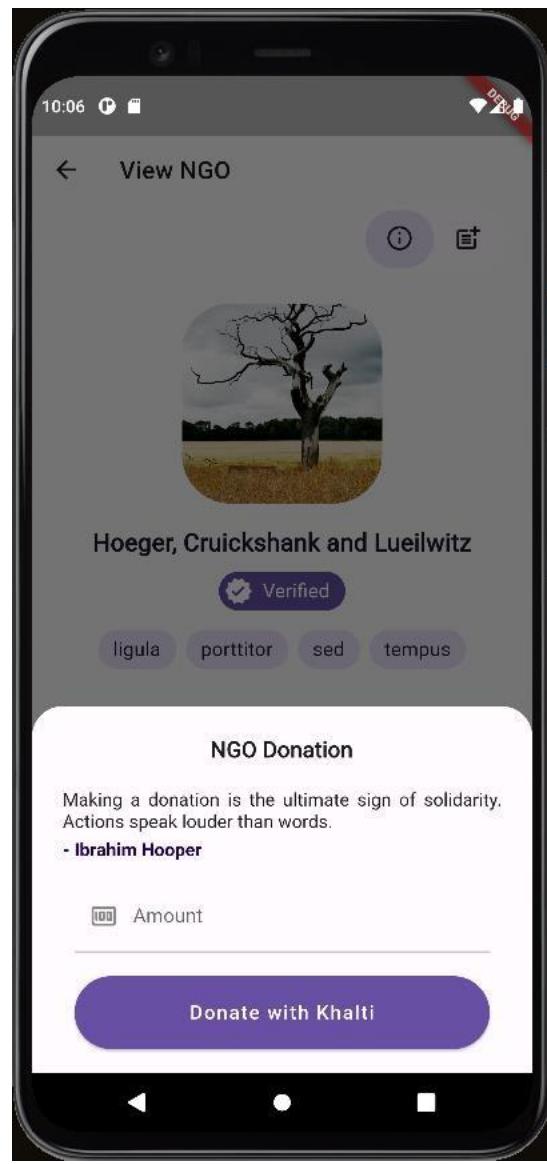
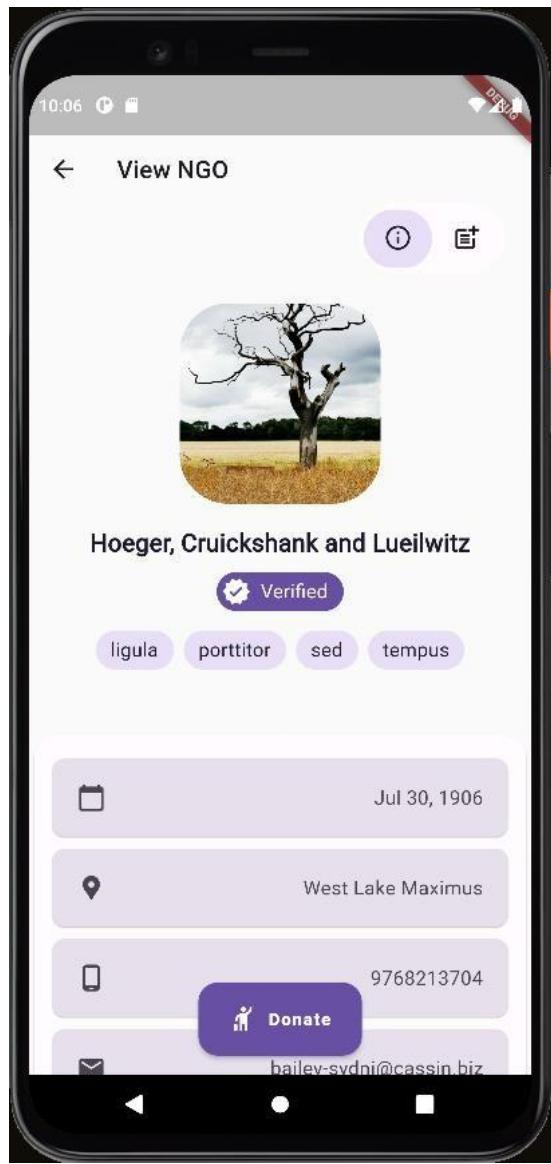
SASAE

Figure 228: Prototype, Update Profile

Figure 229: Prototype, NGOs

Figure 230: Prototype, NGOs filtration

SASAE



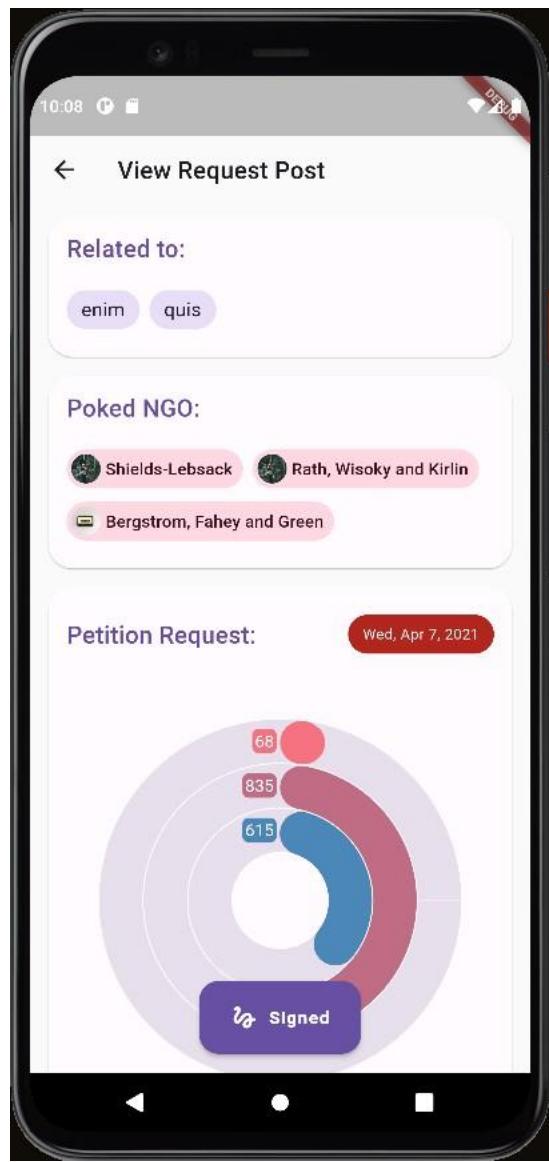
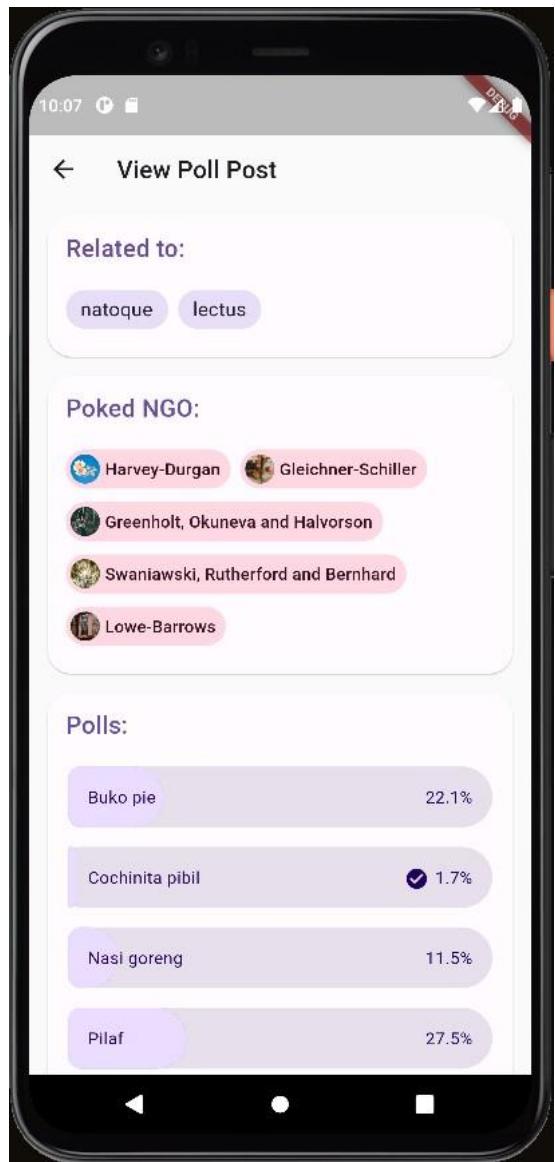
SASAE

Figure 231: Prototype, NGO Profile

Figure 232: Prototype, NGO Donation

Figure 233: Prototype, NGO's Posts

SASAE



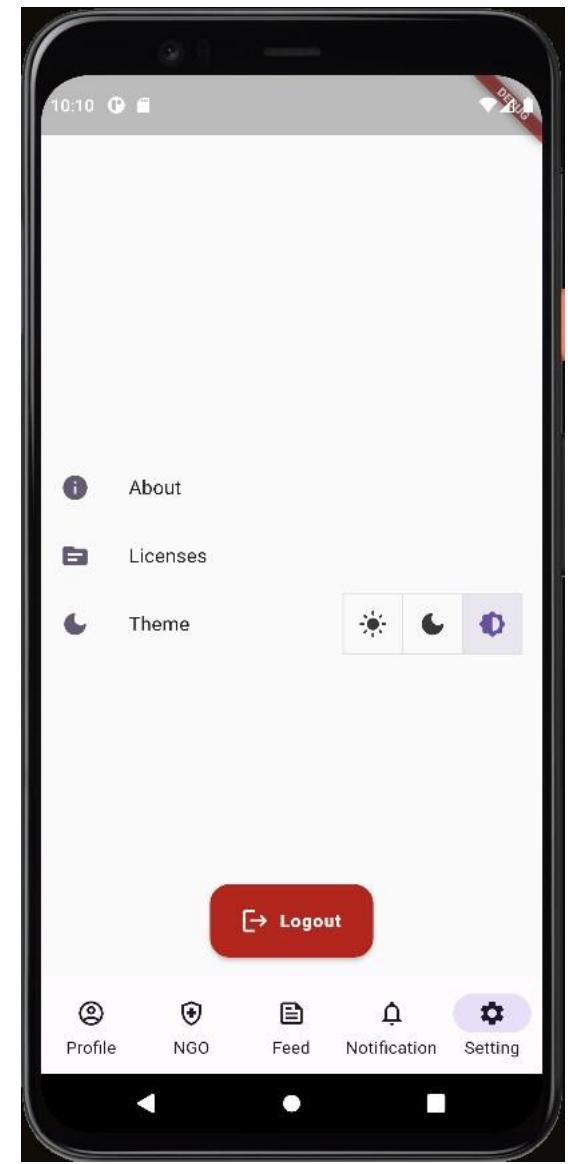
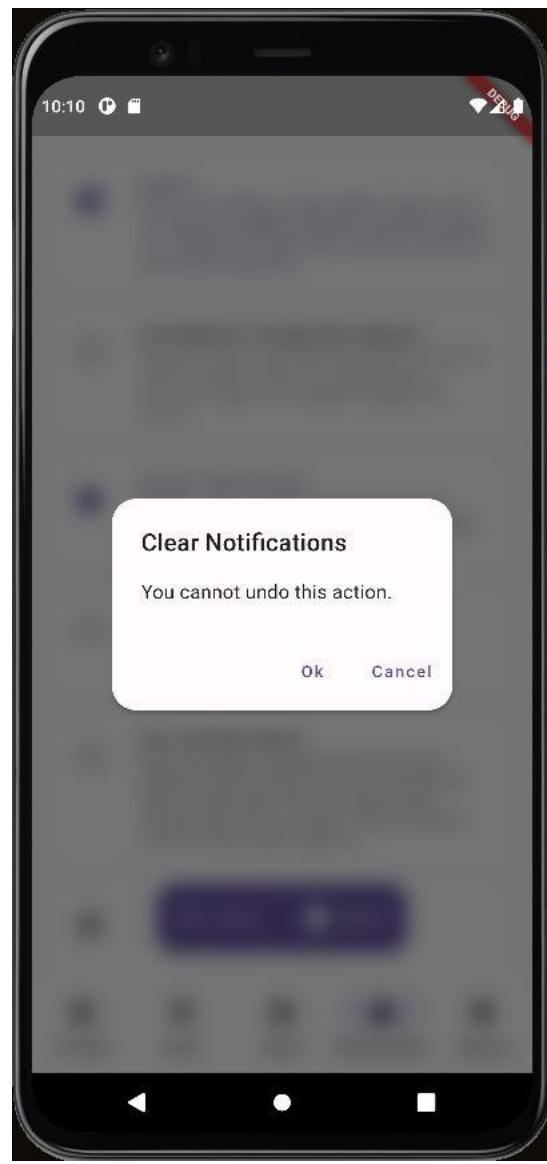
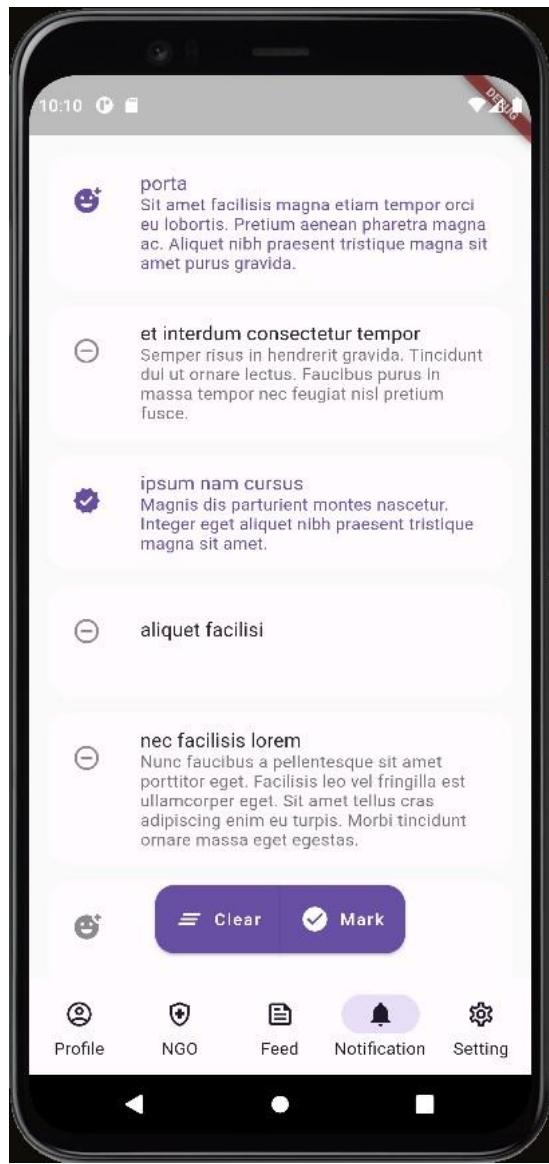
SASAE

Figure 234: Prototype, Poll Post

Figure 235: Prototype, Request Post

Figure 236: Prototype, Normal Post

SASAE



SASAE

Figure 237: Prototype, Notifications

Figure 238: Prototype, Clear Notifications

Figure 239: Prototype, Settings

7.9. APPENDIX I: IMPLEMENTATION

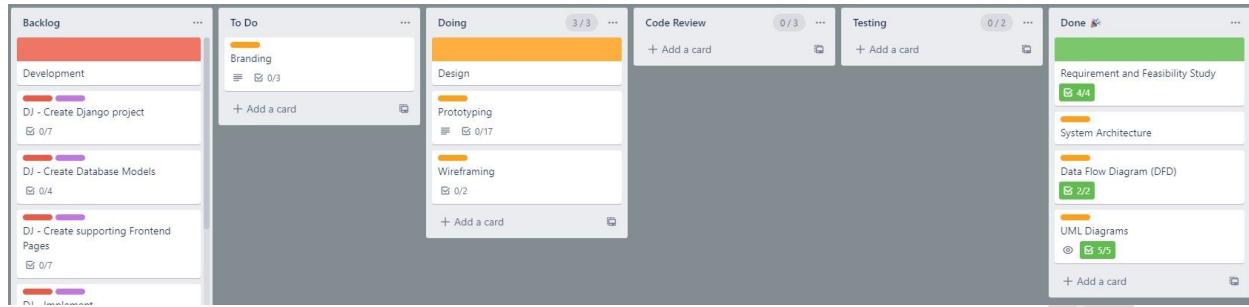


Figure 240: Kanban, Design Phase

After the Requirement and Feasibility Study is done, The Design phase of the project is initiated by dragging the ‘Design’ card from the ‘To Do’ to the ‘Doing’ column section. At first, the System Architecture illustrating the overall technical view of the whole system is created using the Draw.io web app. The system Architecture is defined under the [[2.2 UNDERSTANDING THE SOLUTION](#)] sub-heading. The Wireframe for Admin-Staff Portal and mobile app is also created afterwards. The prototyping of the mobile app was changed from Figma design to an actual Flutter application with dummy data which also falls under the development phase. With a brief understanding of the overall system, various UML diagrams: User Case, High-level Use Case Description. Communication/Collaboration Diagram and Sequence Diagram are designed to model the system design and interactions. The Use Case diagram is under [[3.6.5.1. USE CASE DIAGRAM](#)]. The High-Level Use Case Description is under the [[3.6.5.2. HIGH-LEVEL USE CASE DESCRIPTION](#)]. The Communication/collaboration diagram is under [[3.6.5.4. COMMUNICATION/COLLABORATION DIAGRAM](#)]. The Sequence diagram is under [[3.6.4.5. SEQUENCE DIAGRAM](#)]. The Class diagram is under [[3.6.5.6. CLASS DIAGRAM](#)]. The Data Flow Diagram (DFD) is also created to visualize, analyse, and improve the system processes and movement of data. The DFD is available under [[3.6.5.3. DATA FLOW DIAGRAM \(DFD\)](#)]. Finally, the branding for the project outcome is created that includes an icon, name, and slogan. The branding icon for the overall system is available under [[3.6.1. ICONOGRAPHY](#)].

SASAE

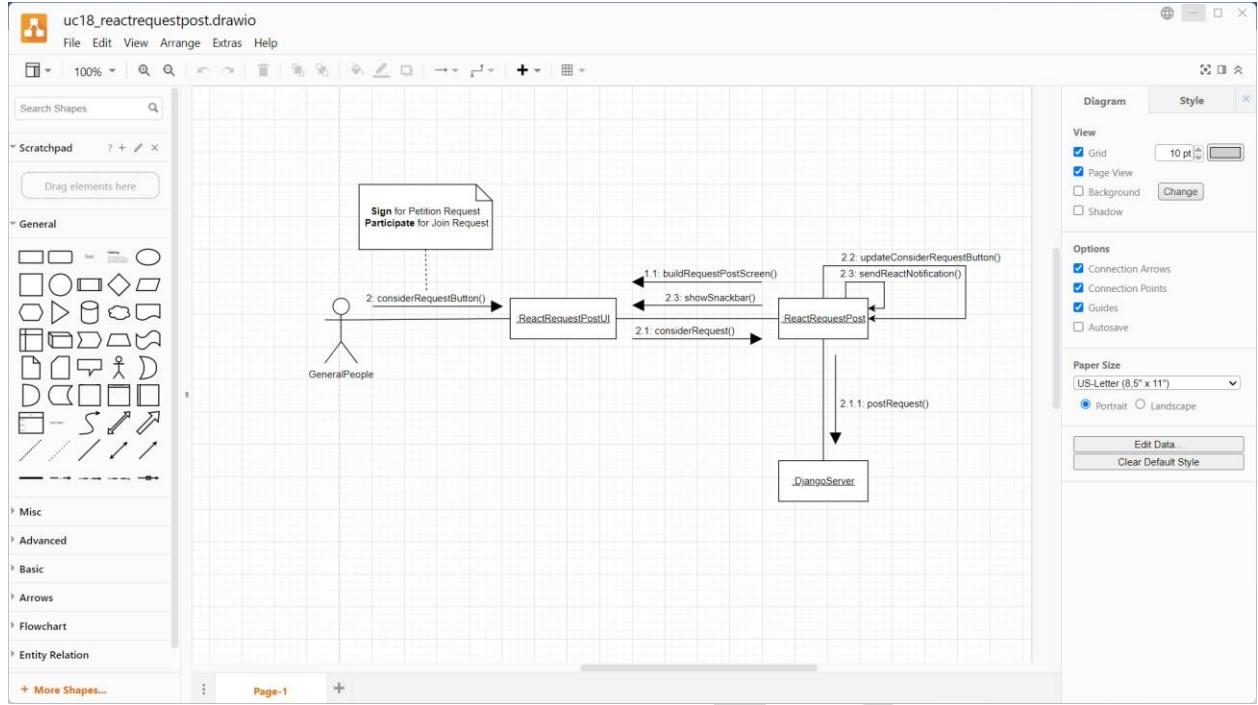


Figure 241: Communication/Collaboration Diagram on Draw.io

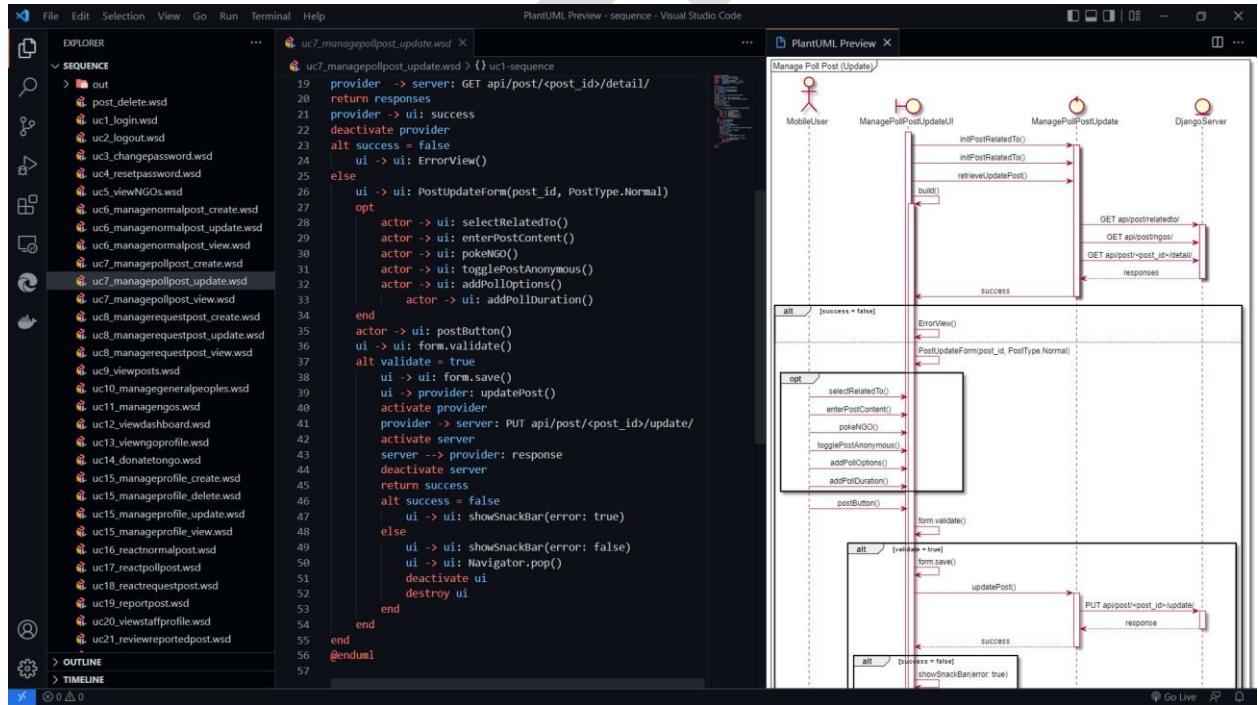


Figure 242: Sequence Diagram using PlantUML

SASAE

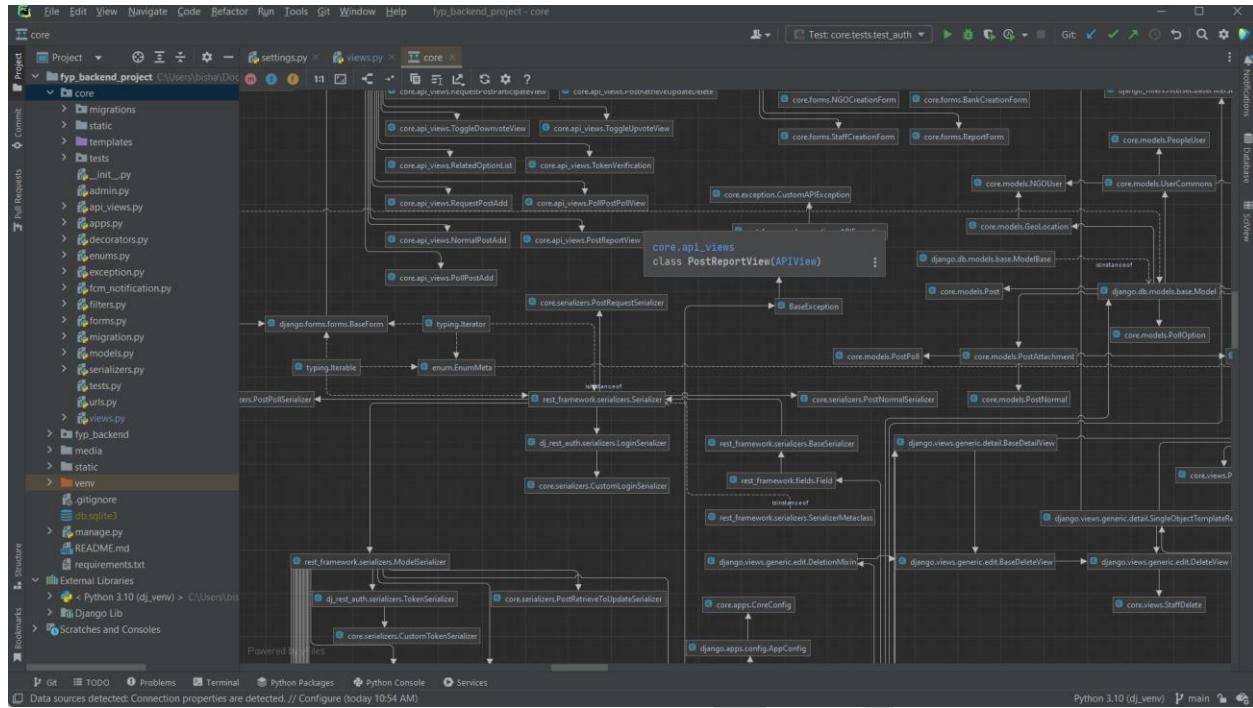


Figure 243: Class Diagram on PyCharm IDE

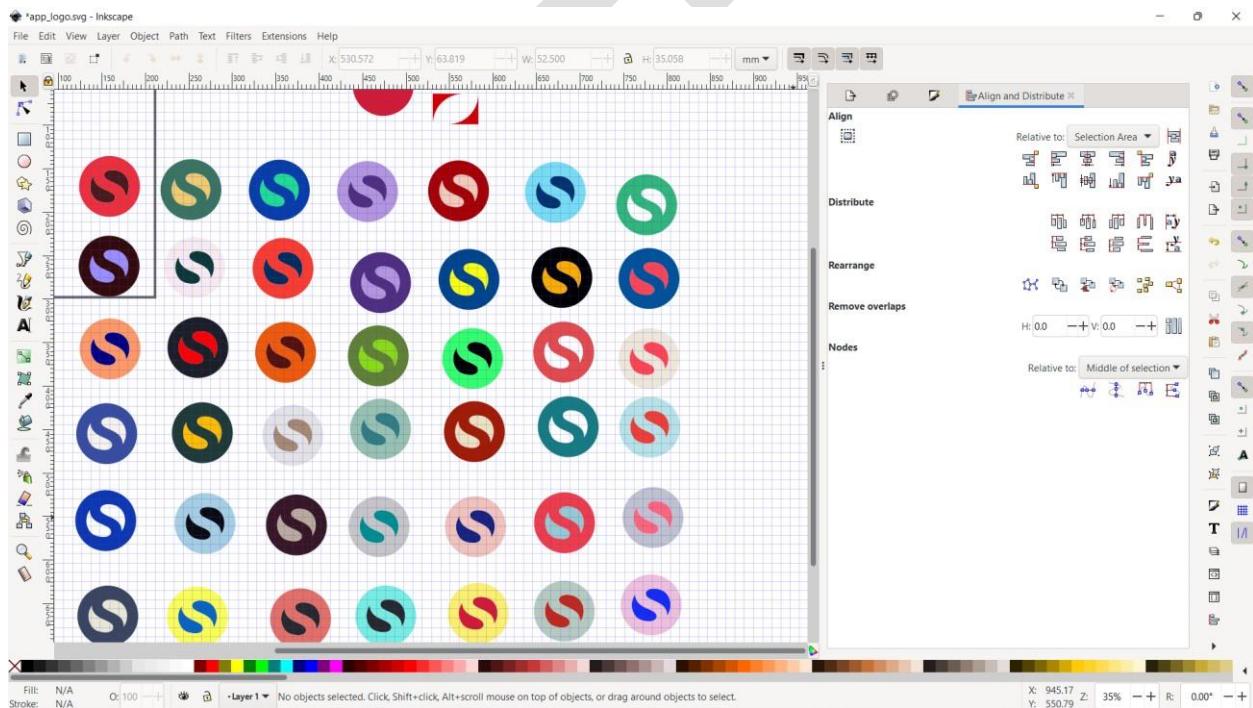


Figure 244: Icon Design on Inkscape

SASAE

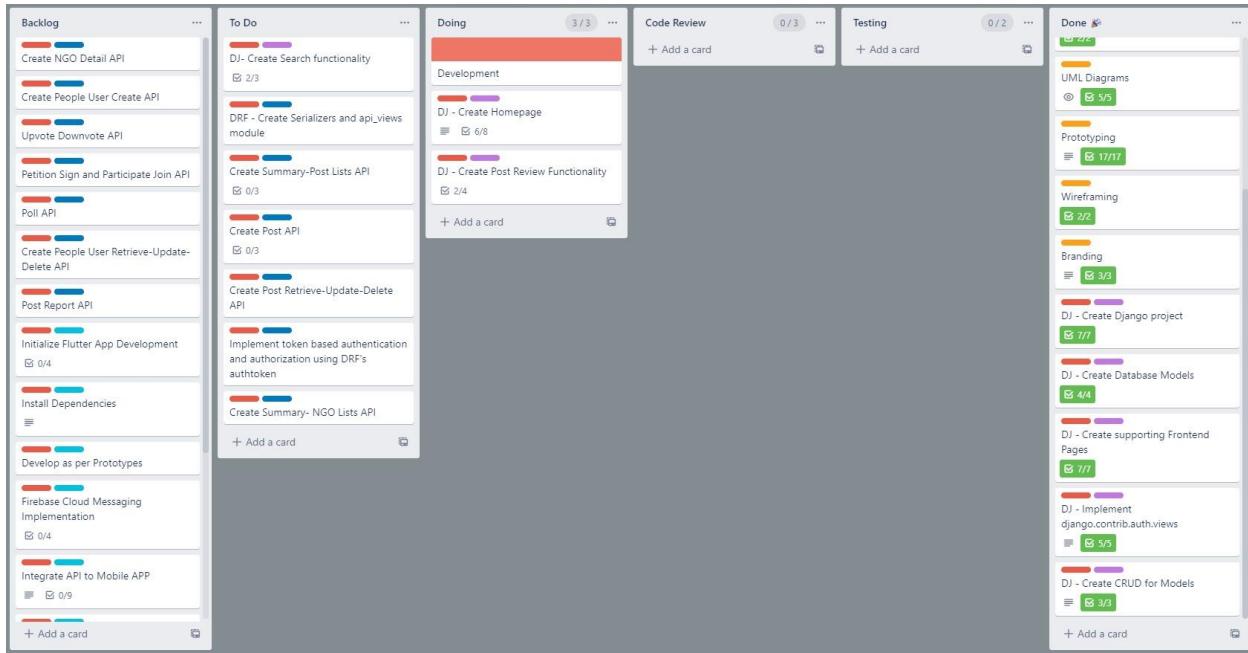


Figure 245: Kanban, Development Phase (Admin Staff Portal)

The Design completion unlocks the door to the development phase. The development phase is carried out by sub-phasing into the smaller development phases each for Admin-Staff Portal, RESTful APIs, and Sasae Mobile App. The ‘Development’ and sub cards are pulled from the ‘To Do’ to ‘Doing’ column section. The development phase is split into sub-phases each for Database, Admin-Staff Portal, RESTful APIs, and Flutter Mobile App. Initially, a Database is designed by defining and analysing all the tables, attributes, data types, relationships (i.e., Cardinality, Modality) etc. in the data dictionary and Entity Relational Diagram (ERD). The Admin-Staff Portal is then developed using the Django framework for backend handling and HTML, CSS, JS, Bootstrap etc. for frontend pages. Beforehand, All the necessary dependencies like Python, Django, Django REST, Pillow, widget-tweaks etc. were installed and a new virtual environment is created for the project’s backend. The below dependencies were used in the development of the Admin-Staff portal and RESTful APIs.

SASAE

```
(dj_venv) PS C:\Users\bisha\Documents\GitHub\fyp_backend_project> pip freeze
asgiref==3.4.1
beautifulsoup4==4.10.0
certifi==2021.10.8
charset-normalizer==2.0.12
dj-rest-auth==2.2.2
Django==3.2.8
django-bootstrap-v5==1.0.8
django-filter==21.1
django-multiselectfield==0.1.12
django-phonenumber-field==6.0.0
django-widget-tweaks==1.4.11
djangorestframework==3.12.4
idna==3.3
phonenumbers==8.12.40
Pillow==9.0.0
pipdeptree==2.2.1
pytz==2021.3
requests==2.27.1
soupsieve==2.3.1
sqlparse==0.4.2
tzdata==2021.5
urllib3==1.26.9
(dj_venv) PS C:\Users\bisha\Documents\GitHub\fyp_backend_project> python --version
Python 3.10.4
```

Figure 246: Project's Backend Dependencies

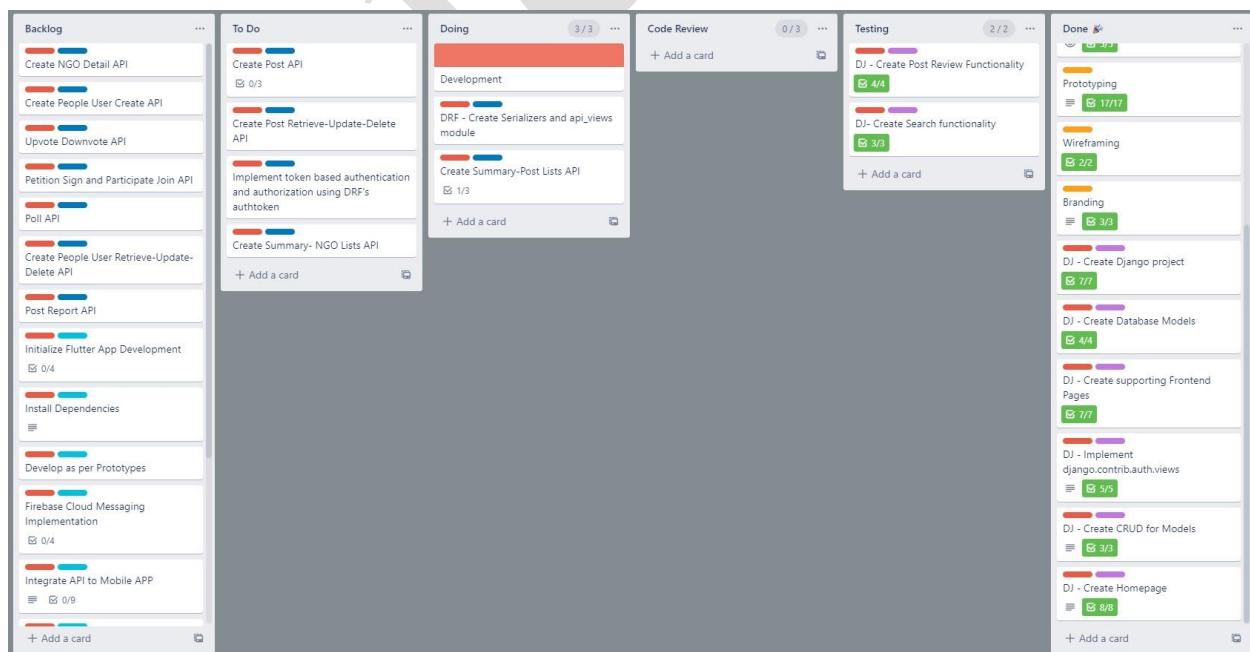


Figure 247: Kanban, Development Phase (RESTful APIs)

SASAE

The screenshot shows the PyCharm IDE interface with the following details:

- Project Structure:** The project is named "typ_backend_project" located at "C:/User/bishal/DS". It contains a "core" directory with "migrations", "static", "templates", and "tests" sub-directories. The "tests" directory contains several files like "init_.py", "base.py", "test_auth.py", "test_home.py", "test Ngo crud.py", "test_people_crud.py", "test_post_report_review.py", and "test_staff_crud.py".
- Code Editor:** The "views.py" file is open, showing Python code for handling staff/admin home pages. The code includes logic for redirecting based on user permissions (admin vs staff vs superuser) and rendering a template with context information.
- Terminal:** The terminal shows a log of HTTP requests from April 29, 2022, at 00:05:16, indicating successful GET requests for various static and media files.
- Status Bar:** The status bar at the bottom right shows "591 CRLF 4 spaces Python 3.10 (dj_venv) P main".

Figure 248: Staff/Admin Home Feature Implementation as per Kanban

Then, All the colour labelled cards that distinguish as Development sub-tasks are completed sequentially along with necessary testing (i.e., Unit Testing, System Testing). Most of the sub-tasks involve creating views and corresponding web pages. During development, some features mentioned in the development sub-tasks were complex and hard to implement, obstructing the development progress. As Kanban is flexible to resolve such problems causing bottlenecks, another queued task was pulled to ‘Doing’ and was implemented through the halted time.

SASAE

The file shown is `post_form.dart` from the `sasae.flutter_app` project. The error occurs at line 79:

```
void setState() or markNeedsBuild()
called during build.
This ScaffoldMessenger widget
cannot be marked as needing to
build because the framework is
already in the process of
building widgets. A widget can
only be marked as needing to be
built during the build phase only if
one of its ancestors is currently
building. This exception is
allowed because the framework
builds parent widgets before
children, which means a dirty
descendant will always be
built. Otherwise, the framework
might not visit this widget
during this build phase.
```

The widget on which `setState()` or `markNeedsBuild()` was called was: `ScaffoldMessenger`

The widget which was currently being built when the offending call was made was: `FutureBuilder<List<dynamic>`

See also: <https://flutter.dev/docs/testing/errors>

Figure 249: Code Error during Development

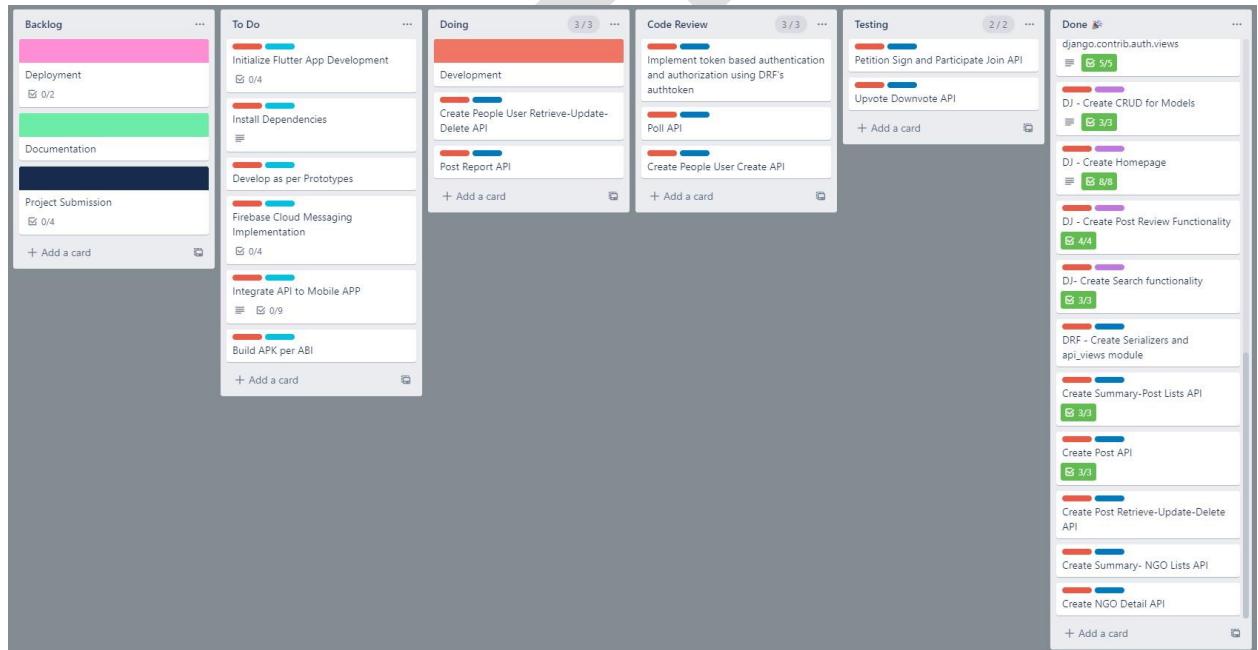


Figure 250: Kanban, Continuous Development and Testing

The Admin-Staff Portal mostly includes CRUD operations on the user: General People,

NGO, and Staff with role-accessibility. It also includes some complex functionality that involves reviewing reported posts and reading the reviews. The portal was almost completed in around late December. Finally, Unit testing and System testing are performed on the Admin-Staff portal. The test plans and tests are available at [d. [TESTING AND ANALYSIS](#)]. Afterwards, The RESTful APIs development was initialized. All the API endpoints required by the frontend mobile application were recognized and created along with the class-based functions to handle them. Per API endpoints, either function-based or class-based views were created along with serializers. The serializers are used to both serialize and de-serialize to and from JSON respectively. DRF's auth token is then implemented to restrict unauthorised HTTP requests to the server. The API request mostly performs CRUD operation on General People and Post. Most Django RESTful APIs are developed parallel to the API integration in mobile apps.

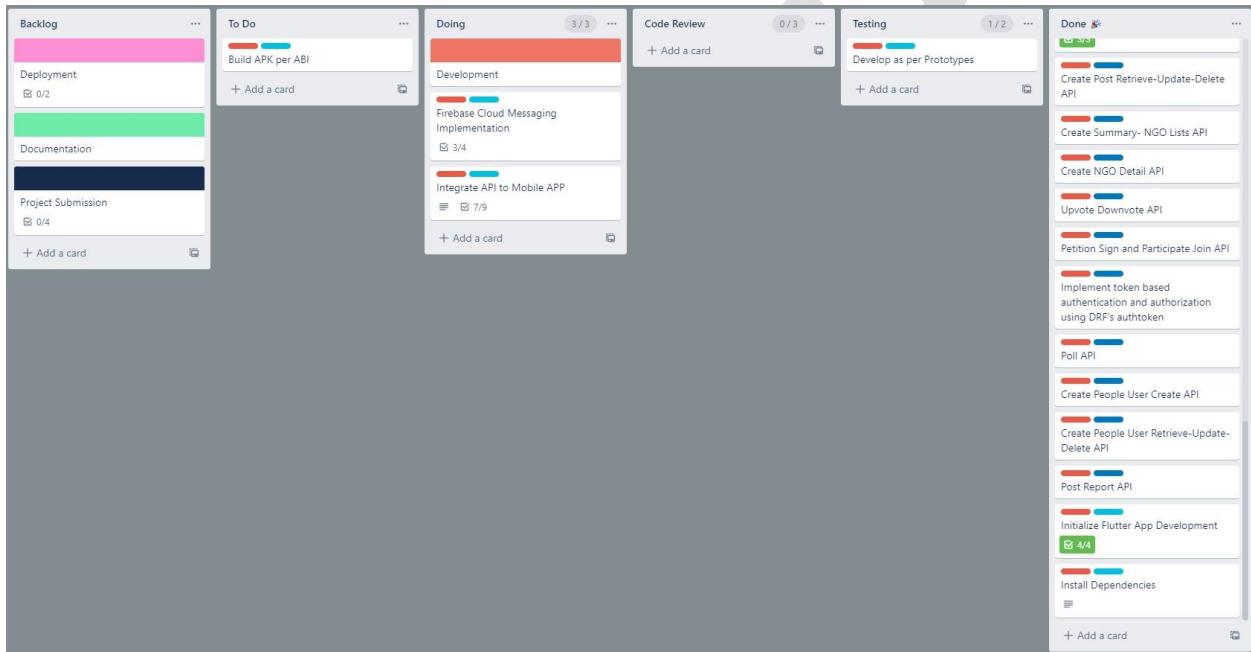


Figure 251: Kanban, Development Phase (Sasae Mobile App)

After the completion of the Admin-Staff Portal, along with the development of REST APIs, the development of the mobile app using the flutter framework is initiated. Every UI component, animation and transition that deals with the change in screen pixel(s) is considered a widget in a flutter. Initially, the mobile app is prototyped using the flutter and faker package. The app is made up of multiple widgets and custom extended/overridden widgets. The prototyped mobile app is then

tested and moved to the API implementation phase. Multiple providers/services are created using Provider, HTTP, and DIO packages to utilize RESTful services from the remote Django server. The UI/UX of the application is then improved. During the development of the mobile app, some common problems related to media queries and device viewports were faced multiple times. Finally, Unit testing and System testing are performed on Mobile App. The test plans and tests are available at [[d. TESTING AND ANALYSIS](#)].

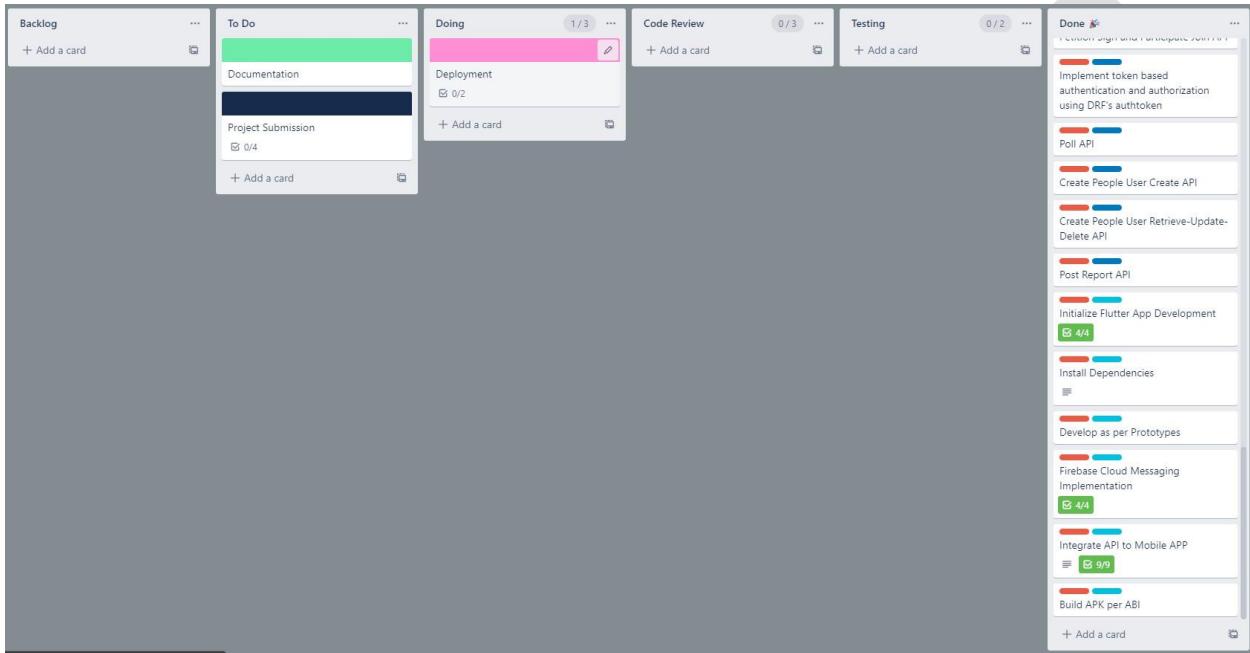


Figure 252: Kanban, Deployment Phase

As per the supervisor's statement, the deployment was not much necessary. The system can be deployed and run locally in the machines, having an internet connection from the same access point. The flutter mobile application was tried to publish in a third-party app store like APKMirror and APKPure, but it failed. The app didn't pass all the requirements. The app will be retried to publish in any app store. For now, the deployment has been paused.

7.10. APPENDIX J: SCREENSHOTS OF THE SYSTEM

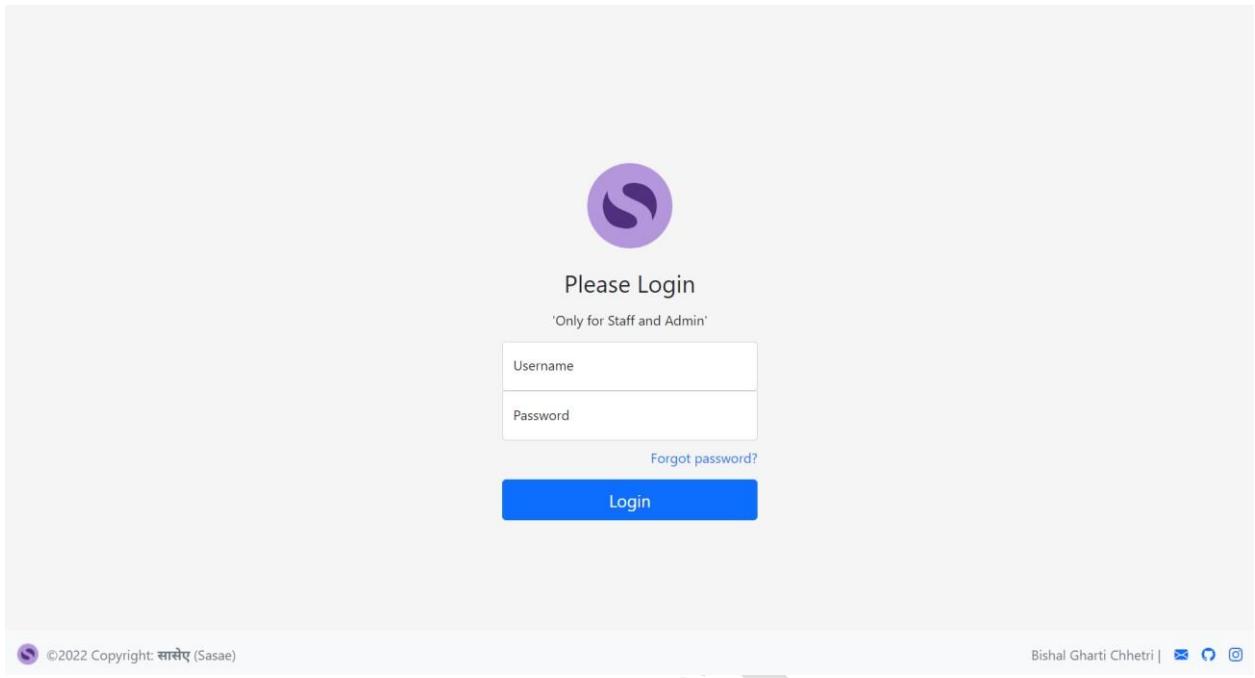


Figure 253: System: Admin-Staff Portal, log in

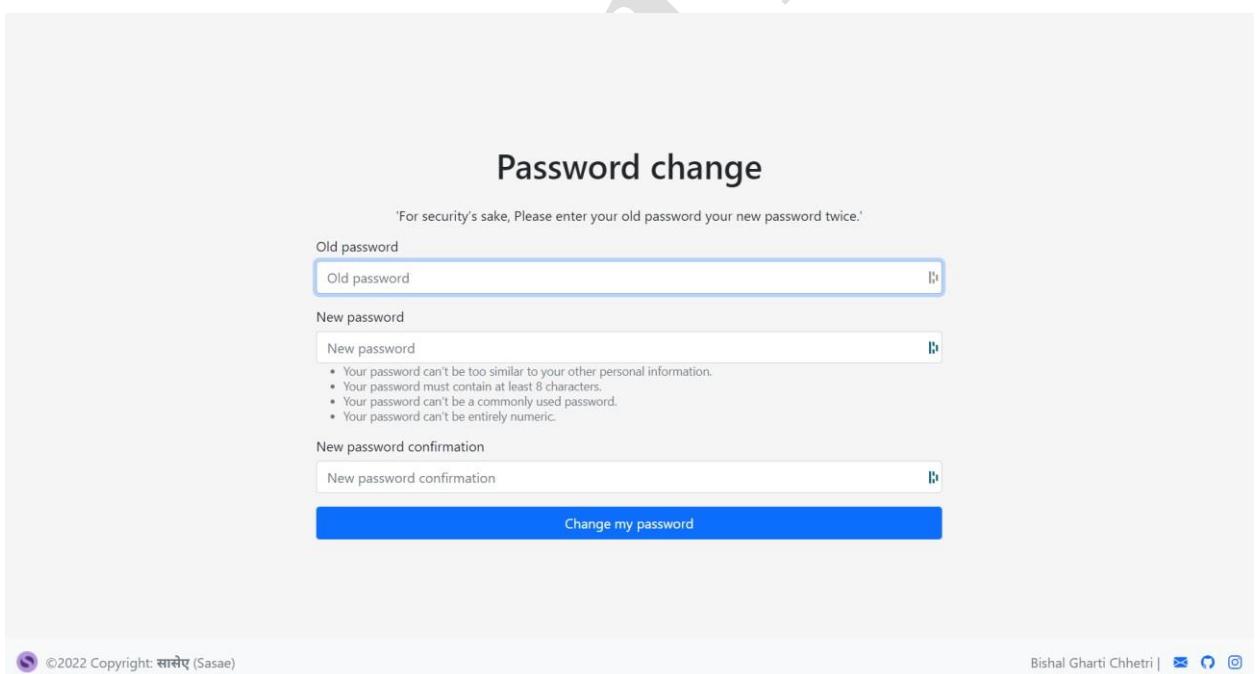


Figure 254: System: Admin-Staff Portal, Password Change

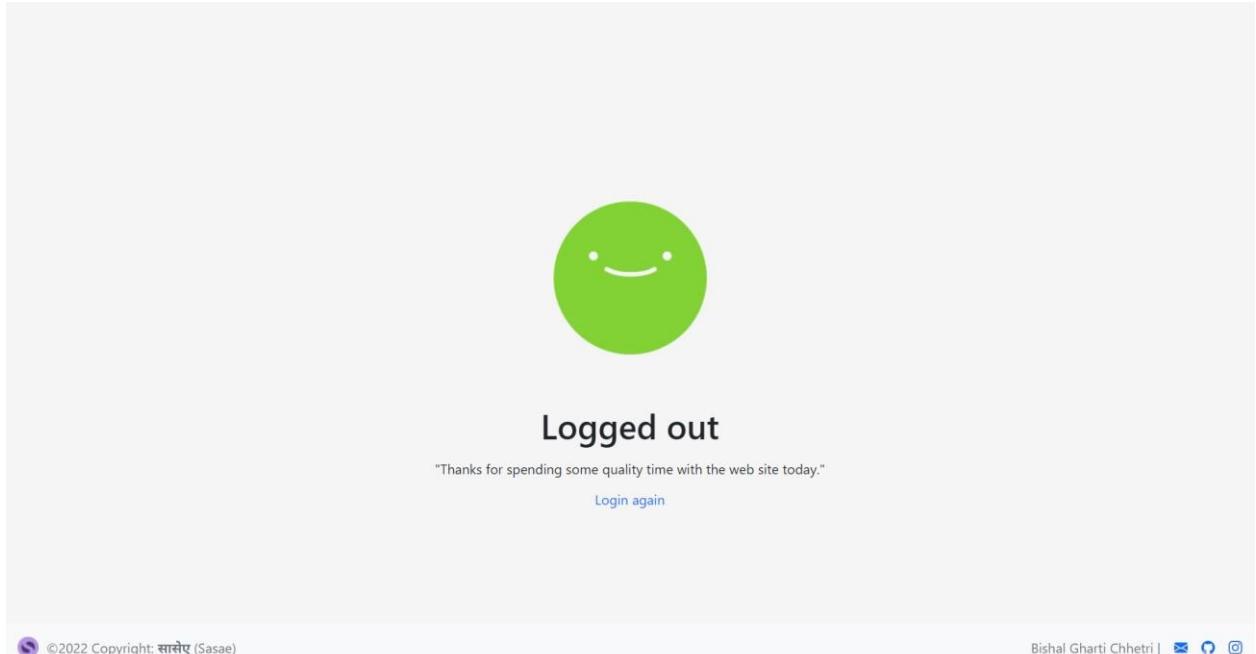


Figure 255: System: Admin-Staff Portal, Logout

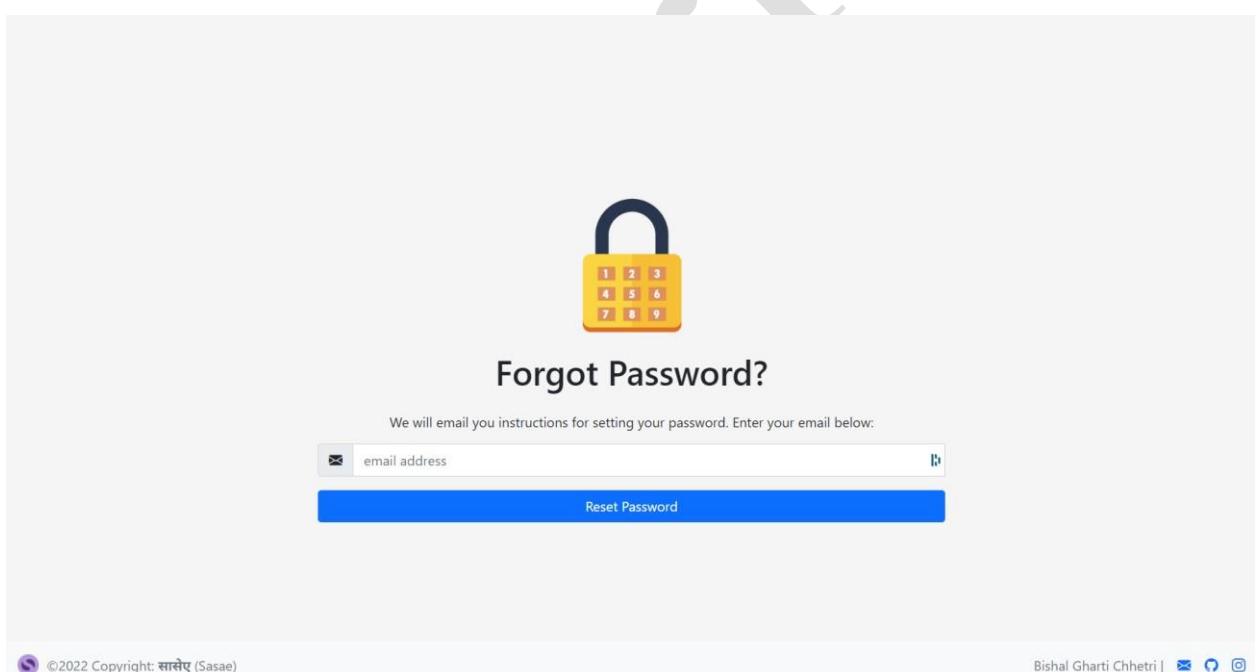


Figure 256: System: Admin-Staff Portal, Password Reset

Password reset confirmation

Please enter your new password twice so we can verify you typed it in correctly.

New password

New password can't be too similar to your other personal information.
 • Your password must contain at least 8 characters.
 • Your password can't be a commonly used password.
 • Your password can't be entirely numeric.

New password confirmation

Change my password

 ©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri | [Email](#) [GitHub](#) [Instagram](#)

Figure 257: System: Admin-Staff Portal, Password Reset Confirmation

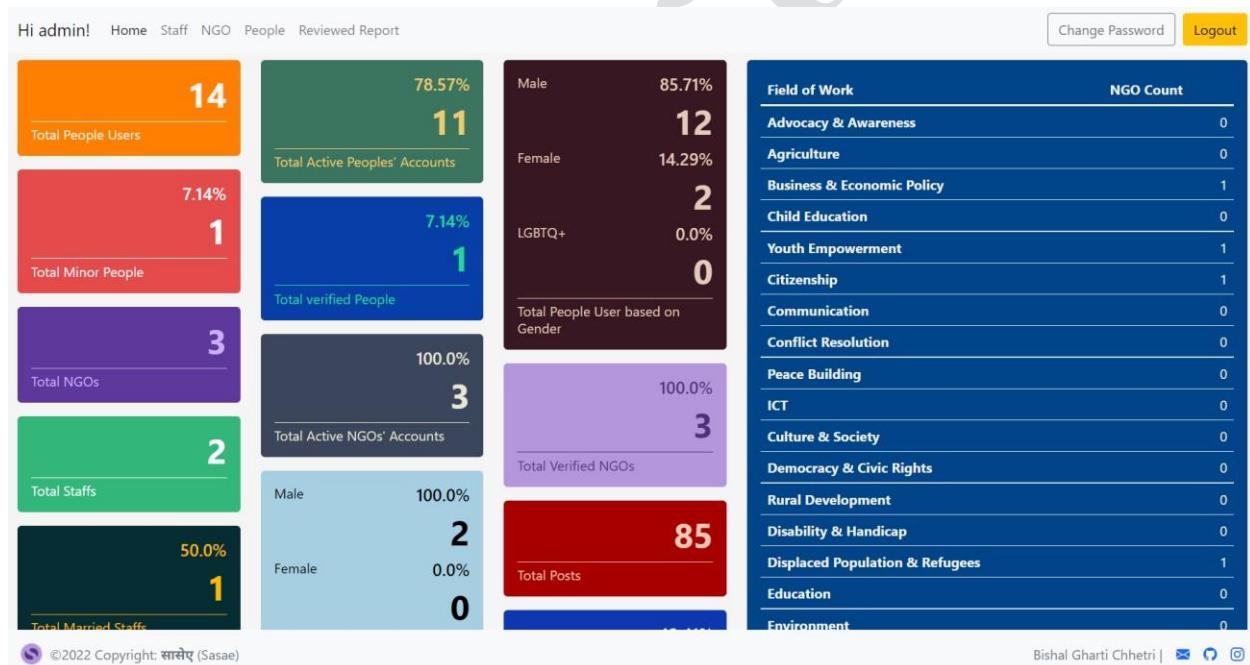


Figure 258: System: Admin-Staff Portal, Admin Home

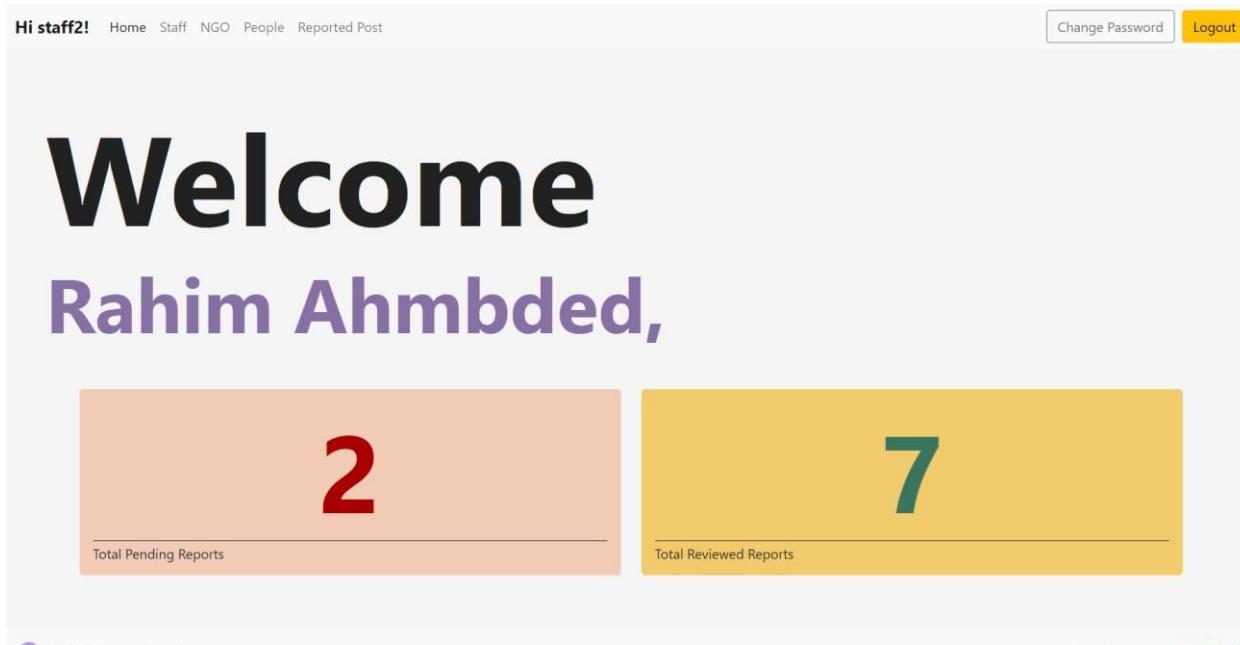


Figure 259: System, Admin-Staff Portal, Staff Home

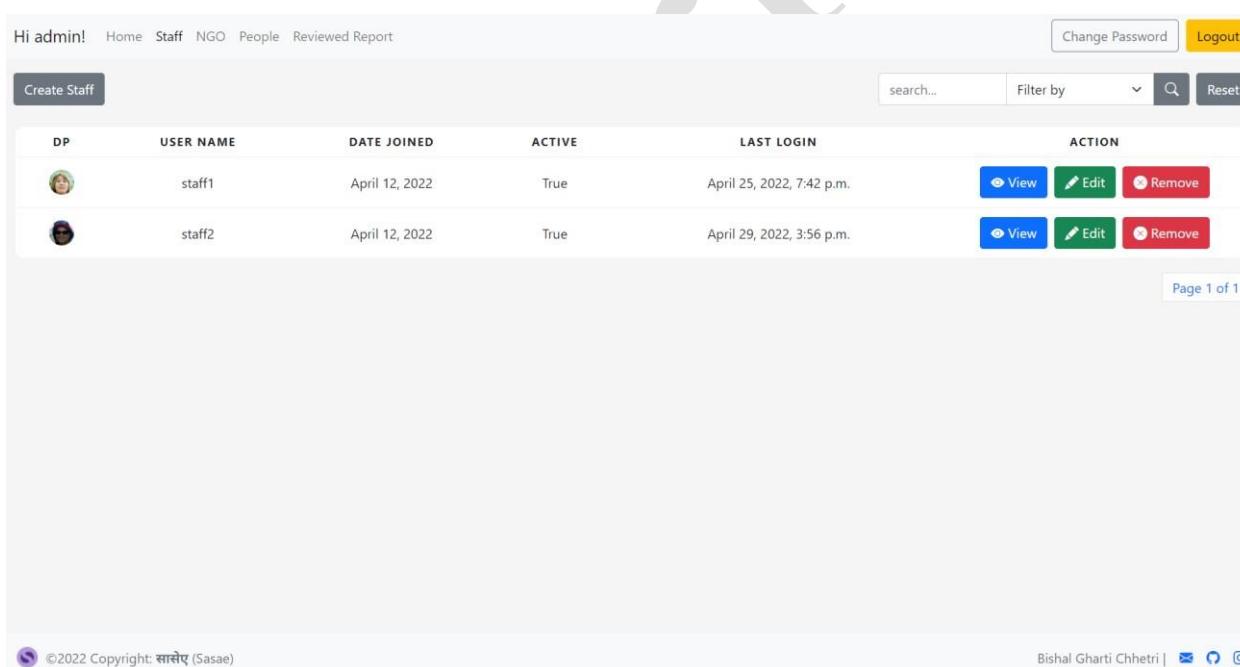


Figure 260: System: Admin-Staff Portal, List of Staff

SASAE

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

Create NGO

DP	NGO NAME	DATE JOINED	ACTIVE	LAST LOGIN	VERIFIED	ACTION		
	INSEC Informal Sector Service Centre	March 27, 2022	True	April 16, 2022, 12:05 p.m.	True	View	Edit	Remove
	IHRICON Institute of Human Rights Communication	March 27, 2022	True	April 2, 2022, 10:12 p.m.	True	View	Edit	Remove
	BishalSanstha	April 25, 2022	True	None	True	View	Edit	Remove

Page 1 of 1

©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri |

Figure 261: System: Admin-Staff Portal, List of NGOs

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

search... Filter by Q Reset

DP	USER NAME	DATE JOINED	ACTIVE	LAST LOGIN	VERIFIED	ACTION		
	bishal	March 27, 2022	True	April 26, 2022, 6:06 p.m.	True	View	Edit	Remove
	mango	March 27, 2022	True	None	False	View	Edit	Remove
	test	March 30, 2022	True	None	False	View	Edit	Remove
	mkbhd1	April 4, 2022	True	None	False	View	Edit	Remove
	ram	April 4, 2022	False	April 26, 2022, 12:09 a.m.	False	View	Edit	Remove
	kamskl	April 4, 2022	False	None	False	View	Edit	Remove
	harilalu	April 23, 2022	True	None	False	View	Edit	Remove
	hari373	April 23, 2022	True	None	False	View	Edit	Remove

Page 1 of 2 next last »

©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri |

Figure 262: System: Admin-Staff Portal, List of General People

SASAE

The screenshot shows a dashboard with two main sections. On the left, a box displays a post from 'staff2!' with the title 'Total Reported Posts: 9'. The post content is a quote about Lorem Ipsum. On the right, another box shows a post from 'staff2!' with the title 'Total Reviewed Posts: 7'. The post content is a quote about reader distraction. Both posts have edit and delete buttons at the top. The bottom of the screen includes a copyright notice and social media links.

Figure 263: System: Admin-Staff Portal, List of Reported Posts

This screenshot shows a detailed view of a reported post. The left panel displays the post's details: posted by 'bishal' on April 23, 2022, at 11:44 a.m., categorized as a 'Poll'. The post content discusses the readability of Lorem Ipsum. Below the content is a poll section with three options (a, b, c) and a reaction counter. The right panel is titled 'Let's Review!' and contains fields for 'Reason' (with 'Reason' typed in), 'Action' (with a dropdown menu), and a large blue 'Take Action' button. At the bottom, there are copyright and social media links.

Figure 264: System: Admin-Staff Portal, Reported Post Detail

SASAE

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

Total Reported Posts: 36 Total Reviewed Posts: 21

He La Re

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ...

PIN March 28, 2022

Co La

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ...

PIRJ April 7, 2022

He Re

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ...

A P PIN April 3, 2022

Re He

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ...

A P PIN April 3, 2022

Wo He

mansdwjhwjw

PIN April 10, 2022

Bu Te

wowow great

P PIN April 8, 2022

Bu Ad Ag

mango amskcookcms

PIN April 3, 2022

DI CI

mklijdds jsshb ajsjd 1

PIRJ April 3, 2022

Co La

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ...

A PIRJ March 27, 2022

He Re La

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ...

PIN March 28, 2022

La Co Ed Tr Ru

the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution ...

A P PIP April 7, 2022

La Co

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ...

PIRJ April 7, 2022

©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)

Figure 265: System: Admin-Staff Portal, List of Reviewed Reports

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

Posted by: **bishal** Date Posted: April 8, 2022, 7:02 p.m. Post Type: **Normal**

Post related to: **Business & Economic Policy Technology**

Post Content: wowow great

Post attached Photo:

Reason

Because of the low incidence and the early diagnosis of endometrial cancer owing to blood loss, we believe that future risk of malignancy should not be regarded as a valid reason for removal of the uterus.

Action

Post Remove

Reviewed by: Neha Singh

©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)

Figure 266: System: Admin-Staff Portal, Reviewed Report Detail

SASAE

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

View NGO

User Name	ngo2
Organization Name	IHRICON Institute of Human Rights Communication
Establishment Date	March 2, 2022
Phone	+9779800657489
Email	ngo2@gmail.com
Address	kalikanagr
Location	32.39238000000000, 109.22300000000000
Fields of Work	Citizenship Displaced Population & Refugees Women's Rights
Epay Account	98007485893
Bank Details	
Verified	True

Display Picture



©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)

Figure 267: System: Admin-Staff Portal, NGO Detail

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

Edit NGO

Organization Name	IHRICON Institute of Human Rights Communication
Establishment date	03/02/2022
Field of work	<input type="checkbox"/> Advocacy & Awareness <input type="checkbox"/> Agriculture <input type="checkbox"/> Business & Economic Policy <input type="checkbox"/> Child Education <input type="checkbox"/> Youth Empowerment <input checked="" type="checkbox"/> Citizenship <input type="checkbox"/> Communication <input type="checkbox"/> Conflict Resolution <input type="checkbox"/> Peace Building <input type="checkbox"/> ICT <input type="checkbox"/> Culture & Society <input type="checkbox"/> Democracy & Civic Rights <input type="checkbox"/> Rural Development <input type="checkbox"/> Disability & Handicap

©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)

Figure 268: System: Admin-Staff Portal, NGO Detail Update

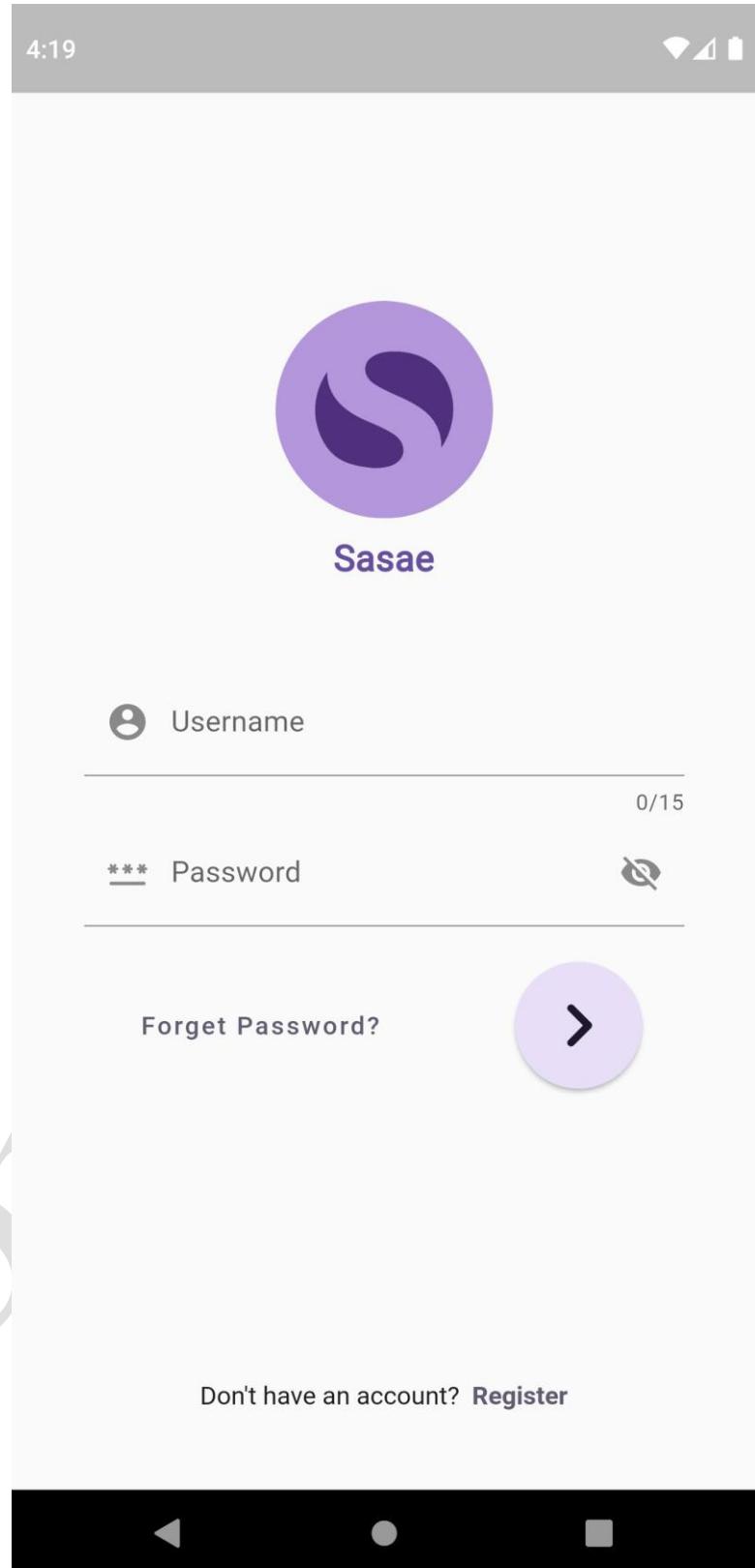


Figure 269: System: Sasae App, log in

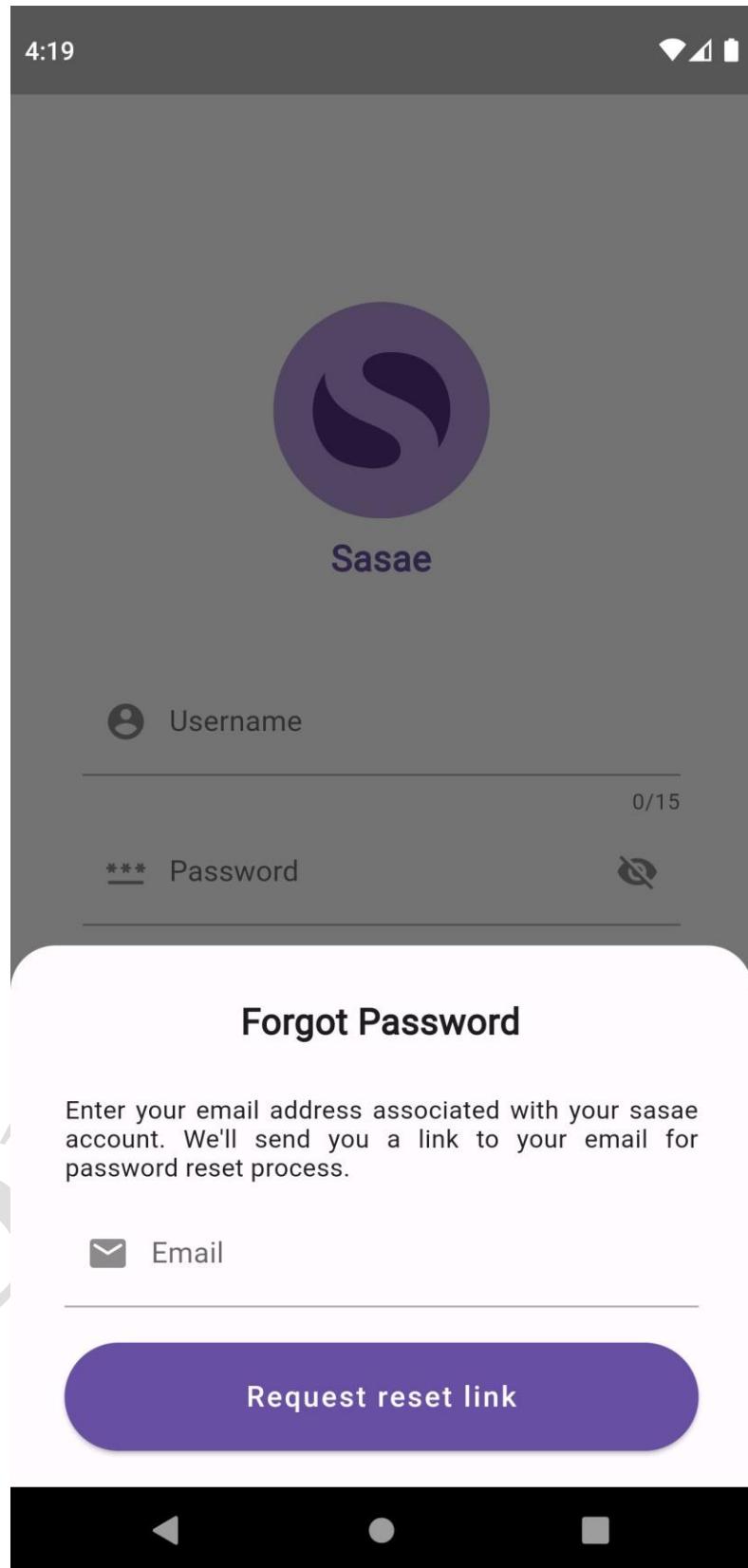


Figure 270: System: Sasae App, Password Reset

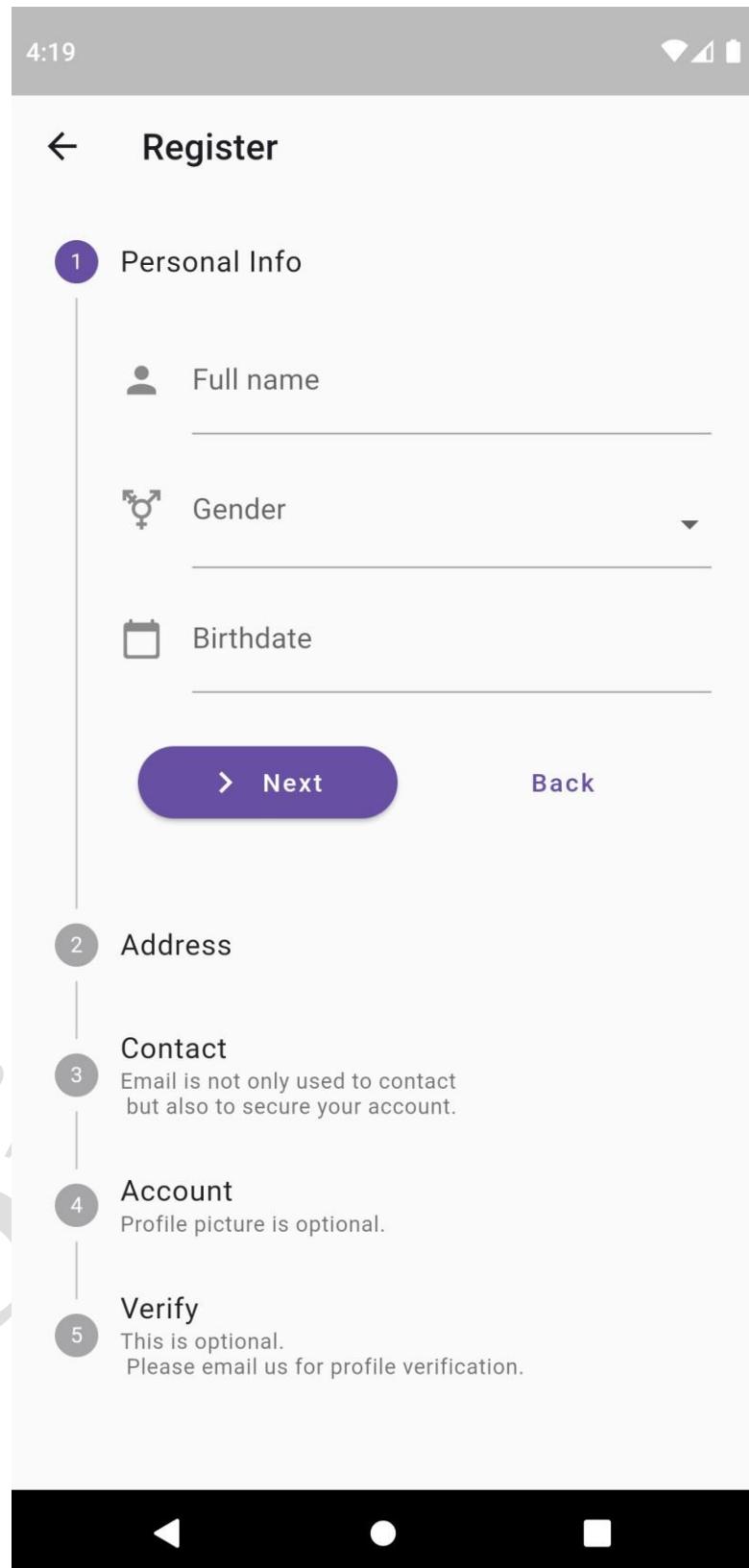


Figure 271: System: Sasae App, General People Registration

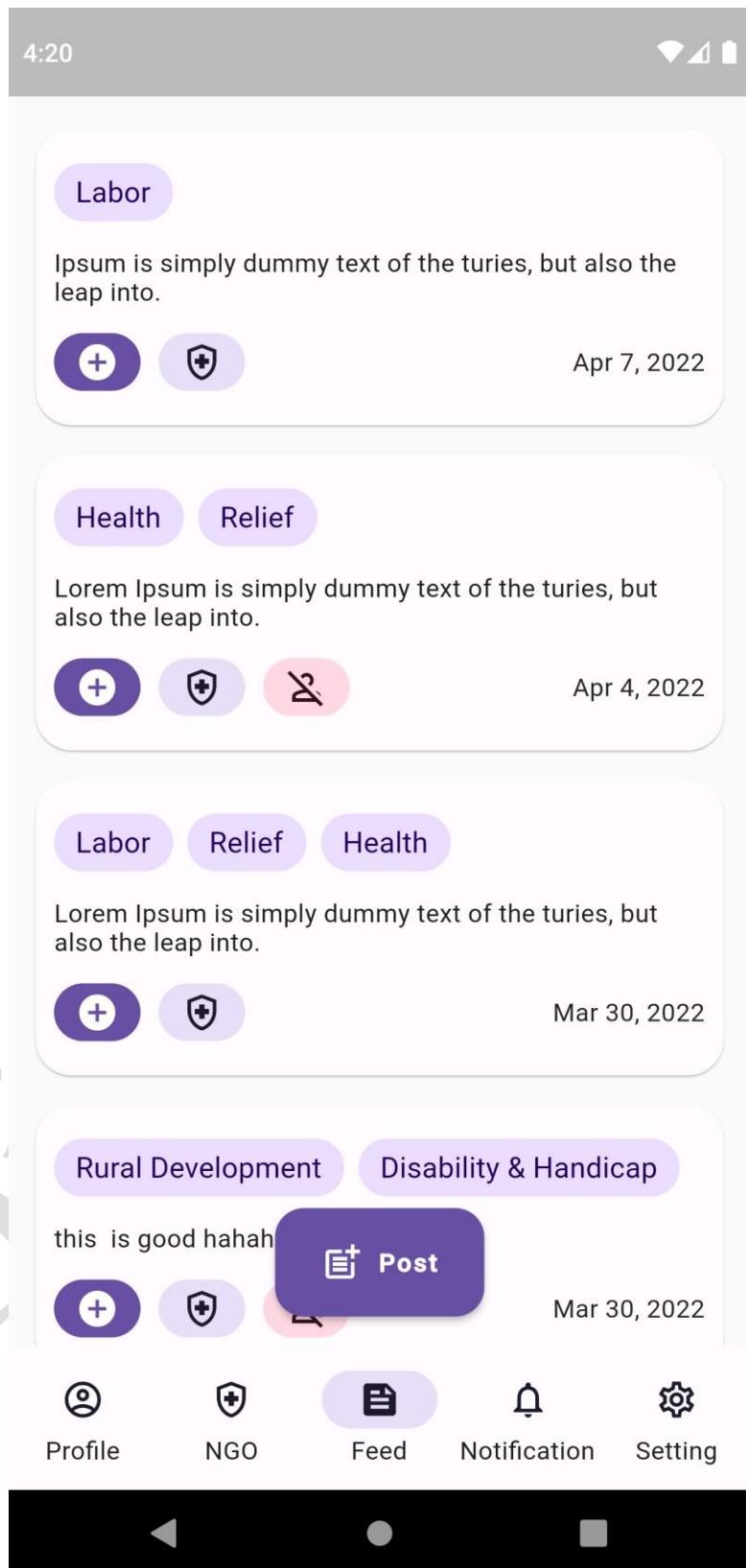


Figure 272: System: Sasae App, List of Posts



Figure 273: System: Sasa App, Post a Post

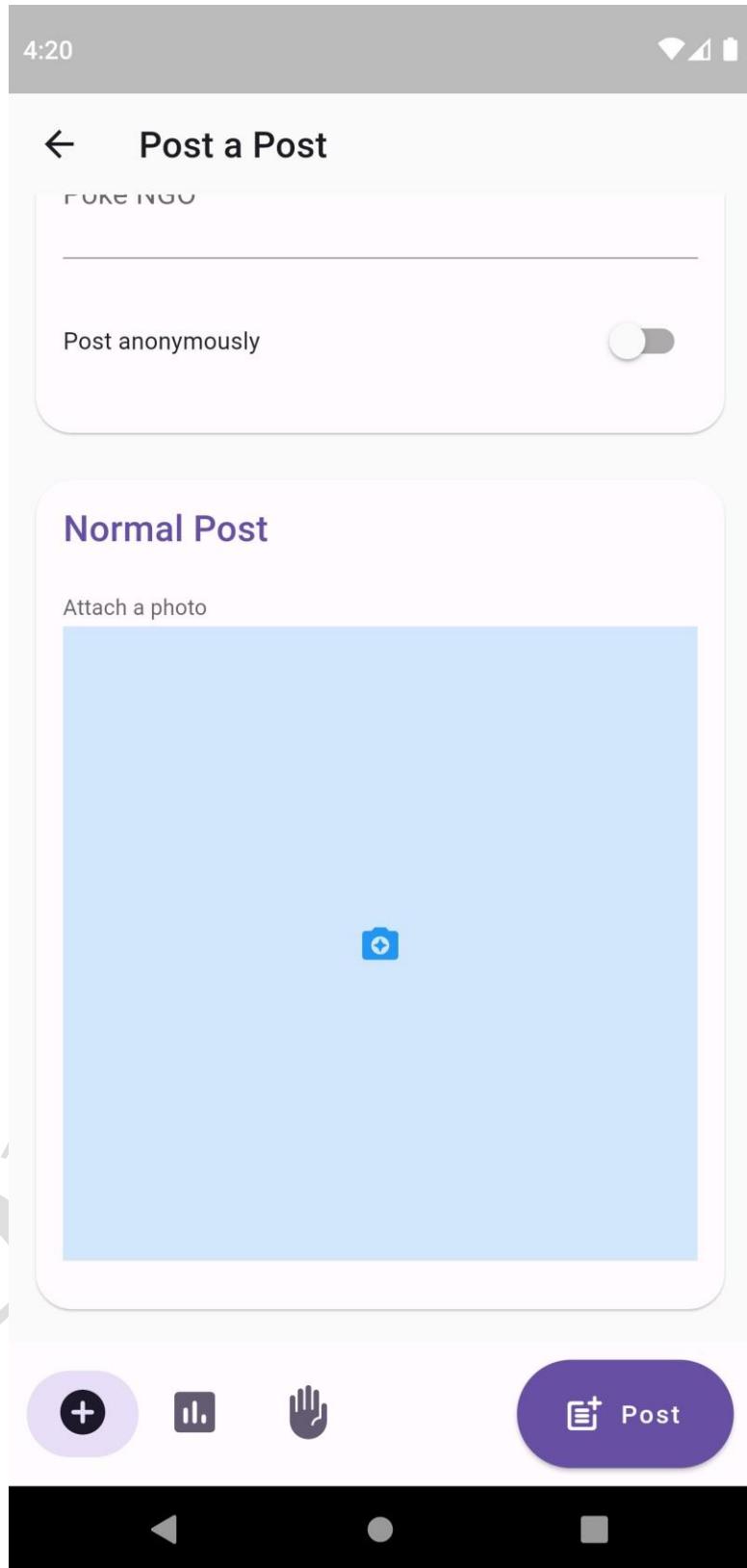


Figure 274: System: Sasae App, Post a Normal Post

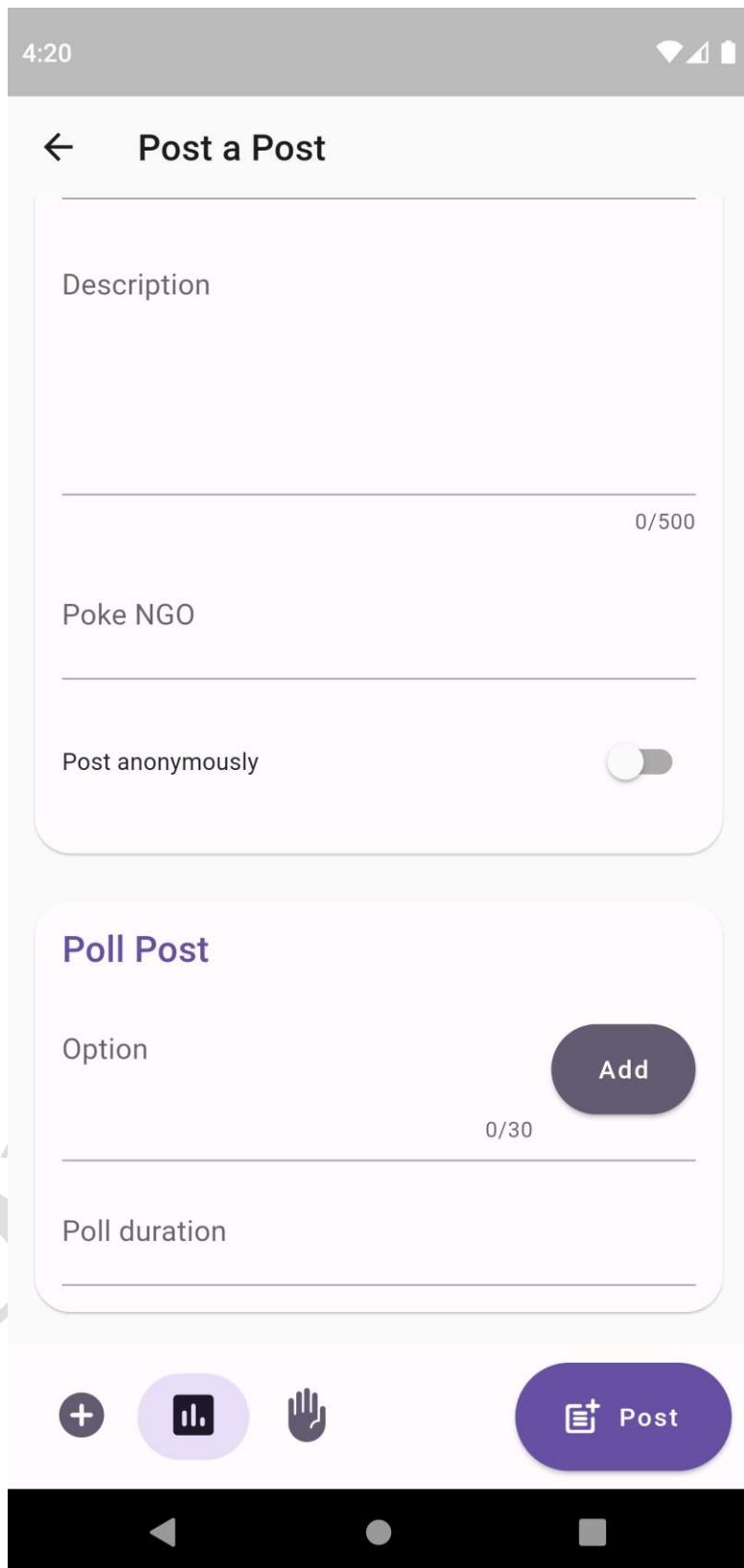


Figure 275: System: Sasae App, Post a Poll Post

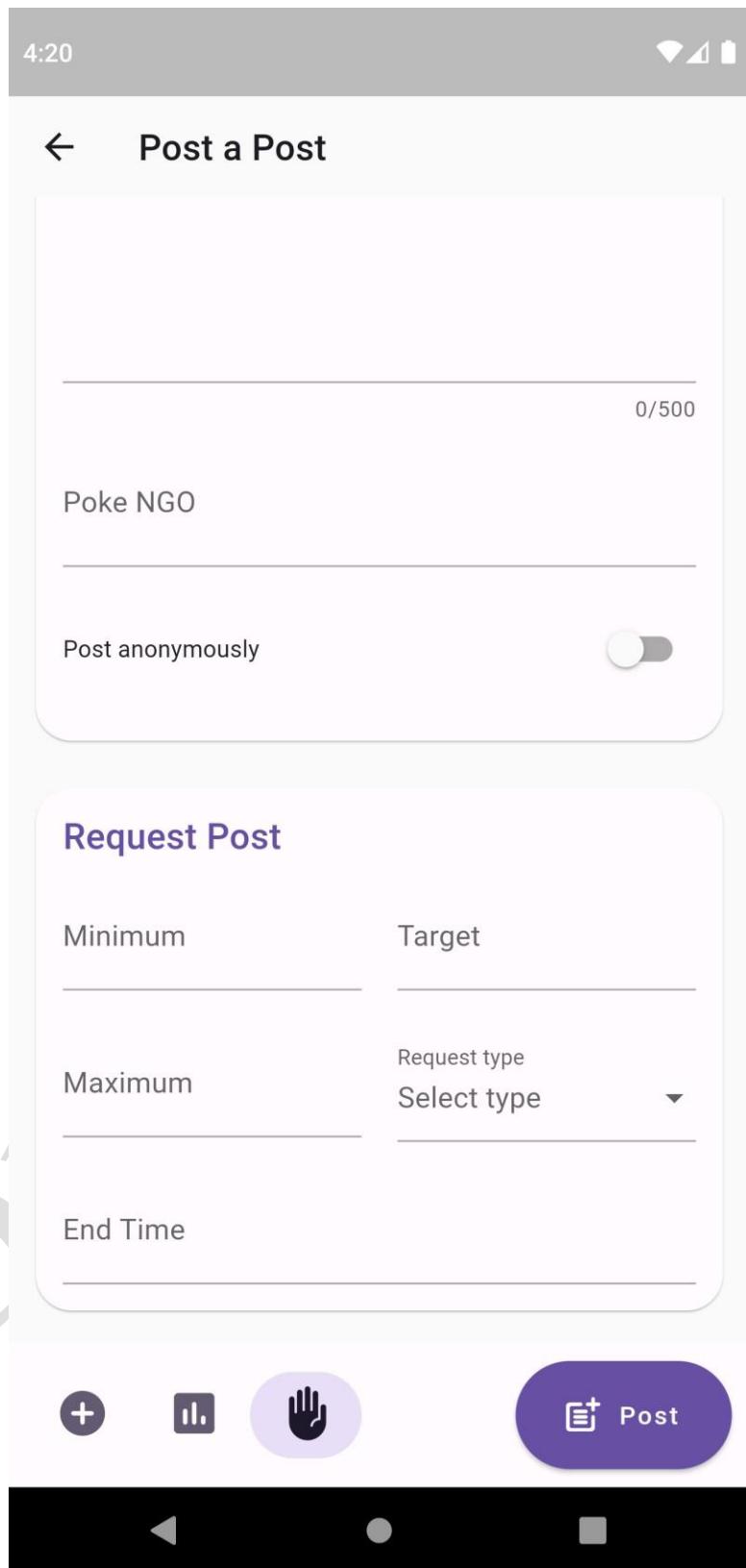


Figure 276: System: Sasae App, Post a Request Post

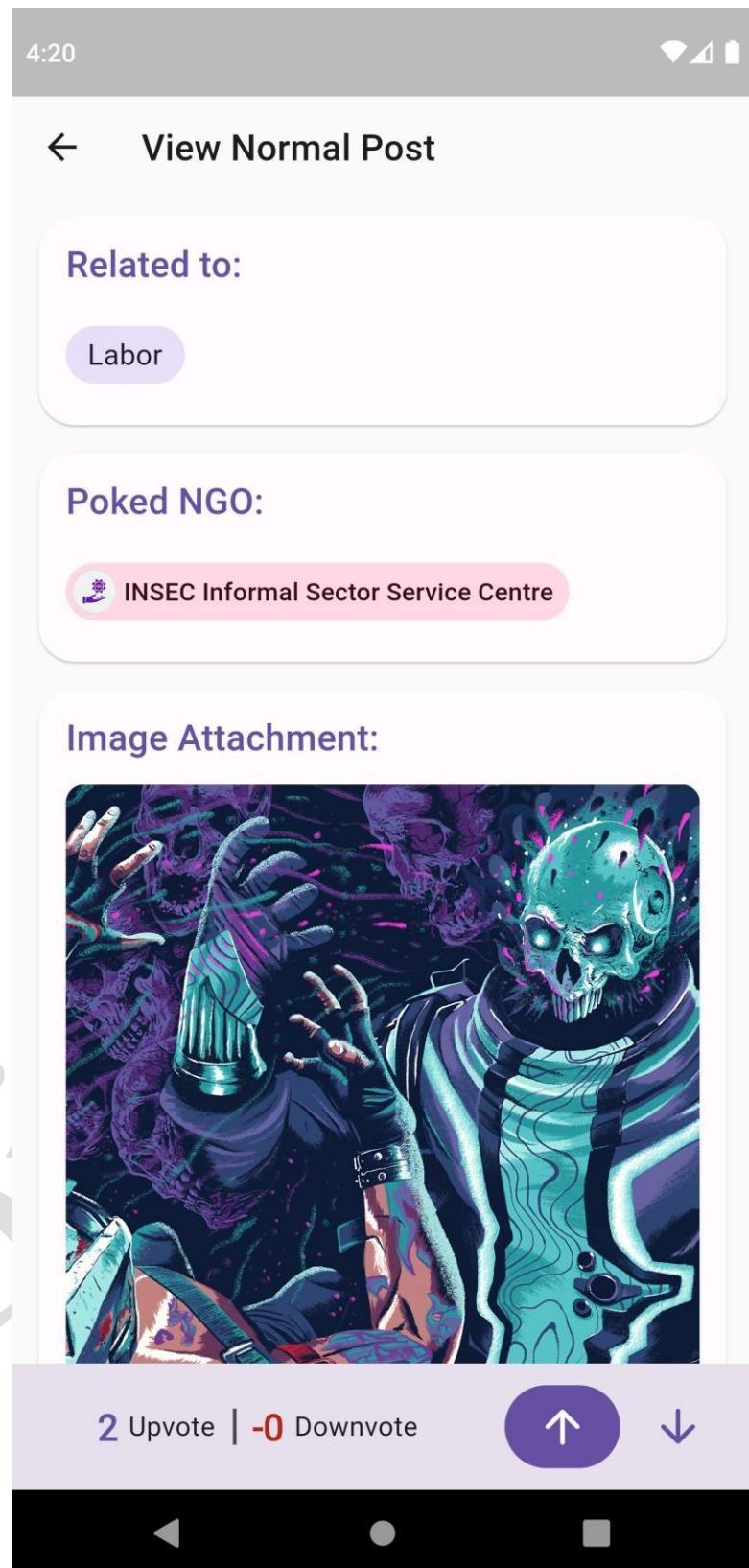


Figure 277: System: Sasae App, Normal Post

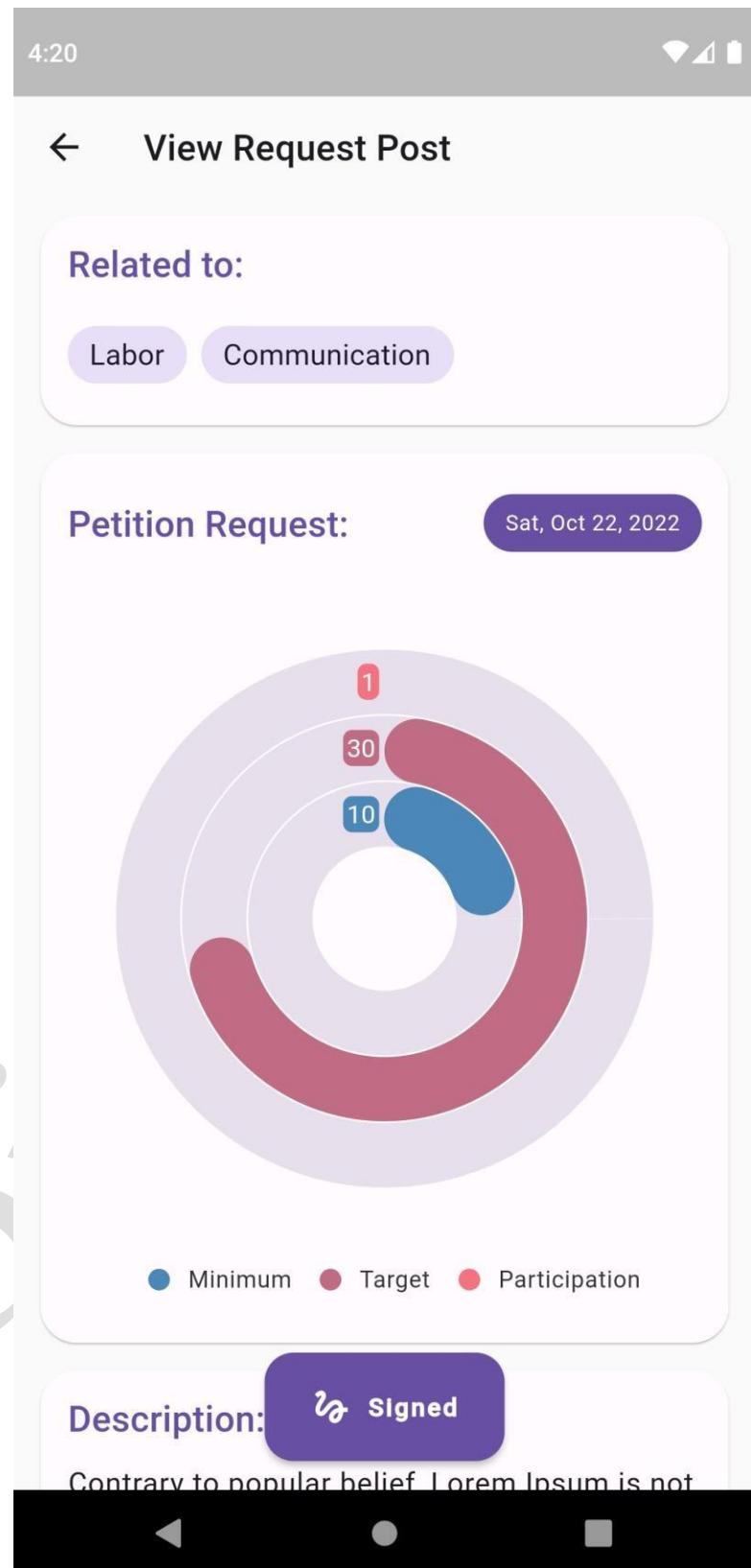


Figure 278: System: Sasae App, Request Post

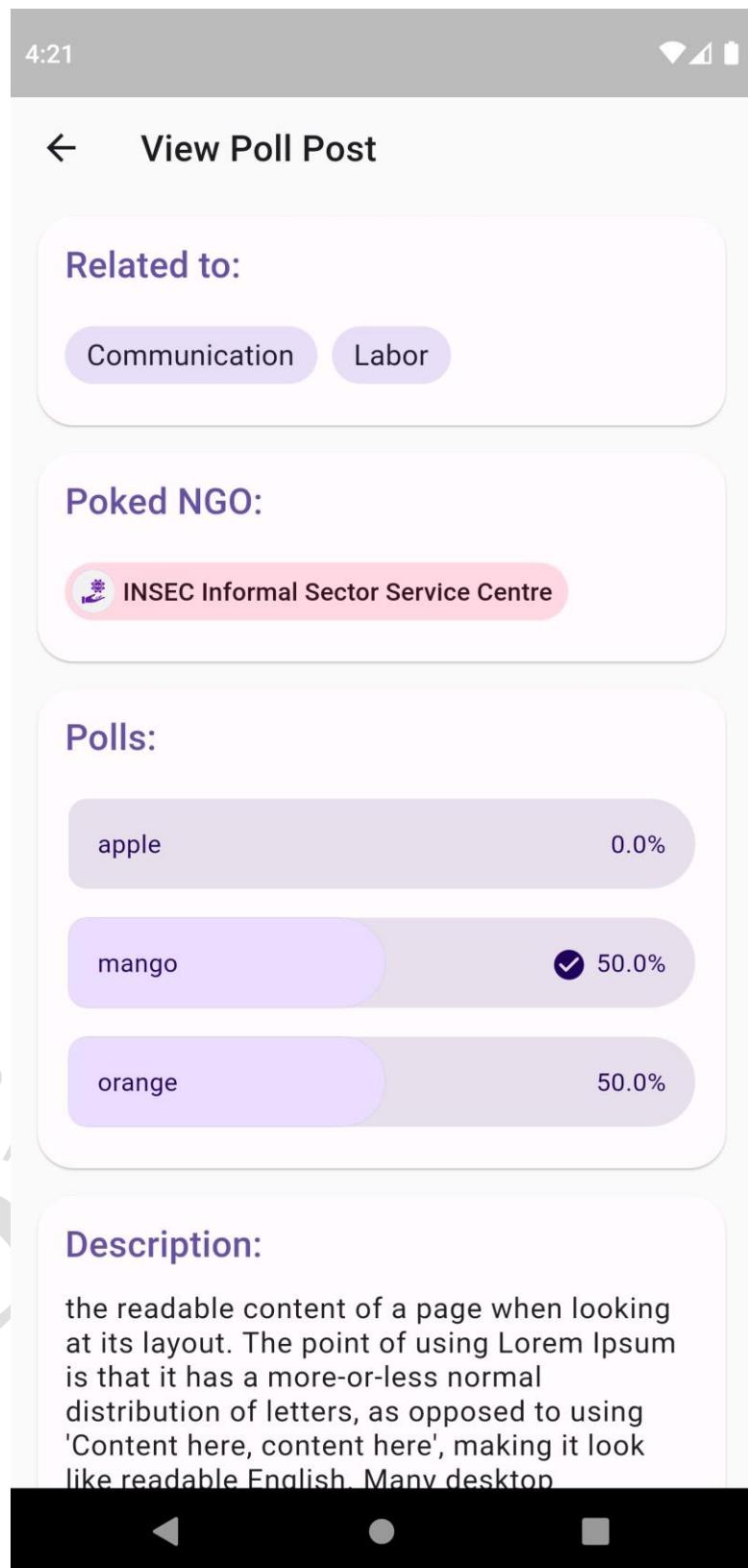


Figure 279: System: Sasa App, Poll Post

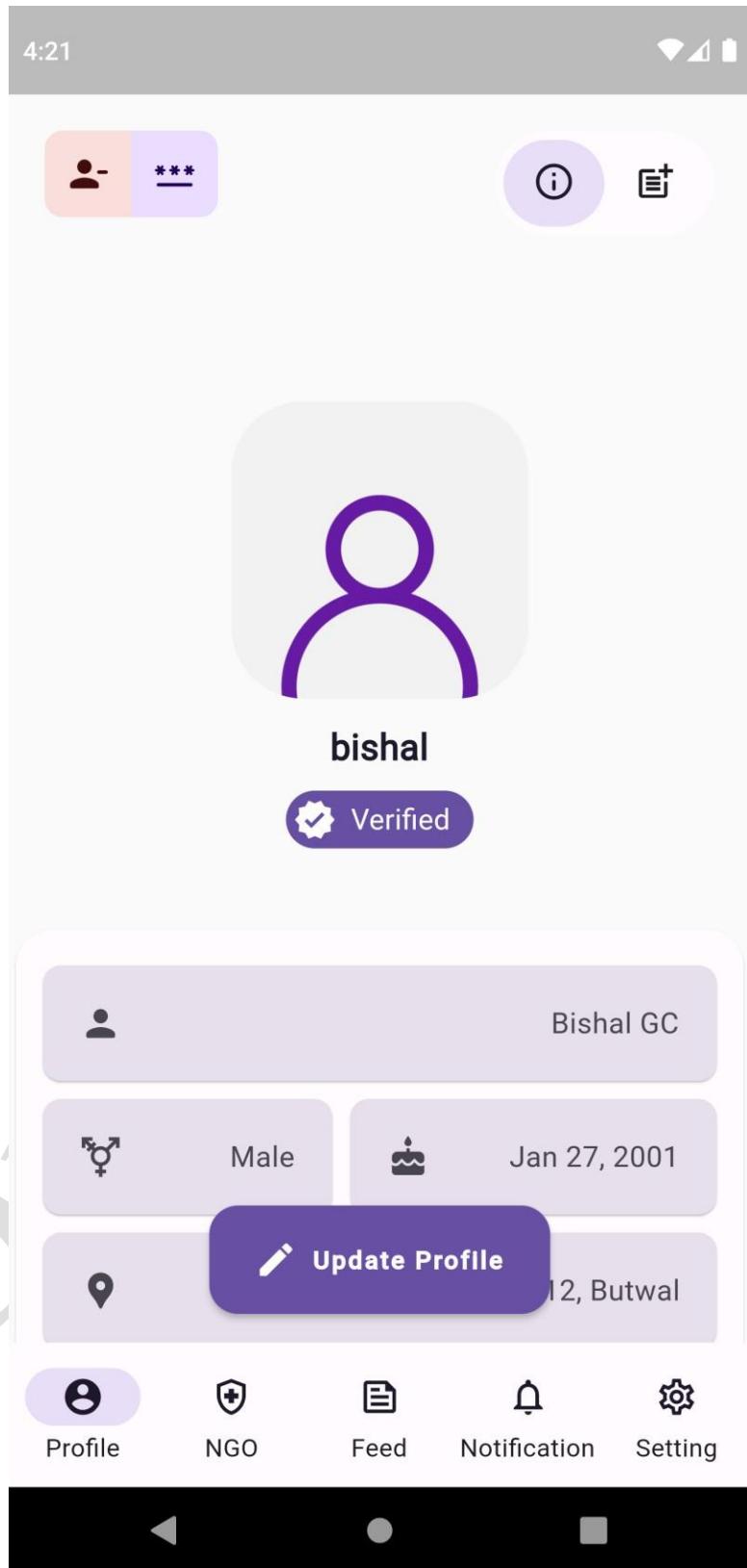


Figure 280: System: Sasae App, General People Profile

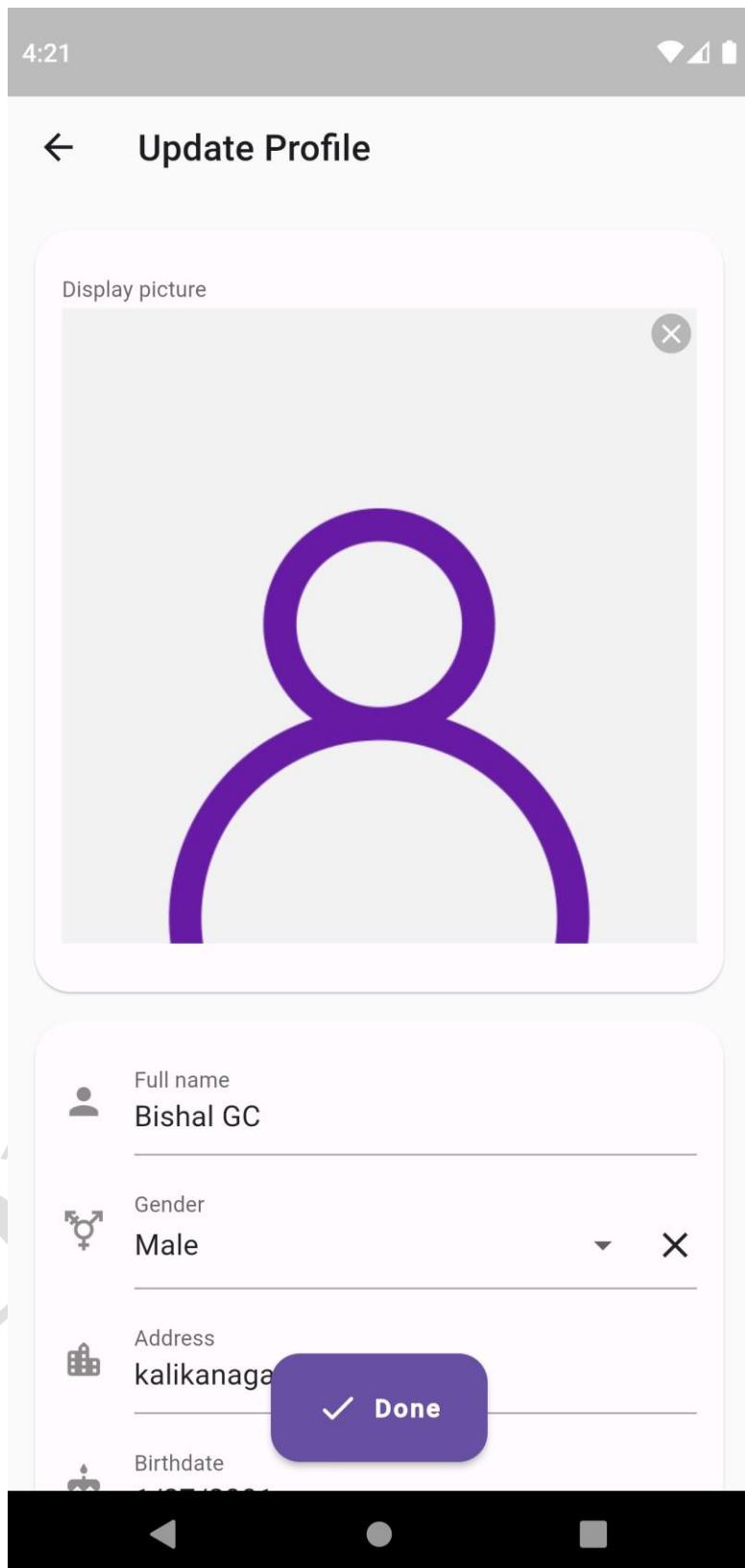


Figure 281: System: Sasae App, Profile Update

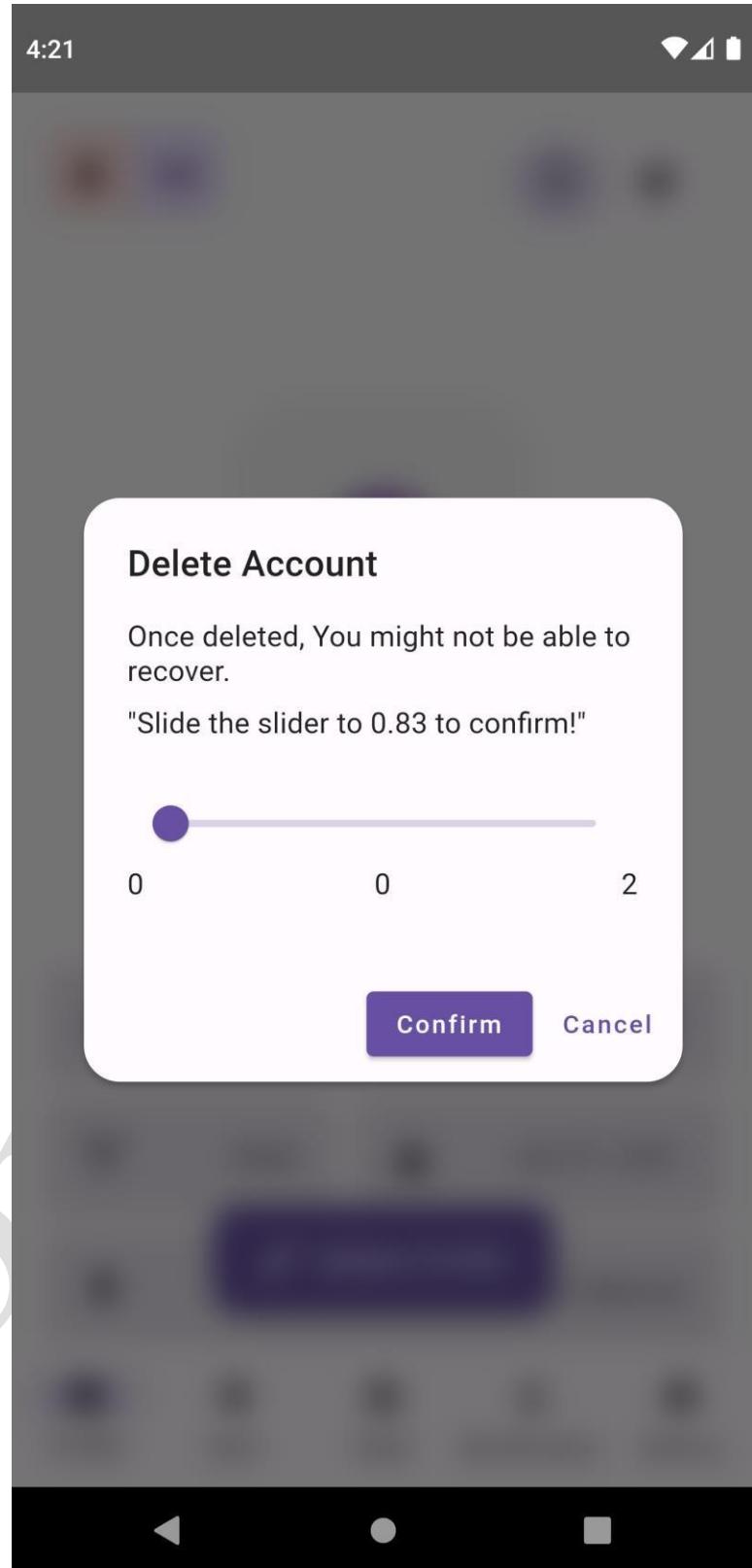


Figure 282: System: Sasae App, General People Account Deletion

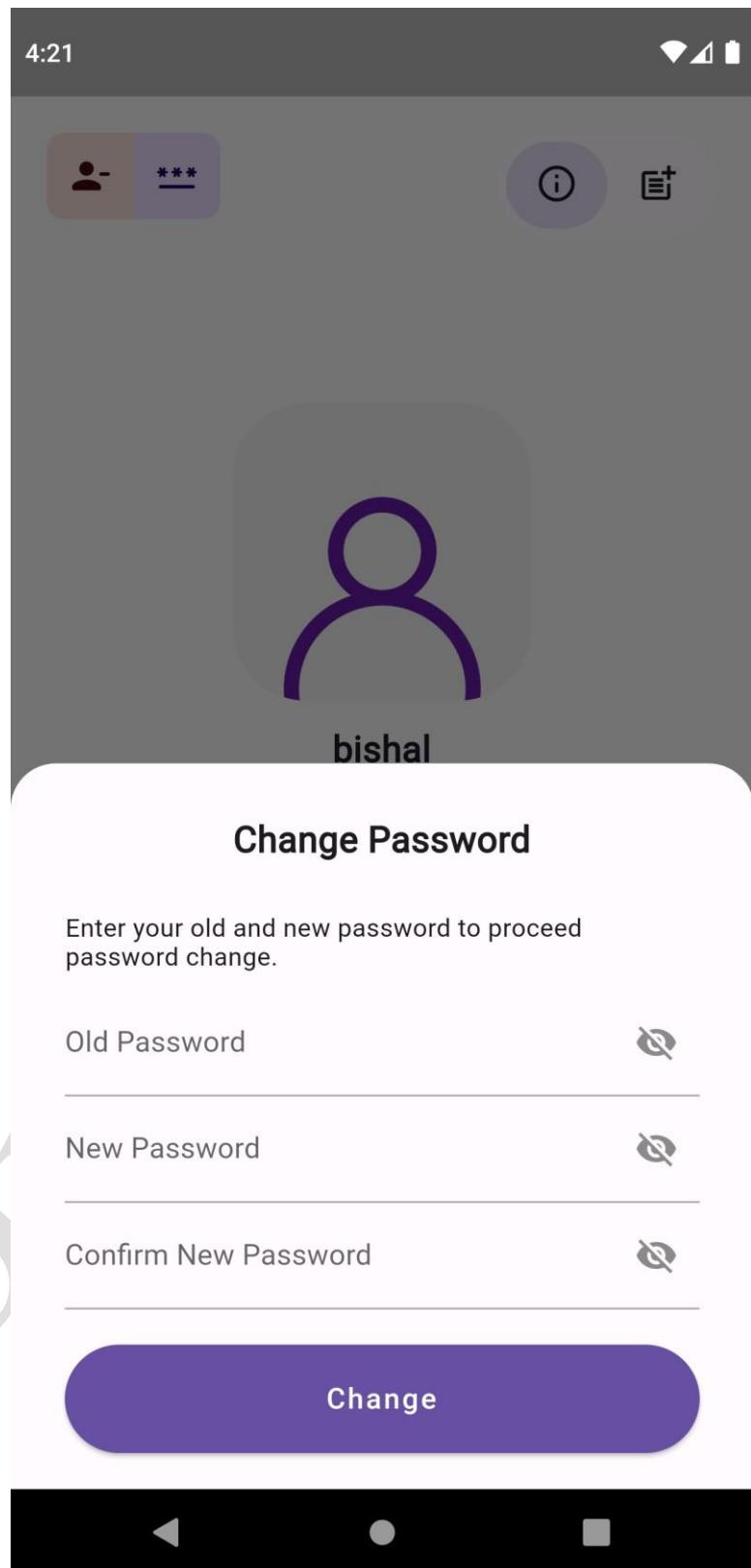


Figure 283: System: Sasae App, Change Password

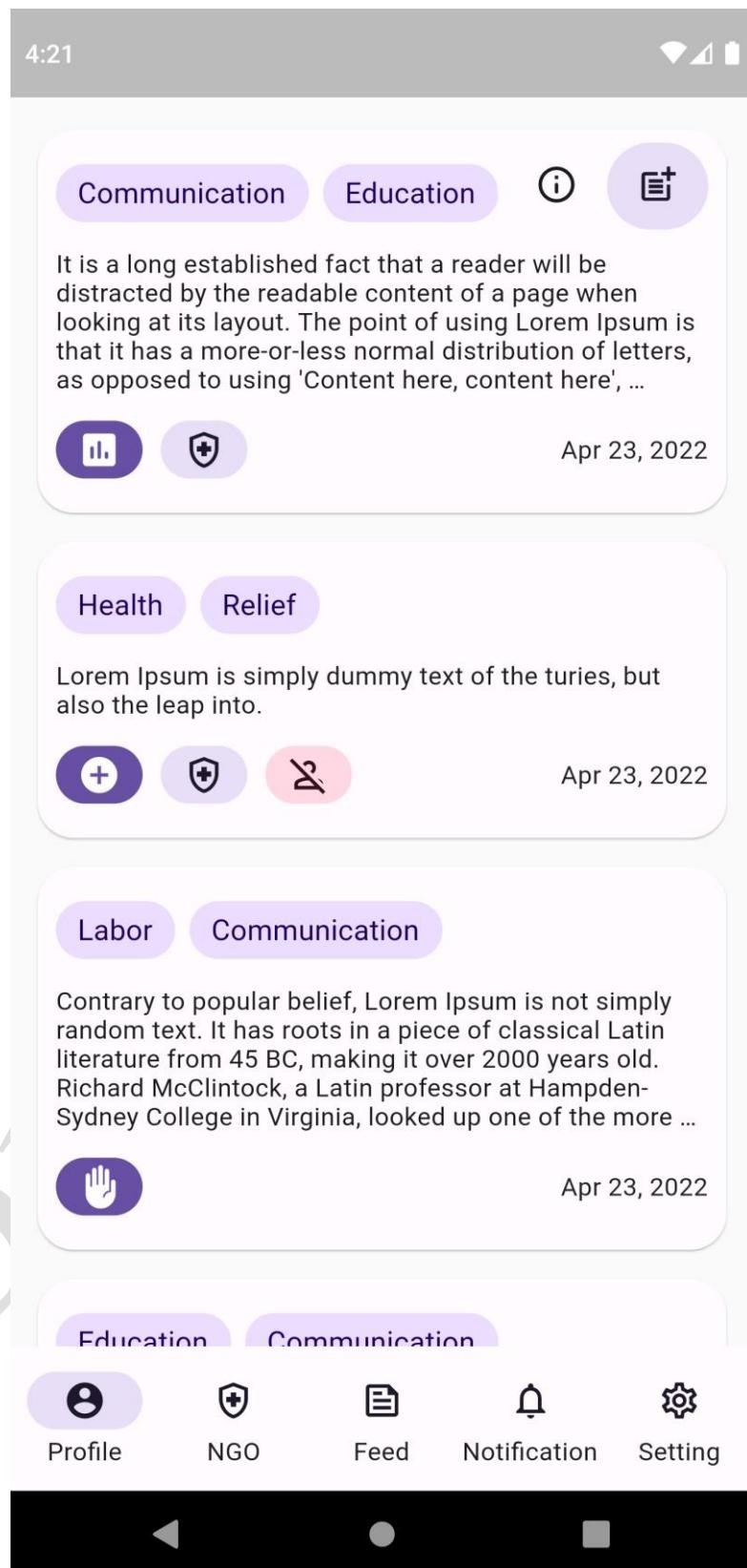


Figure 284: System: Sasae App, User's Post

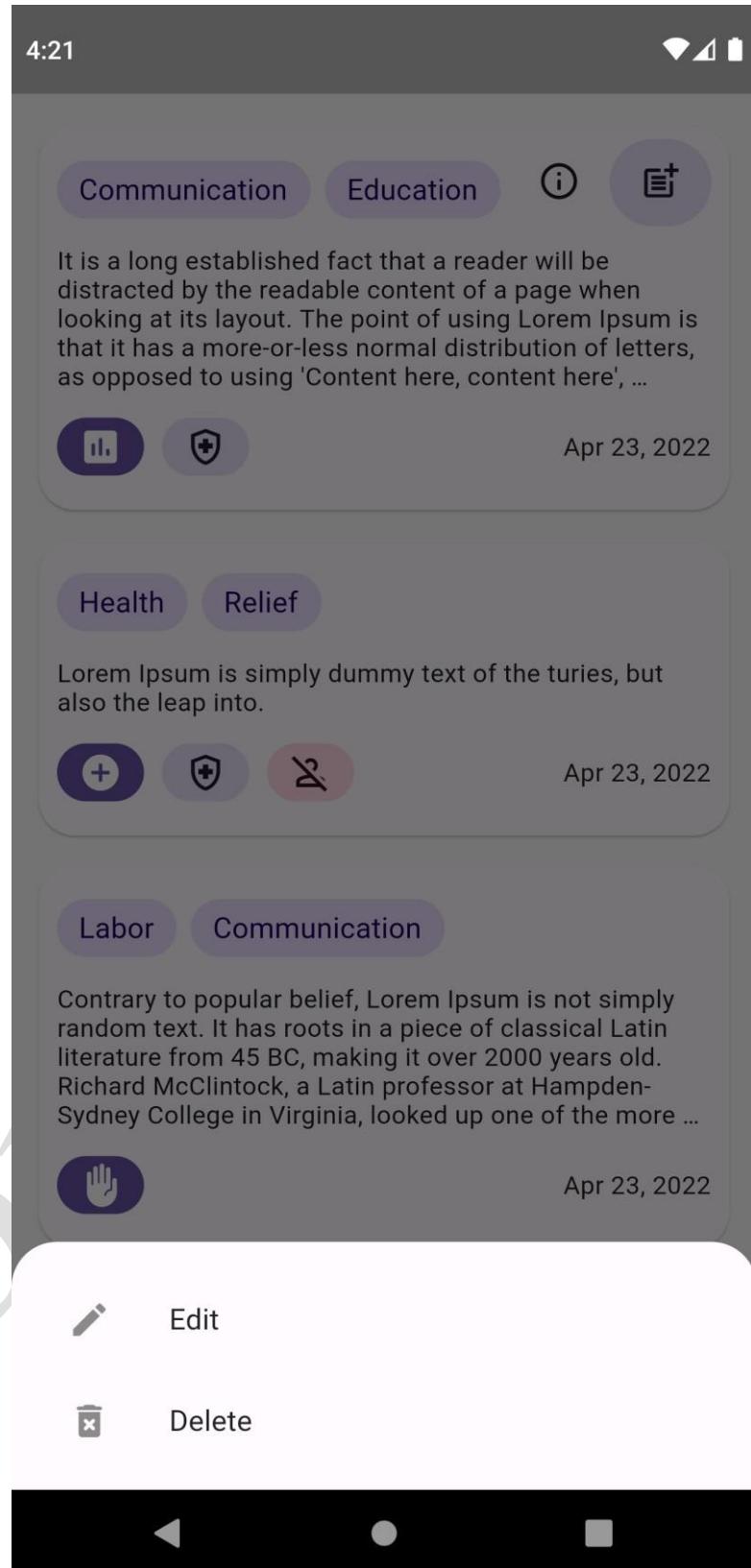


Figure 285: System: Sasae App, Post Edit or Delete

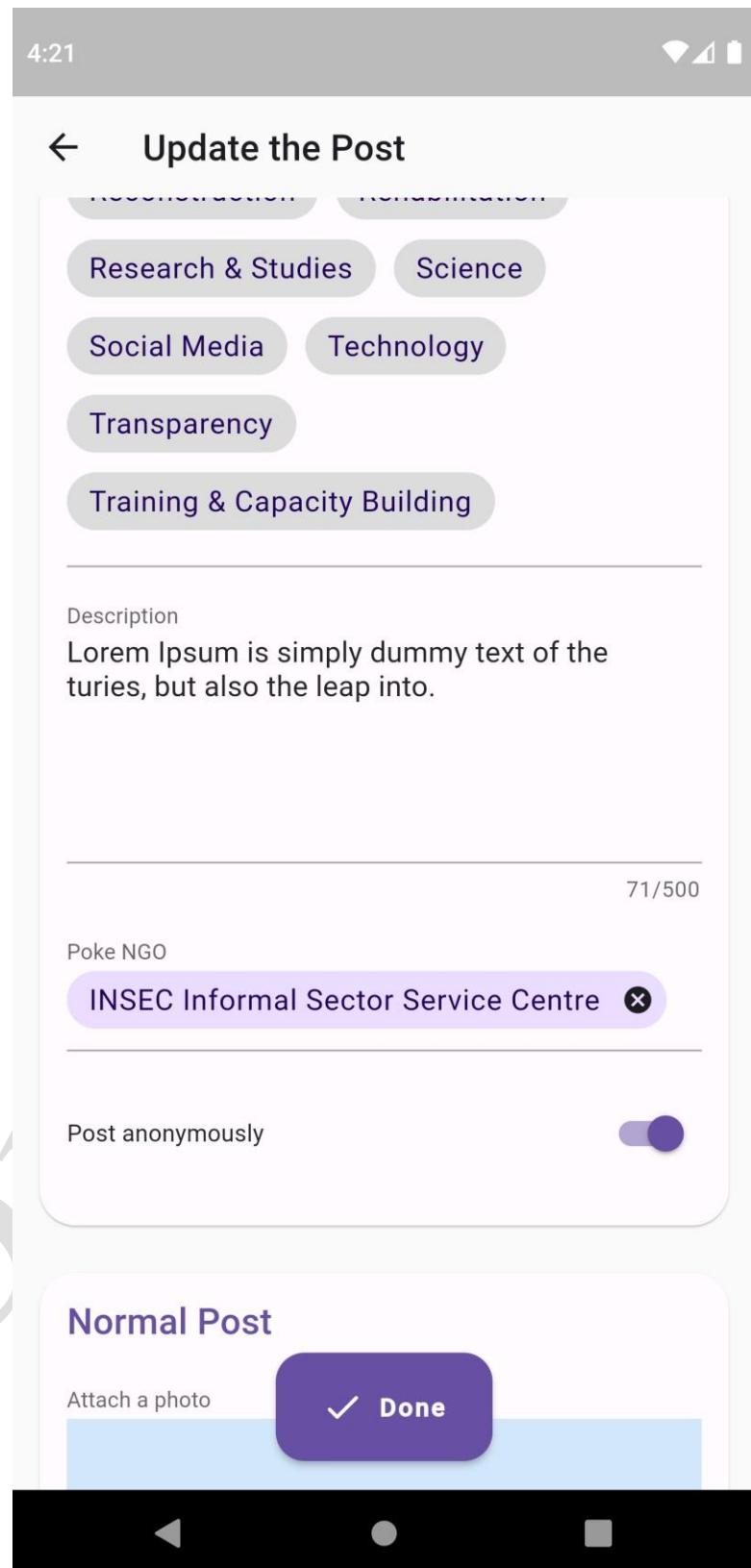


Figure 286: System: Sasae App, Post Update

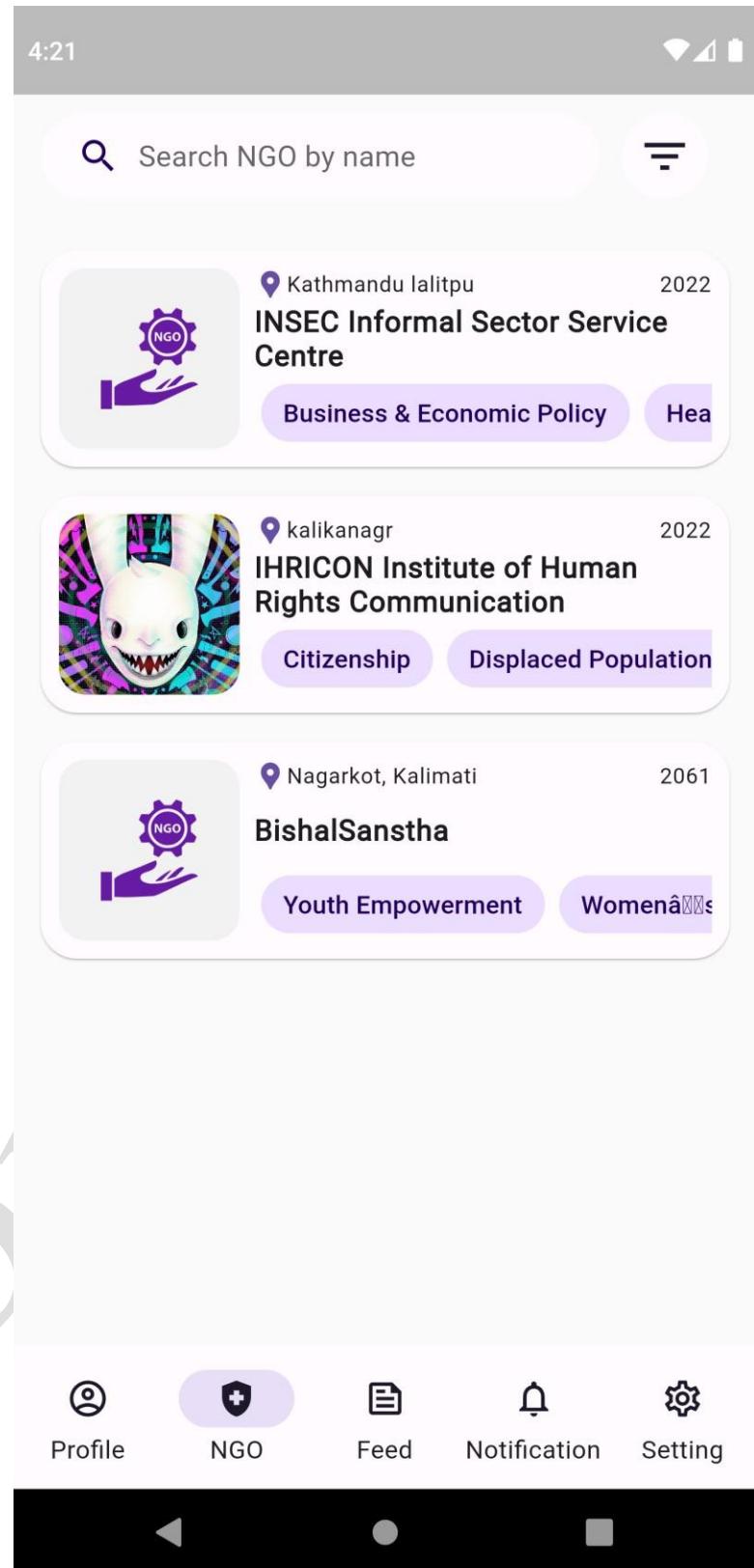


Figure 287: System: Sasae App, List of NGOs

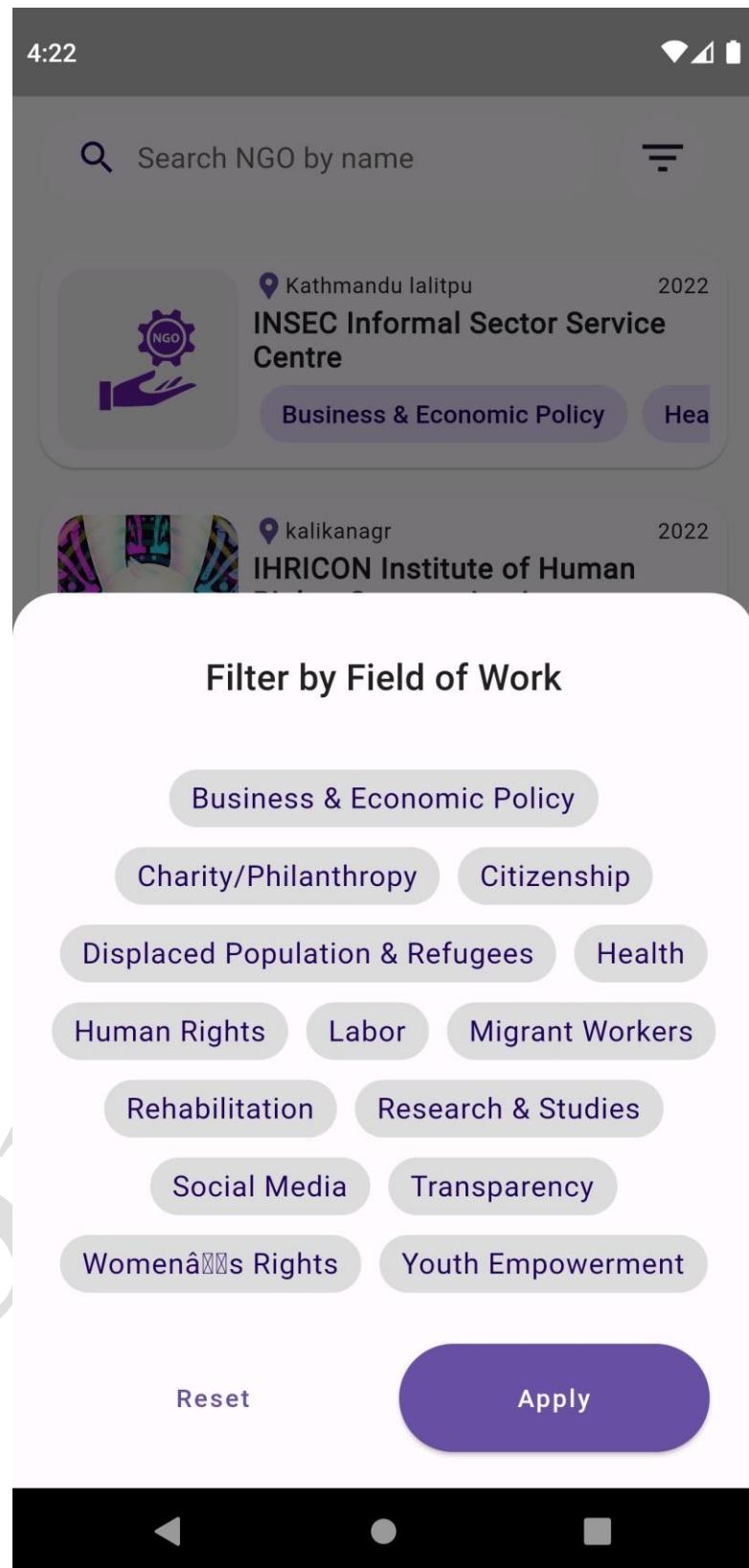


Figure 288: System: Sasae App, NGOs Filtration by Field of Work

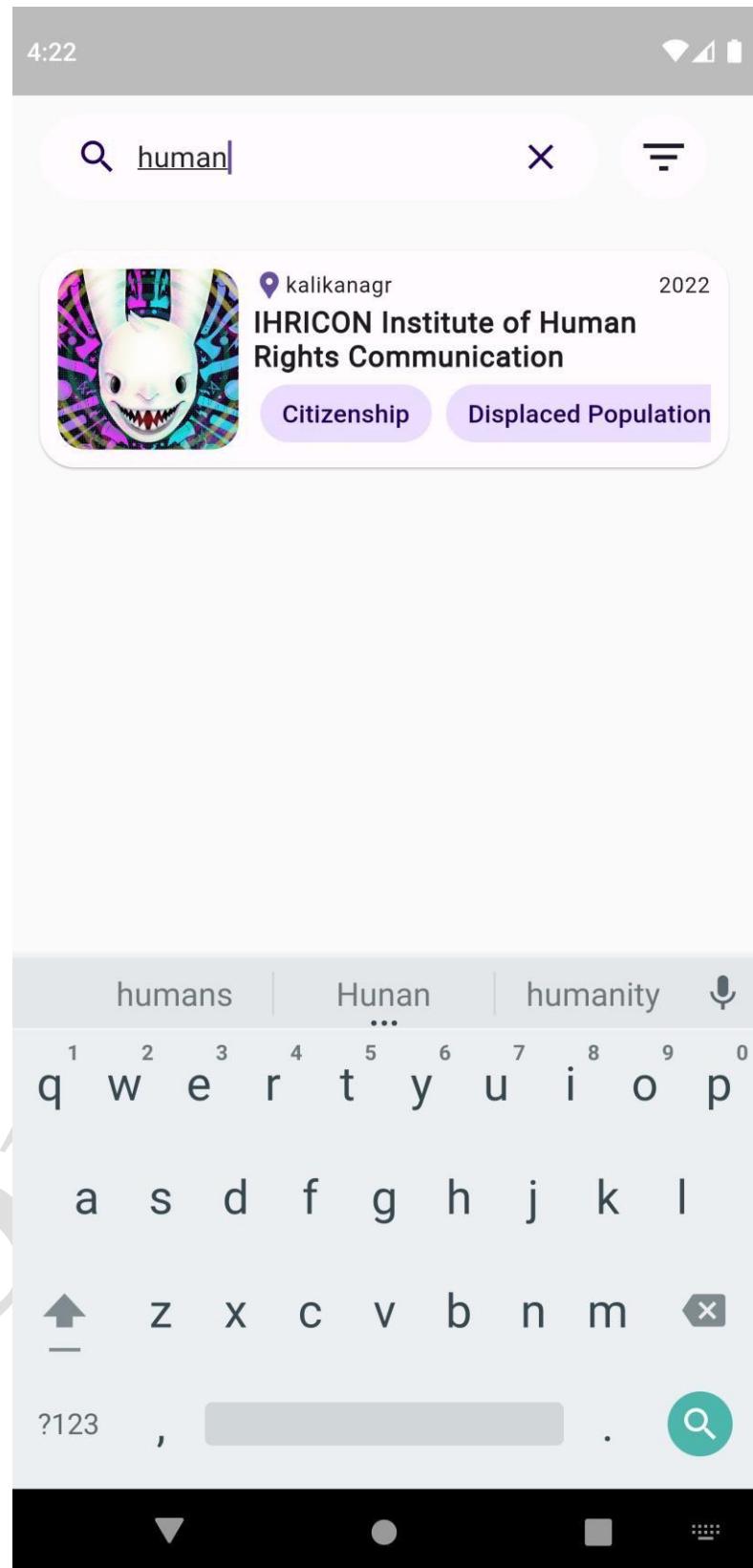


Figure 289: System: Sasae App, Search NGO(s)



Figure 290: System: Sasae App, NGO Profile Screen

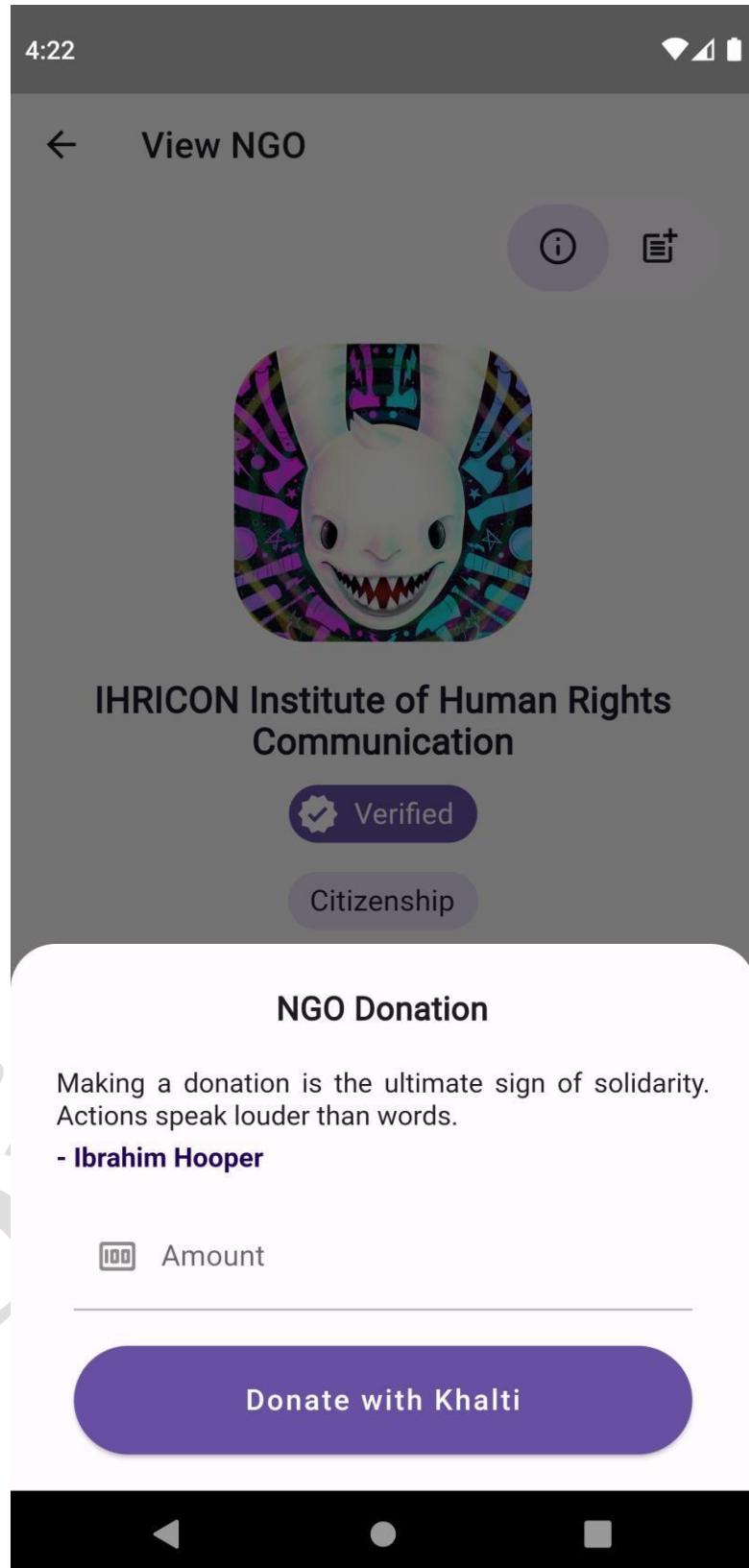


Figure 291: System: Sasae App, Donation to NGO

4:27



← Choose your payment method



KHALTI



CONNECT IPS



MOBILE BANKING

 Khalti Mobile Number *** Khalti MPIN

Amount

Rs. 5,000

PAY

Forgot Khalti MPIN?

RESET KHALTI MPIN

TEST

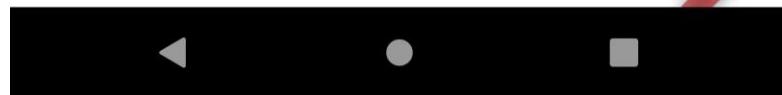


Figure 292: System: Sasae App, Khalti Payment as Donation

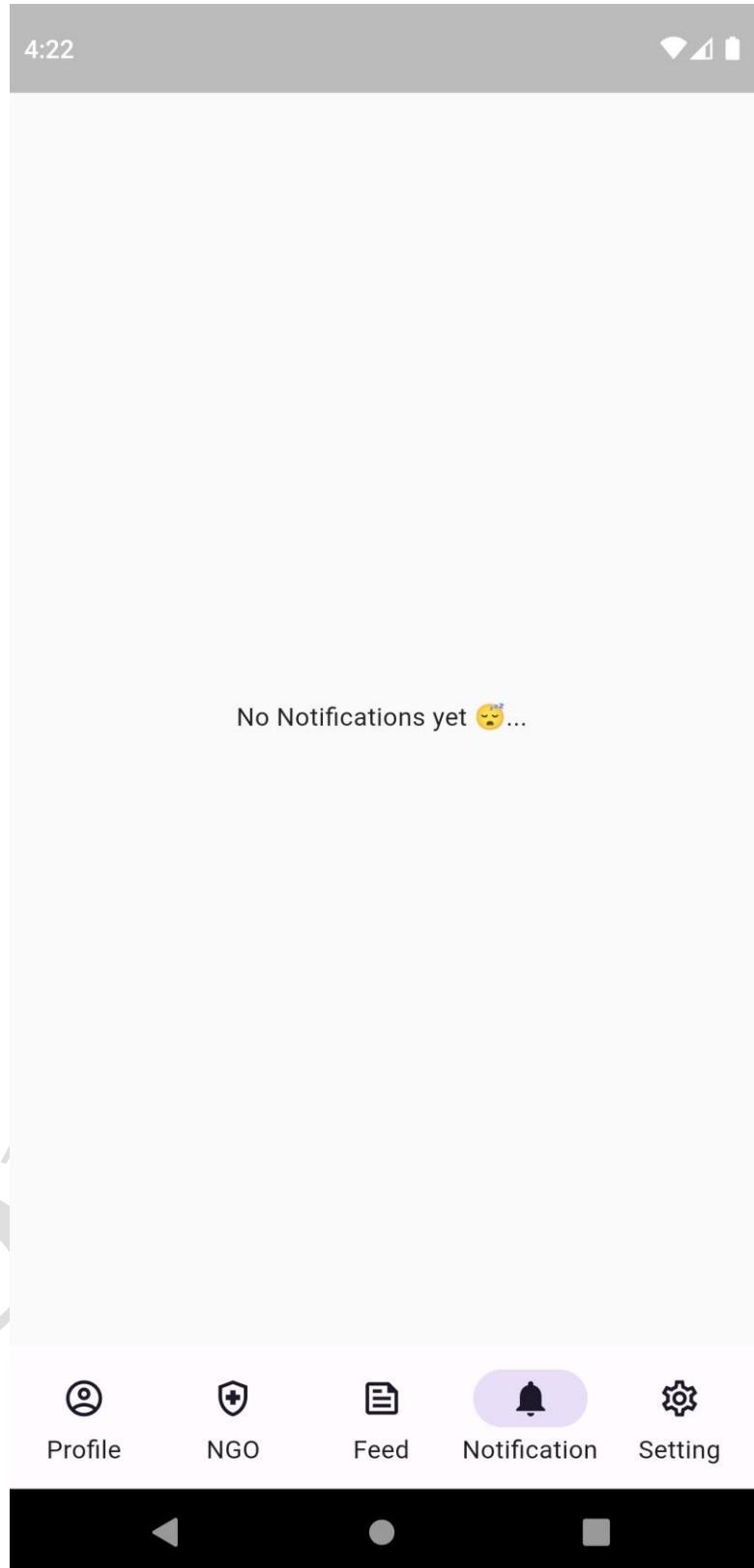


Figure 293: System: Sasae App, Notifications

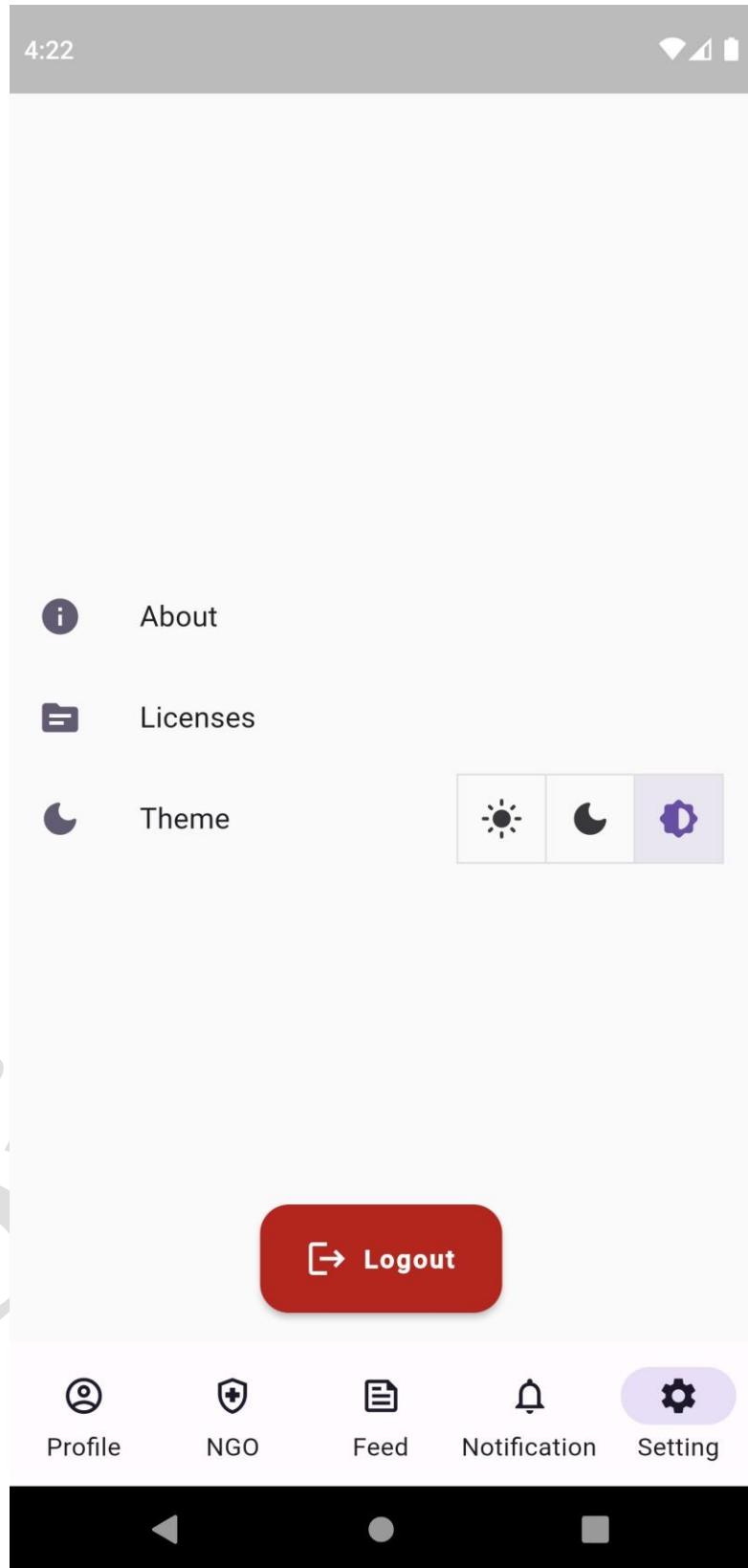


Figure 294: System: Sasae App, Setting

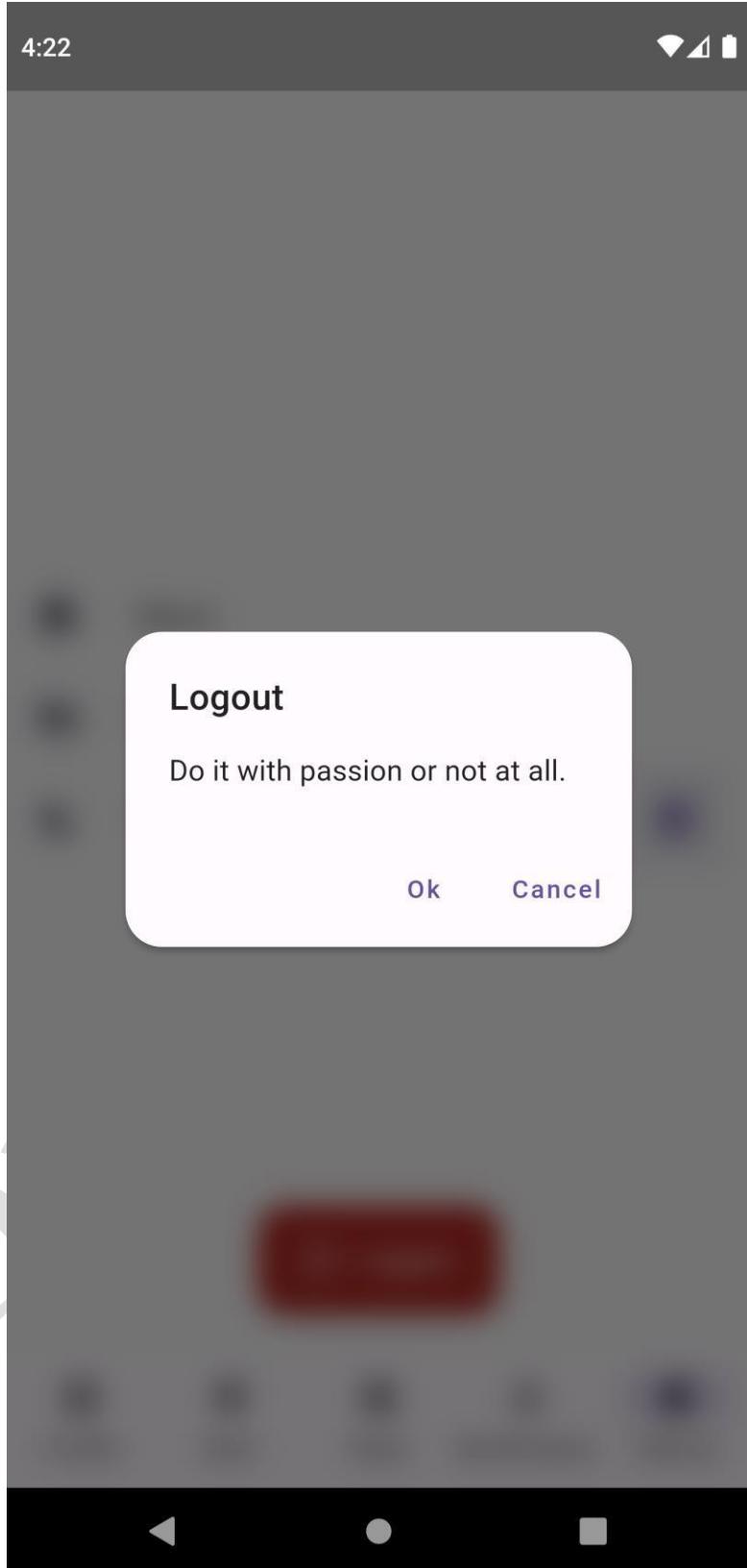


Figure 295: System: Sasae App, Logout

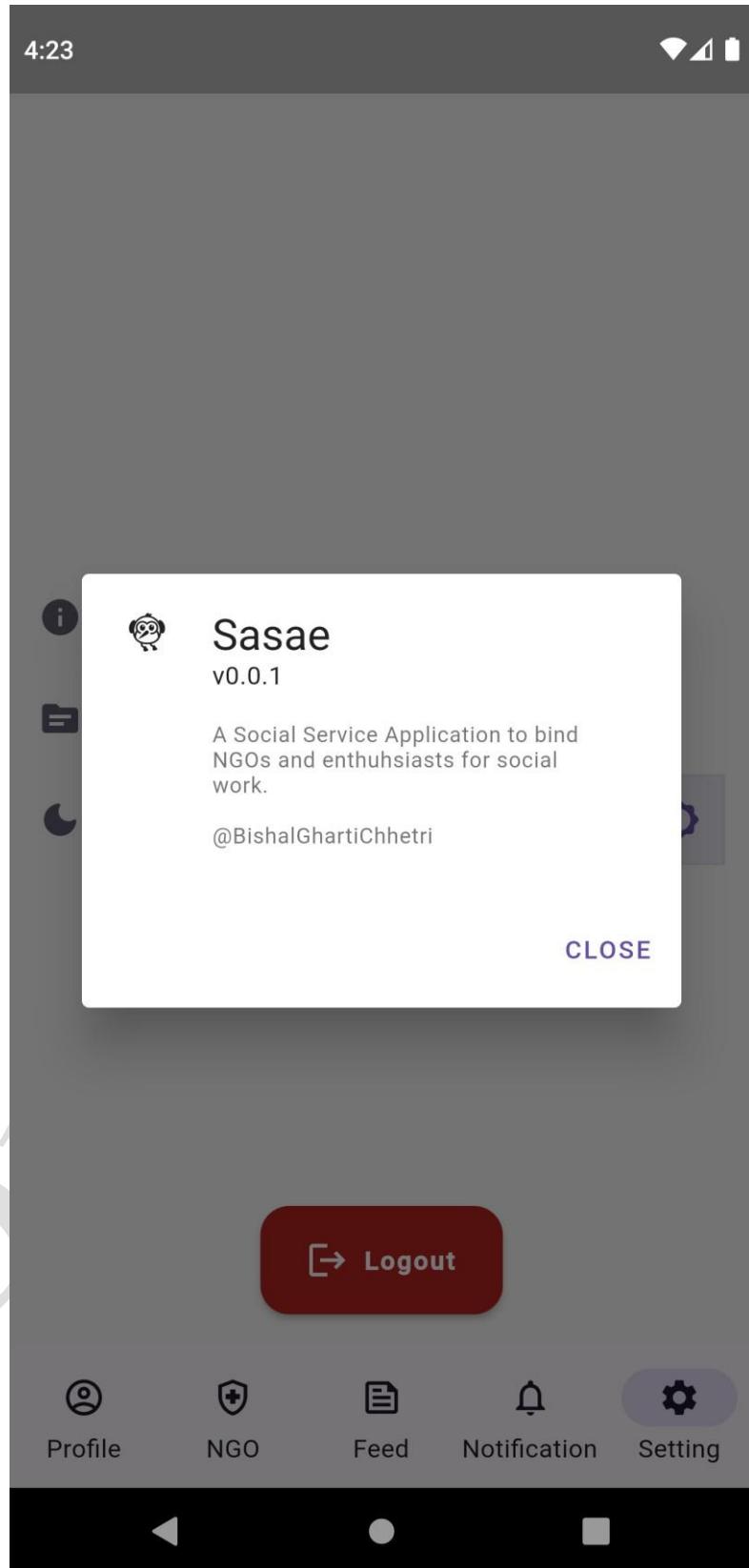
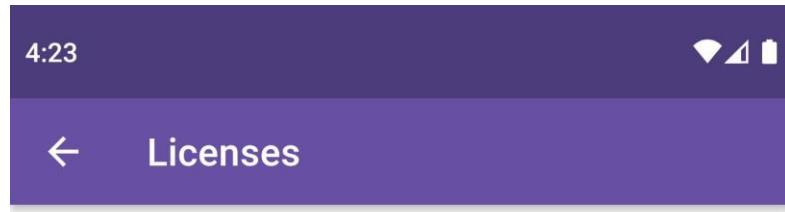


Figure 296: System: Sasae App, About App



StackWalker

2 licenses.

_fe_analyzer_shared

1 license.

abseil-cpp

3 licenses.

accessibility

15 licenses.

adaptive_theme

1 license.

aFileChooser

1 license.

analyzer

1 license.



Figure 297: System: Sasae App, App Licenses

7.11. APPENDIX K: TEST

7.11.1. SYSTEM TESTING

7.11.1.1. ADMIN-STAFF PORTAL TESTING

7.11.1.1.1. TEST STAFF LOGIN

Test Case	TC-2
Objective	To test whether the staff can log in or not
Action	Enter the staff credentials in the login screen and click the login button.
Expected Result	On login, the user must be redirected to the Staff dashboard/home.
Actual Result	The user was redirected to the Staff dashboard/home.
Conclusion	Test Successful

Table 63: System Testing, Admin-Staff Portal TC-2

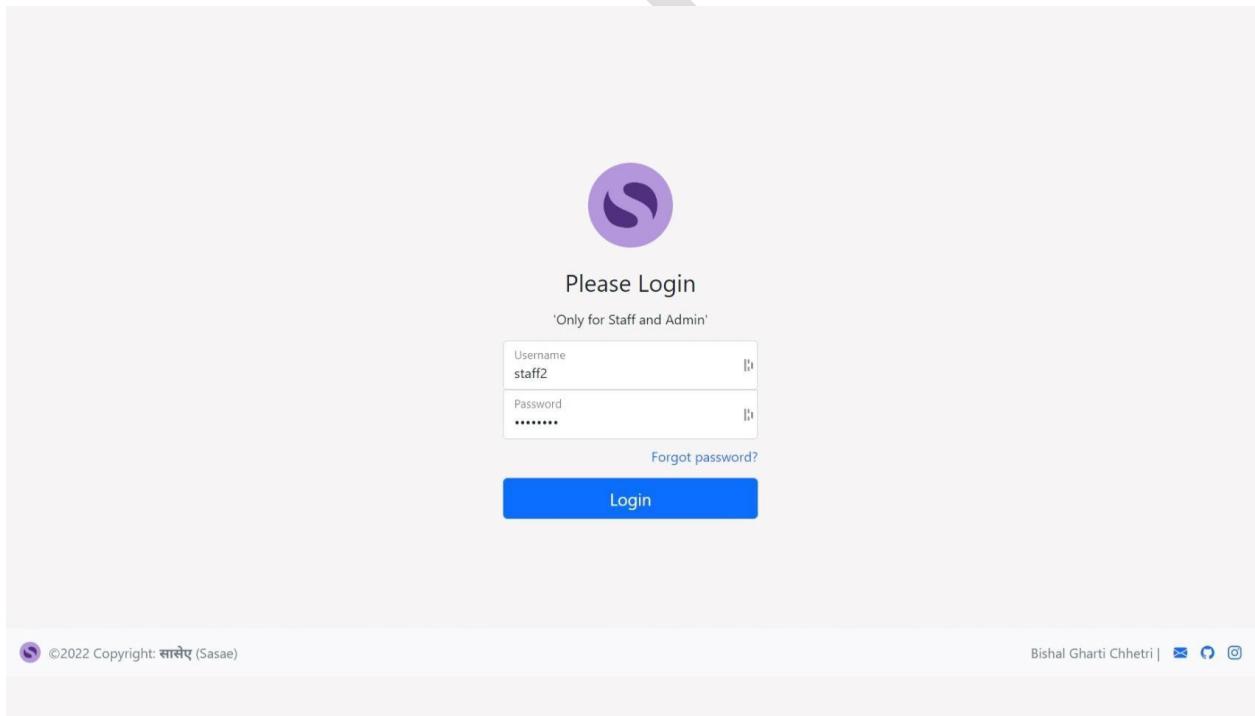


Table 64: Login page with Staff Credentials (Before)

The screenshot shows the SASAE Staff homepage. At the top left, it says "Hi staff2!" with links to Home, Staff, NGO, People, and Reported Post. On the top right are "Change Password" and "Logout" buttons. The main title "Welcome" is in large black font, followed by "Rahim Ahmbded," in purple. Below the title are two boxes: one orange box with the number "3" and "Total Pending Reports", and one yellow box with the number "6" and "Total Reviewed Reports". At the bottom left is a copyright notice: "©2022 Copyright: सासै (Sasae)". At the bottom right are social media links for Bishal Gharti Chhetri.

Table 65: Staff Homepage (After)

7.11.1.1.2. TEST MANAGE STAFF

Test Case	TC-3
Objective	To test the CRUD operations on Staff resource
Action	<ul style="list-style-type: none"> - Create a staff by filling out the form and clicking the register button. - View the newly created staff by clicking the “View” button - Update the staff with new information in the update form and click the “Done” button. - Delete the Staff by clicking the “Remove” button.
Expected Result	CRUD operation must be successful
Actual Result	All CRUD operations were successful
Conclusion	Test Successful

Table 66: System Testing, Admin-Staff Portal TC-3

SASAE

The screenshot shows a web-based administrative interface for managing staff members. At the top, there's a navigation bar with links for Home, Staff, NGO, People, and Reviewed Report. On the right side of the header are buttons for Change Password and Logout. Below the header is a search bar with a placeholder 'search...', a 'Filter by' dropdown, a magnifying glass icon, and a 'Reset' button. A large button labeled 'Create Staff' is positioned above a table. The table has columns for DP (Profile Picture), User Name, Date Joined, Active status, Last Login, and Action buttons. Each row contains two staff entries: staff1 and staff2. The staff1 entry was joined on April 12, 2022, and is active, with last login on April 25, 2022, at 7:42 p.m. The staff2 entry was also joined on April 12, 2022, and is active, with last login on April 25, 2022, at 8:56 p.m. Each row includes three action buttons: 'View' (blue), 'Edit' (green), and 'Remove' (red). At the bottom right of the table area, it says 'Page 1 of 1'. At the very bottom of the page, there's a footer with a copyright notice: '©2022 Copyright: सासै (Sasae)' and social media links for Bishal Gharti Chhetri.

DP	USER NAME	DATE JOINED	ACTIVE	LAST LOGIN	ACTION
	staff1	April 12, 2022	True	April 25, 2022, 7:42 p.m.	<button>View</button> <button>Edit</button> <button>Remove</button>
	staff2	April 12, 2022	True	April 25, 2022, 8:56 p.m.	<button>View</button> <button>Edit</button> <button>Remove</button>

Figure 298: Staffs Page

Add Staff

Username

sneha

Required: 150 characters or fewer. Letters, digits and @//+/-/_ only.

Email address

sneha@gmail.com

Password

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation

Enter the same password as before, for verification.

Full name

Sneha Shrestha

 ©2022 Copyright: सासैए (Sasae)

Bishal Gharti Chhetri |   

Gender

Female



Phone

9779844657384



Address

Lalitpur-10



Display picture

No file chosen

Citizenship photo

No file chosen

Is married

Figure 299: Filled Staff Registration Form (Before)

View Staff

User Name	sneha
Full Name	Sneha Shrestha
Gender	Female
Date of Birth	April 14, 2022
Phone	+9779844657384
Email	sneha@gmail.com
Address	Lalitpur-10
Marital Status	True
Display Picture	

Hi admin! [Home](#) [Staff](#) [NGO](#) [People](#) [Reviewed Report](#)
[Change Password](#) [Logout](#)



Last Login	None
Date Joined	April 25, 2022, 9:19 p.m.
Active	True
Post Reports	0
Reports Reviewed	0

[Edit](#) [Remove](#)

©2022 Copyright: सासैए (Sasae)
Bishal Gharti Chhetri |

Figure 300: Staff Profile Page (After)

SASAE

The screenshot shows the 'Edit Staff' form for a user named Sneha Shrestha. The fields filled are: Full name (Sneha Shrestha), Date of birth (04/14/2022), Gender (Female), Phone (984-4657374), and Address (Kathmandu-09). The 'Display picture' section shows a placeholder image and a 'Choose File' button. At the bottom, there is a checkbox for 'Is married' which is checked. A blue 'Done' button is at the bottom right.

Figure 301: Staff Update Form with Changed Field Data (Before)

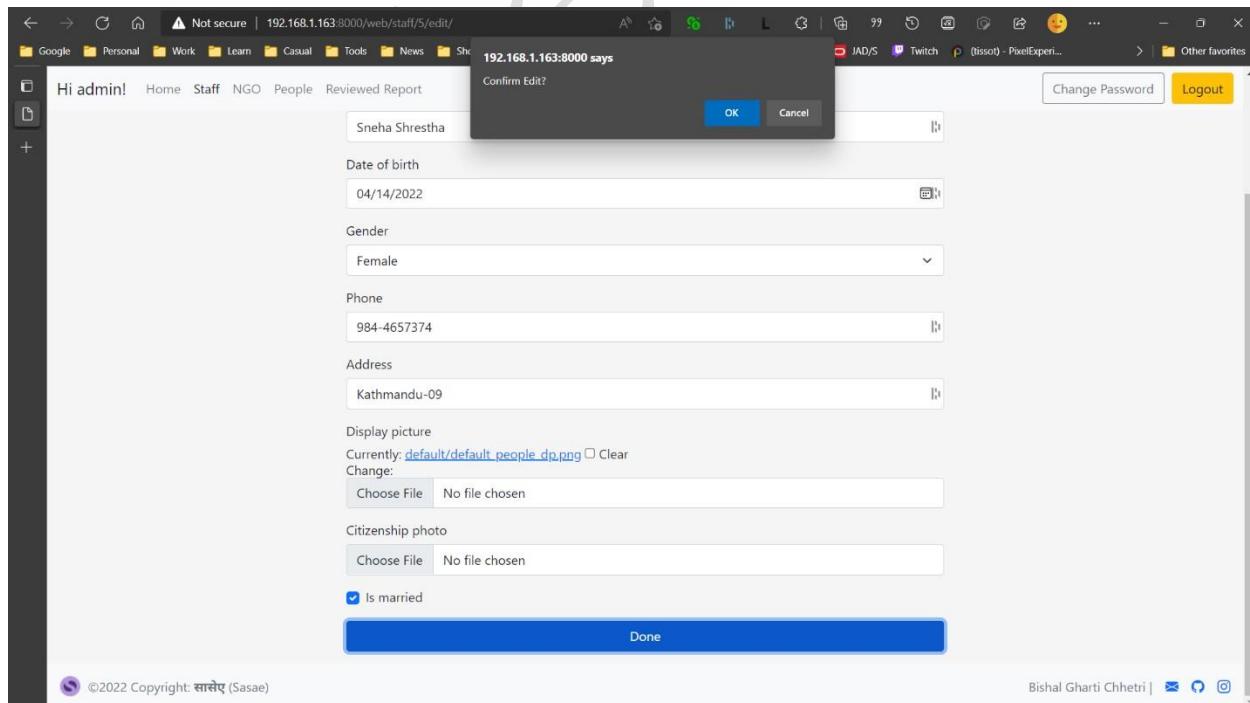


Figure 302: Confirm Update Prompt

View Staff

User Name	sneha
Full Name	Sneha Shrestha
Gender	Female
Date of Birth	April 14, 2022
Phone	+9779844657374
Email	sneha@gmail.com
Address	Kathmandu-09
Marital Status	True

[Display Picture](#)

©2022 Copyright: सासैए (Sasae)

Bishal Gharti Chhetri |



Last Login	None
Date Joined	April 25, 2022, 9:19 p.m.
Active	True
Post Reports	0
Reports Reviewed	0

[Edit](#) [Remove](#)

Figure 303: Staff Profile Page (After)

SASAE

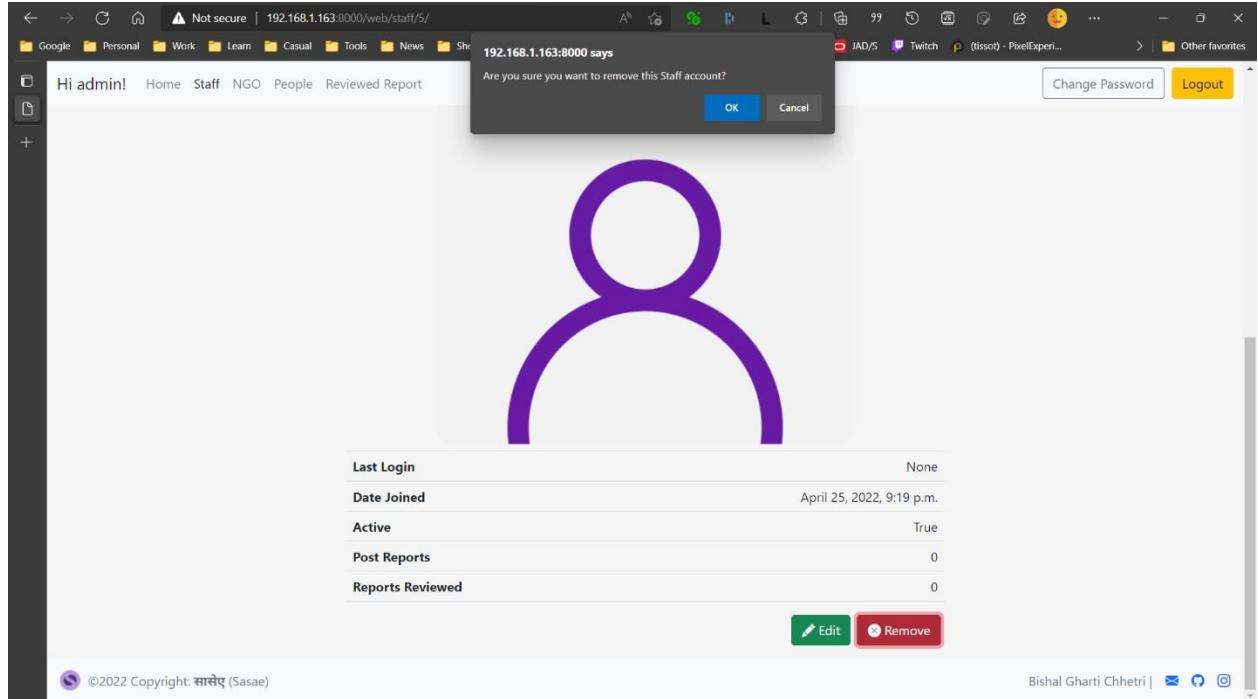


Figure 304: Staff Delete Prompt (Before)

A screenshot of the SASAE application showing the staff list. The URL is 192.168.1.163:8000/web/staffs/. The page displays a table with two rows of staff information:

DP	USER NAME	DATE JOINED	ACTIVE	LAST LOGIN	ACTION
	staff1	April 12, 2022	True	April 25, 2022, 7:42 p.m.	View Edit Remove
	staff2	April 12, 2022	True	April 25, 2022, 8:56 p.m.	View Edit Remove

The footer of the page includes the copyright notice "©2022 Copyright: सासै (Sasae)" and the name "Bishal Gharti Chhetri".

Figure 305: Staffs Page (After)

7.11.1.3. TEST MANAGE NGO

Test Case	TC-4
-----------	------

Objective	To test the CRUD operations on NGO resource
Action	<ul style="list-style-type: none"> - Create an NGO by filling out the form and clicking the “Register” button. - View the newly created NGO by clicking the “View” button - Update the NGO with new information in the update form and click the “Done” button. - Delete the NGO by clicking the “Remove” button.
Expected Result	CRUD operation must be successful
Actual Result	All CRUD operations were successful
Conclusion	Test Successful

Table 67: System Testing, Admin-Staff Portal TC-4

DP	NGO NAME	DATE JOINED	ACTIVE	LAST LOGIN	VERIFIED	ACTION
	INSEC Informal Sector Service Centre	March 27, 2022	True	April 16, 2022, 12:05 p.m.	True	<button>View</button> <button>Edit</button> <button>Remove</button>
	IHRICON Institute of Human Rights Communication	March 27, 2022	True	April 2, 2022, 10:12 p.m.	True	<button>View</button> <button>Edit</button> <button>Remove</button>
	BishalSanstha	April 25, 2022	True	None	True	<button>View</button> <button>Edit</button> <button>Remove</button>

Page 1 of 1

©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)

Figure 306: NGOs Page

Add NGO

Username
 Required: 150 characters or fewer. Letters, digits and @/_+/-/_ only.

Email address

Password
 Your password must be similar to your other account information.

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

Password confirmation
 Enter the same password as before, for verification.

Organization Name

Establishment date

Field of work
 Advocacy & Awareness
 Agriculture
 Business & Economic Policy
 Child Education
 Youth Empowerment
 Citizenship
 Communication
 Conflict Resolution
 Peace Building
 ICT
 Culture & Society
 Democracy & Civic Rights
 Rural Development
 Disability & Handicap

 ©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Twitter](#)

Environment
 Family Care
 Women's Rights
 Governance
 Health
 Human Rights
 Charity/Philanthropy
 Labor
 Law & Legal Affairs
 Migrant Workers
 Relief
 Reconstruction
 Rehabilitation
 Research & Studies
 Science
 Social Media
 Technology
 Transparency
 Training & Capacity Building

Phone

Address

Latitude

Longitude

Display picture
 No file chosen

Epay account
 - Default: Khati as an Electronic Payment Gateway

Social Welfare Council Affl Certificate
 61fd59cb62098.png

PAN Certificate
 987231.jpg

Is verified

Register

Figure 307: NGO Registration Page (Before)

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

View NGO

User Name	horne
Organization Name	Hornni Sanstha
Establishment Date	April 14, 2022
Phone	+9779800465783
Email	hs@gmail.com
Address	Baglung-7
Location	23.032322300000000, -12.232223410000000
Fields of Work	Conflict Resolution, Rural Development
Epay Account	9800465738
	Link Bank
Verified	True
Display Picture	

©2022 Copyright: सासैए (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)



Posted	0 posts
Poked On	0
Last Login	None
Date Joined	April 25, 2022, 9:40 p.m.
Active	True

[Edit](#) [Remove](#)

Figure 308: NGO Profile Page (After)

SASAE

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

Edit NGO

Organization Name
Laurani Sanstha

Establishment date
04/14/2022

Field of work

- Advocacy & Awareness
- Agriculture
- Business & Economic Policy
- Child Education
- Youth Empowerment
- Citizenship
- Communication
- Conflict Resolution
- Peace Building
- ICT
- Culture & Society
- Democracy & Civic Rights
- Rural Development

©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Twitter](#)

Education
 Environment
 Family Care
 Women's Rights
 Governance
 Health
 Human Rights
 Charity/Philanthropy
 Labor
 Law & Legal Affairs
 Migrant Workers
 Relief
 Reconstruction
 Rehabilitation
 Research & Studies
 Science
 Social Media
 Technology
 Transparency
 Training & Capacity Building

Phone
980-0465783

Address
Hetauda-9

Latitude
23.032322300000000

Longitude
-12.232223410000000

Display picture
Currently: [default/default.ngo.default.png](#) Clear
Change:
 Choose File No file chosen

Epay account
9800465738

- Default: Khalti as an Electronic Payment Gateway

Social Welfare Council Affl Certificate
Currently: [ngo/swc/987231.WnMaj.jpg](#) Clear
Change:
 Choose File No file chosen

PAN Certificate
Currently: [ngo/pan/61fd58a9b8a8a.jpg](#) Clear
Change:
 Choose File No file chosen

Is verified

Done

Figure 309: NGO Update Page with changed Field Data (Before)

SASAE

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

View NGO

User Name	home
Organization Name	Laurani Sanstha
Establishment Date	April 14, 2022
Phone	+9779800465783
Email	hs@gmail.com
Address	Hetauda-9
Location	23.032322300000000, -12.232234100000000
Fields of Work	Conflict Resolution, ICT, Culture & Society, Rural Development
Epay Account	9800465738
	Link Bank
Verified	True
Display Picture	
	
©2022 Copyright: सासै (Sasae)	
Bishal Gharti Chhetri Email Facebook Twitter Instagram	
Social Welfare Council Affil Certificate	
	
PAN Certificate	
	
Posted	0 posts
Poked On	0
Last Login	None
Date Joined	April 25, 2022, 9:40 p.m.
Active	True
Edit	Remove

Figure 310: NGO Profile Page (After)

SASAE

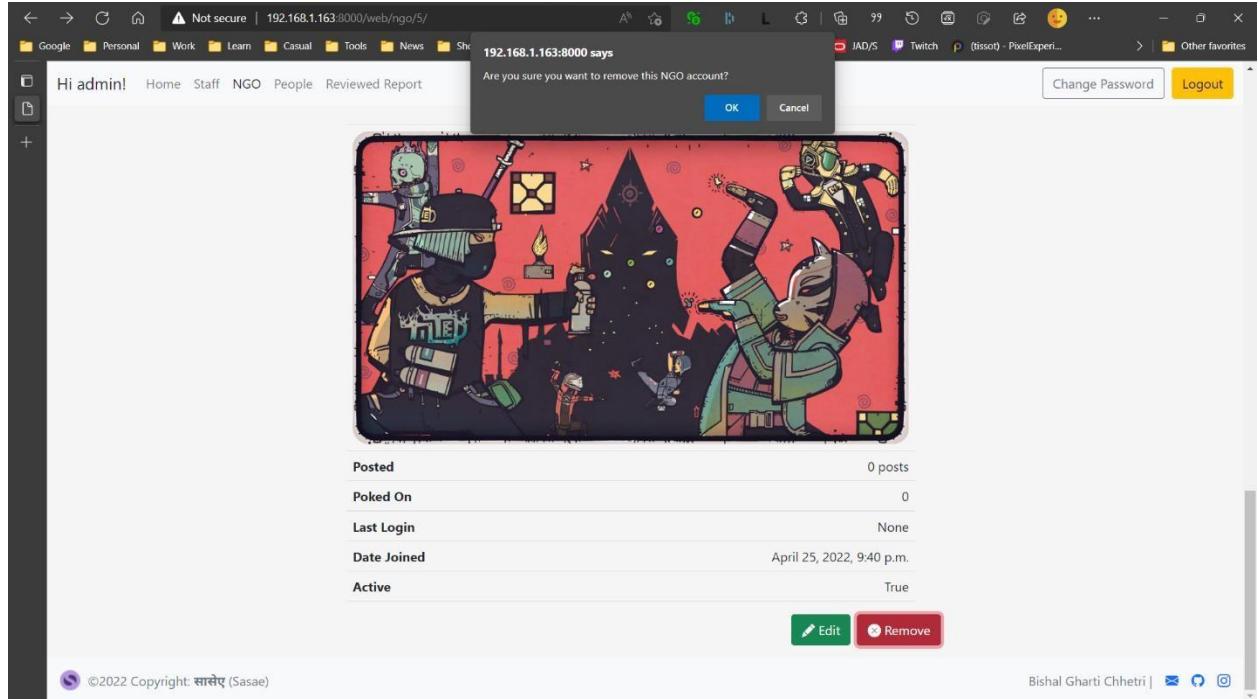


Figure 311: NGO Delete Prompt (Before)

DP	NGO NAME	DATE JOINED	ACTIVE	LAST LOGIN	VERIFIED	ACTION
	INSEC Informal Sector Service Centre	March 27, 2022	True	April 16, 2022, 12:05 p.m.	True	View Edit Remove
	IHRICON Institute of Human Rights Communication	March 27, 2022	True	April 2, 2022, 10:12 p.m.	True	View Edit Remove
	BishalSanstha	April 25, 2022	True	None	True	View Edit Remove

Figure 312: NGOs Page (After)

7.11.1.4. TEST MANAGE GENERAL PEOPLE

Test Case	TC-5
-----------	------

SASAE

Objective	To test the Read, Update, and Delete operations on General People resource
Action	<ul style="list-style-type: none"> - Create a staff by filling out the form and clicking the register button. - View the newly created staff by clicking the “View” button - Update the staff with new information in the update form and click the “Done” button. - Delete the Staff by clicking the “Remove” button.
Expected Result	CRUD operation must be successful
Actual Result	All CRUD operations were successful
Conclusion	Test Successful

Table 68: System Testing, Admin-Staff Portal TC-5

DP	USER NAME	DATE JOINED	ACTIVE	LAST LOGIN	VERIFIED	ACTION
	bishal	March 27, 2022	True	April 25, 2022, 5:14 p.m.	True	
	mango	March 27, 2022	True	None	False	
	sujata	March 29, 2022	True	April 25, 2022, 6:04 p.m.	False	
	test	March 30, 2022	True	None	False	
	mkbhd1	April 4, 2022	True	None	False	
	ram	April 4, 2022	True	April 13, 2022, 11:50 p.m.	False	
	kamskl	April 4, 2022	False	None	False	
	harilalu	April 23, 2022	True	None	False	

search... Filter by

Page 1 of 2 | [next](#) [last »](#)

©2022 Copyright: सासै (Sasae) Bishal Gharti Chhetri |

Figure 313: General People Page

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

View Person

User Name	sujata
Full Name	Sujata Magar
Gender	Female
Date of Birth	March 23, 2000
Phone	+9779800465783
Email	mands@gmail.com
Address	Kathmandu, Maitidevi
Verified	False

Display Picture



©2022 Copyright: सासैए (Sasae) Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)



Posted	1 posts
Last Login	April 25, 2022, 6:04 p.m.
Date Joined	March 29, 2022, 7:43 p.m.
Active	True

[Edit](#) [Remove](#)

Figure 314: General People Profile

Hi admin! Home Staff NGO People Reviewed Report

Change Password Logout

Edit Person

Full name
Sujata Magar

Date of birth
03/23/2000

Gender
Female

Phone
980-0467485

Address
Delhi, Nepal

Display picture
Currently: [display_picture/19030070_YCKXDRMYHD.jpg](#) Clear
Change:
 No file chosen

©2022 Copyright: सासैए (Sasae)

Bishal Gharti Chhetri |

Is verified

Figure 315: Update General People Form with changed Field Data (Before)

SASAE

Hi admin! Home Staff NGO People Reviewed Report

[Change Password](#)

[Logout](#)

View Person

User Name	sujata
Full Name	Sujata Magar
Gender	Female
Date of Birth	March 23, 2000
Phone	+9779800467485
Email	mands@gmail.com
Address	Delhi, Nepal
Verified	True

Display Picture



©2022 Copyright: सासैए (Sasae)

Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)



Citizenship photo>



Posted	1 posts
Last Login	April 25, 2022, 6:04 p.m.
Date Joined	March 29, 2022, 7:43 p.m.
Active	True

[Edit](#) [Remove](#)

Figure 316: General People Profile Page (After)

312

BISHAL GHARTI CHHETRI

SASAE

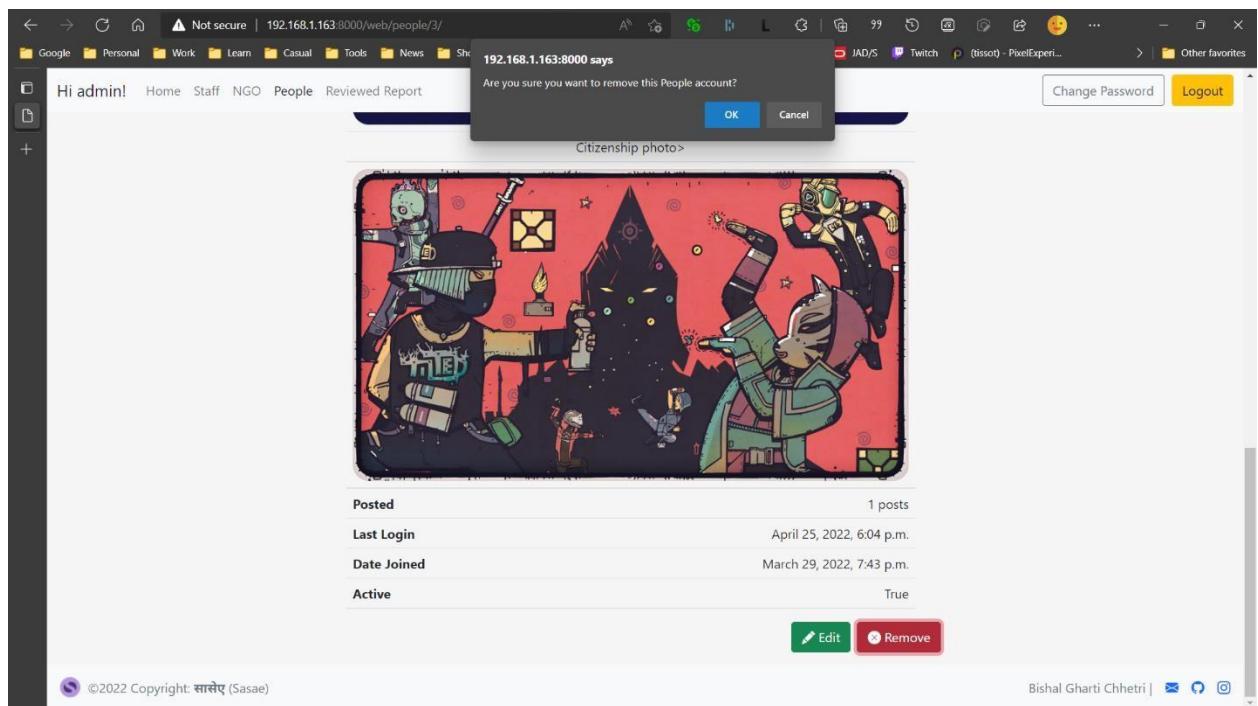


Figure 317: General People Delete Prompt (Before)

A screenshot of the 'People' page in the SASAE application. The page lists users with their details and actions. The users listed are:

DP	USER NAME	DATE JOINED	ACTIVE	LAST LOGIN	VERIFIED	ACTION
	bishal	March 27, 2022	True	April 25, 2022, 5:14 p.m.	True	
	mango	March 27, 2022	True	None	False	
	test	March 30, 2022	True	None	False	
	mkbhd1	April 4, 2022	True	None	False	
	ram	April 4, 2022	True	April 13, 2022, 11:50 p.m.	False	
	kamskl	April 4, 2022	False	None	False	
	harilalu	April 23, 2022	True	None	False	
	haril373	April 23, 2022	True	None	False	

At the bottom, there is a navigation bar with 'Page 1 of 2', 'next', and 'last »'. The footer includes the copyright notice '©2022 Copyright: सासै (Sasae)' and social media links.

Figure 318: General People Page (After)

7.11.1.5. TEST REVIEW REPORTED POST

Test Case	TC-6
-----------	------

Objective	To test whether the reported post can be reviewed or not.
Action	After clicking a reported post, fill the report form that includes the reason text field and action dropdown button and click the “Take Action” button
Expected Result	The user must be redirected to a list of remaining reported posts for review with a count of less than 1
Actual Result	The user was redirected to a list of reported posts lesser than before
Conclusion	Test Successful

Table 69: System Testing, Admin-Staff Portal TC-6

The screenshot shows the SASAE Admin-Staff Portal interface. At the top, there is a navigation bar with links for Home, Staff, NGO, People, and Reported Post. On the right side of the header, there are buttons for Change Password and Logout. Below the header, the page title is "Reported Posts". It displays three reported posts:

- Post 1:** Total Reported Posts: 9. Status: La (Last). Content: Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making Reporting status: Cu, Re, Hu. Date: April 7, 2022.
- Post 2:** Total Reviewed Posts: 6. Status: Cu (Current). Content: There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words Reporting status: Re, Hu. Date: March 29, 2022.
- Post 3:** Status: Re (Reviewed). Content: It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The Reporting status: Hu. Date: April 23, 2022.

At the bottom left, there is a copyright notice: ©2022 Copyright: सासै (Sasae). At the bottom right, there are social media icons for Facebook, Twitter, and LinkedIn, along with the name Bishal Gharti Chhetri.

Figure 319: Reported Posts Page (before)

SASAE

The screenshot shows a user interface for reviewing a reported post. At the top, there's a navigation bar with links for Home, Staff, NGO, People, and Reported Post. On the right, there are buttons for Change Password and Logout.

Post Details:

- Posted by: ngo1
- Date Posted: March 30, 2022, 12:11 a.m.
- Post Type: Request, Petition

Post related to: Training & Capacity Building, Culture & Society, Research & Studies, Human Rights

Post Content:

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary.

Total Signs: 1 | Request ends on: March 30, 2022, noon

Poked to NGOs: n1-ngo1, n2-ngo2

Review Section:

Reason: Porta facilisis tortor dignissim taciti pellentesque tellus quisque sem, vitae viverra commodo nisi aliquam accumsan euismod, eleifend mollis justo ante a nisl imperdiet. Commodo ut dapibus tempor montes sodales mus viverra auctor porttitor tempus ridiculus posuere sed torquent, congue curae suscipit lacinia fringilla voluptat tincidunt semper lobortis

Action: Post Remove

Buttons: Take Action

At the bottom, there's a footer with a copyright notice: ©2022 Copyright: सासै (Sasae) and social media links for Bishal Gharti Chhetri.

Figure 320: Reviewing the Reported Post

The screenshot shows the reported post page after it has been reviewed. At the top, there's a navigation bar with links for Home, Staff, NGO, People, and Reported Post. On the right, there are buttons for Change Password and Logout.

Total Reported Posts: 9 | **Total Reviewed Posts: 7**

Post Preview:

La Co
Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ...
P|R|P April 7, 2022

Post Preview:

Co Ed
It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The ...
P P|P April 23, 2022

At the bottom, there's a footer with a copyright notice: ©2022 Copyright: सासै (Sasae) and social media links for Bishal Gharti Chhetri.

Figure 321: Reported Post Page (After)

7.11.1.6. TEST READ REPORTED POST

Test Case	TC-7
Objective	To test to read the reviewed reports.

Action	Click the ‘Reported Posts’ nav item to list all the reviewed reports. Click a report to read the review of the reported post along with the staff name who acted on it.
Expected Result	The user must be redirected to the review report page.
Actual Result	The user was redirected to the review report page.
Conclusion	Test Successful

Table 70: System Testing, Admin-Staff Portal TC-7

SASAE

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

Total Reported Posts: 36 Total Reviewed Posts: 19

He La Re Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, ... PIN March 28, 2022	Co La Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ... PRJ April 7, 2022	He Re Lorem Ipsum is simply dummy text of the turies, but also the leap into. A P PIN April 3, 2022	Re He Lorem Ipsum is simply dummy text of the turies, but also the leap into. A P PIN April 3, 2022
Wo He mansdwjhwjw PIN April 10, 2022	Bu Te wowow great P PIN April 8, 2022	Bu Ad Ag mango amskcookcms PIN April 3, 2022	Di Ci mklijdds jsshb ajjsjd 1 PRJ April 3, 2022
Co La Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ... A PRJ March 27, 2022	He Re La Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, ... PIN March 28, 2022	La Co Ed Tr Ru the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution ... A P PIN April 7, 2022	La Co Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ... PRJ April 7, 2022
 ©2022 Copyright: सासैए (Sasae) ysryfztzugxjgxugx A P PIN April 13, 2022	Co Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making ... PRJ April 7, 2022	Ed Co It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The ... P PIN March 29, 2022	Bishal Gharti Chhetri    Lorem Ipsum is simply dummy text of the turies, but also the leap into. P PIN March 30, 2022

Figure 322: Reviewed Reports Page

Hi admin! Home Staff NGO People Reviewed Report Change Password Logout

Posted by: **bishal** Date Posted: April 3, 2022, 4:55 p.m. Post Type: **Normal**

Post related to: Health Relief

Post Content:
Lorem Ipsum is simply dummy text of the turies, but also the leap into.

Post attached Photo:

Poked to NGOs:

n1-ngo1

Let's Review!

Up-Votes	0
Down-Votes	0
Reports	1

Reason
I don't know. This needs to be removed ASAP.

Action
Post Remove

Reviewed by: Neha Singh

Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)

©2022 Copyright: सासै (Sasae)

Figure 323: Read Reviewed Post Page

7.11.1.1.7. TEST LOGOUT

Test Case	TC-8
Objective	To test logging out
Action	Click the ‘Logout’ button that is at the top-right of the navbar. And confirm the prompt.
Expected Result	The user must be redirected to a Logout message page.
Actual Result	The user was logged out to a Logout message page.
Conclusion	Test Successful

Table 71: System Testing, Admin-Staff Portal TC-8

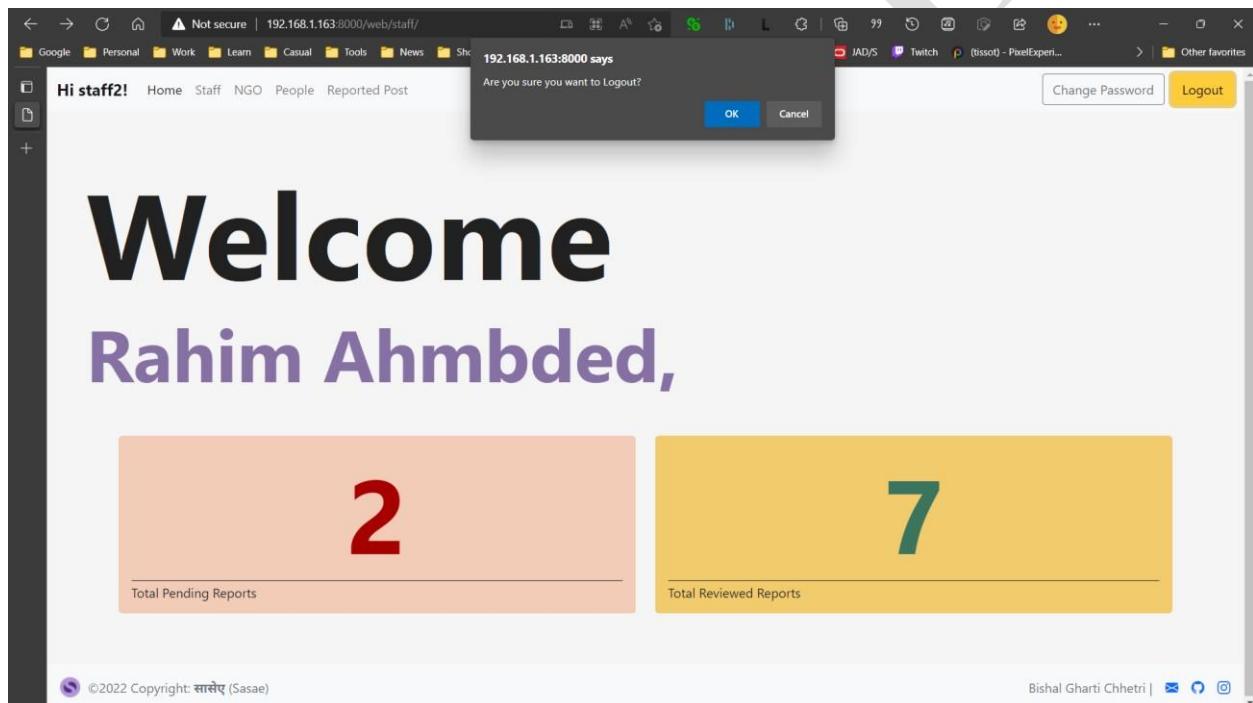


Figure 324: Logout Prompt (Before)

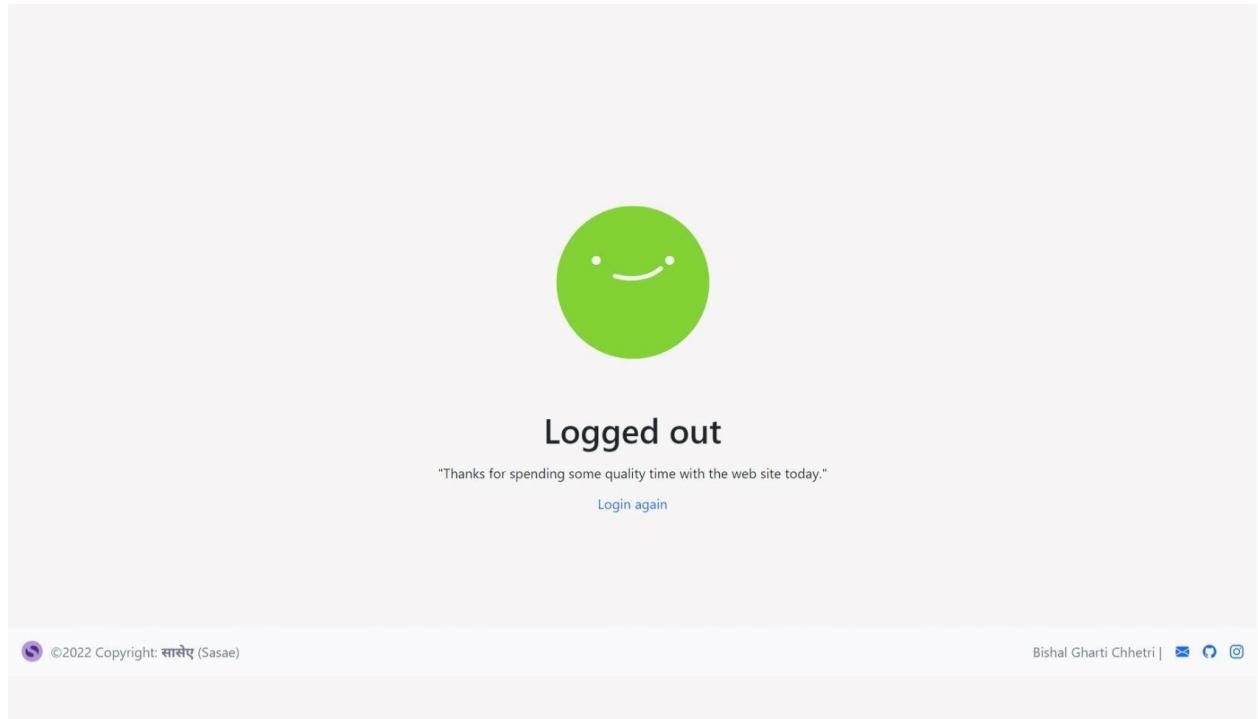


Figure 325: Successful Logout Message Page (After)

7.11.1.1.8. TEST CHANGE PASSWORD

Test Case	TC-8
Objective	To test logging out
Action	Click the ‘Change Password’ button that is at the top-right of the navbar. Fill out the rendered form containing the old password, and new password and confirm a new password. Then click ‘Change my Password’.
Expected Result	The user must be redirected to the ‘Change password successful’ message page.
Actual Result	The user was redirected to the ‘Change password successful’ message page.
Conclusion	Test Successful

Table 72: System Testing, Admin-Staff Portal TC-9

©2022 Copyright: सासैए (Sasae)

Bishal Gharti Chhetri | [Email](#) [Facebook](#) [Instagram](#)

Figure 326: Password Change Page with filled fields (Before)

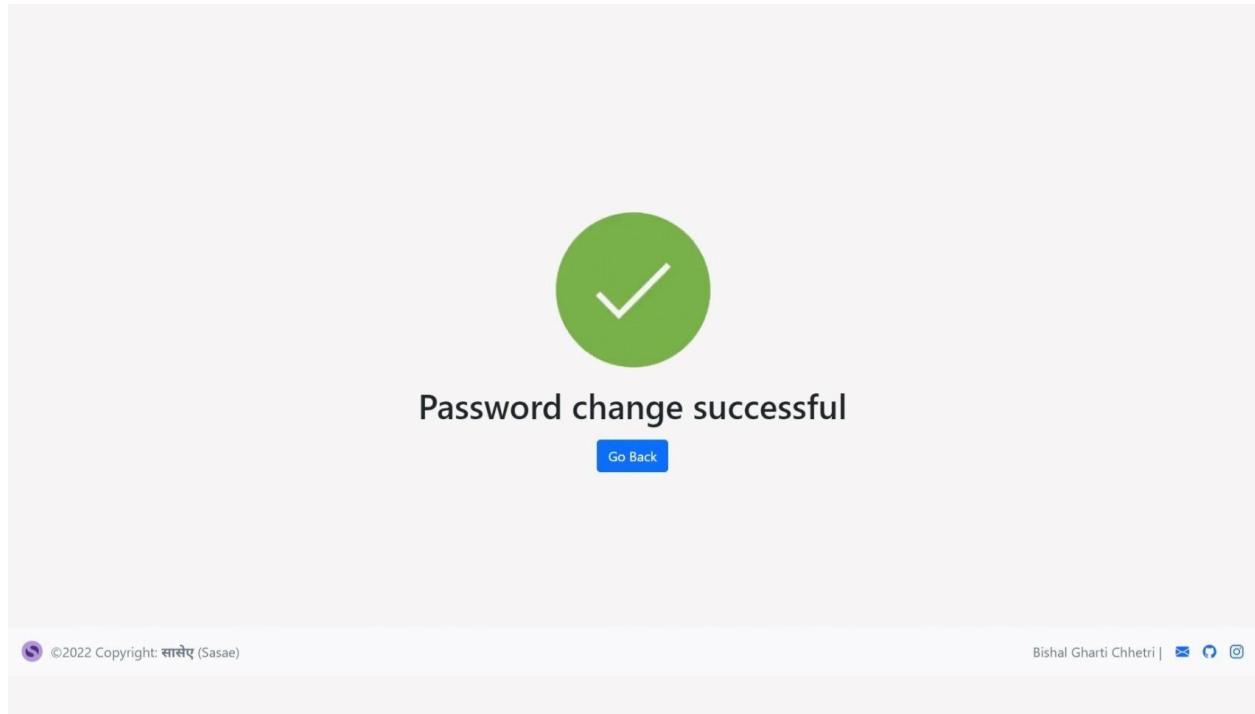


Figure 327: Password Change Successful Message Page (After)

7.11.1.2. SASAE MOBILE APP TESTING

7.11.1.2.1. TEST GENERAL PEOPLE LOGIN

Test Case	TC-2
Objective	To test logging into the app
Action	From the Login screen, Enter the credentials and click the 'Right Arrow' icon button.
Expected Result	The user must be navigated to the Feed page.
Actual Result	The user was navigated to the Feed page.
Conclusion	Test Successful

Table 73: System Testing, Sasae Mobile App TC-2

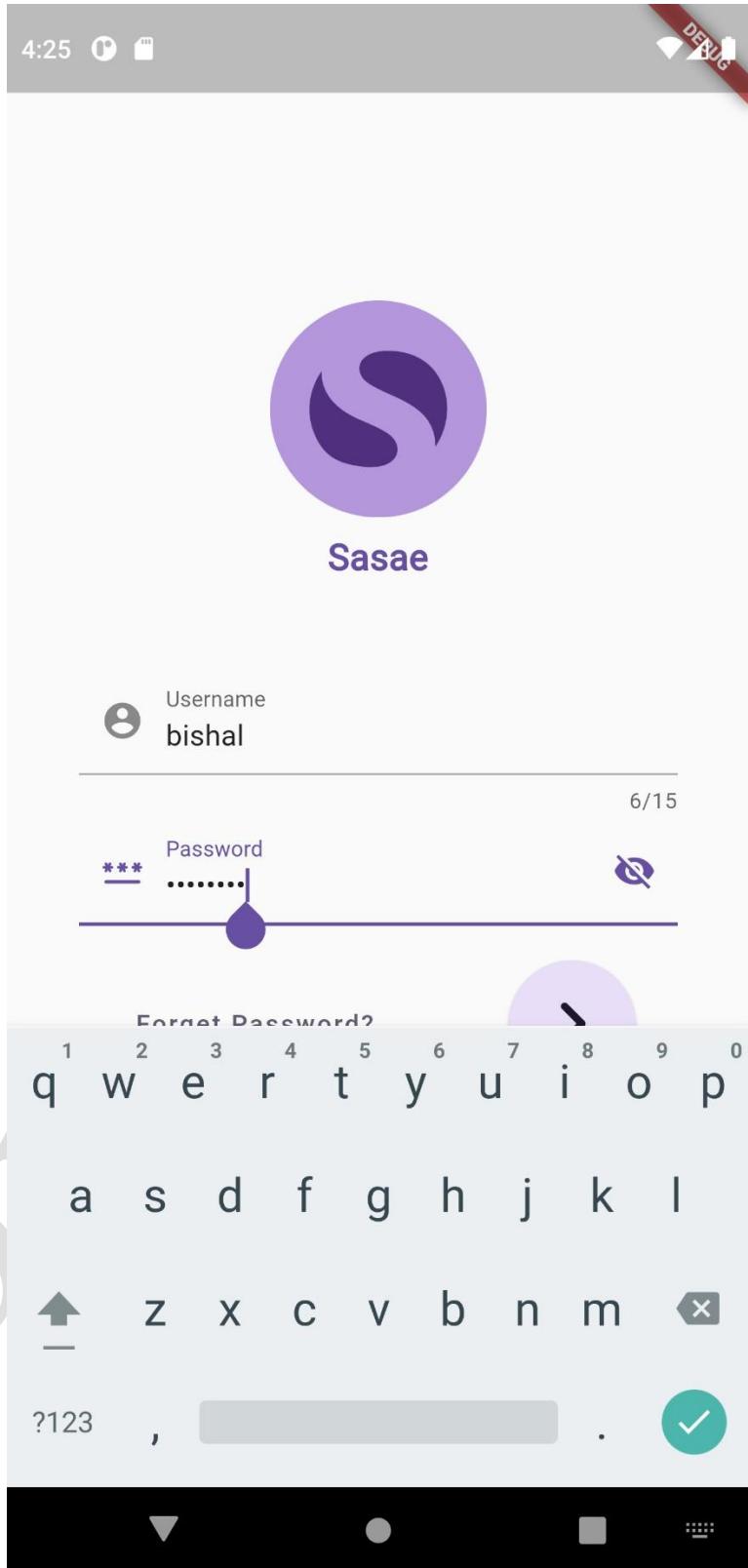


Figure 328: Mobile, Login Screen with filled credentials (Before)

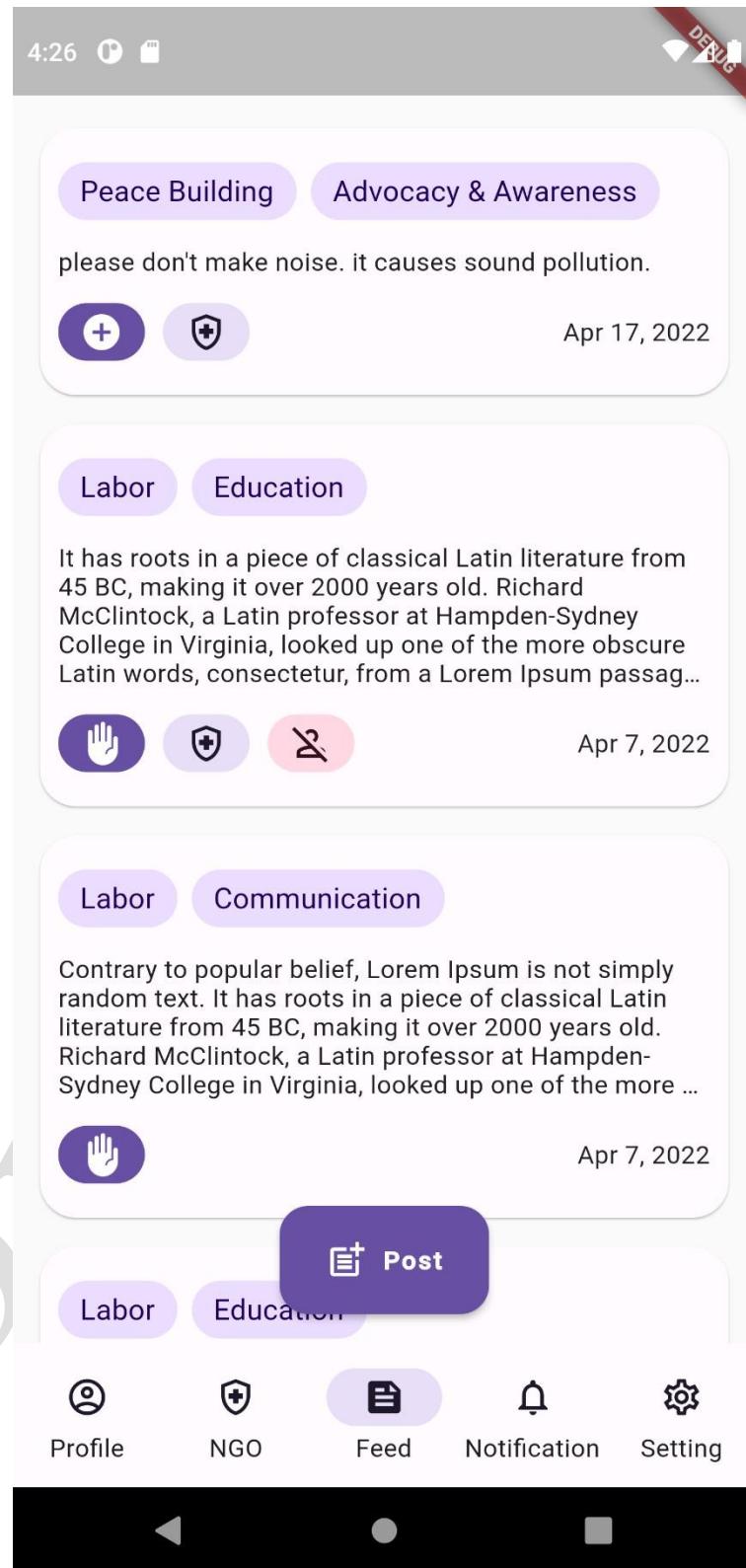


Figure 329: Mobile, Posts Screen (After)

7.11.1.2.2. TEST CHANGE PASSWORD

Test Case	TC-3
Objective	To test the password change feature
Action	From the Profile Page, Click the ‘password’ icon button. Fill in the old password, and new password and confirm the new password in the appeared modal sheet. Then hit the ‘change’ button.
Expected Result	The modal sheet must be popped with a ‘success’ message in the snack bar.
Actual Result	‘Success’ message was shown to the User.
Conclusion	Test Successful

Table 74: System Testing, Sasae Mobile App TC-3

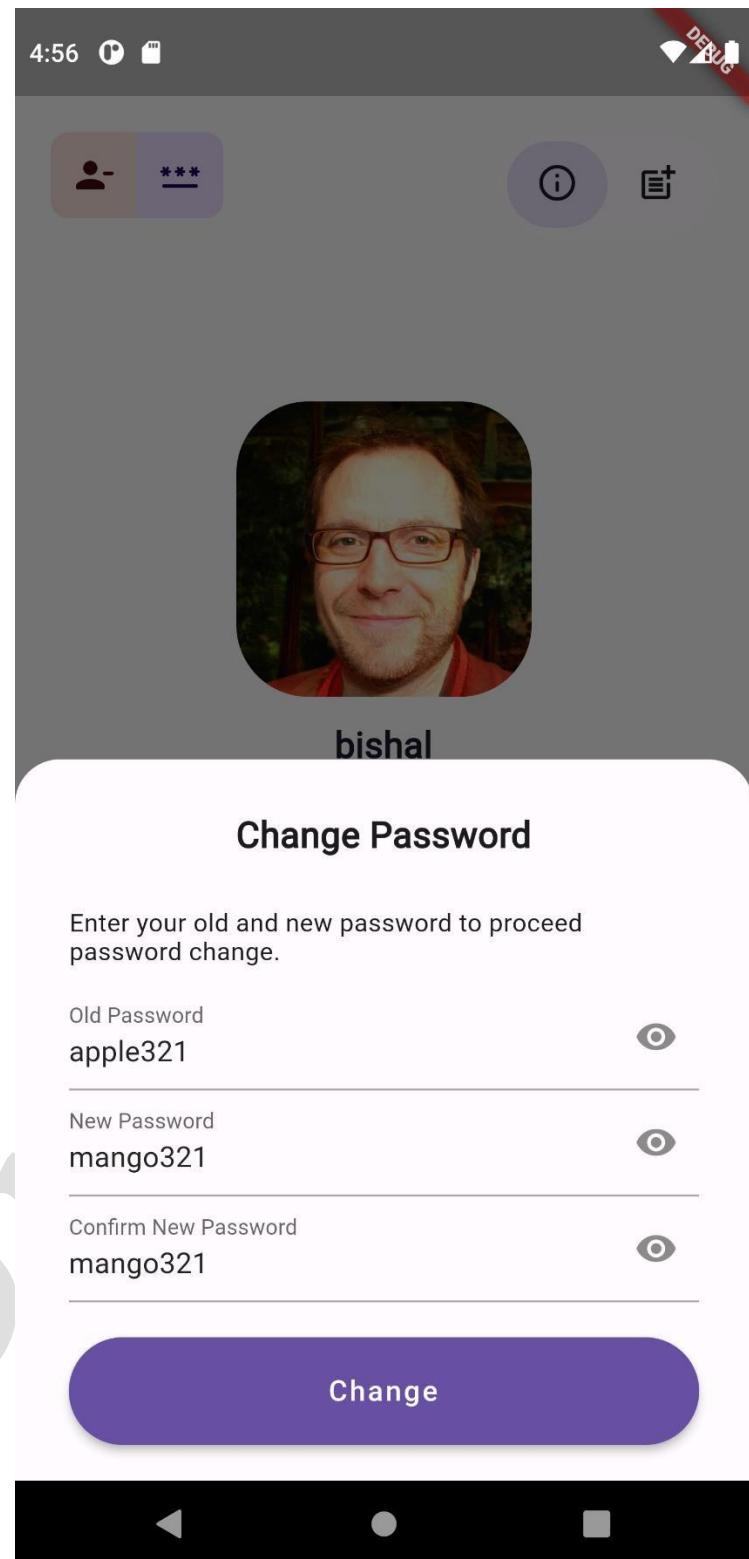


Figure 330: Mobile, Change Password Modal Bottom Sheet with filled data (Before)

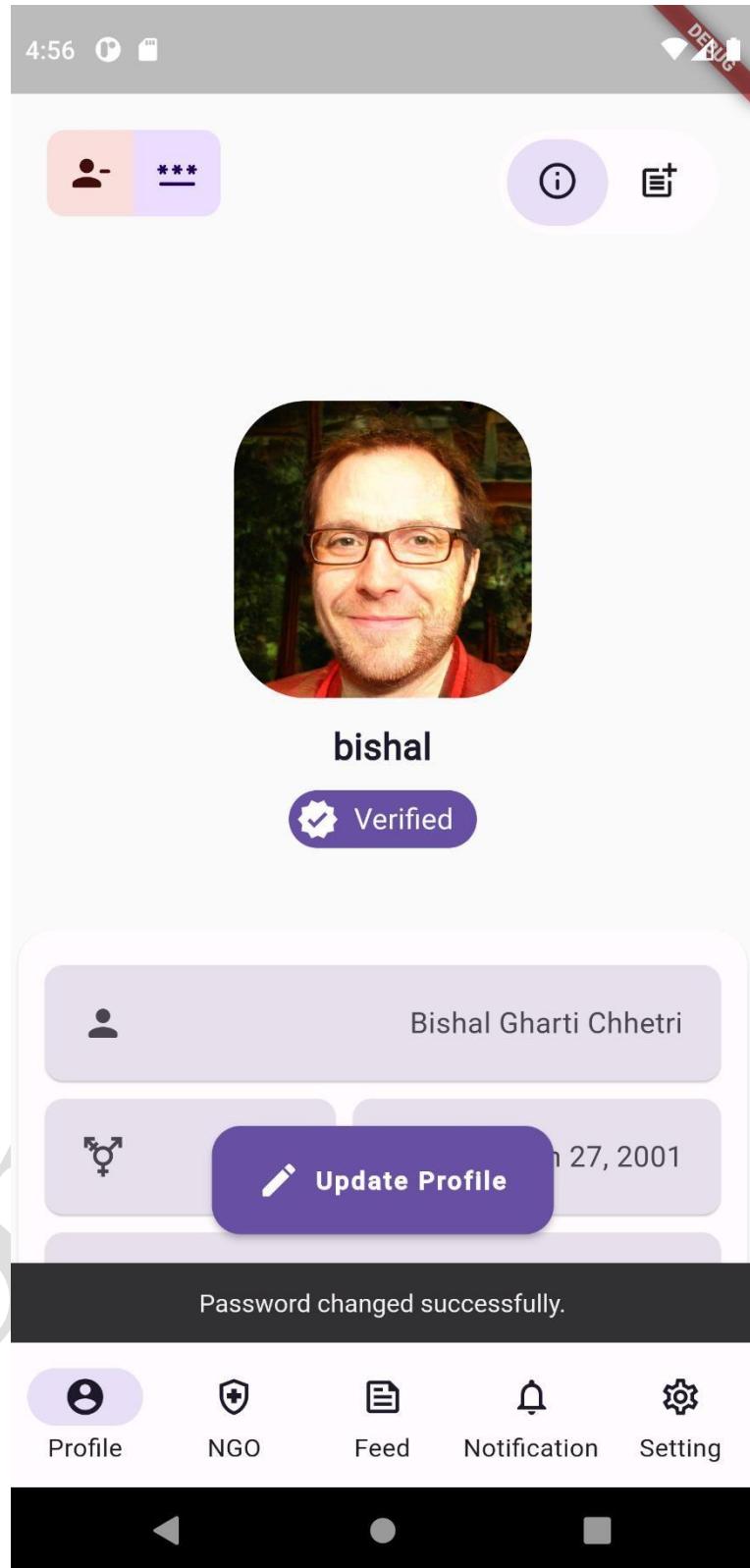


Figure 331: Mobile, General People Profile (After)

7.11.1.2.3. TEST VIEW NGO(S)

Test Case	TC-4
Objective	To test the view feature on NGO(s)
Action	<ul style="list-style-type: none"> - From the Home screen, navigate to the NGO page to view a list of NGOs. - Click an NGO option to view its profile in detail.
Expected Result	Users must be able to view a list of NGOs and an individual NGO.
Actual Result	The user was able to view NGO(s)
Conclusion	Test Successful

Table 75: System Testing, Sasae Mobile App TC-4

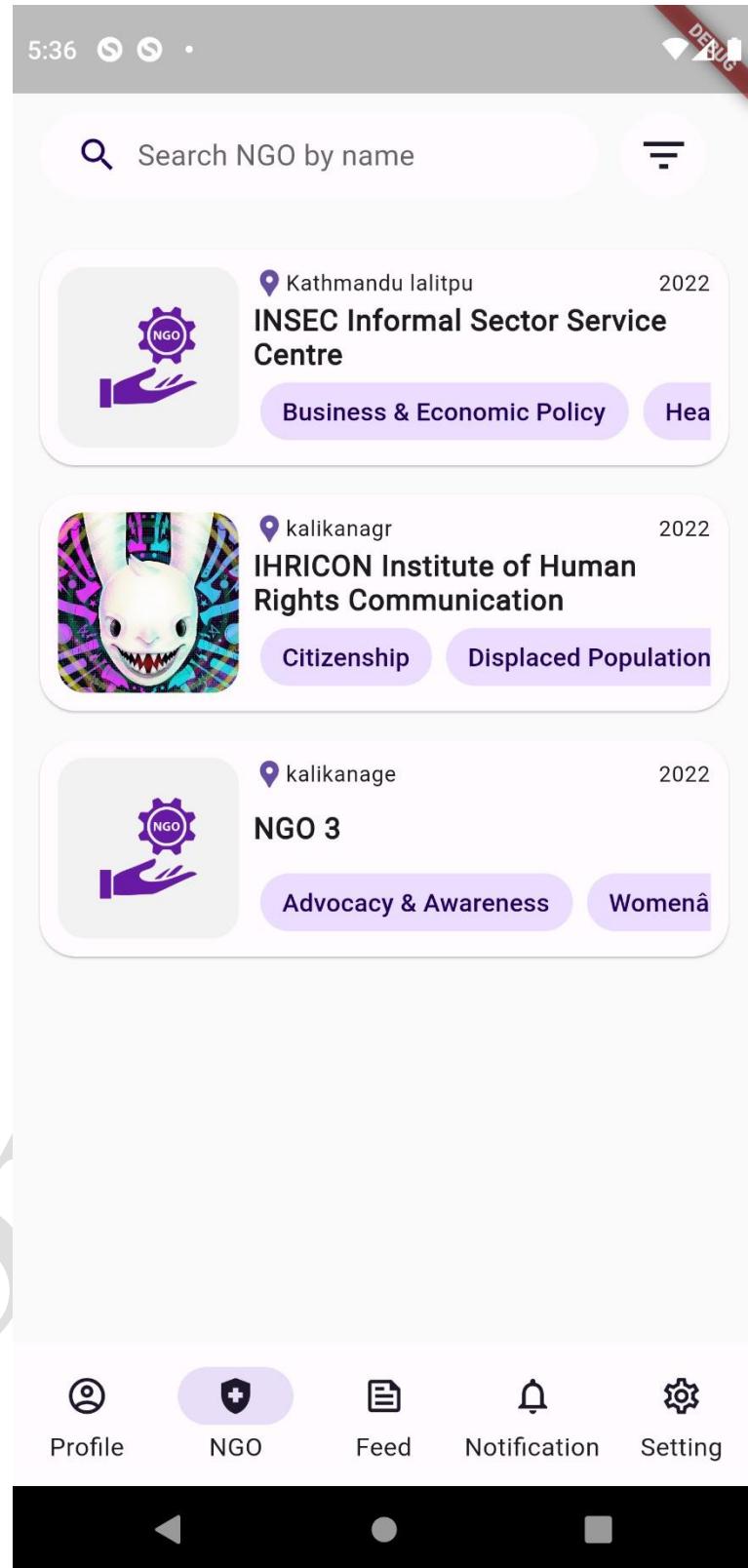


Figure 332: Mobile, View NGOs



Figure 333: Mobile, NGO Profile

7.11.1.2.4. TEST FILTER NGO BY FIELD OF WORK

Test Case	TC-5
Objective	To test NGOs filtration by field of work
Action	From the NGOs page, click the ‘filter’ icon button. Select field of work(s) from the modal sheet and then hit the ‘Apply’ button.
Expected Result	Users must see filtrated list of NGOs as per the selected field of work(s).
Actual Result	The user sees the filtrated list of NGOs.
Conclusion	Test Successful

Table 76: System Testing, Sasae Mobile App TC-5

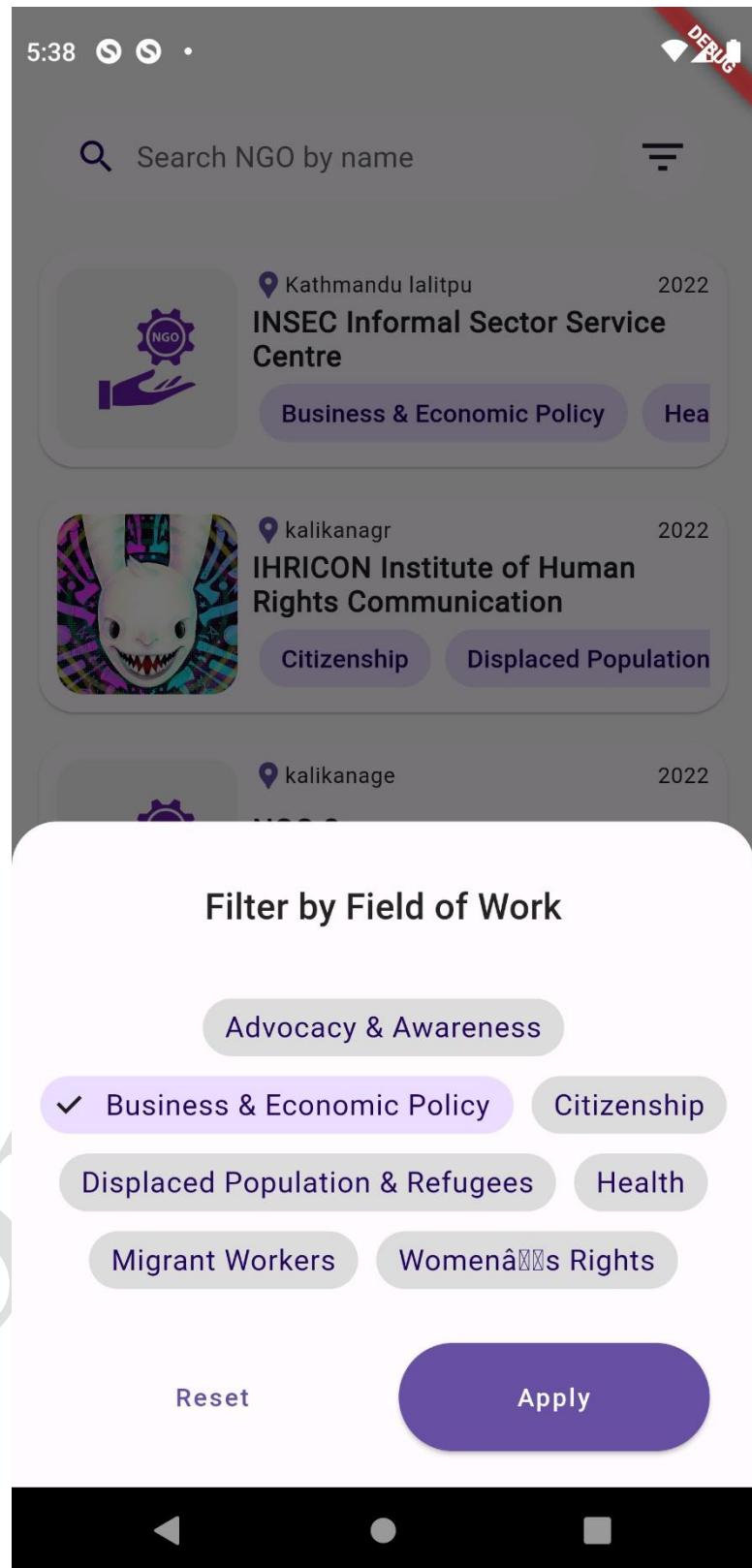


Figure 334: Mobile, NGOs Filtration by Field of Work (Before)

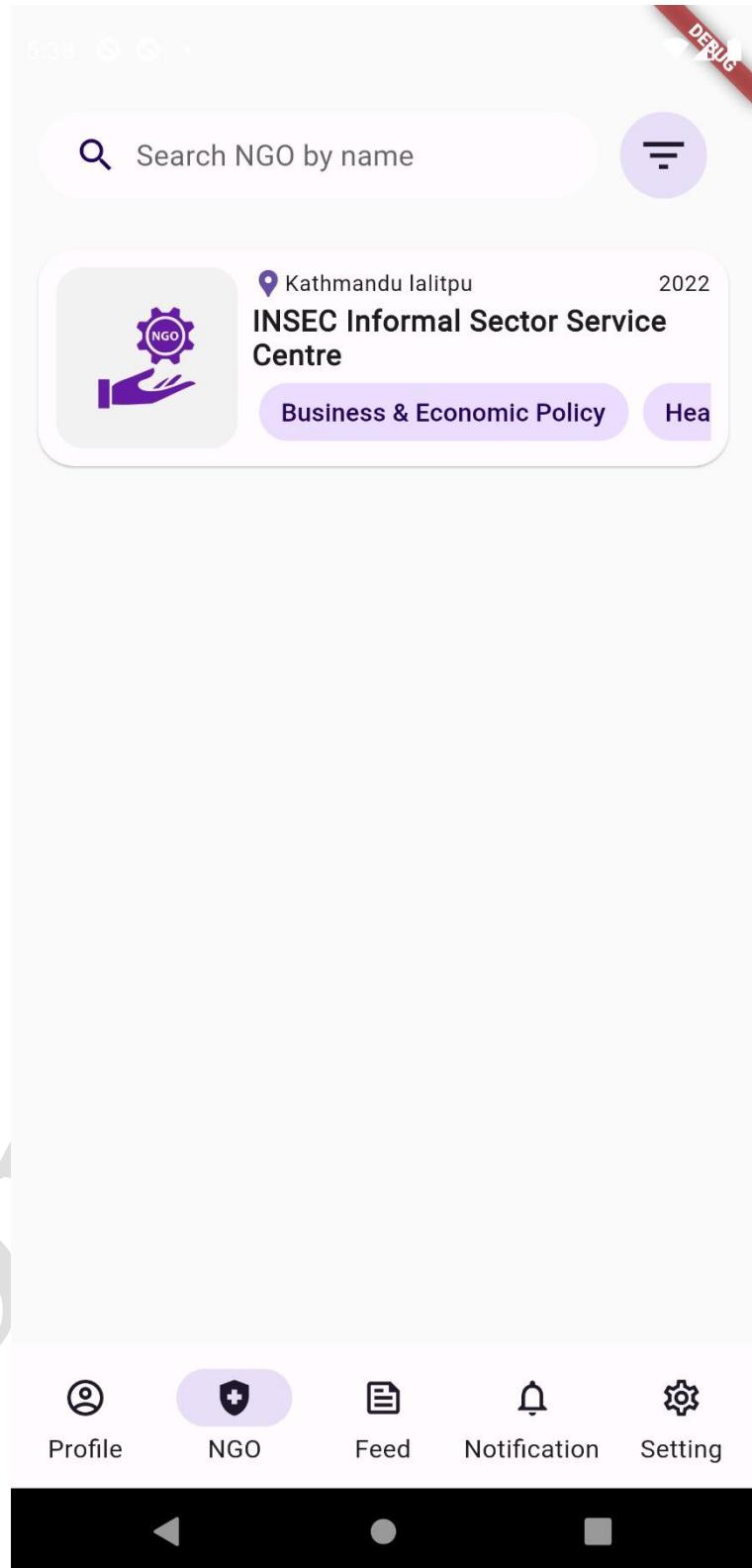
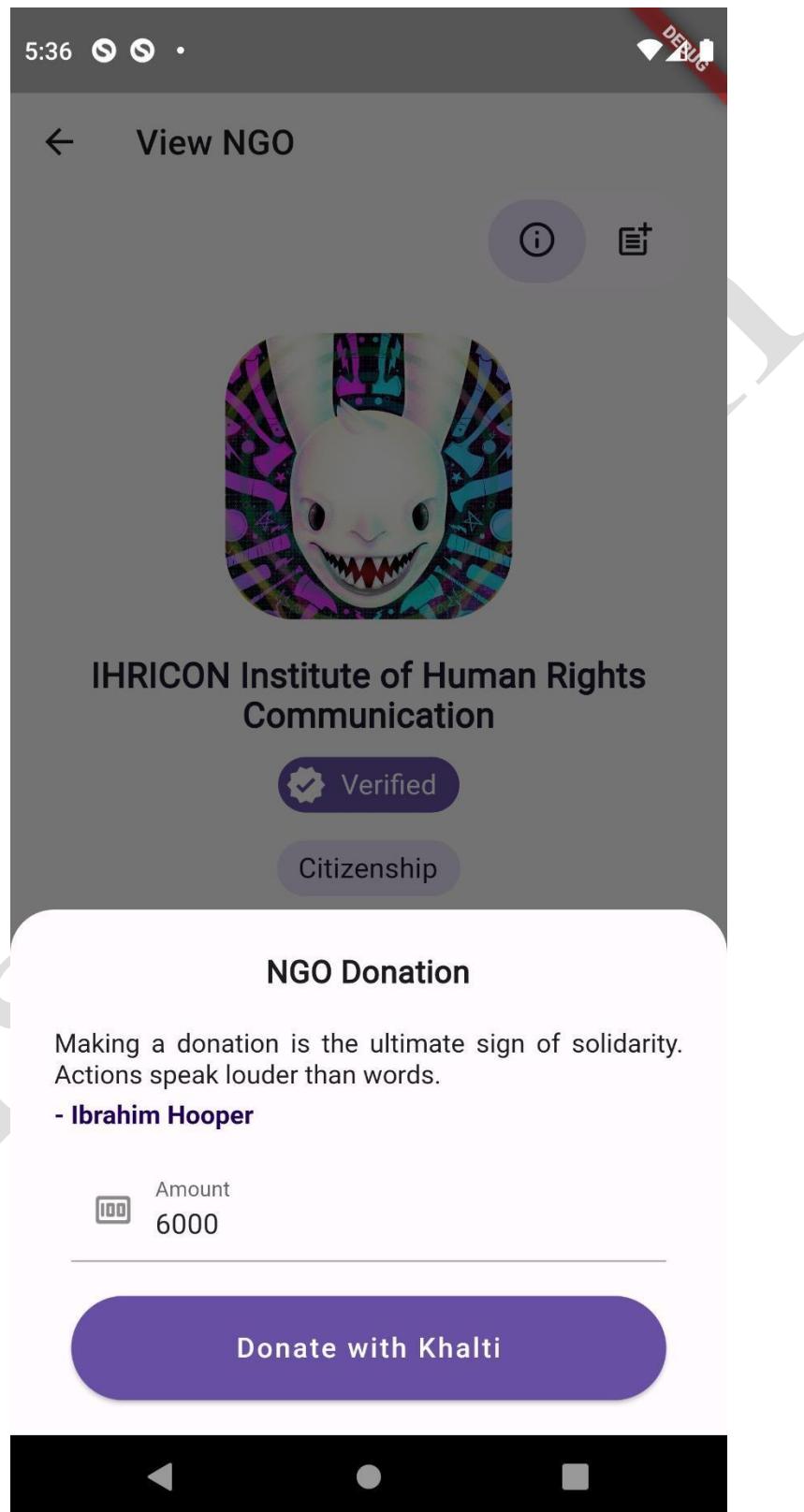


Figure 335: Mobile, Updated NGOs List (After)

7.11.1.2.5. TEST DONATION

Test Case	TC-6
Objective	To test donation to an NGO
Action	From a verified NGO profile, click on the ‘Donate’ button. Enter the amount of donation in the text field of the modal sheet. Then click the ‘Donate with Khalti’ button to navigate to the payment screen. Enter the Khalti account credential to proceed with the donation.
Expected Result	Users must be able to donate via the Khalti payment screen.
Actual Result	User successfully donates to NGO.
Conclusion	Test Successful

Table 77: System Testing, Sasae Mobile App TC-6



Figure

336: Mobile, NGO Donation with filled Amount (Before)

12:28 ◻ ◻ •

DEBUG

← Choose your payment method



KHALTI



CONNECT IPS



MOBILE BANKING



Khalti Mobile Number



0000000005

Khalti MPIN

*** *****

Amount

Rs. 6,000

PAY

Forgot Khalti MPIN?

RESET KHALTI MPIN

TEST



336

Figure 337: Mobile, Khalti Payment Screen



Figure

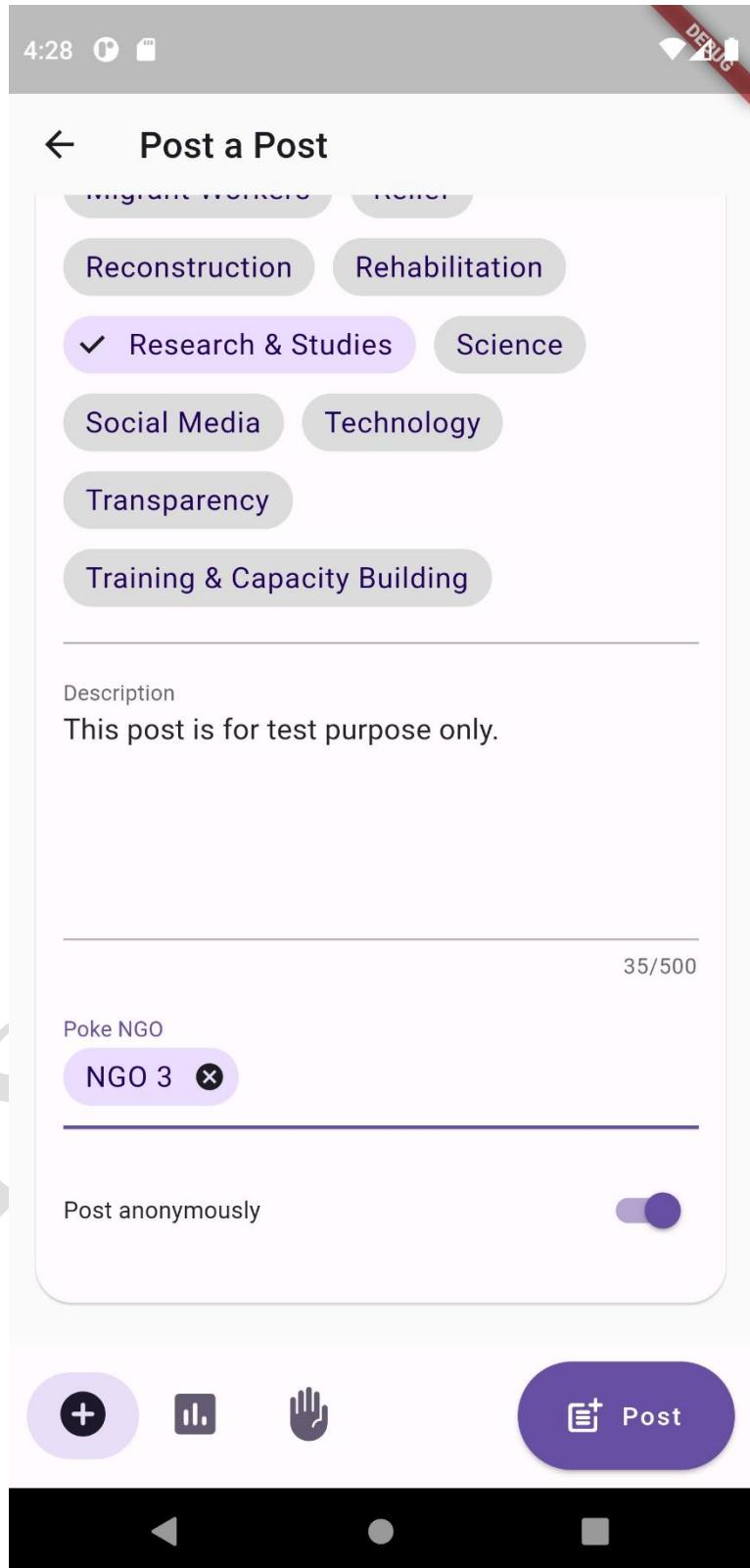


338: Mobile, NGO Profile with Success Message (After)

7.11.1.2.6. TEST MANAGE POST

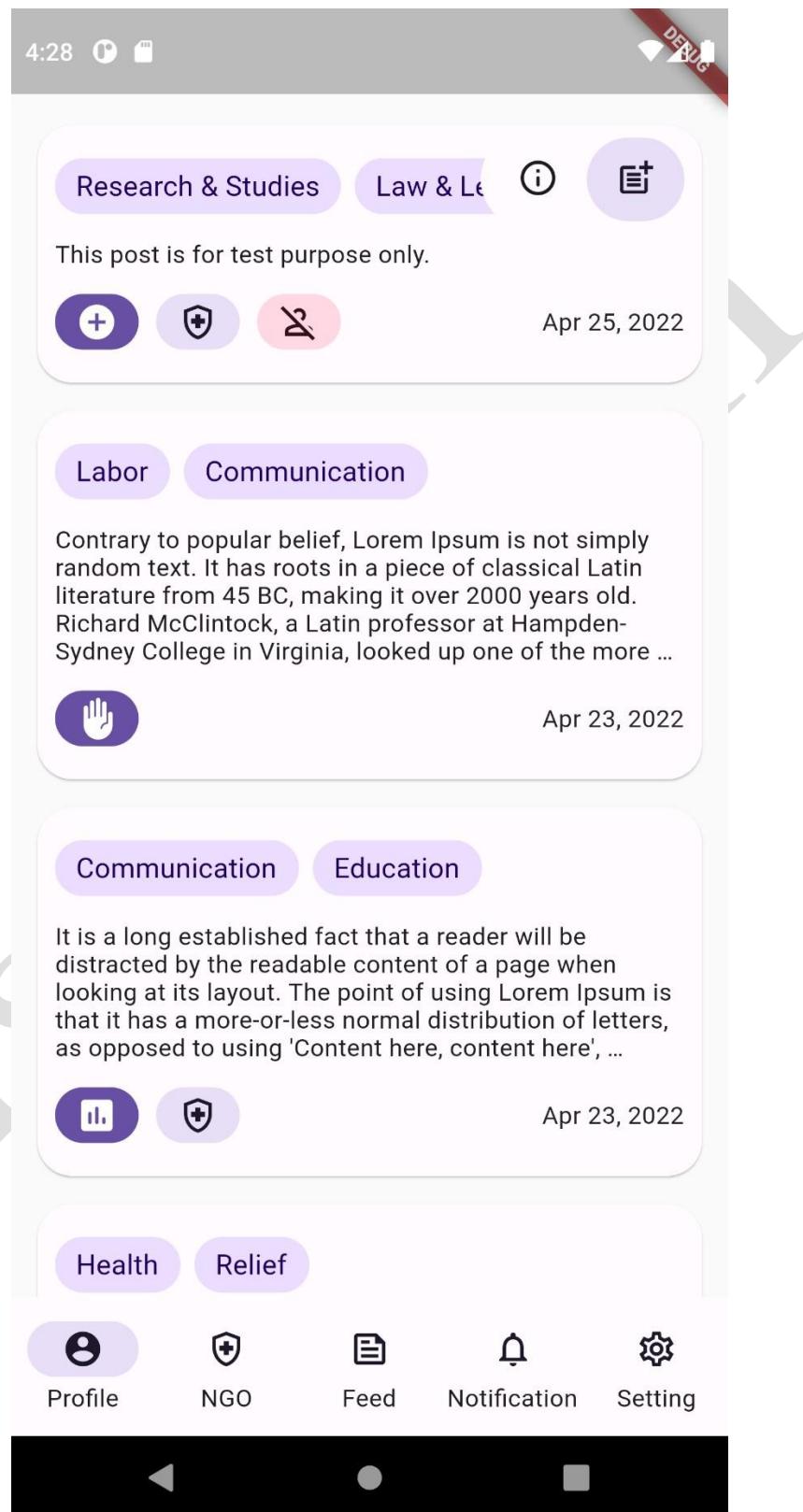
Test Case	TC-7
Objective	To test Post management (CRUD)
Action	<ul style="list-style-type: none"> - From the Feed page, click on ‘Post’ to get navigated to the Post Form Screen. Choose the Post Type and fill all the required fields. Hit the ‘Post’ button to Post. - Click the user post tab located at the top-right button of the profile page. Long press the desired Post to pop a modal sheet with edit and delete options. - Click Edit to navigate to the Post update form. Fill all the changes required fields with valid data and then hit the ‘Done’ button to update the post. - Click the Delete button to prompt a confirmation dialogue. Click ‘OK’ to delete the post.
Expected Result	Users must be able to perform CRUD operations on General People resources.
Actual Result	The user was able to perform all CRUD operations.
Conclusion	Test Successful

*Table 78: System Testing, Sasae Mobile App TC-7**Figure*



339: Mobile, Create Post Screen with filled Data (Before)

SASAE



Figure

Figure 340: Mobile, User Posts Tab (After)



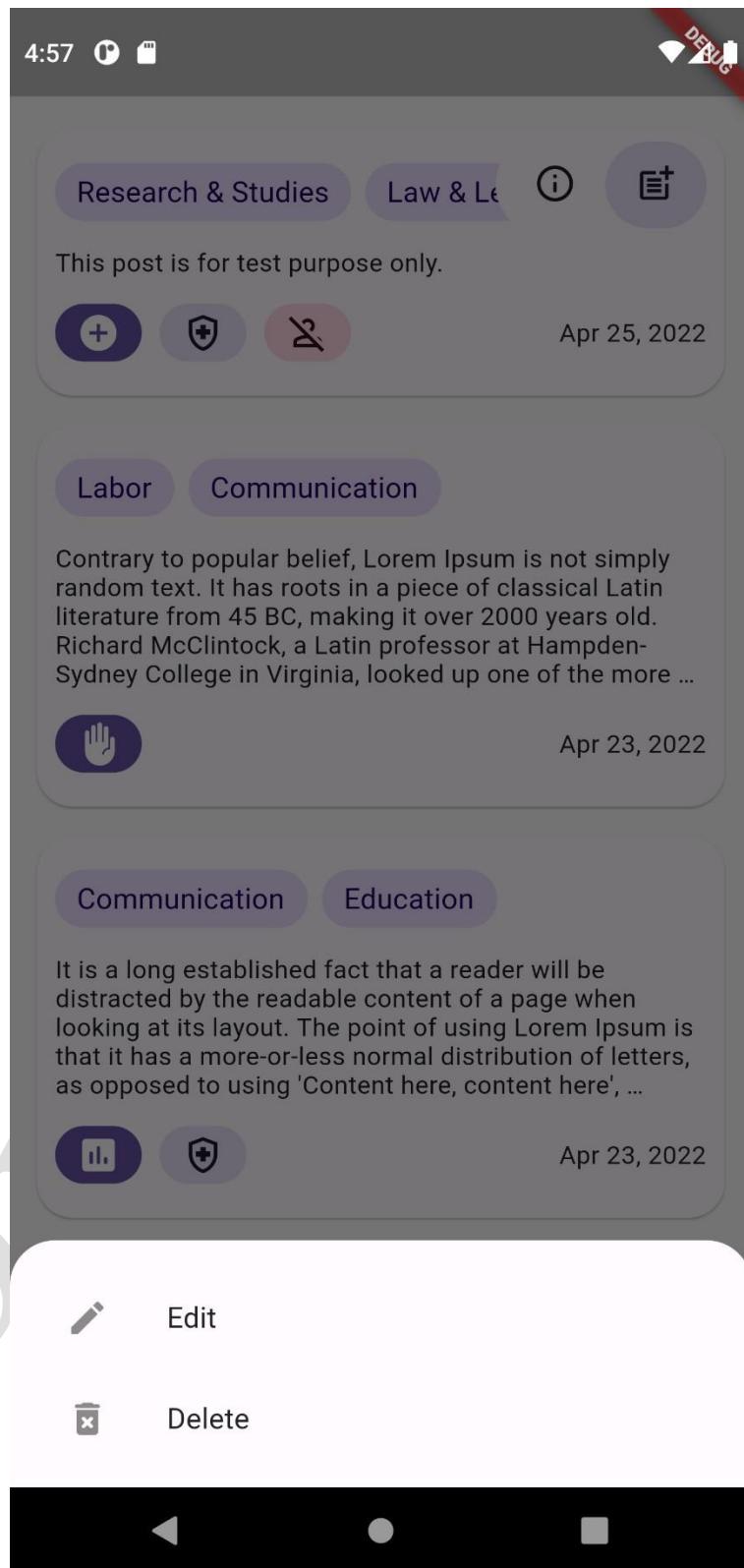


Figure 341: Mobile, Edit-Delete Post Modal Sheet

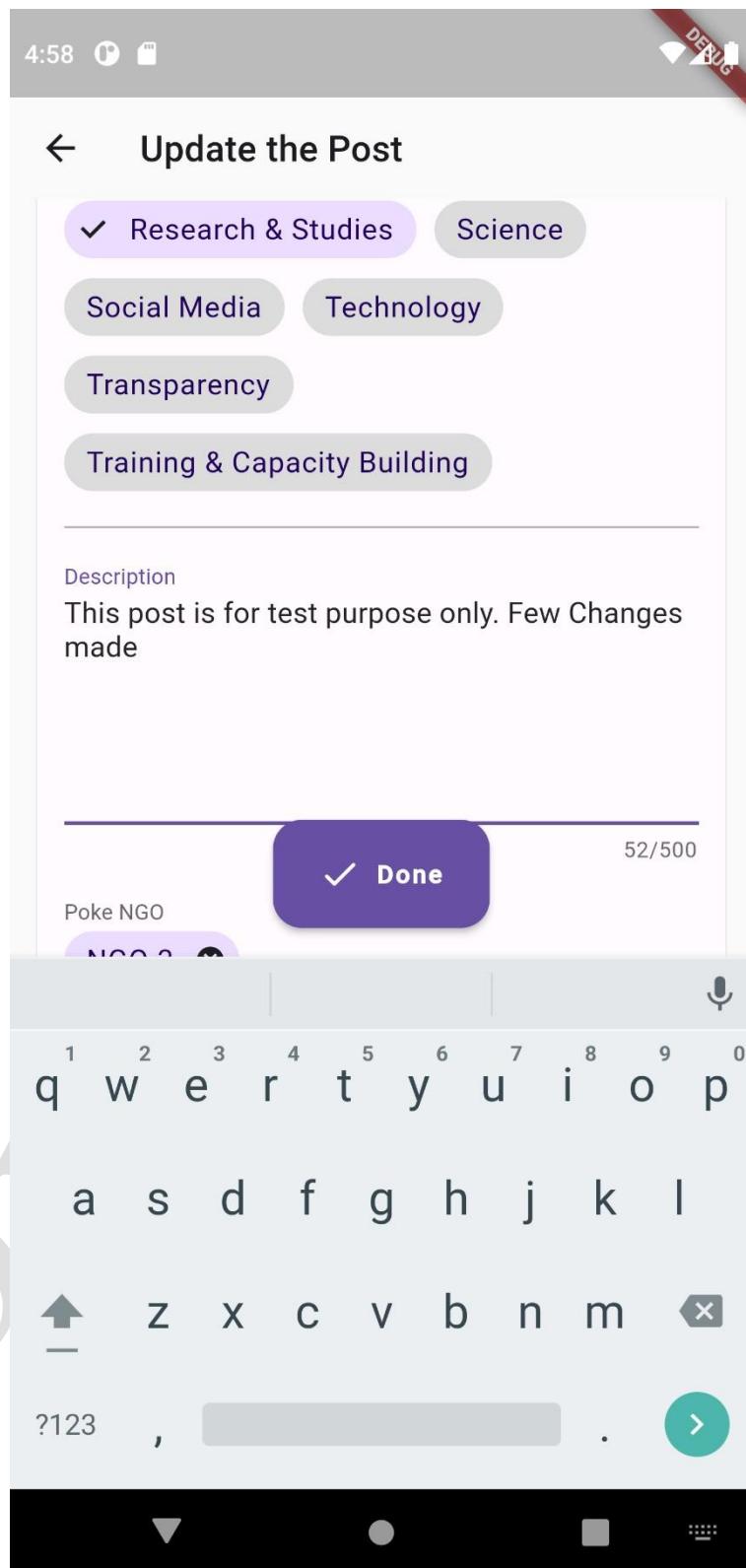


Figure 342: Mobile, Post Update Form Screen with changed Data (Before)

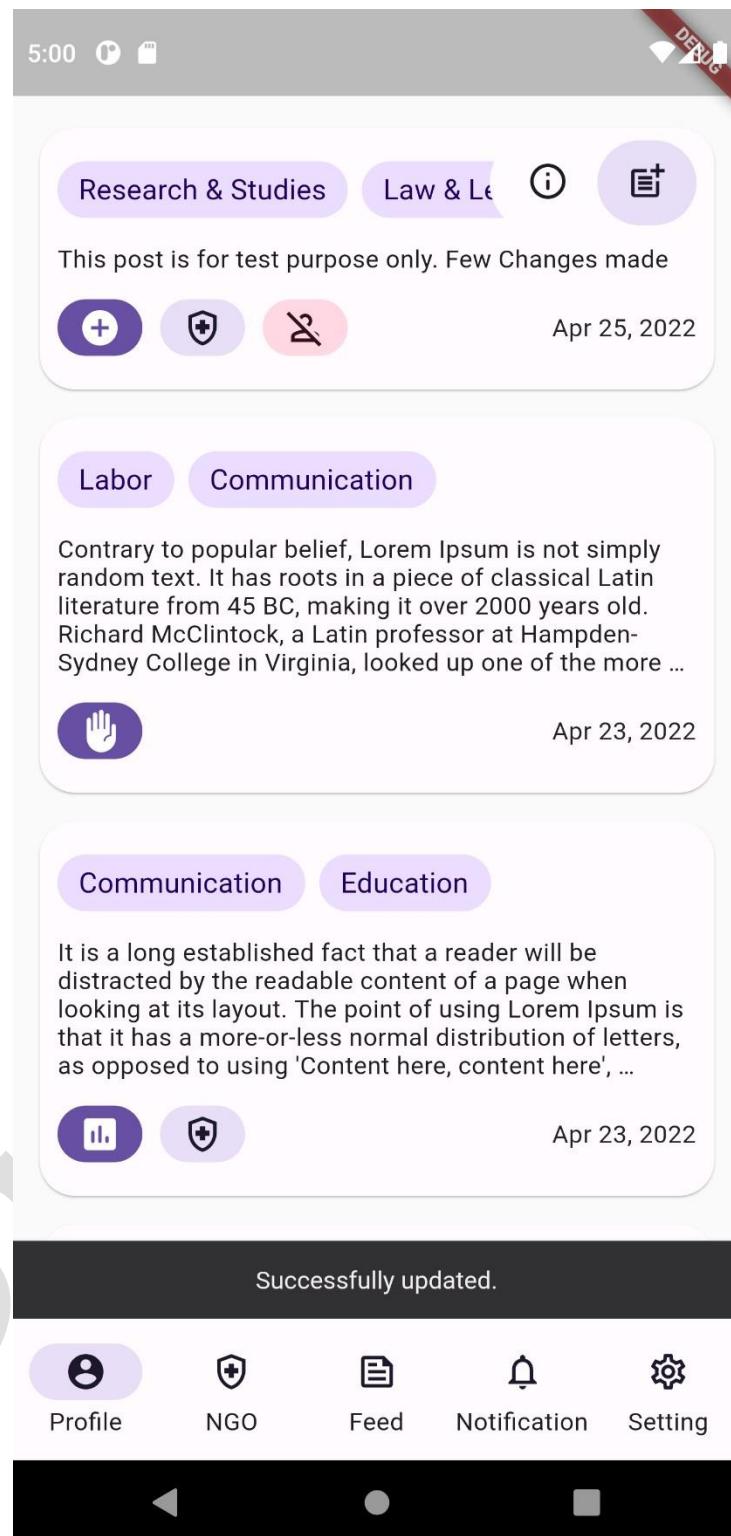


Figure 343: Mobile, User's Posts Tab (After)

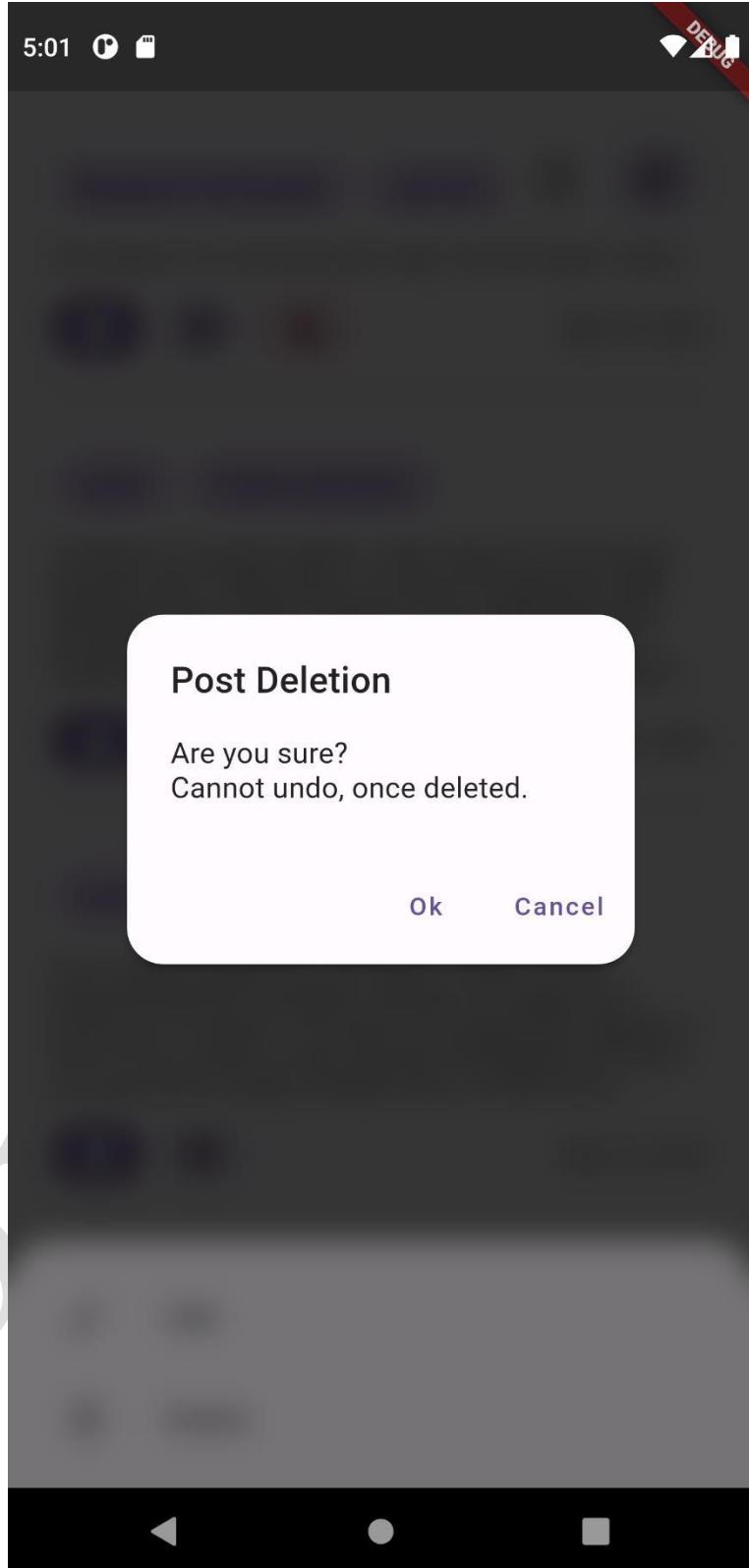


Figure 344: Mobile Post Delete Dialog (Before)

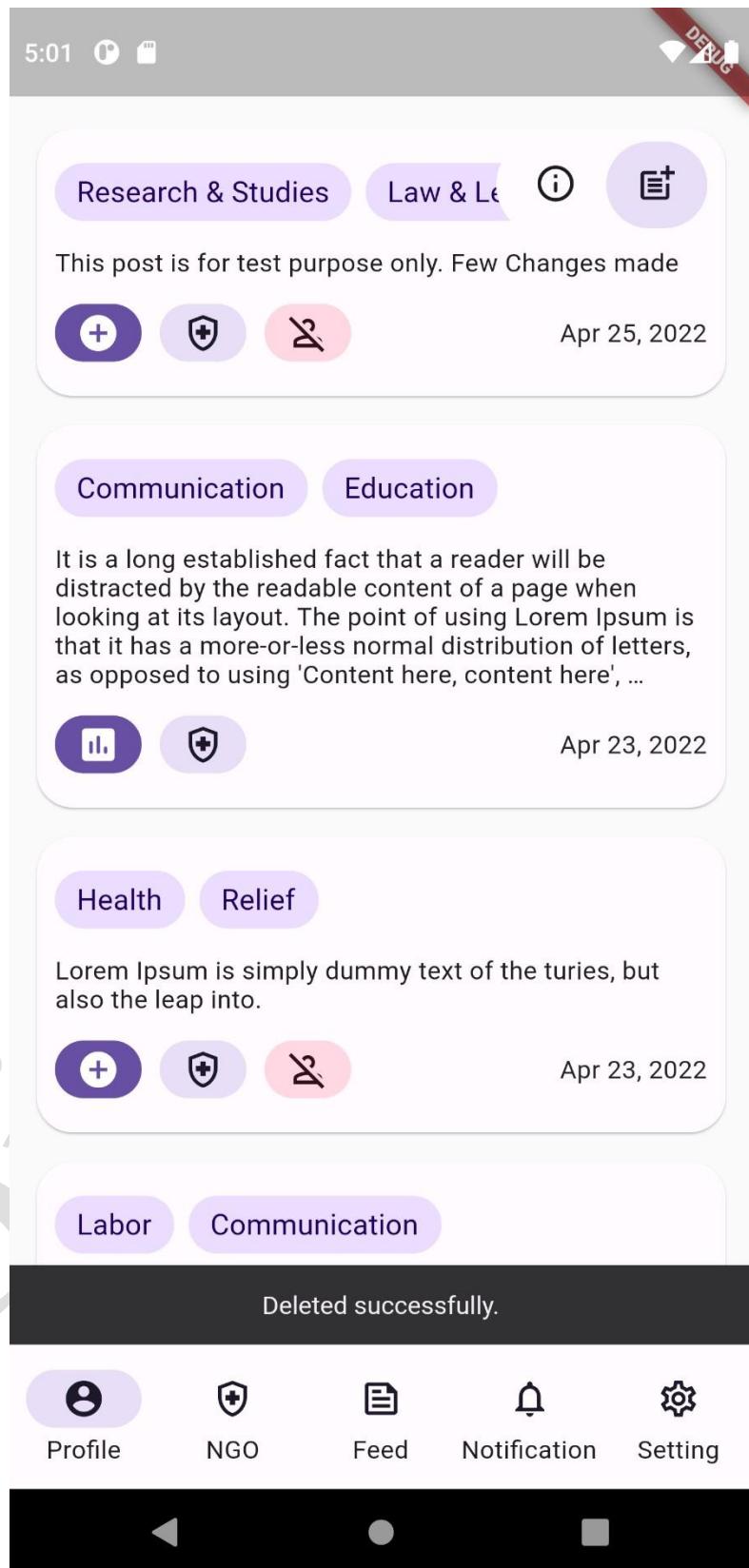


Figure 345: Mobile, User's Post Tab with Success Message (After)

7.11.1.2.7. TEST VIEW POST(S)

Test Case	TC-8
Objective	To test view feature on Post(s)
Action	The Home screen or Feed page shows a list of Posts. Click on any post to view it in detail.
Expected Result	Users must be able to view a list of Posts or an individual Post.
Actual Result	The user did view Post(s).
Conclusion	Test Successful

Table 79: System Testing, Sasae Mobile App TC-8

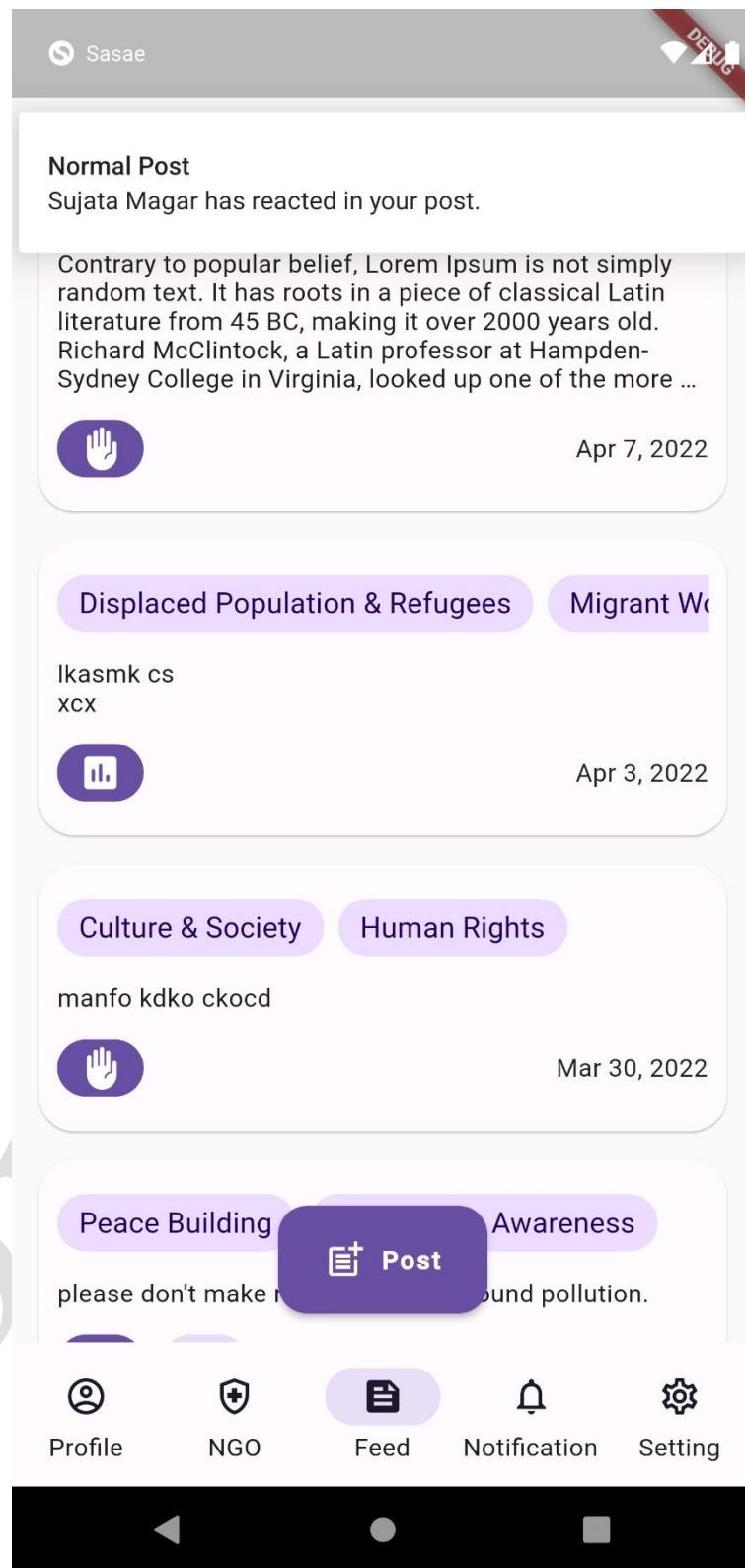


Figure 346: Mobile, Posts Page

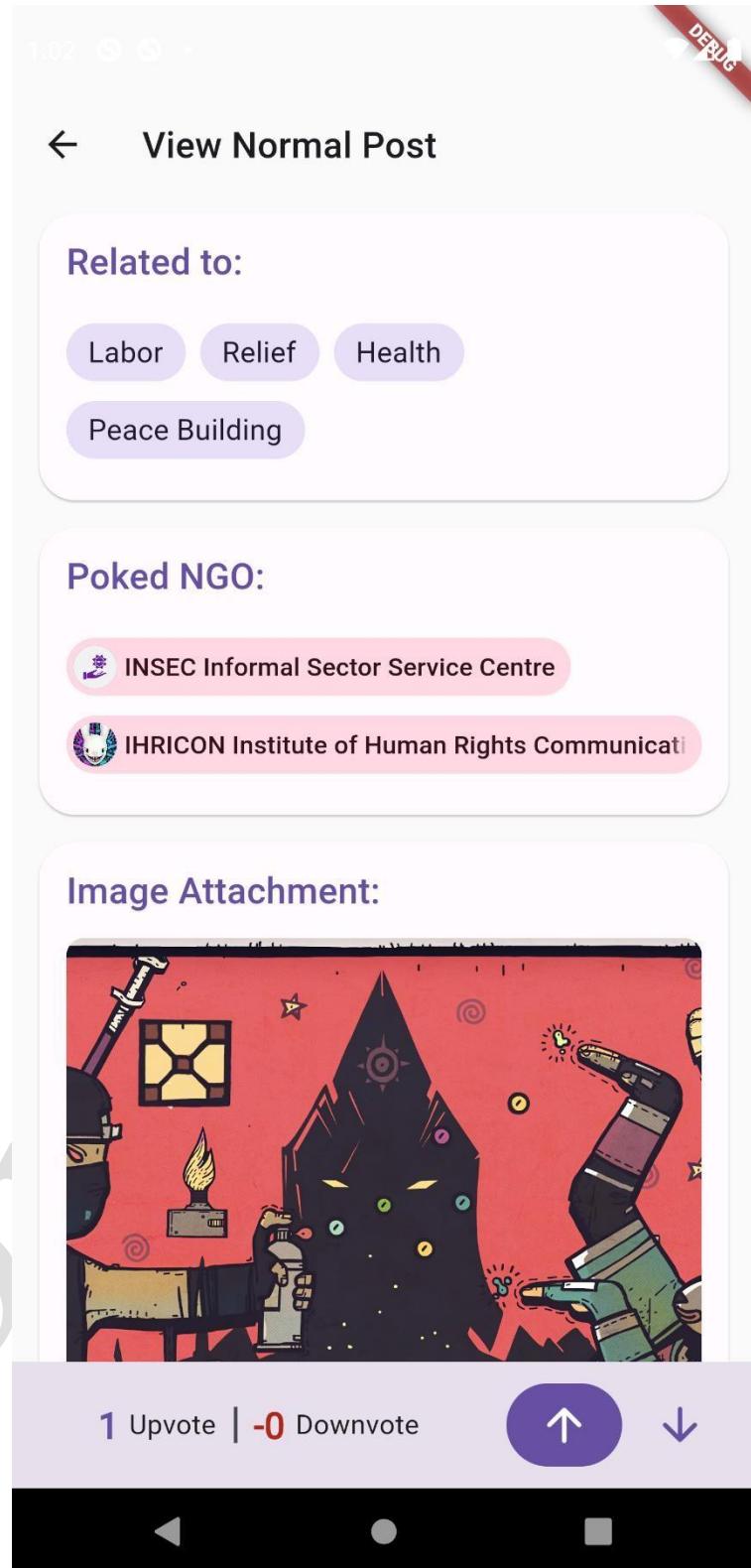


Figure 347: Mobile, Normal Post Screen

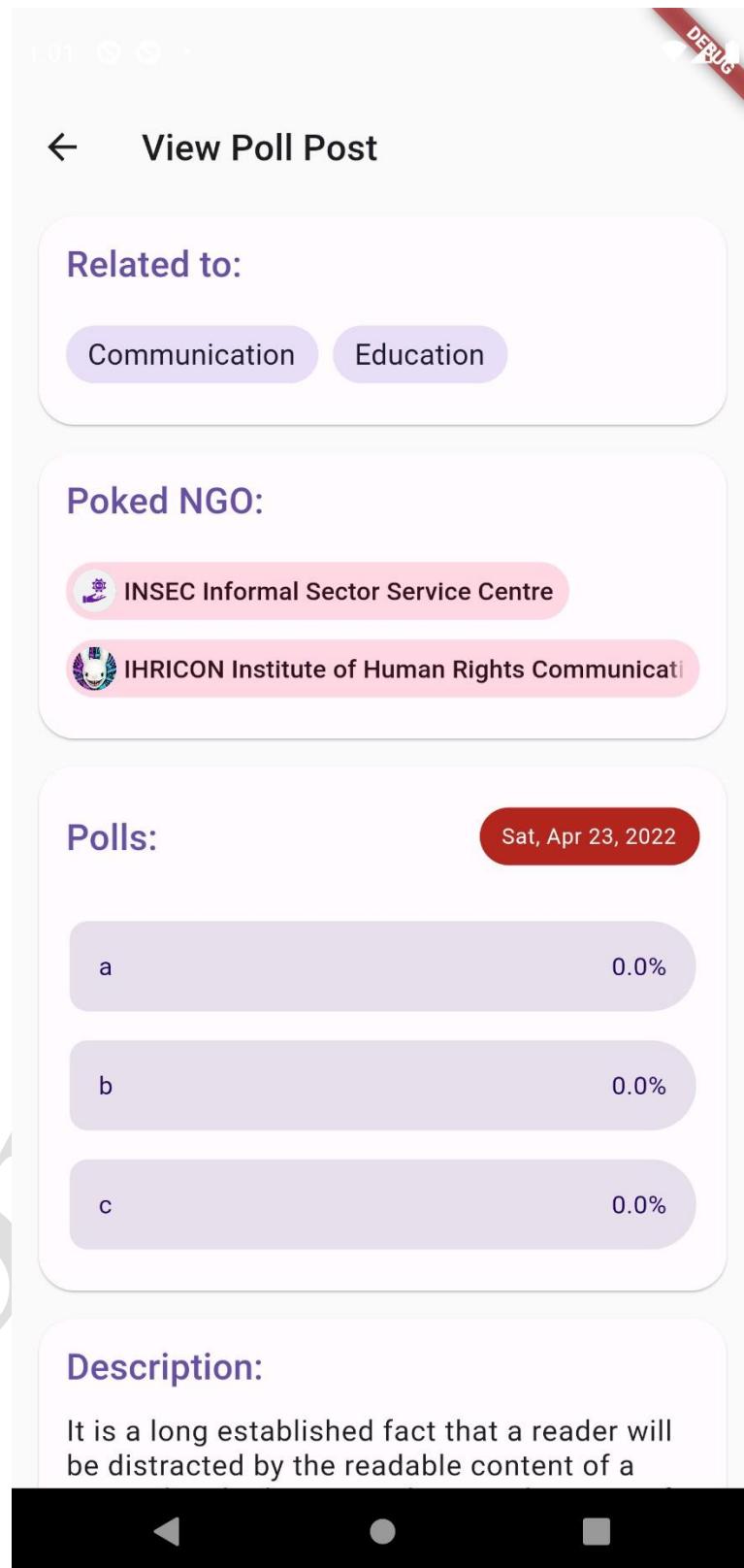


Figure 348: Mobile, Poll Post Screen

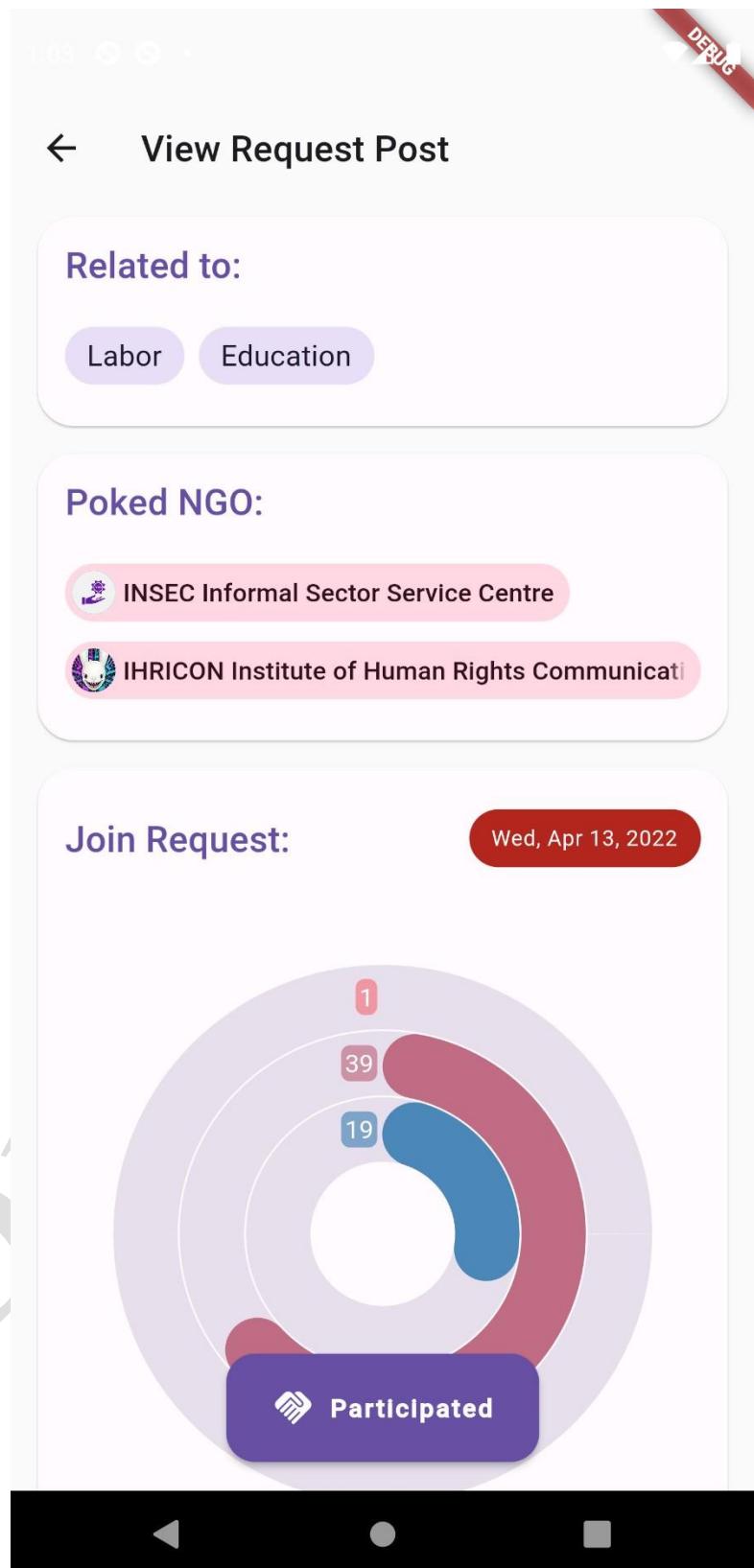


Figure 349: Mobile, Request Post Screen

7.11.1.2.8. TEST REACT POST

Test Case	TC-9
Objective	To test Post reaction regardless of the type
Action	<p>From the Feed page, select distinct Posts by type and react to them.</p> <ul style="list-style-type: none"> - Upvote/downvote on Normal Post. - Poll on Poll Post - Sign/Participate on Request Post of petition or join type respectively.
Expected Result	Users must be able to react to each different type of posts
Actual Result	Users reacted to different Posts.
Conclusion	Test Successful

Table 80: System Testing, Sasae Mobile App TC-9

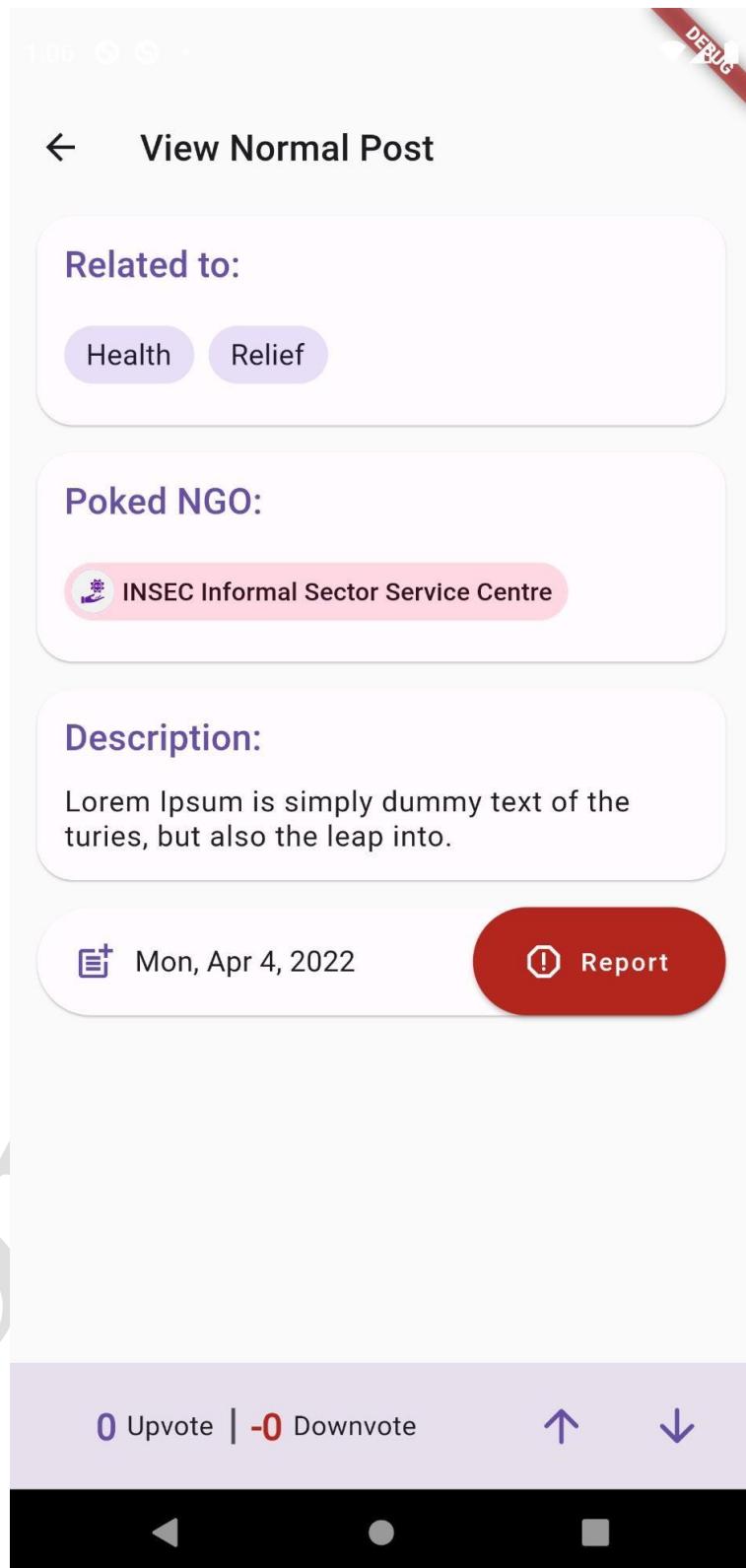


Figure 350: Mobile, Normal Post Screen (Before)

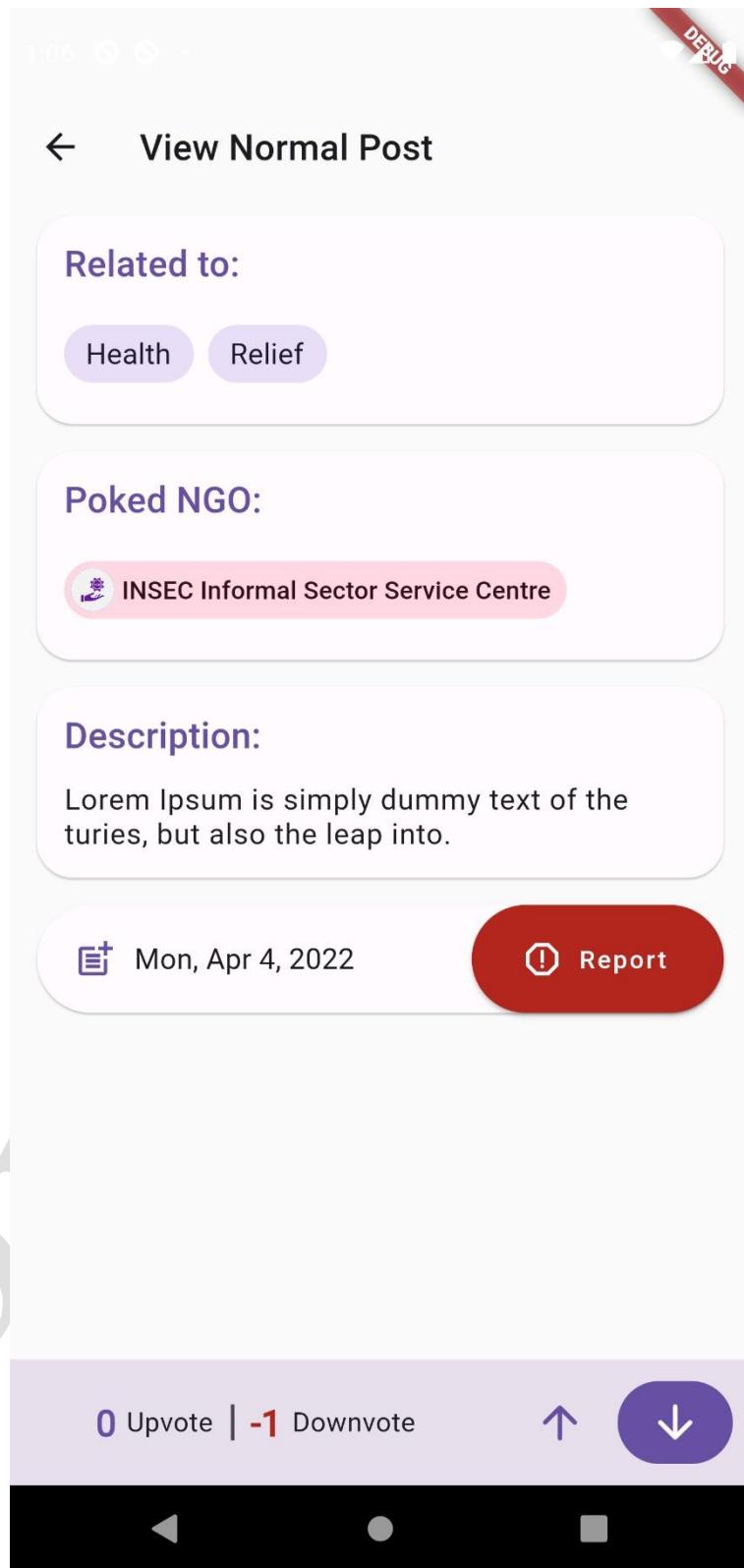


Figure 351: Mobile, Normal Post Screen (After)

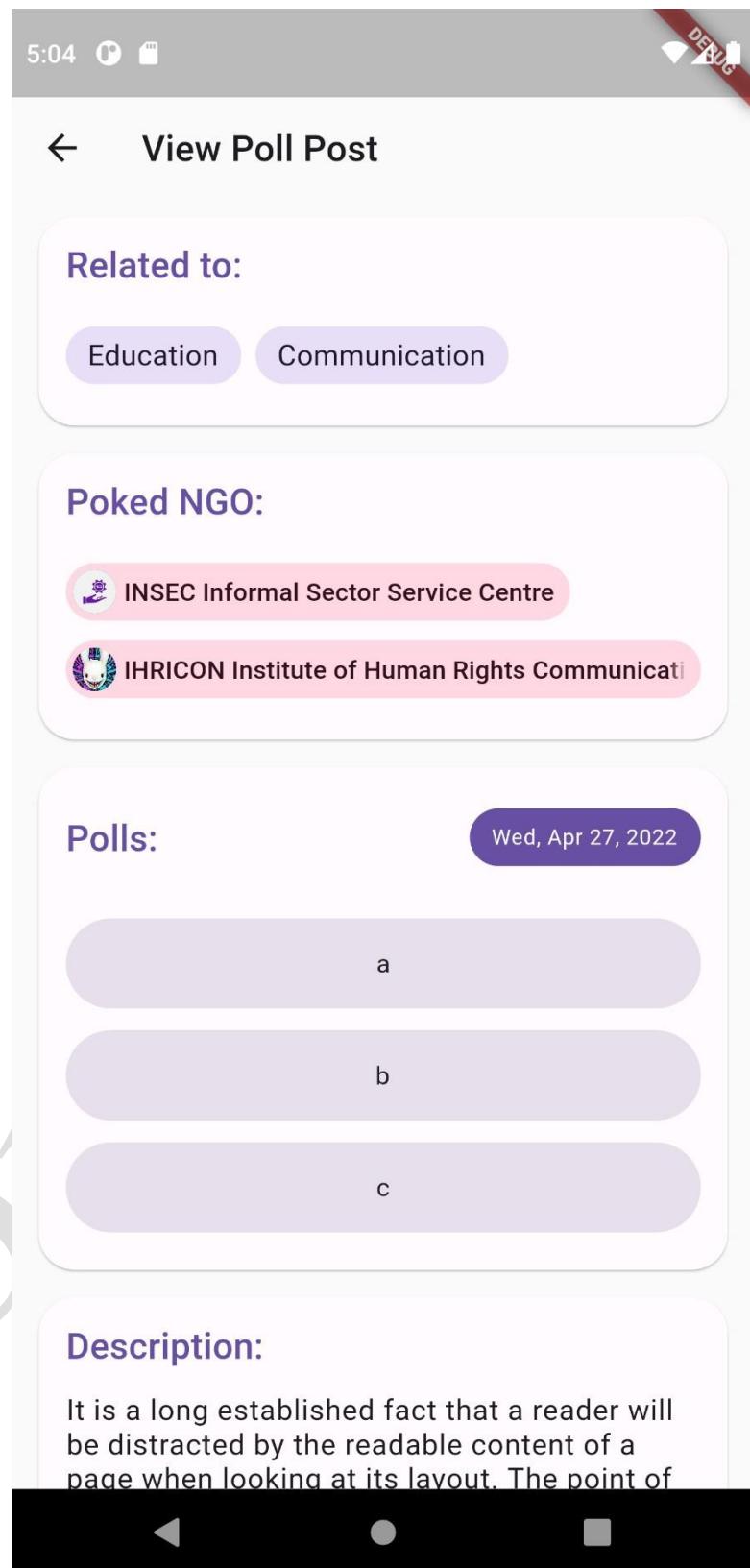


Figure 352: Mobile, Poll Post Screen (Before)

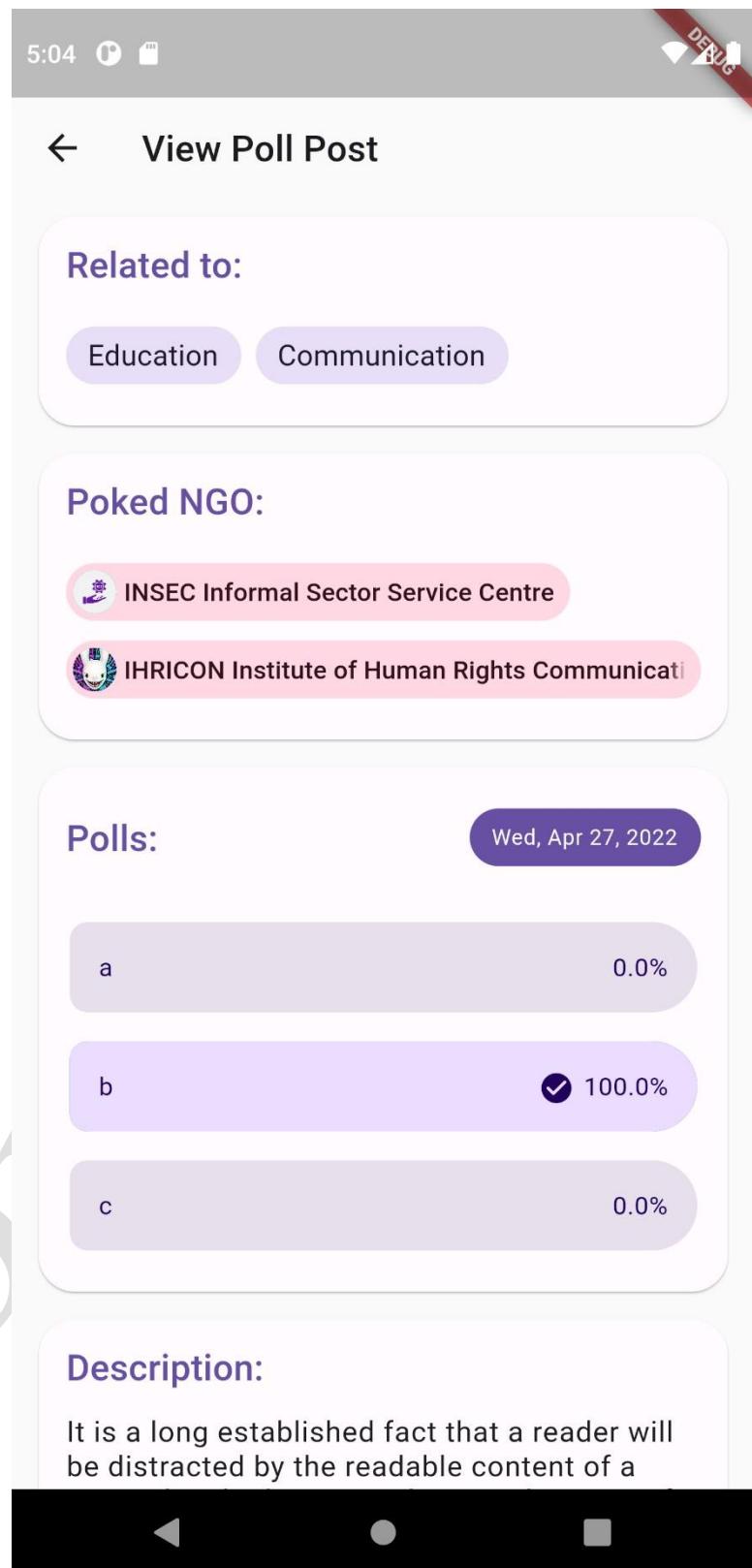


Figure 353: Mobile, Poll Post Screen (After)

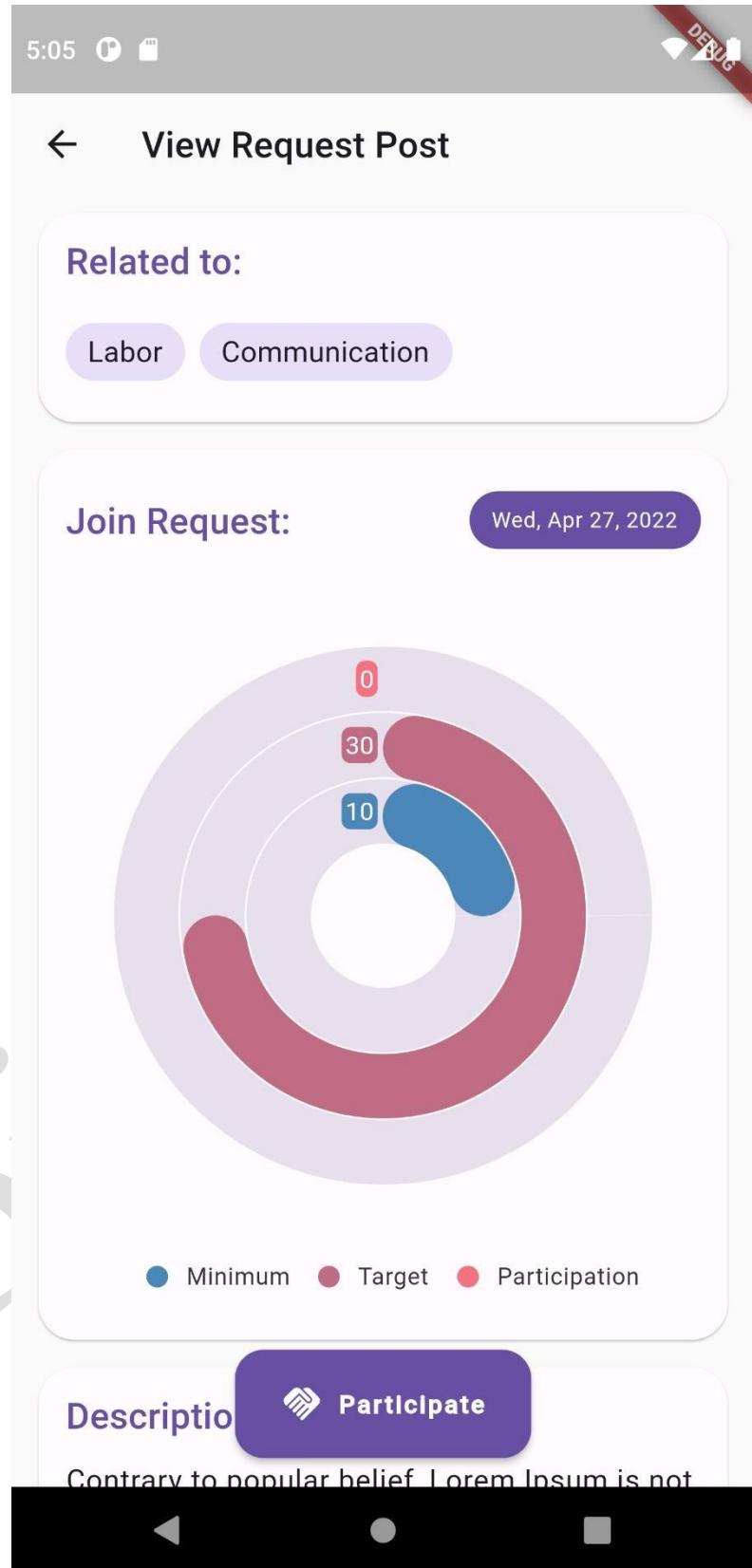


Figure 354: Mobile, Request Post Screen (Before)

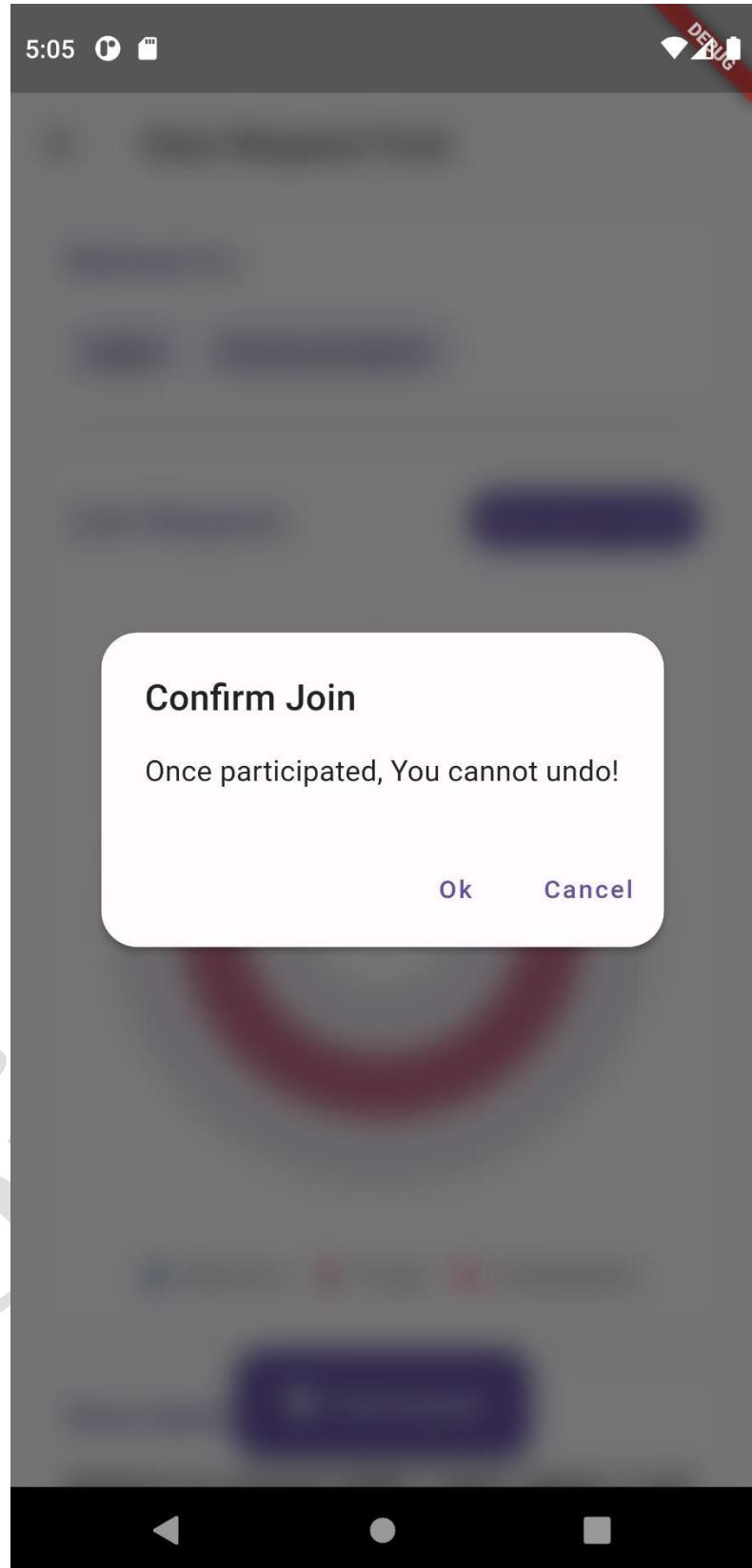


Figure 355:Mobile, Request Post Participate Prompt

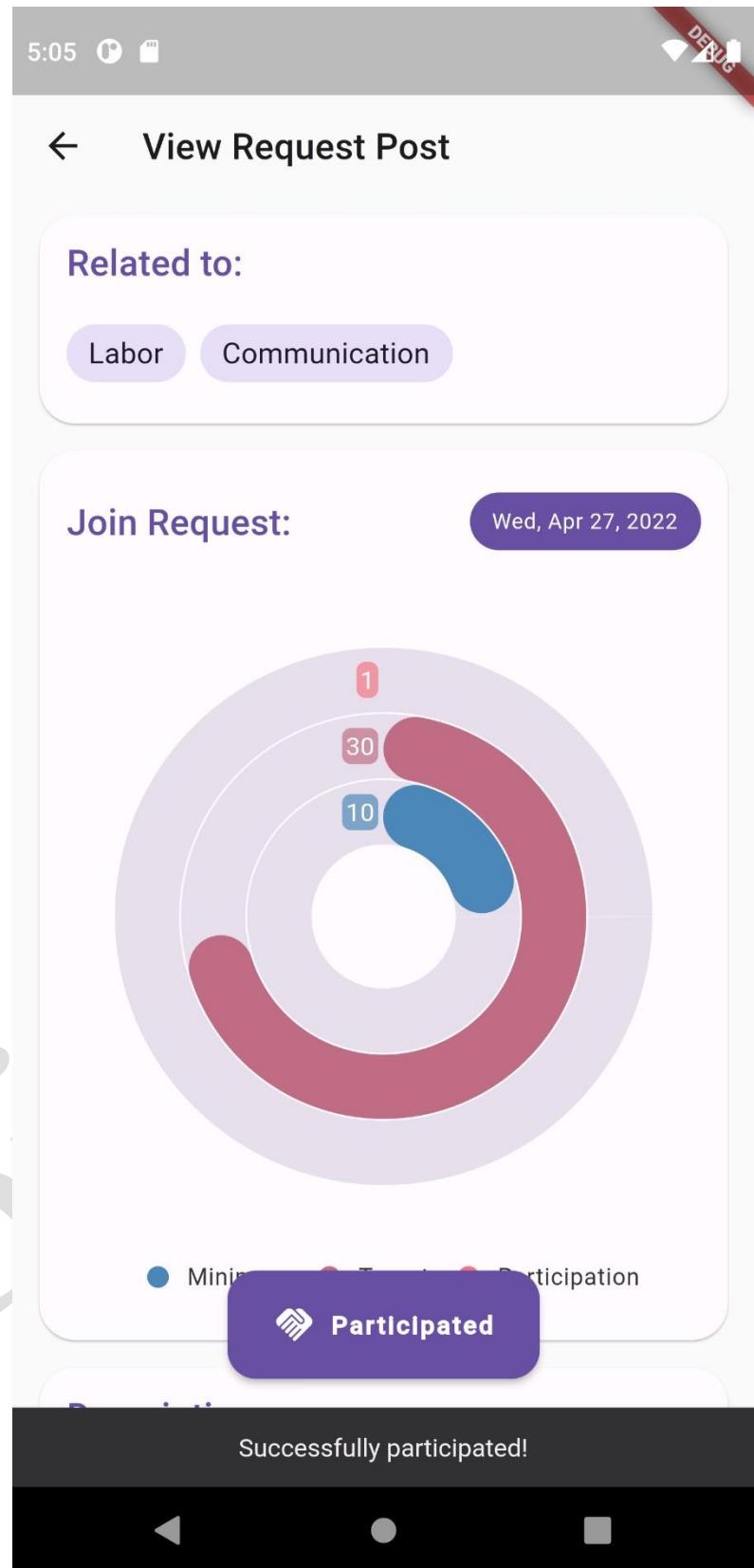


Figure 356: Mobile, Request Post Screen (After)

7.11.1.2.9. TEST POST-REACTION NOTIFICATION

Test Case	TC-10
Objective	To test notification on Post reaction.
Action	From the Feed page, Click a Post of a different User (i.e., NGO, General People). React to the Post after the Post detail is fully loaded.
Expected Result	The Post author must be notified of the Post reaction.
Actual Result	Post author was notified.
Conclusion	Test Successful.

Table 81: System Testing, Sasae Mobile App TC-10

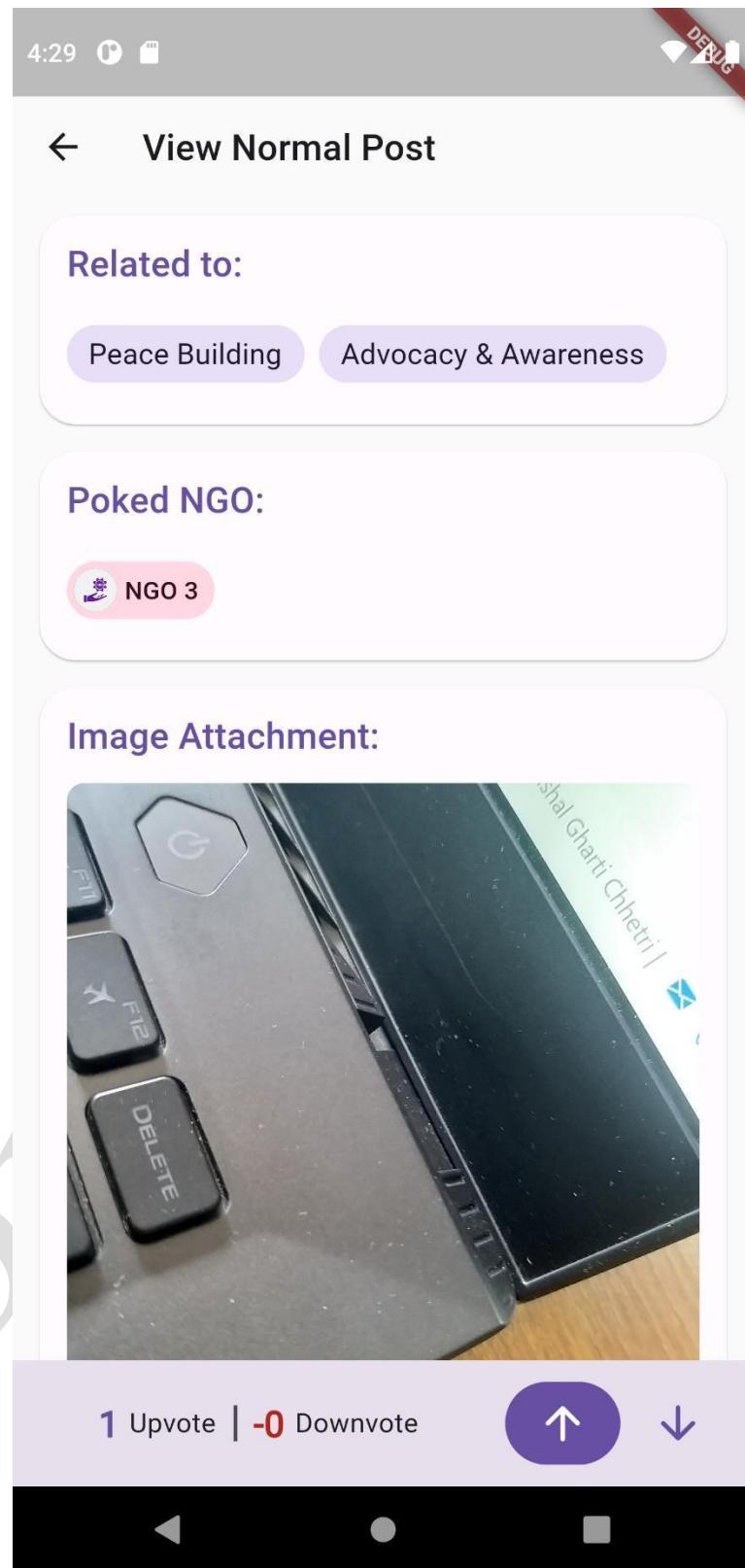


Figure 357: Mobile, Normal Post Upvoted (Before)

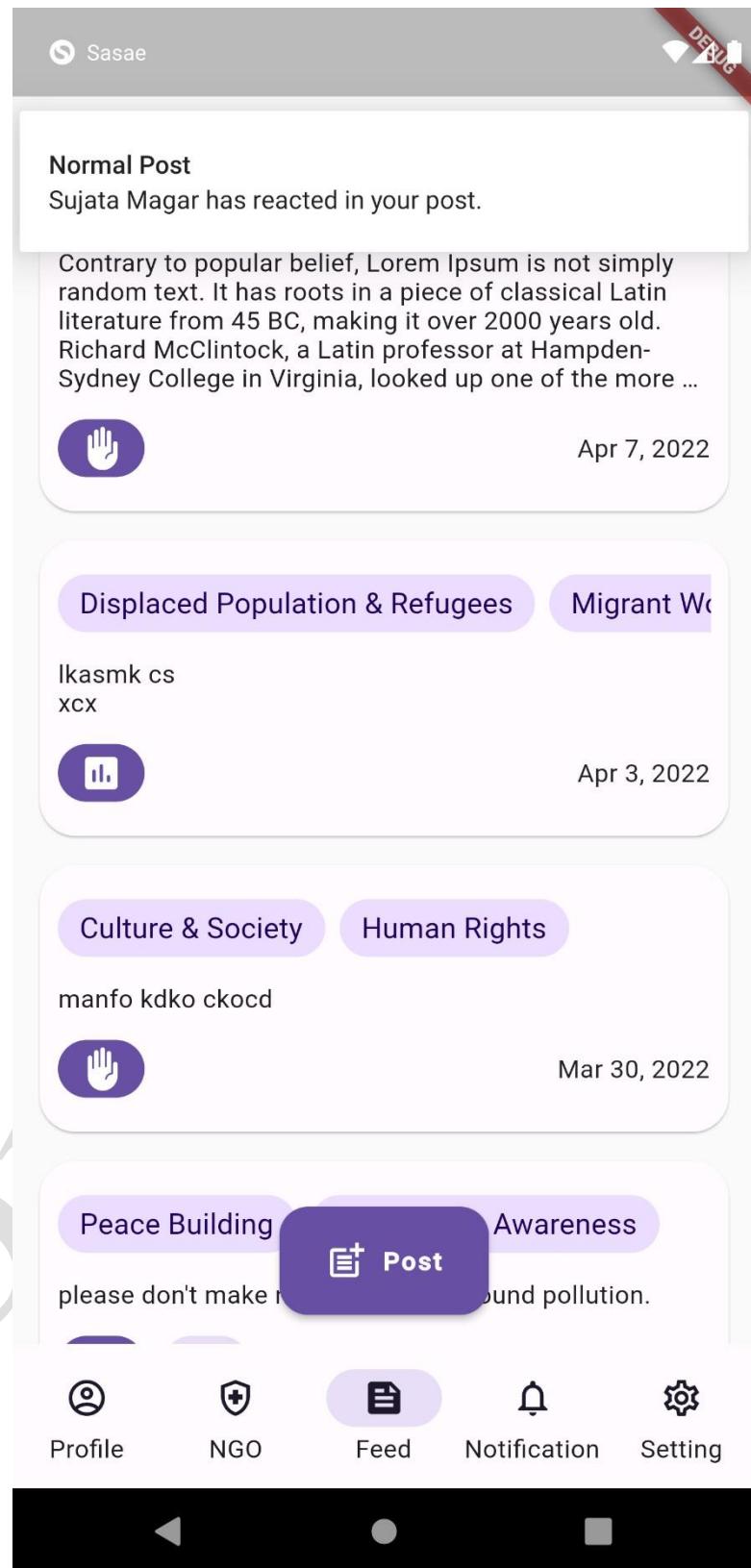


Figure 358: Mobile, Post Reaction Push Notification. (After)

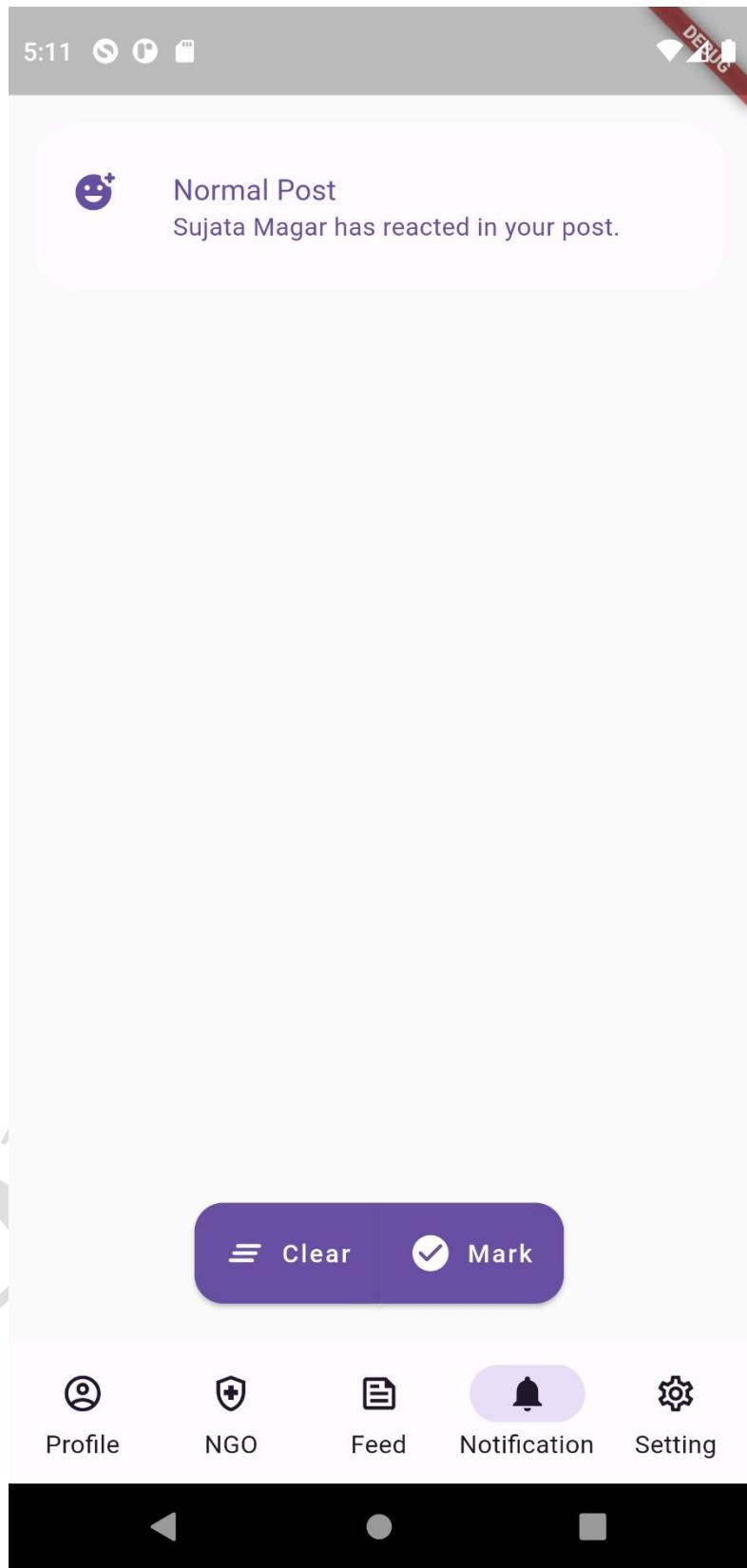


Figure 359: Mobile, Notification Page (After)

7.11.1.2.10. TEST POST REMOVE NOTIFICATION

Test Case	TC-11
Objective	To test notification on removed post
Action	On the Admin-staff portal with staff logged in, click on 'Reported Post'. On the Reported Post page, click any post to review. Fill the Review form with the action set to 'Post Remove'. Then click the 'Take Action' button.
Expected Result	The user must get a notification with a title and body containing the reason.
Actual Result	The user gets a notification.
Conclusion	Test Successful

Table 82: System Testing, Sasae Mobile App TC-11

The screenshot shows the 'Review Reported Posts' screen. At the top, there's a navigation bar with 'Hi staff1!', 'Home', 'Staff', 'NGO', 'People', 'Reported Post', 'Change Password', and 'Logout'. Below the navigation, a post is displayed with the following details:

- Posted by: **bishal**
- Date Posted: March 29, 2022, 9:35 p.m.
- Post Type: **Poll**

Under 'Post related to:', there are two categories: 'Education' and 'Communication'.

The 'Post Content' section contains a paragraph about Lorem Ipsum. Below it, a poll is shown with three options: 'a', 'b', and 'c'. Option 'c' has a blue progress bar indicating 100% reaction.

In the 'Poked to NGOs:' section, two NGOs are listed: 'n1-ngo1' and 'n2-ngo2'.

On the right side, a 'Let's Review!' panel is open. It includes a 'Reports' tab (with 1 report), a 'Reason' section containing placeholder text, an 'Action' dropdown set to 'Post Remove', and a large blue 'Take Action' button.

At the bottom left, there's a copyright notice: '©2022 Copyright: सासैए (Sasae)'. At the bottom right, there are social media icons for LinkedIn, Facebook, and Instagram.

Figure 360: Web, Review Reported Posts (Before)

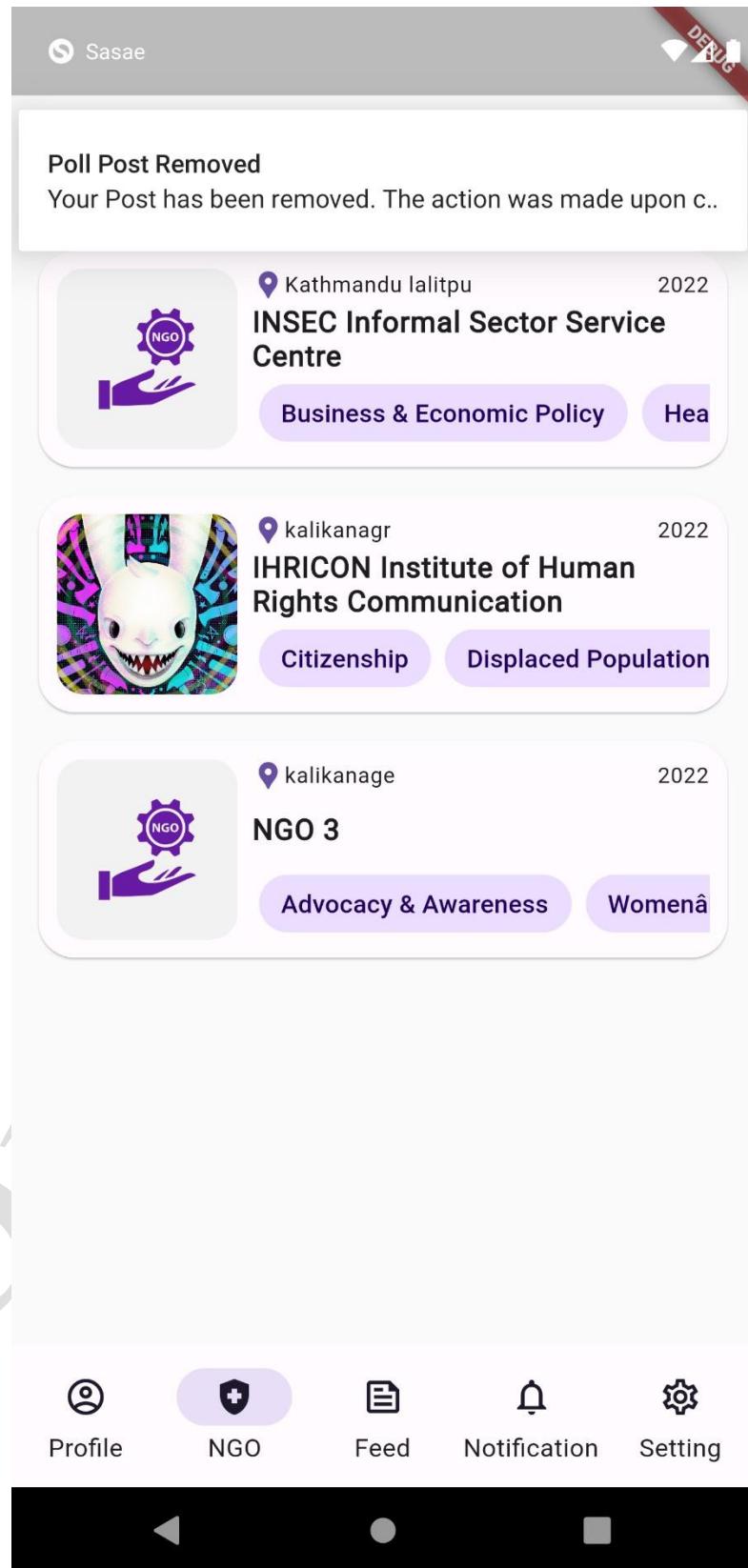


Figure 361: Mobile, Push Notification for Reported Post Remove (After)

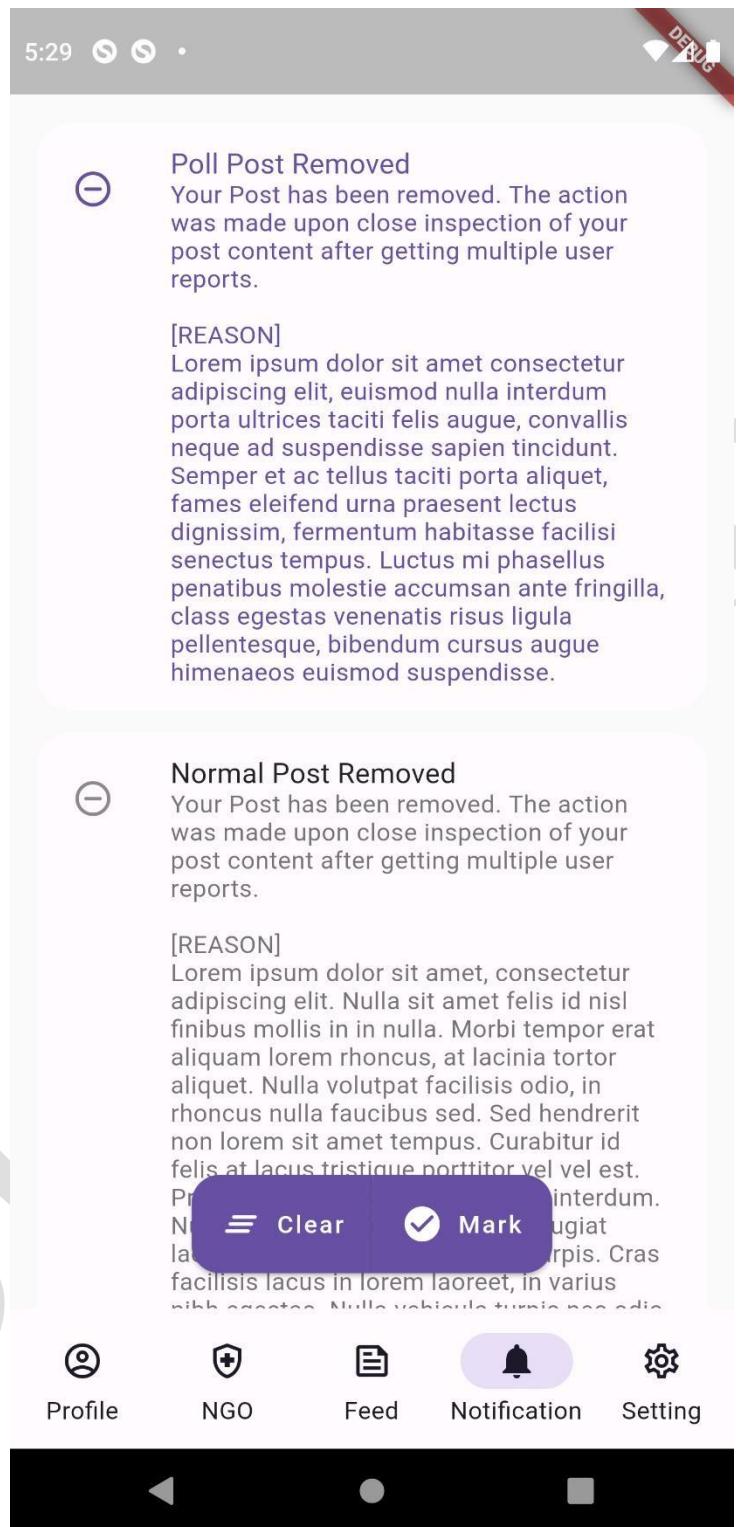


Figure 362: Mobile, Notification Page (After)

7.11.1.2.11. TEST LOGOUT

Test Case	TC-12
Objective	To test logging into the app
Action	From the Login screen, Enter the credentials and click the 'Right Arrow' icon button.
Expected Result	The user must be navigated to the Feed page.
Actual Result	The user was navigated to the Feed page.
Conclusion	Test Successful

Table 83: System Testing, Sasae Mobile App TC-12

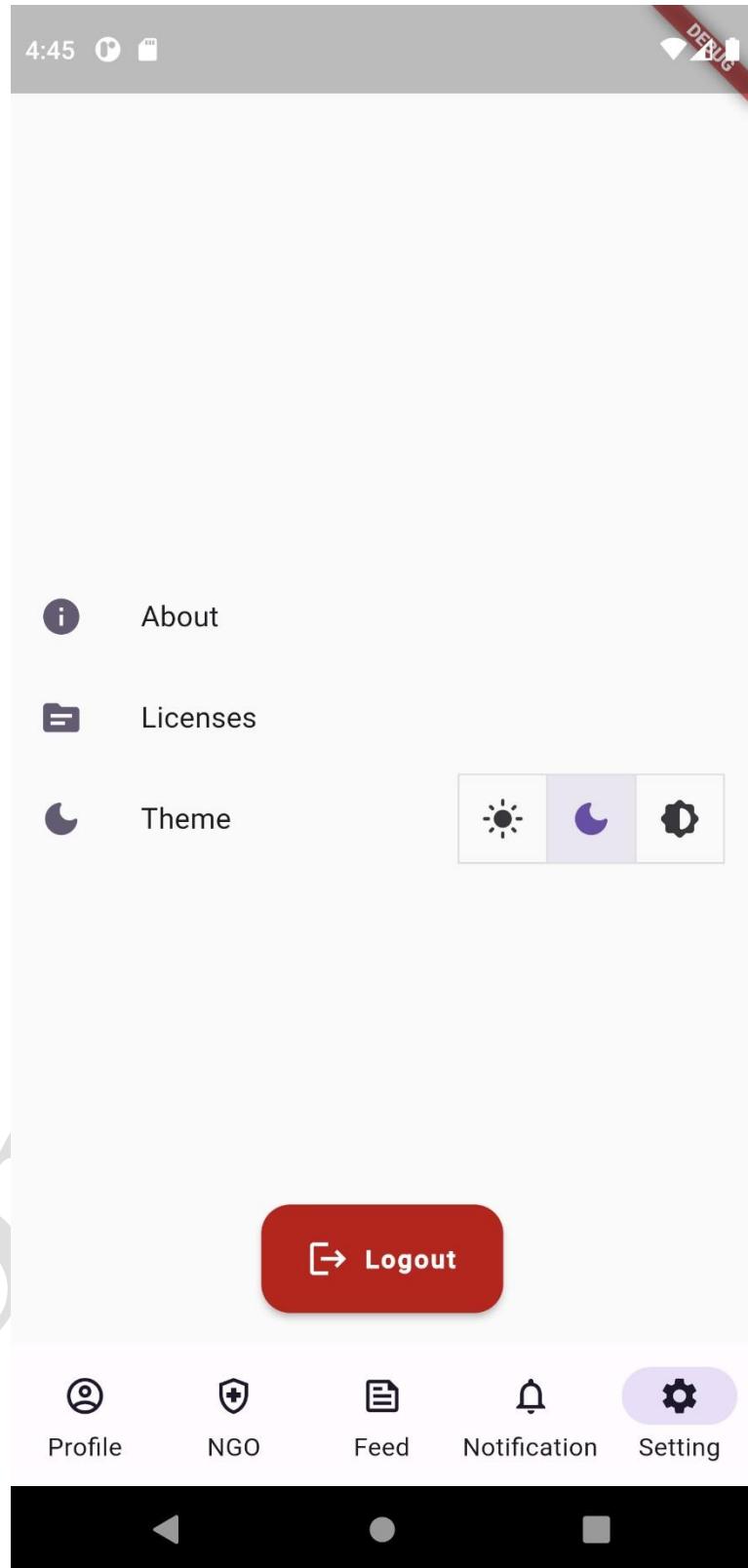


Figure 363: Mobile, Setting Page

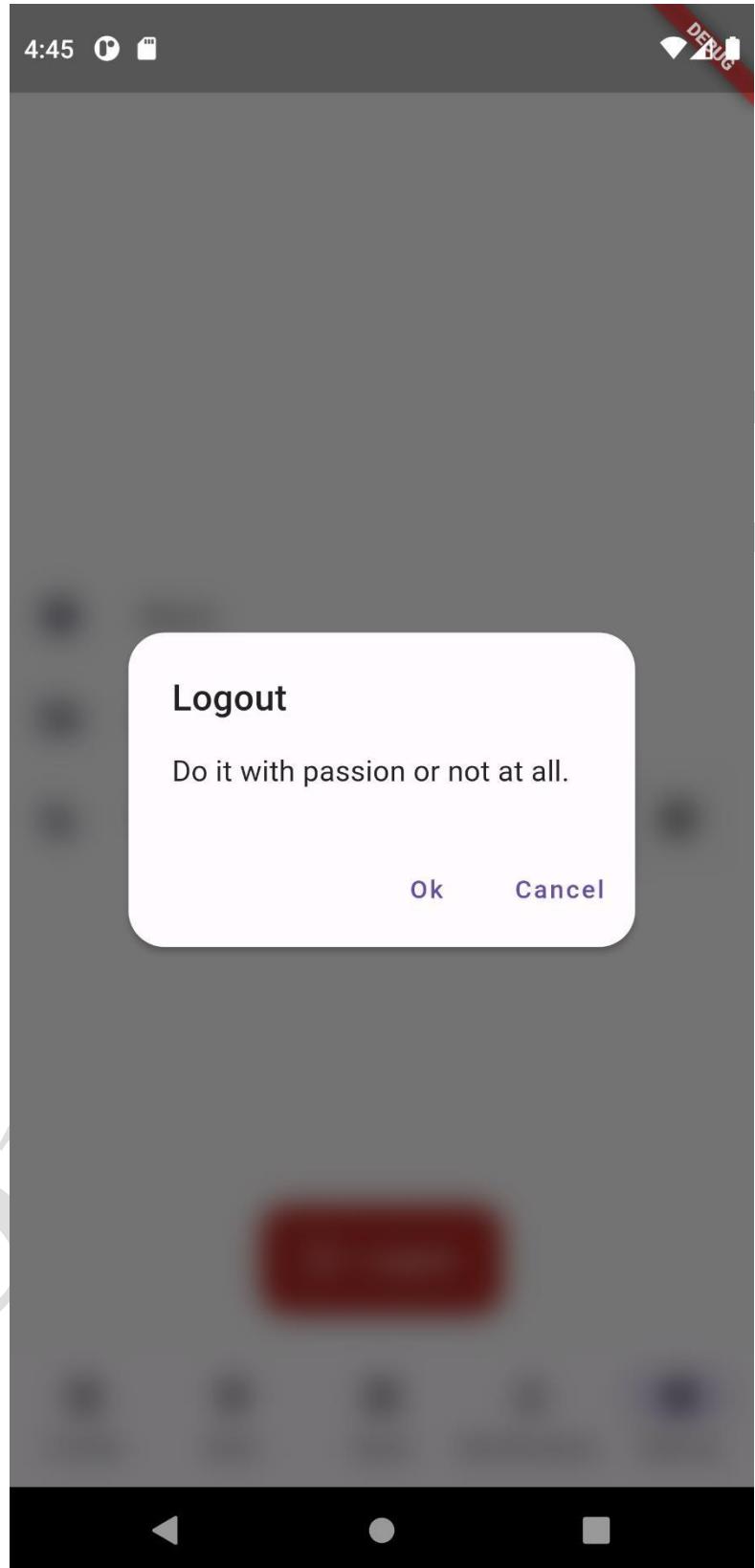


Figure 364: Mobile, Logout Prompt (Before)

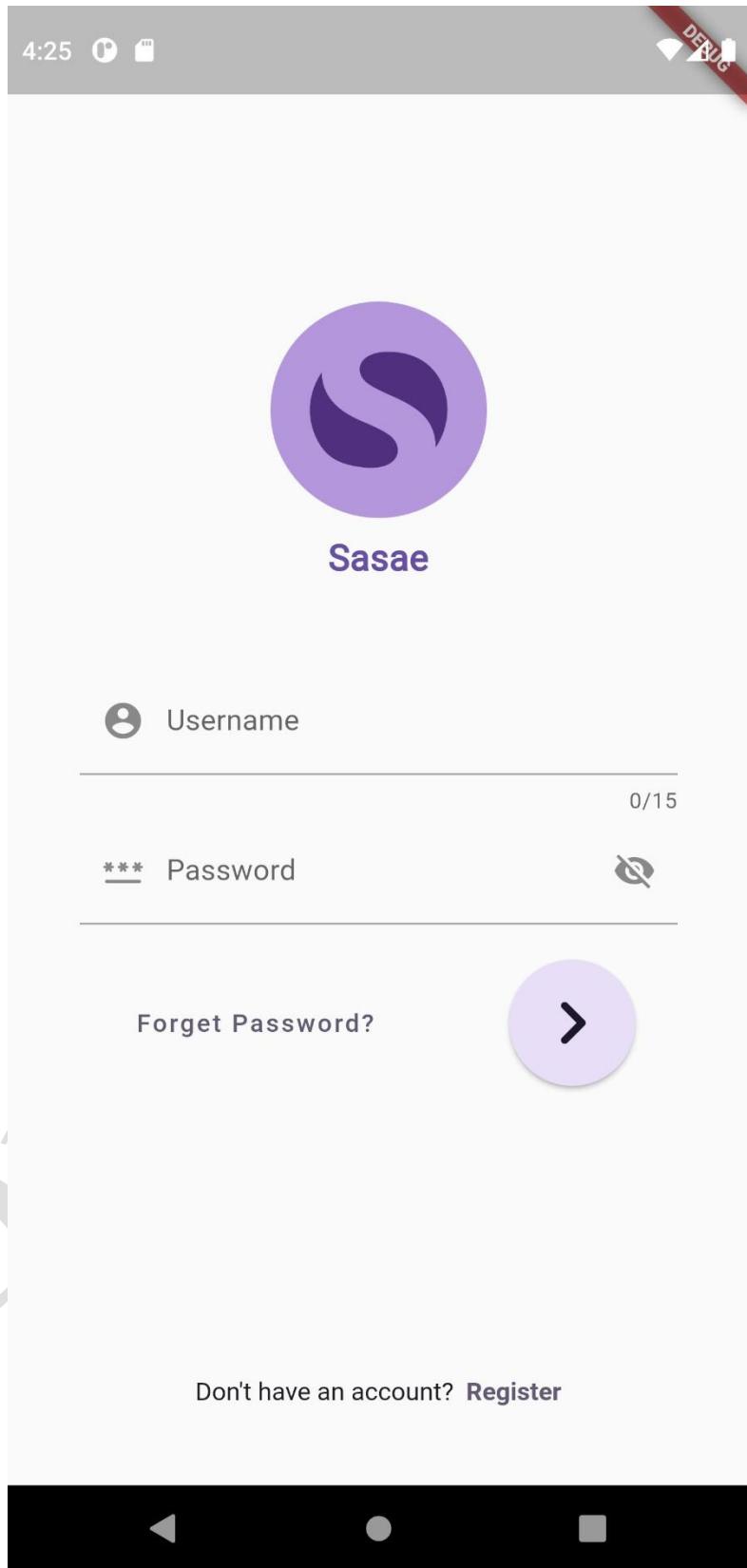


Figure 365: Mobile, Login Screen (After)

7.12. APPENDIX L: USER FEEDBACK

7.12.1. USER FEEDBACK FORM

The screenshot shows a Google Forms survey titled "User Feedback Form". The form consists of several questions:

- Question 1:** "What is your Name? *". Type response field.
- Question 2:** "Rate the UI/UX *". A 5-point rating scale from "Poor" to "Excellent".
- Question 3:** "Rate the Overall Features?". A 5-point rating scale from "Poor" to "Excellent".
- Question 4:** "Have you faced any issues yet? *". Radio button options: Yes, No, Maybe.
- Question 5:** "Any feedback to the project?". Long-answer text field.

The top right corner of the interface shows various navigation and settings icons.

Figure 366: User Feedback Form

7.12.2. SAMPLE OF FILLED USER FEEDBACK FORMS

The screenshot shows a Google Forms interface with the following details:

- Responses:** 10 (highlighted in blue)
- Accepting responses:** Enabled (purple switch)
- Individual:** Selected view mode
- User Feedback Form:**
 - What is your Name? ***: Nawraj Kafle
 - Rate the UI/UX ***: Rating 5 (Excellent)
 - Rate the Overall Features?**: Rating 4 (Good)
 - Have you faced any issues yet? ***: Maybe (selected)
 - Any feedback to the project?**: Fundraising in next update
- Submitted:** 29/04/2022, 00:50

Figure 367: Filled User Feedback Form Sample

7.12.3. USER FEEDBACK RESULT

Rate the UI/UX

10 responses

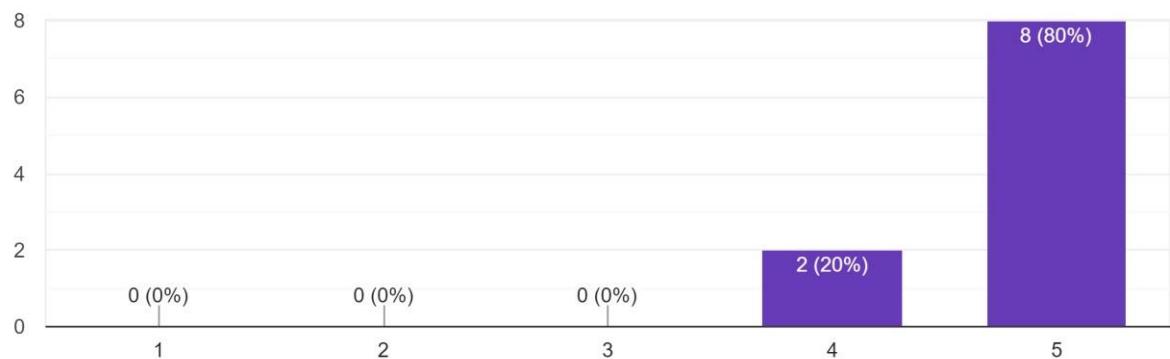


Figure 368: User Feedback, Question-2 Result

Rate the Overall Features?

10 responses

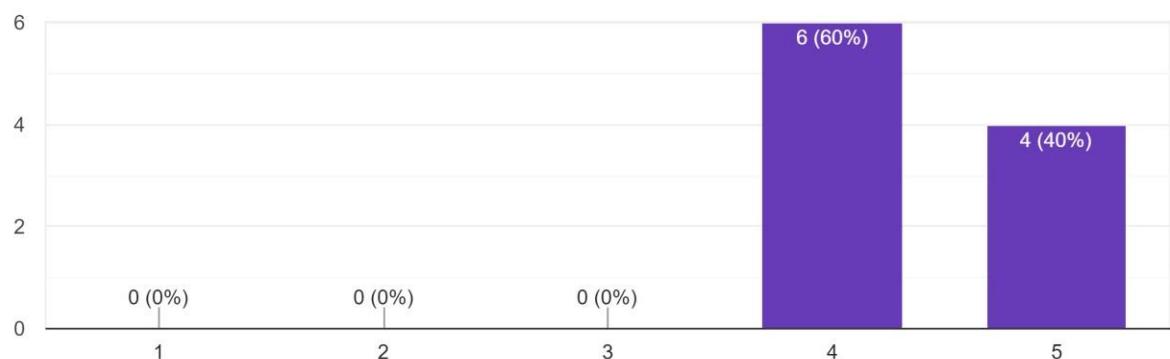


Figure 369: User Feedback, Question-3 Result

Have you faced any issues yet?

10 responses

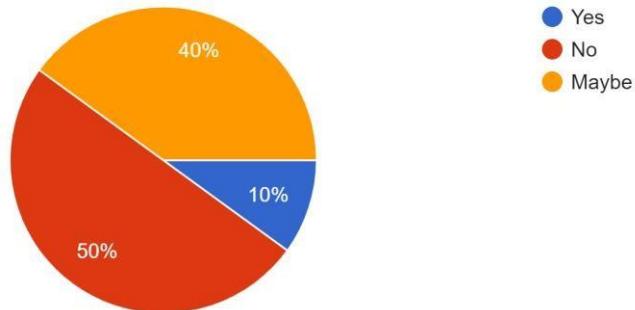


Figure 370: User Feedback, Question-4 Result

Any feedback to the project?

6 responses

No

Fundraising in next update

Login with Facebook/Google

Home screen widget please

Pink theme

Figure 371: User Feedback, Question-5 Result

7.13. APPENDIX M: FUTURE WORK

7.13.1. READINGS FOR FUTURE WORK

- ReactJS: [Getting Started – React \(reactjs.org\)](https://reactjs.org/)
- Tailwind CSS: [Installation: Tailwind CLI - Tailwind CSS](https://tailwindcss.com/installation/tailwindcli)
- Spring Framework: [Spring Framework Documentation](https://spring.io/projects/spring-framework)
- ASP.NET Core: [ASP.NET Core Documentation — ASP.NET documentation](https://docs.microsoft.com/en-us/aspnet/core)

(jakeydocs.readthedocs.io)

- MVC Architecture: [Model–view–controller - Wikipedia](#)
- MVVM Architecture: [Model–view–viewmodel - Wikipedia](#)
- RESTful API: [What is REST - REST API Tutorial \(restfulapi.net\)](#)
- Fetch API: [Fetch API - Web APIs | MDN \(mozilla.org\)](#)
- SOAP vs REST: [SOAP vs REST APIs: Which Is Right For You? | SoapUI](#)
- GraphQL: [Introduction to GraphQL | GraphQL](#)
- Web Design Principle: [10 Principles Of Good Web Design — Smashing Magazine](#)
- Learn MongoDB: [MongoDB Documentation](#)
- Learn PostgreSQL: [PostgreSQL: Documentation](#)
- Material Design Principle: [12 Absolute Principles of Material Design \(creative-tim.com\)](#)
- flutter_osm_plugin: [flutter_osm_plugin - Dart API docs \(pub.dev\)](#)
- Learn Clean Code: [Clean Code: A Handbook of Agile Software Craftsmanship \[Book\] \(oreilly.com\)](#)
- Learn Refactoring: [Refactoring \(martinfowler.com\)](#)
- Minimal Design: [What Is Minimalist Design and How to Apply It | Elementor](#)
- Learn AWS: [Learn AWS with Training and Certification | Cloud Skills Courses and Programs | AWS \(amazon.com\)](#)
- Learn Git and GitHub: [Git - git Documentation \(git-scm.com\)](#)
- Learn Docker: [Docker Documentation | Docker Documentation](#)