Bilkent University

Department of Computer Science

Senior Design Project

KEBAP TYCOON

[www.kebaptycoon.com]

Analysis Report

Group Members

Can Akgün
Doğancan Demirtaş
Kıvanç Kamay
Sinem Sav
Ulaş Sert

Supervisor

Hakan Ferhatosmanoğlu

Jury Members

Uğur Güdükbay
Selim Aksoy

Innovation Expert
Armağan Yavuz

Table Of Contents

Table Of Figures

1. Introduction

One of the contemporary sectors included in computer science field is surely entertainment sector. Today, the game industry has a big portion in economics; it employs dozens of people for different positions starting from game designer, game developer and all the way to the marketing executive. In this big market, business simulation games which are known as tycoon games became one of the genres that entertain people by setting an environment to simulate management of business for them.

We live in a world of scarcity meaning there are limited resources for the unlimited wants of humans. This is one of the fundamental problems in economics and business life. Thus, in a way, business in the real world already is a sort of game. But if one wants to play that game, it’s actually much easier to just turn on the PC and play one of the many great business tycoon simulations.

Within this genre, there are numerous games from different industrial areas and food is one of them. However, none of them is local and represents Turkish way of food. This is why we decided to build a Turkish food business game, "Kebap Tycoon" namely. This game will represent humorous sides of Turkish food and restaurant culture in which Turkish people can enjoy.

2. Current System

In current tycoon games, management and specifically micromanagement remain in the forefront with appealing graphics. Simulation of this management, however, is hold as a single-player game where players manage their any type of business and try to drum up their business. However, when we think about real world business, we see that interaction between people, namely between managements, play an important role in business life. Thus, changing this current single-player pattern in tycoon games to multiplayer is an innovative approach and while entertaining Turkish people with "Kebap Tycoon", making it multiplayer, as it is in real life, is our aim.

3. Proposed System

3.1 Overview

Our senior design project is a cross platform application called "Kebap Tycoon" which is a challenging game where players can found their restaurant chains and put effort to be the best in this sector by making decisions on both preferences of sales and locality strategies. The target user group will mostly depend on residences of Turkey as the game will represent instantaneous aspects of Turkish culture. After starting the game with a small booth (meatball or any type of booth that player selects at the beginning of the game), players will be able to enhance their business up to grand restaurant chains of Turkish kebab. Within this business improvement process, players will be able to see and pursue other restaurant owners in their server instance in order to create a race environment in the game as well as by virtue of Facebook login option players will have the opportunity of being in a competitive game ambience where they can invite their friends on Facebook. Another aspect of the multiplayer feature of the game is that players will be able to sell and buy restaurants to/from their friends. This report contains an overview of the game by describing functional, non-functional and pseudo requirements, use-case diagrams, scenarios and dynamic models including sequence and state diagrams. Finally, user interface mock-ups are included to give an idea about GUI of the game.

3.2 Functional Requirements

- The game will provide an interactive tutorial at the very beginning of the game in which players can see different features.
- Players will be able to use Facebook and Google login and in case they don't want to use one of these, they can register the game filling a short form with their information.
- Players will be able to see their stock status and buy new materials necessary for the production of kebabs and maintenance of the restaurant.
- Players will be able to create new recipes and determine the prices of the kebabs and other products.
- Players will be able to hire and fire different qualified employees (chef, waiter etc.) necessary for their restaurant.

- Players will be able to see which kebab or products customers liked the most.
- Players will be able to see the restaurant's and the dish's popularity.
- Players will be able to buy different types of restaurant items and design the restaurant.
- Players will be able to see monthly reports about their restaurants.
- Players will be able to increase their restaurant's popularity with their recipes and items they buy.
- Players will be able to add their friends by using their nicknames or using Facebook.
- Players will be able to see their added friends' status on game including their profit, current restaurants etc.
- Players will be able to buy restaurants as it is (with its current inventory and furniture) from their friends.
- Players will be able to sell their restaurants as it is (with its current inventory and furniture) to their friends.
- Players will be able to experience random good events where they gain money from.
- Players will be able to experience random bad events where they lose money. (ex. flood, robbery, fire).
- Players will be able to unlock items depending on their levels which are calculated by experience points.
- Players will be able to advertise their restaurant which will increase their revenue.
- Players will be able to open as many restaurants as they wish in case they have enough money.

3.3 Non-Functional Requirements

- **User Friendly Interface:** The game will provide a user-friendly interface with a qualified isometric design.
- **Usability:** The game will provide easy and efficient usability with the help of initial tutorial and also with online documentation. The user-friendly interface will also increase the usability of the game.
- **Stability:** We plan to implement this game aiming having minimum number of bugs and glitches. The bugs and glitches occurred in the game will be fixed as soon as possible.

- **Reliability:** The game will consistently perform correct outputs and mean time between failures will be as maximum as possible.
- **Availability:** The game will operate any time that player has access to internet. The game will be available for Android, iOS and desktop platforms.
- **Security:** Personal information of users required in registration session will be kept safe. Personal information will not be shared with other players or third parties.

3.4 Pseudo Requirements

- The game will be implemented in Java and libGDX framework will be used.
- The game will be available for Android, iOS and desktop platforms.
- The information about the users and their game status will be kept in a cloud environment. For the backend service and cloud, Parse will be used.

3.5 System Models

3.5.1 Scenarios

1) Form Registration

Participating Actor: Mustafa

Entry Condition: There must be no logged in player and an internet connection must be present.

Exit Condition: The registration is finished successfully or cancelled.

Flow of Events 1:

- Mustafa opens the game and clicks on “Registration with form” button.
- He completes the form by filling the username, password and email fields.
- He clicks on the “Finish” button.
- The system checks if the information is unique and greets the player with “Registration successfully done!” message.
- Mustafa moves on to the login screen.

Flow of events 2:

- Mustafa opens the game and clicks on “Registration with form” button.
- He completes the form by filling the username, password and email fields.
- He clicks on the “Finish” button.
- The system checks if the information is unique. However, it is not unique. Thus, a message that says “There is already a user with this information!” appears.
- Mustafa repeats the second event.

2) Facebook & Google Registration

Participating Actor: Kemal

Entry Condition: There must be no logged in user, the user must have Facebook or Google accounts and an internet connection is required.

Exit Condition: The registration is finished successfully or cancelled.

Flow of Events 1:

- Kemal opens the game and clicks on either "Registration with Facebook" or "Registration with Google" button.
- Depending on the choice, either Facebook API or Google API greets the player.
- The system checks if the user has registered before and "Registration successfully done!" message appears if everything went well.
- Mustafa moves on to the game.

Flow of Events 2:

- Kemal opens the game and clicks on either "Registration with Facebook" or "Registration with Google" button.
- Depending on the choice, either Facebook API or Google API greets the player.
- The system checks if the user has registered before and "An error occurred!" message appears if something went wrong.
- Mustafa moves on to the registration screen.

3) Login

Participating Actress: Irmak

Entry Condition: There must be no logged in player, the user must have registered before and an internet connection is required.

Exit Condition: The player logins and starts playing or it is cancelled.

Flow of Events 1:

- Irmak opens the game and login screen greets her.
- She tries logging in using the login button.
- The system checks if she has registered before.
- She moves on to playing.

Flow of Events 2:

- Irmak opens the game and login screen greets her.
- She tries logging in using the login button.

- The system checks if she has registered before and finds that Irmak has not any account.
- She is directed to registration screen.

4) Tutorial

Participating Actress: Deniz

Entry Condition: The player must have logged in and never played the game before.

Exit Condition: The player either plays the tutorial or does not play since it is played before.

Flow of Events 1:

- Deniz logins to the game.
- After the login session, the system finds out that it is her first time in the game.
- She plays the tutorial in the game.
- After that, she continues to play on her own.

Flow of Events 2:

- Deniz logins to the game.
- After the login session, the system finds out that she has played the tutorial before.
- She continues to play the game without playing the tutorial again.

5) Hire Employee:

Participating Actor: Hakan

Entry Condition: The player opens the menu and selects "Hire Employee" tab.

Exit Condition: The player hires the employee or cannot hire due to some reason.

Flow of Events 1:

- While playing, Hakan opens up the menu and selects "Employee" tab.
- He traverses the list and finds himself the most suitable employee.
- He tries to hire the employee and successfully hires.
- He continues to play.

Flow of Events 2:

- While playing, Hakan opens up the menu and selects "Employee" tab.
- He traverses the list and finds himself the most suitable employee.
- He tries to hire the employee but cannot hire because he does not have the required experience.
- He continues looking at the list.

6) Buy New Restaurant:

Participating Actress: Cansın

Entry Condition: The player presses the “Get New Restaurant” button.

Exit Condition: The player either buys a restaurant or cannot buy.

Flow of Events 1:

- Cansın thinks that another restaurant will increase her profit.
- She presses the “Get New Restaurant” button.
- She tries to buy a new restaurant and successfully buys.
- She starts developing that restaurant.

Flow of Events 2:

- Cansın thinks that another restaurant will increase her profit.
- She presses the “Get New Restaurant” button.
- She tries to buy a new restaurant and cannot buy since she has not enough money.
- She continues playing with her initial restaurant.

7) Buy An Item:

Participating Actor: Utku

Entry Condition: The player opens up the menu and hits the “Shopping” tab.

Exit Condition: The player either manages to buy an item or cannot.

Flow of Events 1:

- Utku thinks his restaurant is expanding and more customers come each day. Hence, adding more tables and chairs would be beneficial.
- He opens up the menu and clicks on the “Shopping” tab.
- He tries to buy a table and successfully buys.
- Utku finds a proper area and places the item.

Flow of Events 2:

- Utku thinks his restaurant is expanding and more customers come each day. Hence, adding more tables would be beneficial.
- He opens up the menu and clicks on the “Shopping” tab.
- He tries to buy a table but cannot buy since he does not have the necessary level.
- Utku returns to the shopping list and looks for other items.

8) Give An Advertisement:

Participating Actress: Naz

Entry Condition: The player opens up the menu and clicks on "Advertisement" tab.

Exit Condition: The player gives a certain type of advertisement or cannot.

Flow of Events 1:

- Naz feels that her business is getting more and more popular and considers giving an advertisement.
- She opens the menu and hits the "Advertisement" tab.
- She finds the best medium to give an advertisement and successfully does that.
- Naz continues to play the game.

Flow of Events 2:

- Naz feels that her business is getting more and more popular and considers giving an advertisement.
- She opens the menu and hits the "Advertisement" tab.
- She finds the best medium to give an advertisement. However, she cannot do that because she lacks the required amount of money.
- Naz continues to play the game.

9) Buy Stock:

Participating Actor: Barış

Entry Condition: The player opens up the menu and hits the "Stock" tab.

Exit Condition: The player successfully buys items or cannot do that.

Flow of Events 1:

- Barış observes his restaurant had many customers today and he needs to refill his stocks for tomorrow.
- He opens the menu and hits the "Stock" tab.
- He tries to buy what is missing and manages to do that action.
- Barış continues to play the next day.

Flow of Events 2:

- Barış observes his restaurant had many customers today and he needs to refill his stocks for tomorrow.
- He opens the menu and hits the "Stock" tab.
- He tries to buy what is missing but cannot do that since his money is low.

- Barış continues to play the next day.

<u>10) Create A Recipe:</u>

Participating Actress: Zeynep

Entry Condition: The player opens up the menu and selects the “Recipe” tab.

Exit Condition: The player creates the recipe or cannot.

Flow of Events 1:

- Zeynep thinks she can increase her business’ popularity by creating new recipes.
- She opens up the menu and selects the “Recipe” tab.
- She tries to create a recipe and successfully does that.
- Zeynep continues to play the game.

Flow of Events 2:

- Zeynep thinks she can increase her business’ popularity by creating new recipes.
- She opens up the menu and selects the “Recipe” tab.
- She tries to create a recipe but cannot since her level is not enough.
- Zeynep continues to play the game.

<u>11) See Reports:</u>

Participating Actor: Murat

Entry Condition: The player opens the menu and presses the “Report” tab.

Exit Condition: The player sees the reports or cannot.

Flow of Events 1:

- Murat claims his restaurant is doing great and wants to prove that.
- He opens the menu and presses the “Report” tab.
- He displays his restaurant’s reports.
- Murat closes the tab and continues to play.

Flow of Events 2:

- Murat claims his restaurant is doing great and wants to prove that.
- He opens the menu and presses the “Report” tab.
- He cannot display his restaurant’s reports because it is his first day and there is not any report to show.
- Murat closes the tab and continues to play.

12) Add A Friend:

Participating Actresses: Ezgi and Hazal

Entry Condition: The player opens the menu and selects “Friends” tab.

Exit Condition: The player adds a friend or cannot do that.

Flow of Events 1:

- Ezgi thinks that the game will be more enjoyable with a friend and learns that her best friend Hazal plays Kebap Tycoon as well.
- She opens the menu and selects “Friends” tab.
- From the tab, she writes Hazal’s information and the system finds her.
- She successfully adds Hazal as a friend and continues playing.

Flow of Events 2:

- Ezgi thinks that the game will be more enjoyable with a friend and learns that her best friend Hazal plays Kebap Tycoon as well.
- She opens the menu and selects “Friends” tab.
- From the tab, she writes Hazal’s information. However, the system cannot find Hazal because Ezgi wrote her information wrong.
- She returns to “Friends” screen.

13) Display A Friend:

Participating Actors: Mert and Kerem

Entry Condition: The player opens the menu and selects “Friends” tab.

Exit **Condition:** The player displays his friend’s business.

Flow of Events:

- Mert and Kerem play Kebap Tycoon for a long time. Mert wants to see which player is doing better.
- He opens the menu and selects “Friends” tab.
- From the tab, he clicks on his friend’s name and the system shows Kerem’s information.
- Unfortunately, Mert sees that Kerem is better than him. Hence, Mert continues to play to beat him.

3.5.2 Use-Case Model

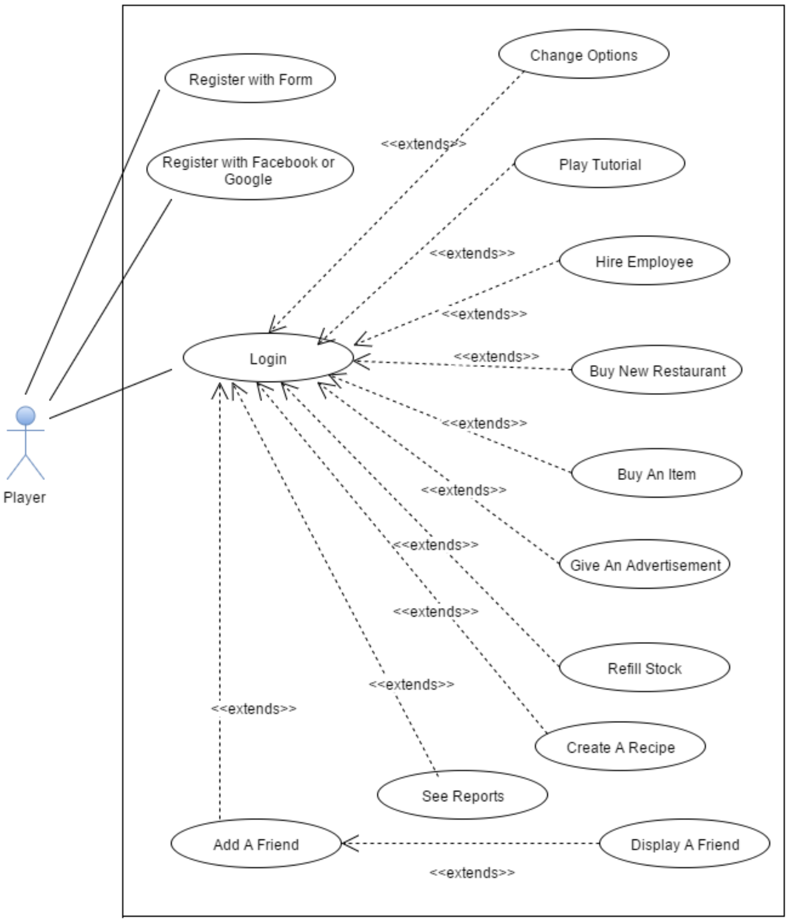
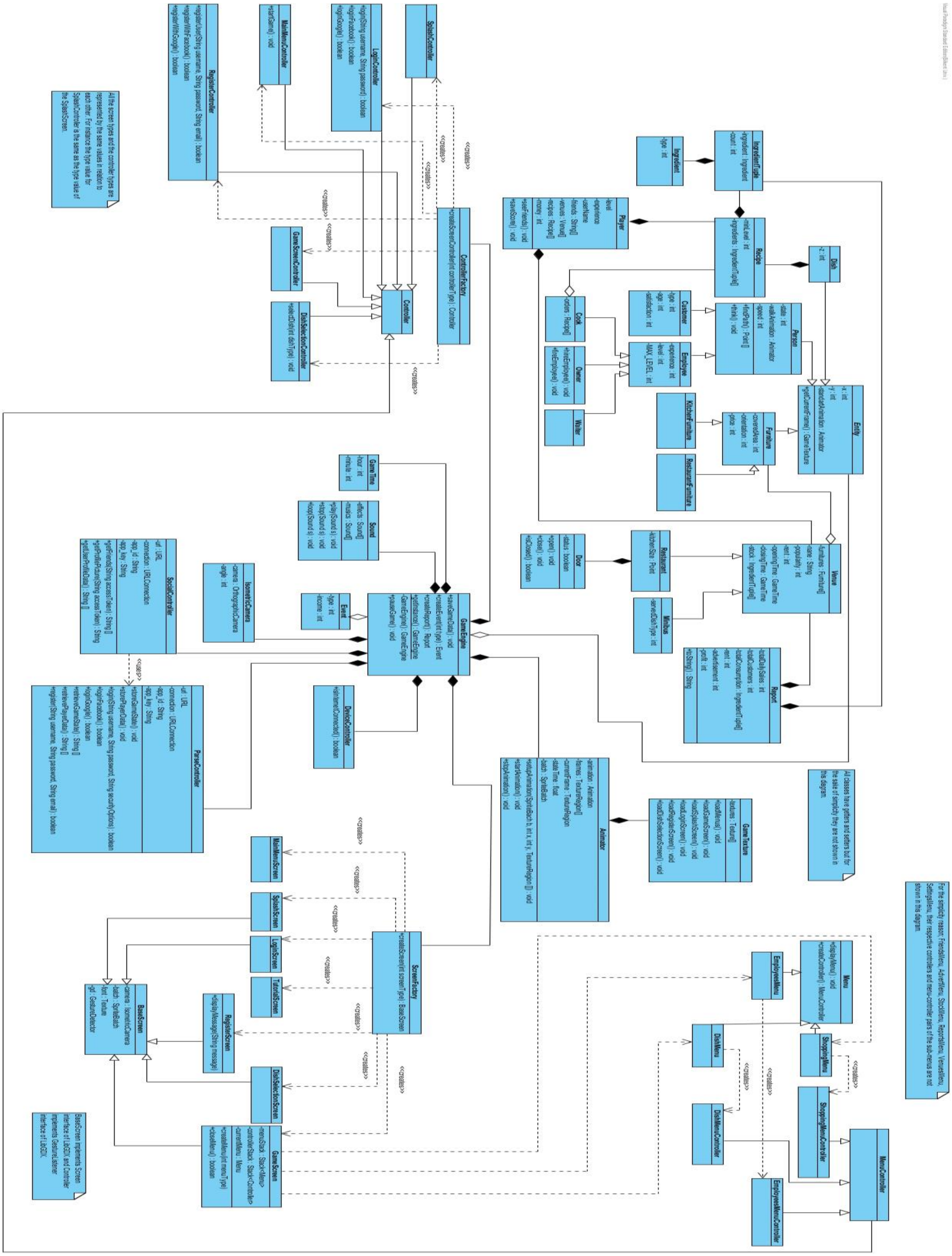


Figure 1 *– Use Case Model*

3.5.3 Class Diagram (Also see Appendix A)



All the screen types and the controller types are represented by the same values in relation to each other. For instance the type value for SplashController is the same as the type value of the SplashScreen.

All classes have getters and setters but for the sake of simplicity they are not shown in this diagram.

For the simplicity reason, FriendsMenu, AdvertMenu, StockMenu, ReportsMenu, VenuesMenu, SettingsMenu, their respective controllers and menu-controller pairs of the sub-menus are not shown in this diagram.

BaseScreen implements Screen interface of LibGDX and Controller implements GestureListener interface of LibGDX.

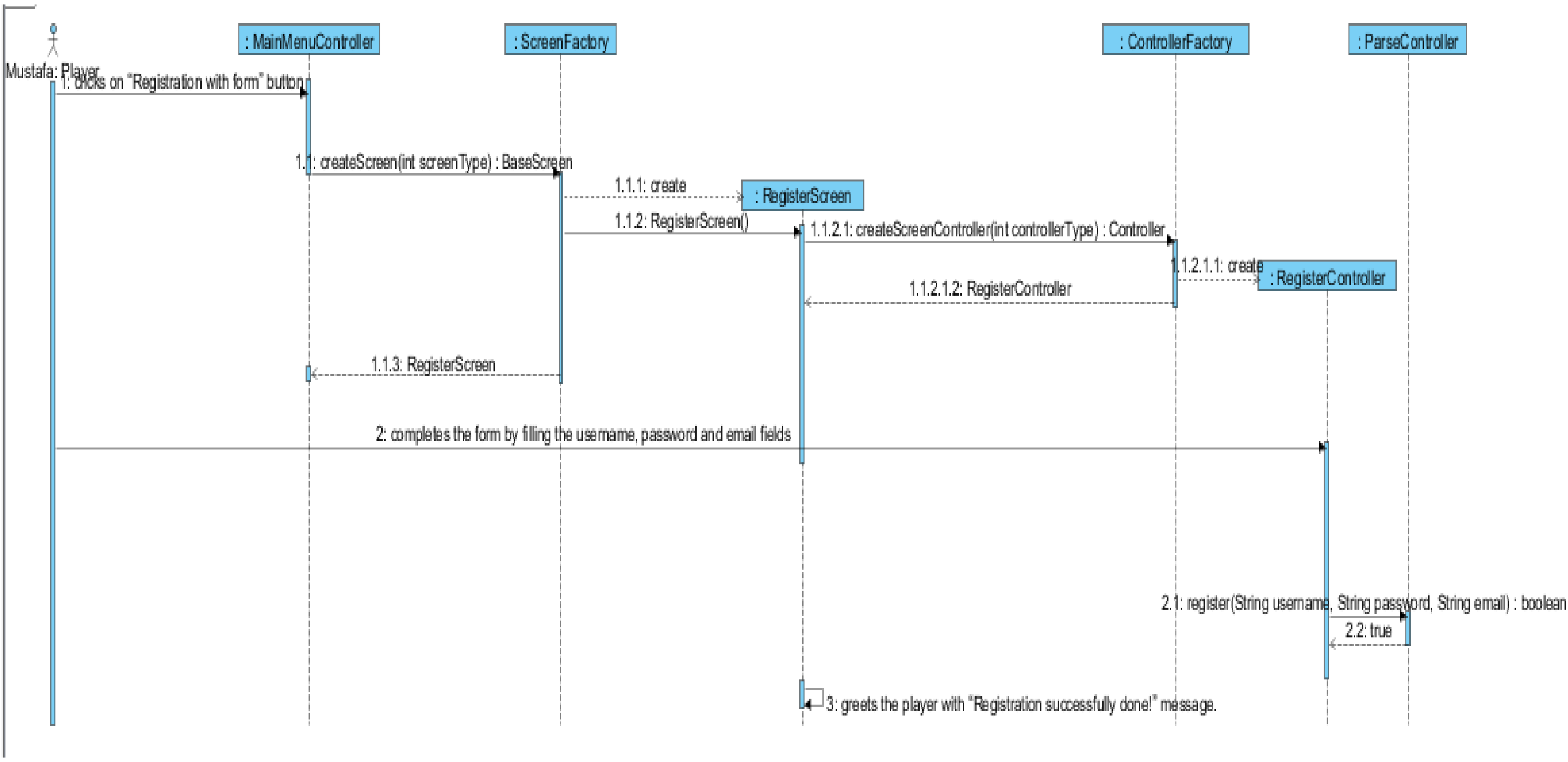
Figure 2 *– Class Diagram*

3.5.4 Dynamic Models

3.5.4.1 Sequence Diagrams

1) Form Registration:

Description: First of all, Mustafa clicks on “Registration with form” button. Then MainMenuController is activated and createScreen(…) method is called. ScreenFactory class creates a RegisterScreen object. At last, RegisterController handles the events and Mustafa is greeted with RegsiterScreen. Using the form, he fills up the necessary places. Moreover, RegisterController calls register(…) method in order to check entered information from database.



Mustafa: Player
: MainMenuController
: ScreenFactory
: ControllerFactory
: ParseController
: RegisterScreen
: RegisterController
1: clicks on "Registration with form" button
1.1: createScreen(int screenType) : BaseScreen
1.1.1: create
1.1.2: RegisterScreen()
1.1.2.1: createScreenController(int controllerType) : Controller
1.1.2.1.1: create
1.1.2.1.2: RegisterController
1.1.3: RegisterScreen
2: completes the form by filling the username, password and email fields
2.1: register(String username, String password, String email) : boolean
2.2: true
3: greets the player with "Registration successfully done!" message.

Figure 3 *– Form Registration Sequence Digram*

2) Login:

Description: Irmak clicks on “Login” button. Like in previous diagram; ScreenFactory, LoginScreen, ControllerFactory and LoginController classes handle the request and she is shown LoginScreen. After she fills the information about logging, ParseController checks if the information is true via login(…) method. If so, a GameScreen is created and shown to Irmak.

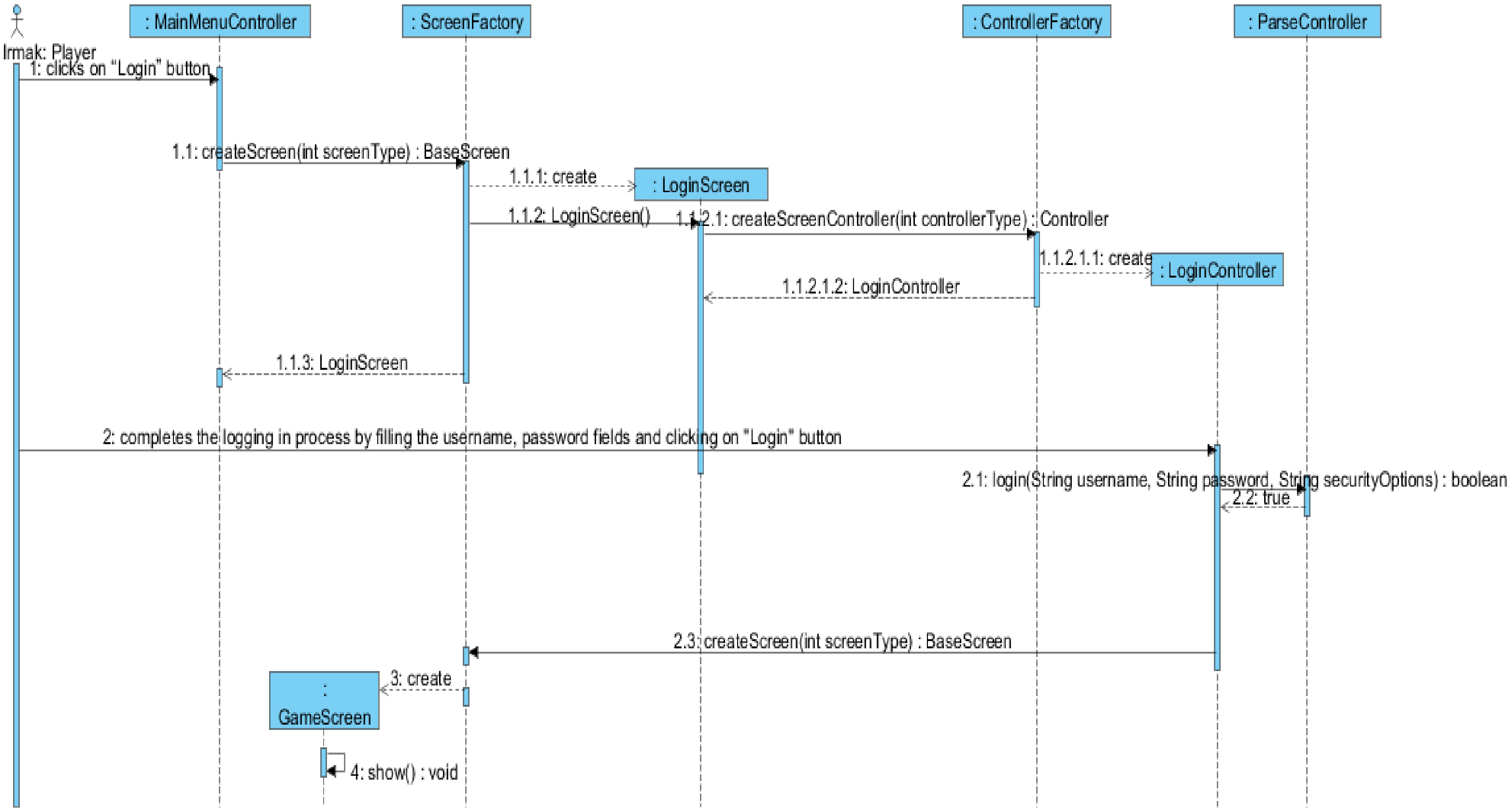
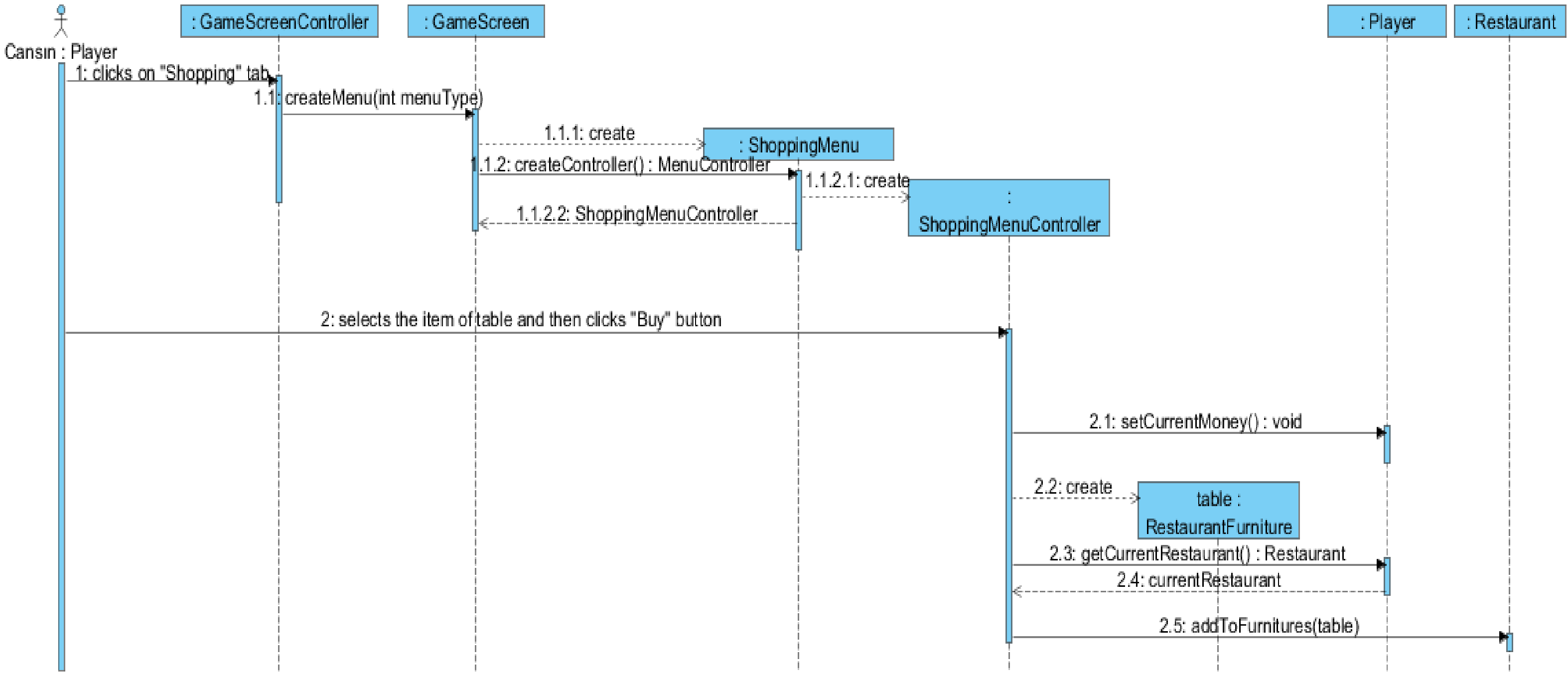

Irmak: Player
: MainMenuController
: ScreenFactory
: ControllerFactory
: ParseController
1: clicks on "Login" button
1.1: createScreen(int screenType) : BaseScreen
1.1.1: create
: LoginScreen
1.1.2: LoginScreen()
1.1.2.1: createScreenController(int controllerType) : Controller
1.1.2.1.1: create
: LoginController
1.1.2.1.2: LoginController
1.1.3: LoginScreen
2: completes the logging in process by filling the username, password fields and clicking on "Login" button
2.1: login(String username, String password, String securityOptions) : boolean
2.2: true
2.3: createScreen(int screenType) : BaseScreen
3: create
: GameScreen
4: show() : void

__Figure 4__ – Login Sequence Digram

3) Buy An Item:

Description: Cansın, firstly clicks on the “Shopping” tab and after that, selects the item that she wants to buy. By tapping the “Buy” button, her current amount of money is updated and the item that she selected is created by ShoppingMenuController class. Finally, the item is added to her restaurant which is a Restaurant object.



Cansın : Player
: GameScreenController
: GameScreen
: Player
: Restaurant
1: clicks on "Shopping" tab
1.1: createMenu(int menuType)
1.1.1: create
: ShoppingMenu
1.1.2: createController() : MenuController
1.1.2.1: create
: ShoppingMenuController
1.1.2.2: ShoppingMenuController
2: selects the item of table and then clicks "Buy" button
2.1: setCurrentMoney() : void
2.2: create
table : RestaurantFurniture
2.3: getCurrentRestaurant() : Restaurant
2.4: currentRestaurant
2.5: addToFurnitures(table)

Figure 5 *– Buy an Item Sequence Digram*

4) Create A Recipe:

Description: At first, Zeynep opens the “Dish” tab and chooses hamburger from her unlocked items. Moreover, she tries to increase the amount of onion in the hamburger. To do this, RecipeMenuController first calls getRecipe(…) method on Player object. Then, using the IngredientTuple class, hamburger’s onion quantity is changed by calling setCount(…) method.

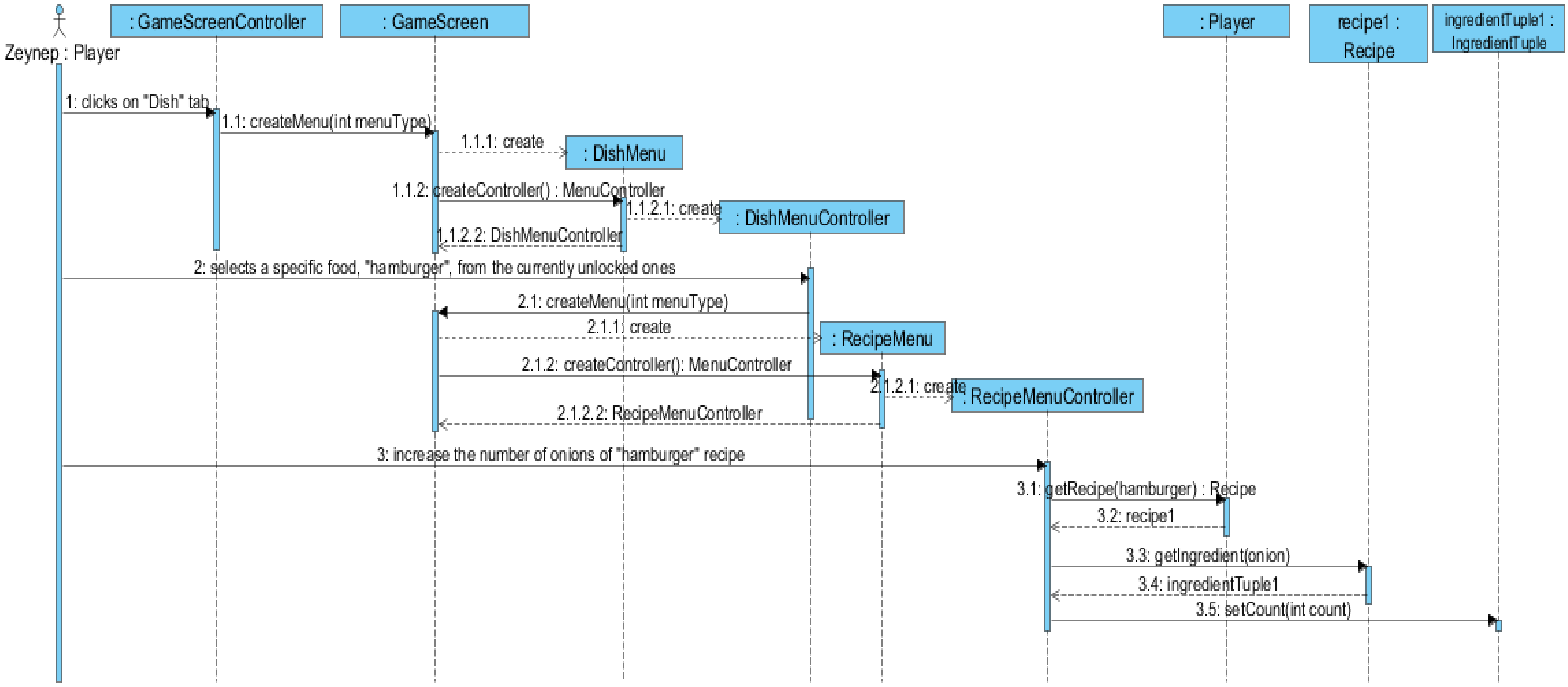

Zeynep : Player
: GameScreenController
: GameScreen
: Player
recipe1 : Recipe
ingredientTuple1 : IngredientTuple
1: clicks on "Dish" tab
1.1: createMenu(int menuType)
1.1.1: create
: DishMenu
1.1.2: createController() : MenuController
1.1.2.1: create
: DishMenuController
1.1.2.2: DishMenuController
2: selects a specific food, "hamburger", from the currently unlocked ones
2.1: createMenu(int menuType)
2.1.1: create
: RecipeMenu
2.1.2: createController(): MenuController
2.1.2.1: create
: RecipeMenuController
2.1.2.2: RecipeMenuController
3: increase the number of onions of "hamburger" recipe
3.1: getRecipe(hamburger) : Recipe
3.2: recipe1
3.3: getIngredient(onion)
3.4: ingredientTuple1
3.5: setCount(int count)

Figure 6 *– Create A Recipe Sequence Digram*

3.5.4.2 State Diagrams

1) Register:

When the player opens Kebap Tycoon, a choice is made for the type of registration. On one hand, if the player chooses registering with form, the form must be filled with valuable information. If so, there is not any problem. However, if there is a problem, the player is returned back to filling the form again. On the other hand, when the player chooses registering with either Facebook or Google, the specific API is waited for a return. If a problem occurs, the state goes to its initial position.

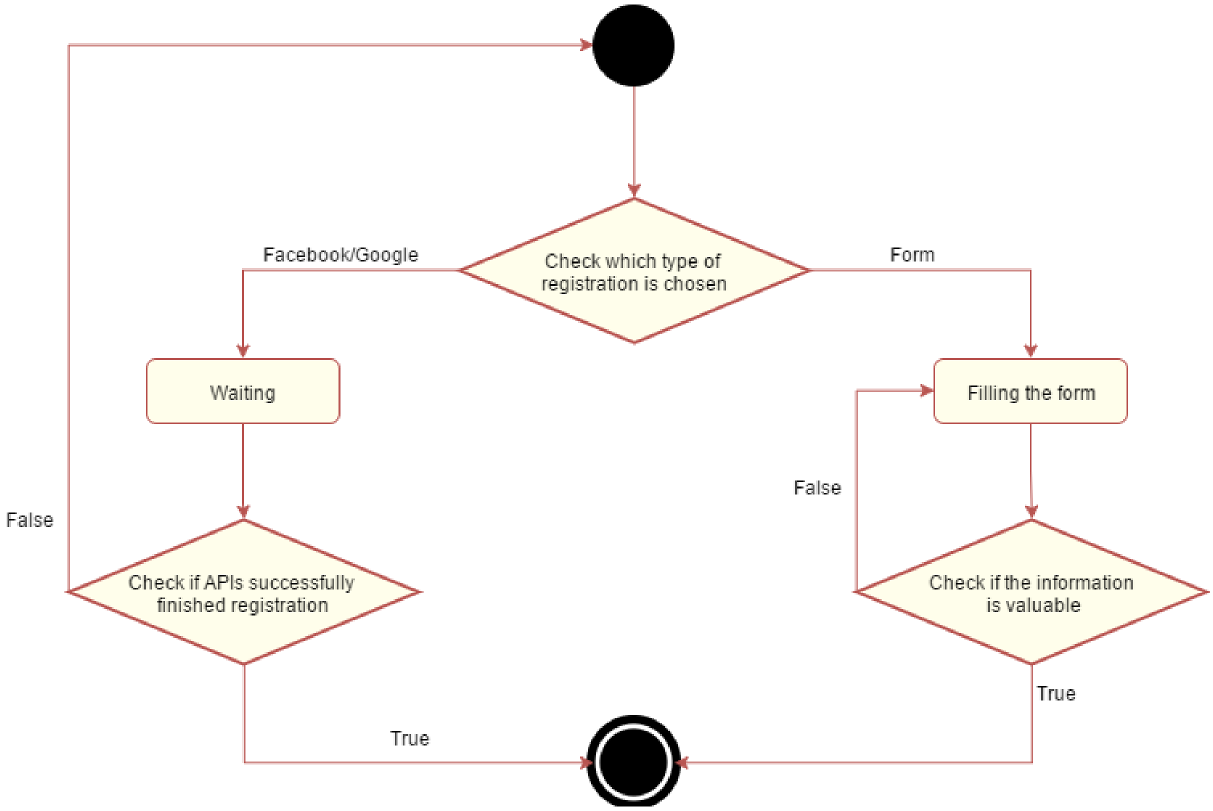

Facebook/Google
Check which type of registration is chosen
Form
Waiting
Filling the form
False
False
Check if APIs successfully finished registration
Check if the information is valuable
True
True

***Figure 7** – Register State Diagram*

2) Login:

When the game starts, it checks the previous condition for whether there is a remembered user or not. If so, the player can either logout or start playing. However, if there is not any logged in person, the player can either register or login. By doing that, the player will be in “Logged in” state.

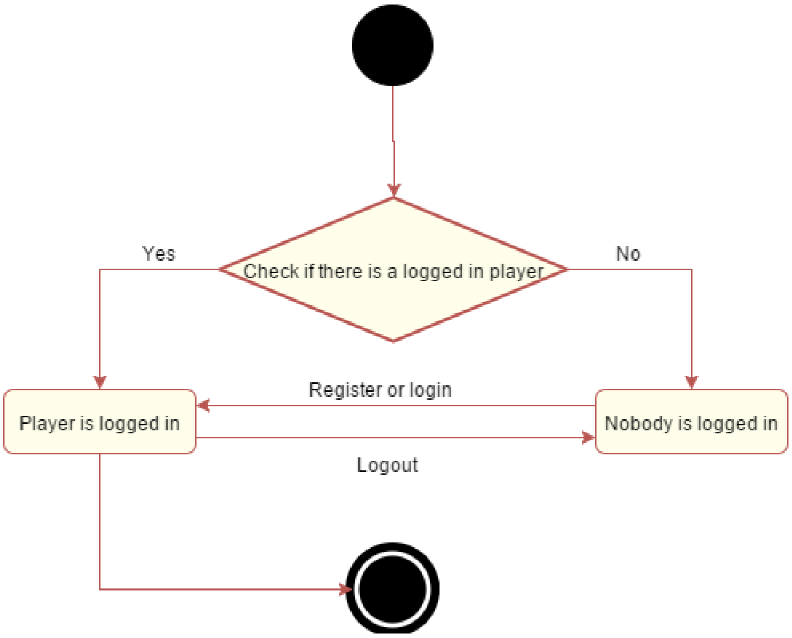


Figure 8 *– Login State Diagram*

3) In Game Functions:

In the actual gameplay, there are several functionalities of the game that the player can do. Fortunately, most of them are very similar in terms of state diagrams. Therefore, we can examine one of them to understand the whole state table. In the beginning, the player is in the "Playing" state. From this state, he can navigate to several functions. All of them start with opening a menu, doing that function and returning back to the "Playing" state. When the player wants to exit, the game saves and then quits.

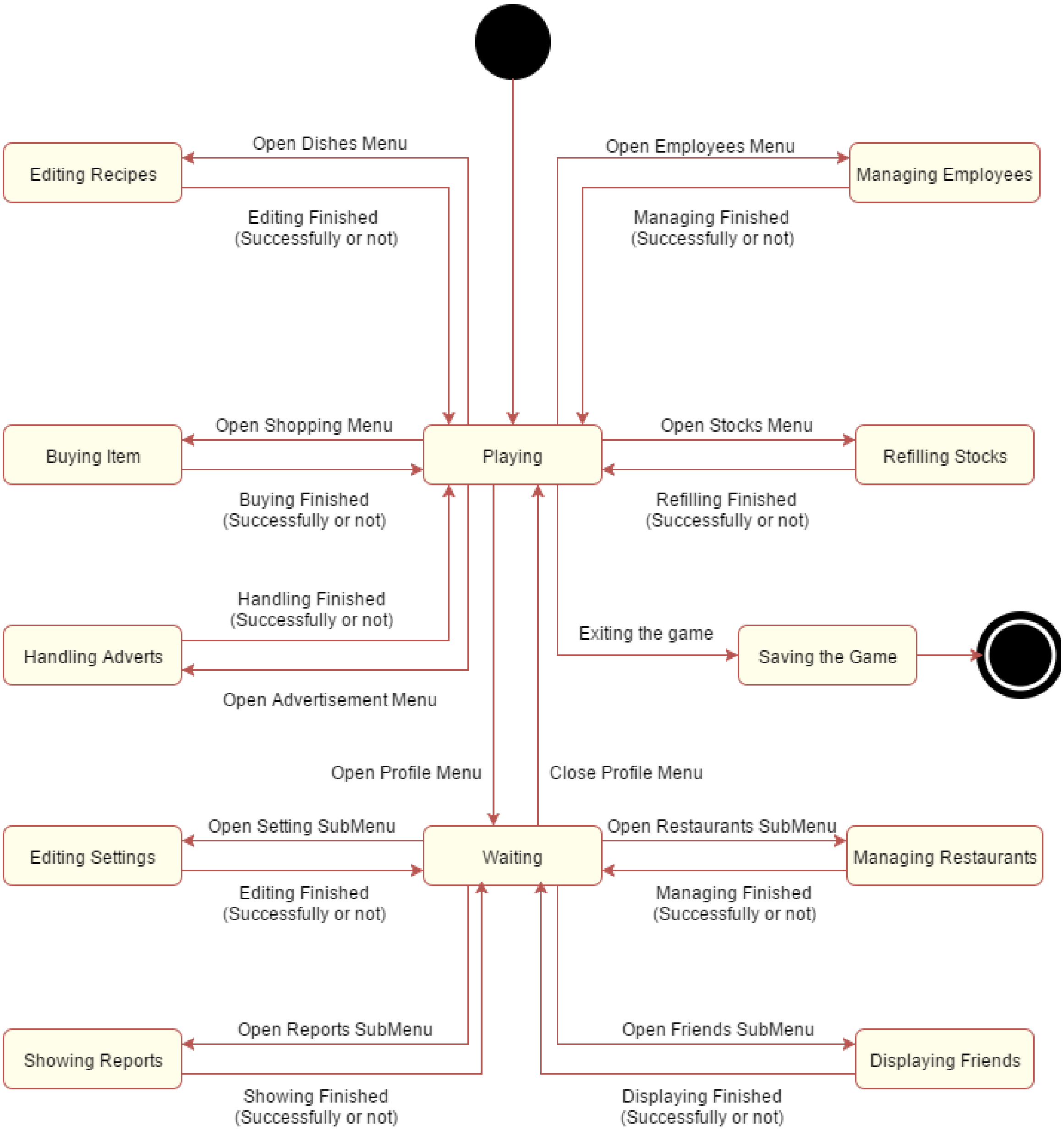


Figure 9 *– Register State Diagram*

3.5.4.3 Activity Diagram(Also see Appendix B)

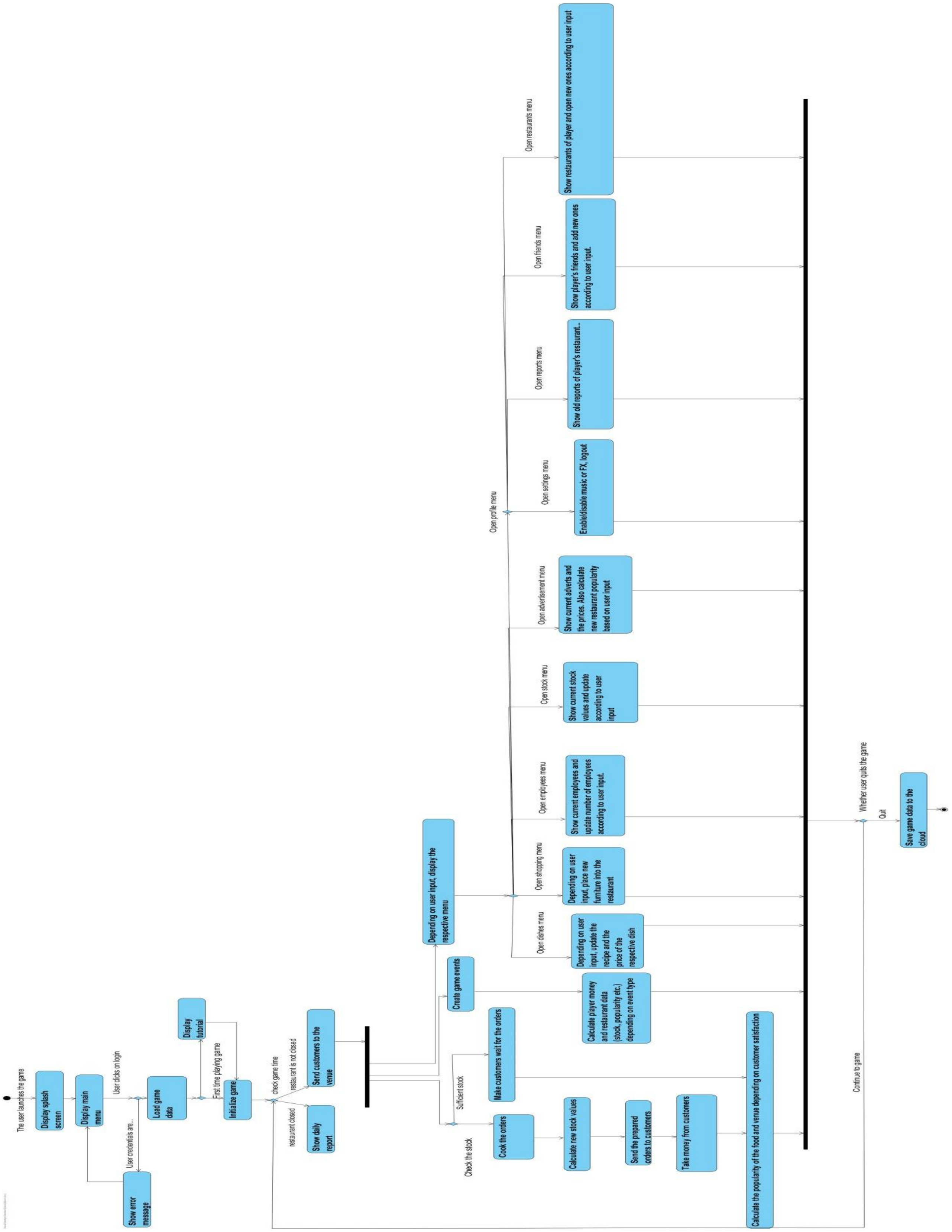

The user launches the game
Display splash screen
Display main menu
User clicks on login
User credentials are...
Show error message
Load game data
First time playing game
Display tutorial
Initialize game
check game time
restaurant is not closed
restaurant closed
Send customers to the venue
Show daily report
Create game events
Calculate player money and restaurant data (stock, popularity etc.) depending on event type
Depending on user input, display the respective menu
Open dishes menu
Depending on user input, update the recipe and the price of the respective dish
Open shopping menu
Depending on user input, place new furniture into the restaurant
Open employees menu
Show current employees and update number of employees according to user input.
Open stock menu
Show current stock values and update according to user input
Open advertisement menu
Show current adverts and the prices. Also calculate new restaurant popularity based on user input
Open profile menu
Open settings menu
Enable/disable music or FX, logout
Open reports menu
Show old reports of player's restaurant...
Open friends menu
Show player's friends and add new ones according to user input.
Open restaurants menu
Show restaurants of player and open new ones according to user input
Check the stock
Sufficient stock
Make customers wait for the orders
Cook the orders
Calculate new stock values
Send the prepared orders to customers
Take money from customers
Calculate the popularity of the food and venue depending on customer satisfaction
Continue to game
Whether user quits the game
Quit
Save game data to the cloud

Figure 10 *– Activity Digram*

3.5.5 User Interface

Main Menu Screen:

The first screen that player will see is the main menu screen (see Figure 11). In this menu, player can directly start playing the game in case s/he is already registered or can register and make a Facebook login from bottom. Lastly, if player does not have a Facebook account, clicking on "Facebook hesabın yok mu?" and fill a form to register.

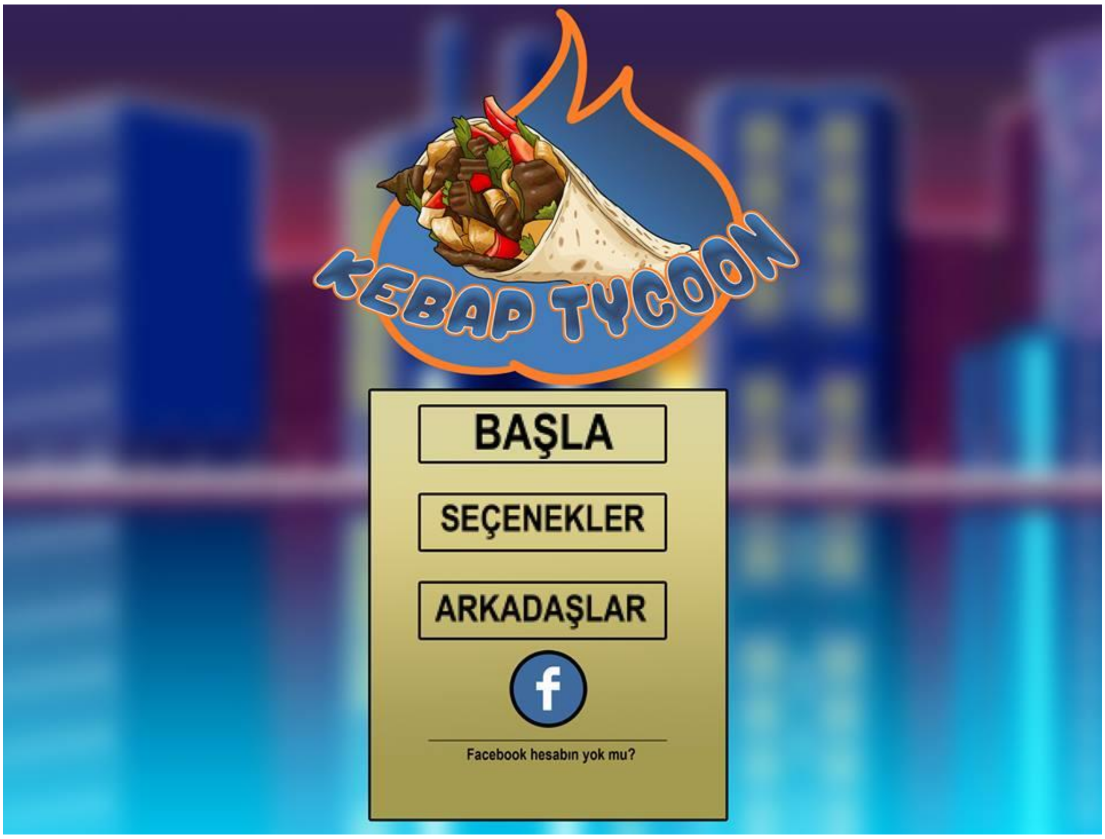


Figure 11 *– Main Menu Screen Mock-Up*

<u>Initial Dish Selection Menu:</u> After login operation, if player is playing the game for the first time, this menu (see Figure 12) will appear and player can select one of the dish and character options to start his/her booth.

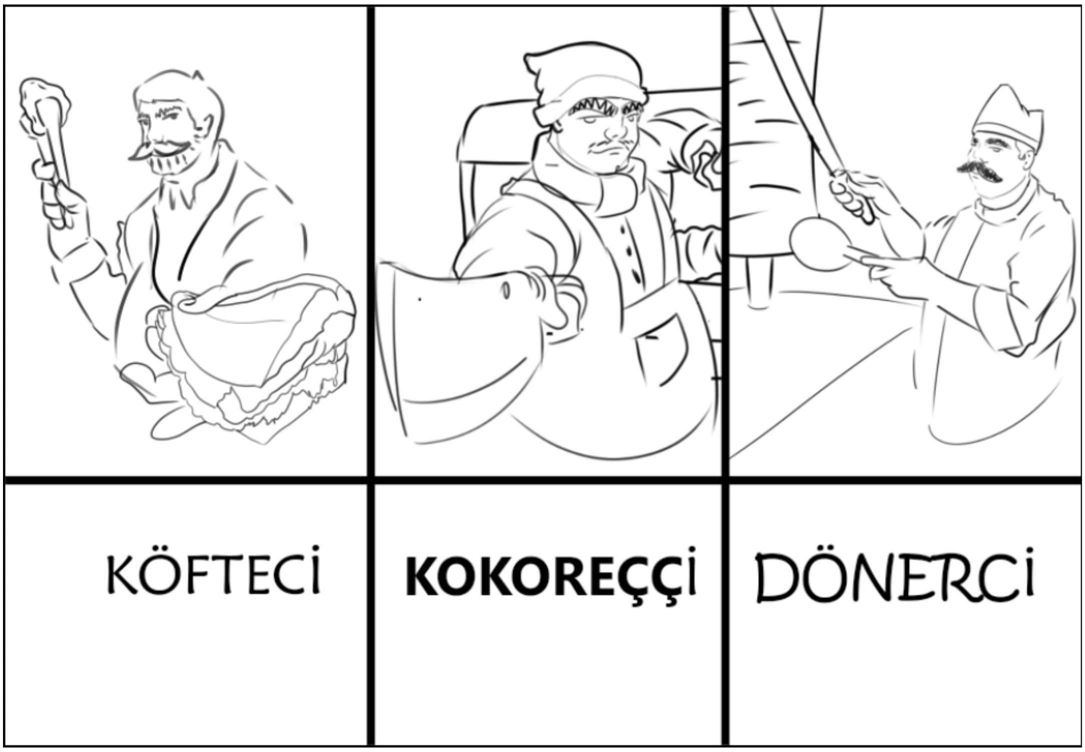


Figure 12 *– Initial Dish Selection Menu Mock-Up*

Game Screen: After login operation and dish/hero selection, player can start playing with the main game screen(see Figure 13). In this screen, player can see his/her current money, date and time in the game and also select one of the menus from bottom. In addition, current status of the booth or restaurant, guests, the flow of work(waiters, chefs) can be also seen.

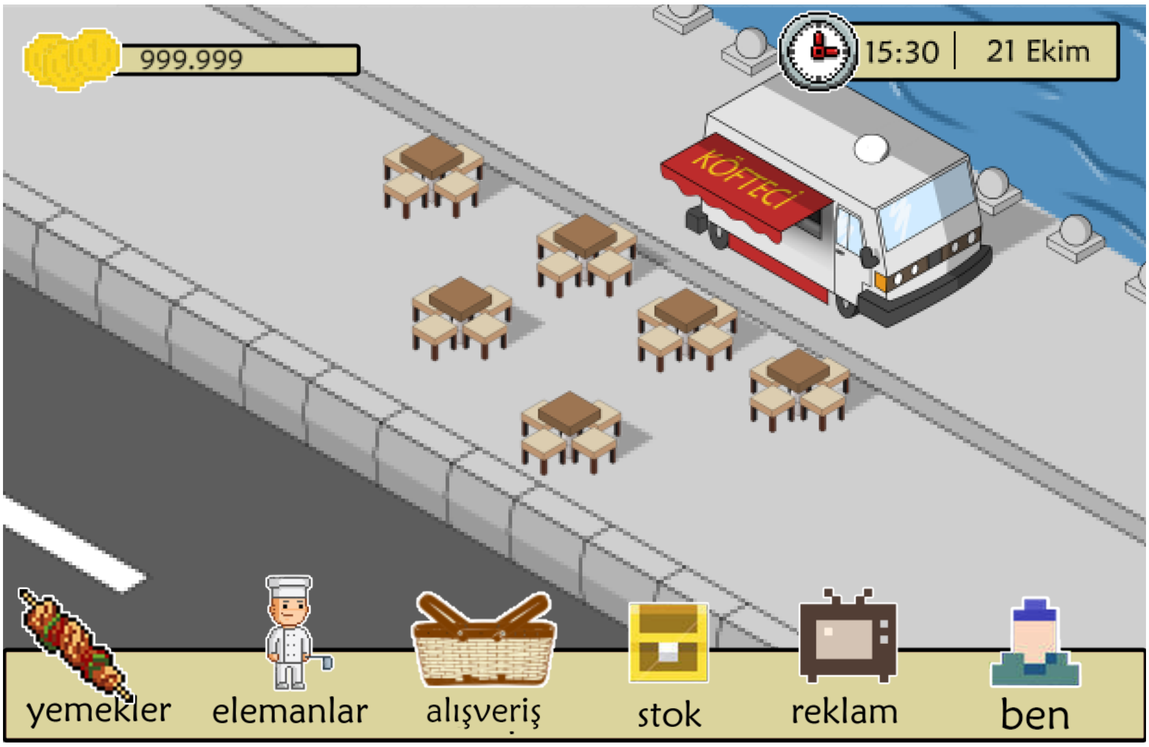


***Figure 13** – Game Screen Mock-Up*

Profile Menu: In the game screen which is previously described, player can select one of the menus from bottom. In Figure 14, profile is selected as an example and clicking on any sub-menu, player can get information about his/her profile namely; restaurants, reports and friends (buy/sell restaurant operation between friends will be managed from this Friends sub-menu).

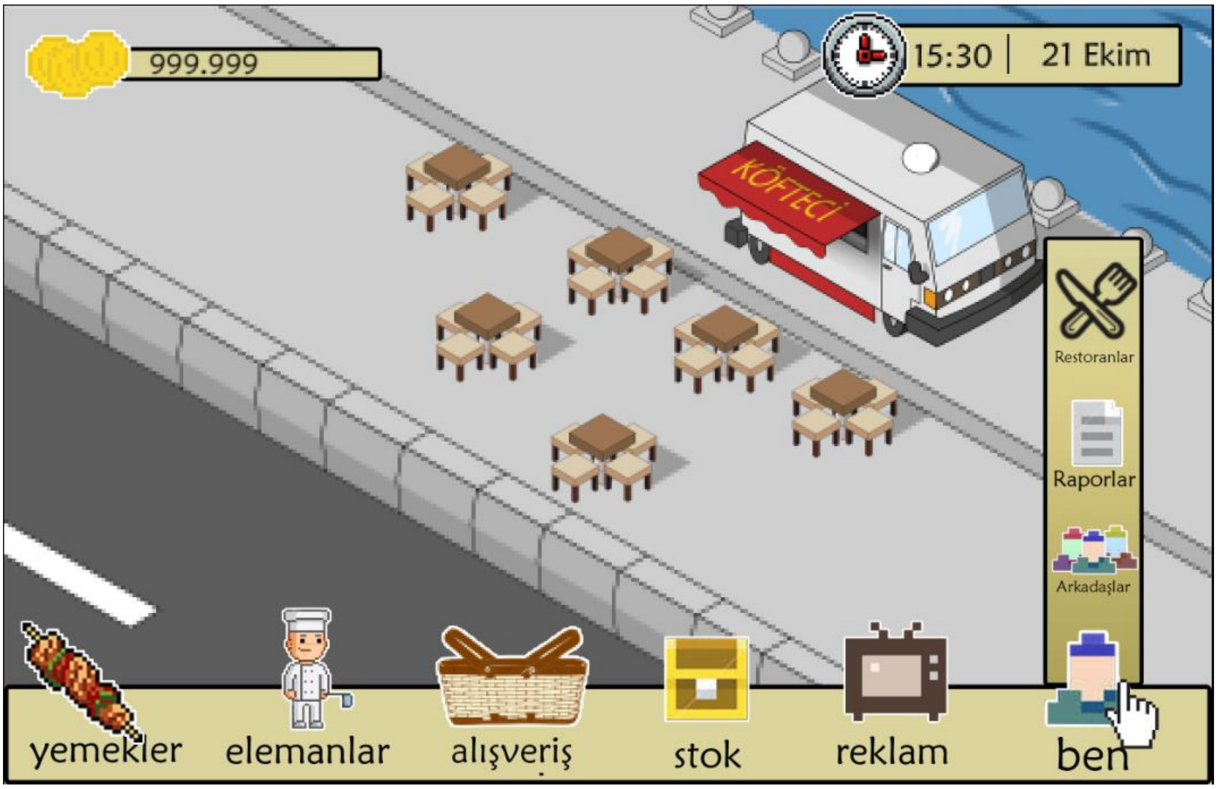


Figure 14 *– Profile Menu Mock-Up*

Current Dishes Menu: From this menu(see Figure 15), player can see the current dishes he can sell, their prices and also locked dished. These locked dishes will be unlocked depending on player's experience in the game.

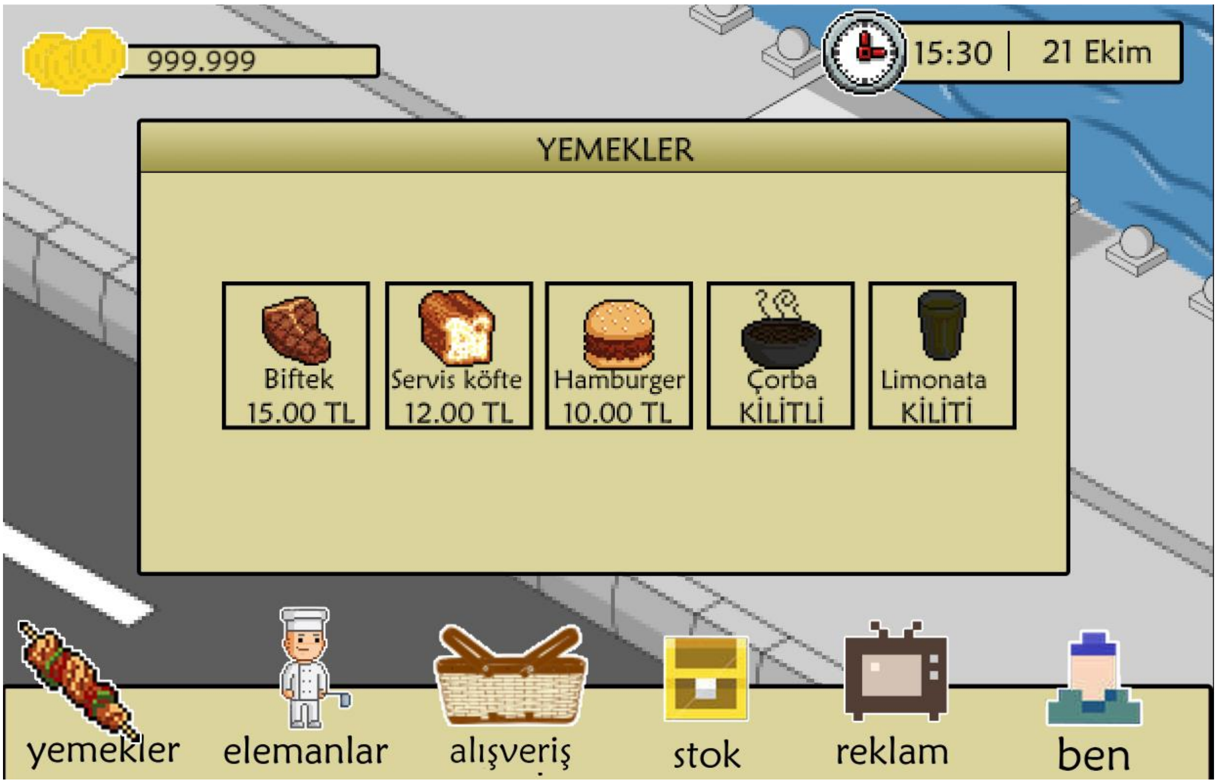


***Figure 15** – Current Dishes Menu Mock-Up*

<u>**Recipe Menu:**</u> With this menu, player will be able to prepare his own recipe on a specific food. Figure 16 shows hamburger recipe as a specific example. Player can increase or decrease the amount of ingredients in hamburger and price will change accordingly.

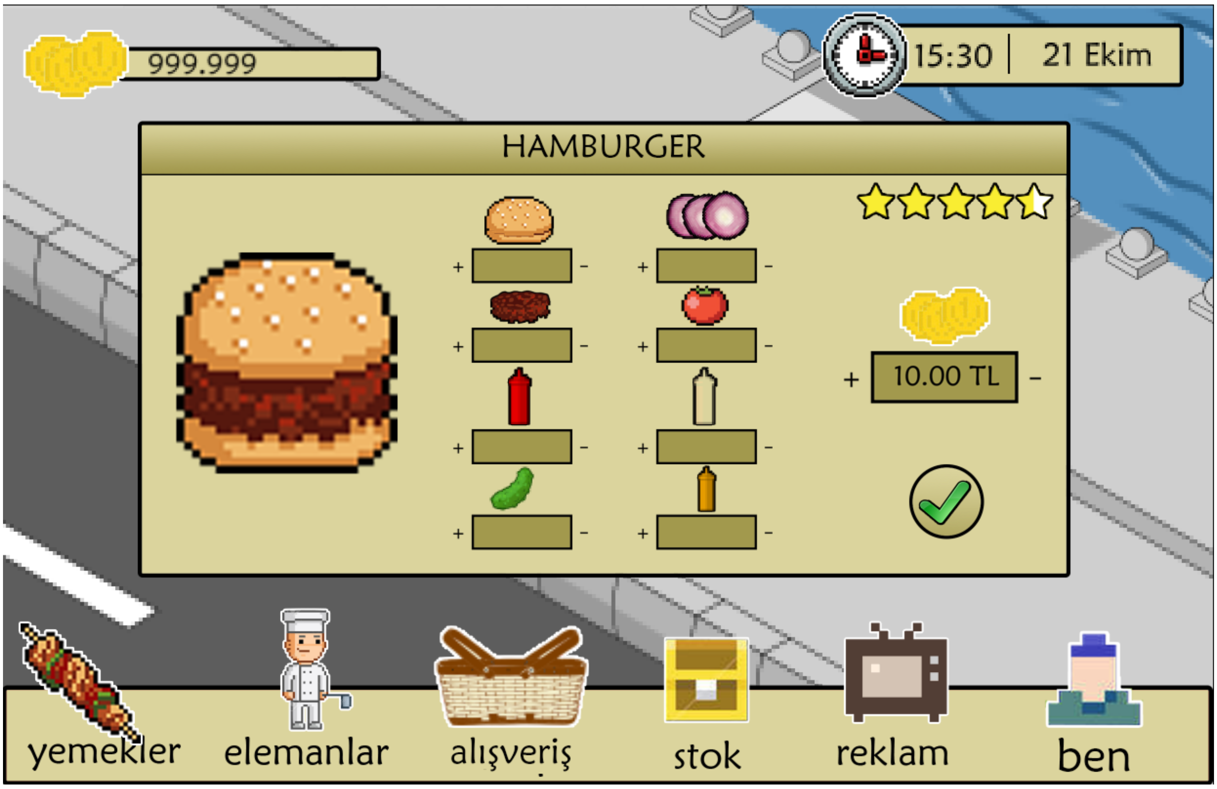


Figure 16 *– Recipe Menu Mock-Up*

4. Glossary

LibGDX: libGDX is a game-development application framework written in the Java programming language. It allows for the development of desktop and mobile games by using the same code base.

Parse: Parse is a cloud system that is helpful for backend services.

Glitch: A short-term fault on the software system.

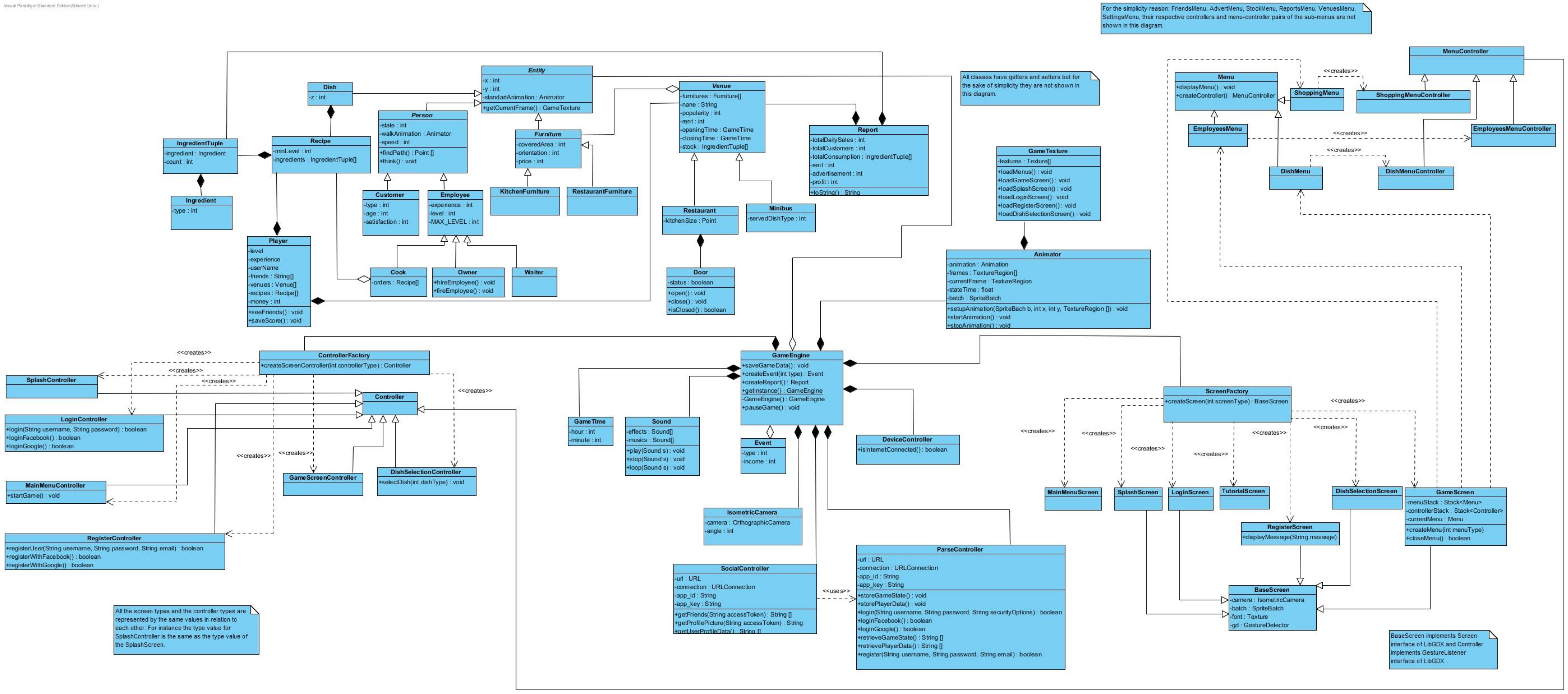
5.References

libgdx.badlogicgames.com,. 'Libgdx'. N.p., 2015. Web. 30 Oct. 2015.

parse.com,. 'Parse'. N.p., 2015. Web. 28 Oct. 2015.

Atwood, Jeff. 'Understanding Model-View-Controller'.Blog.codinghorror.com. N.p., 2008. Web. 1 Nov. 2015.

6. Appendix A



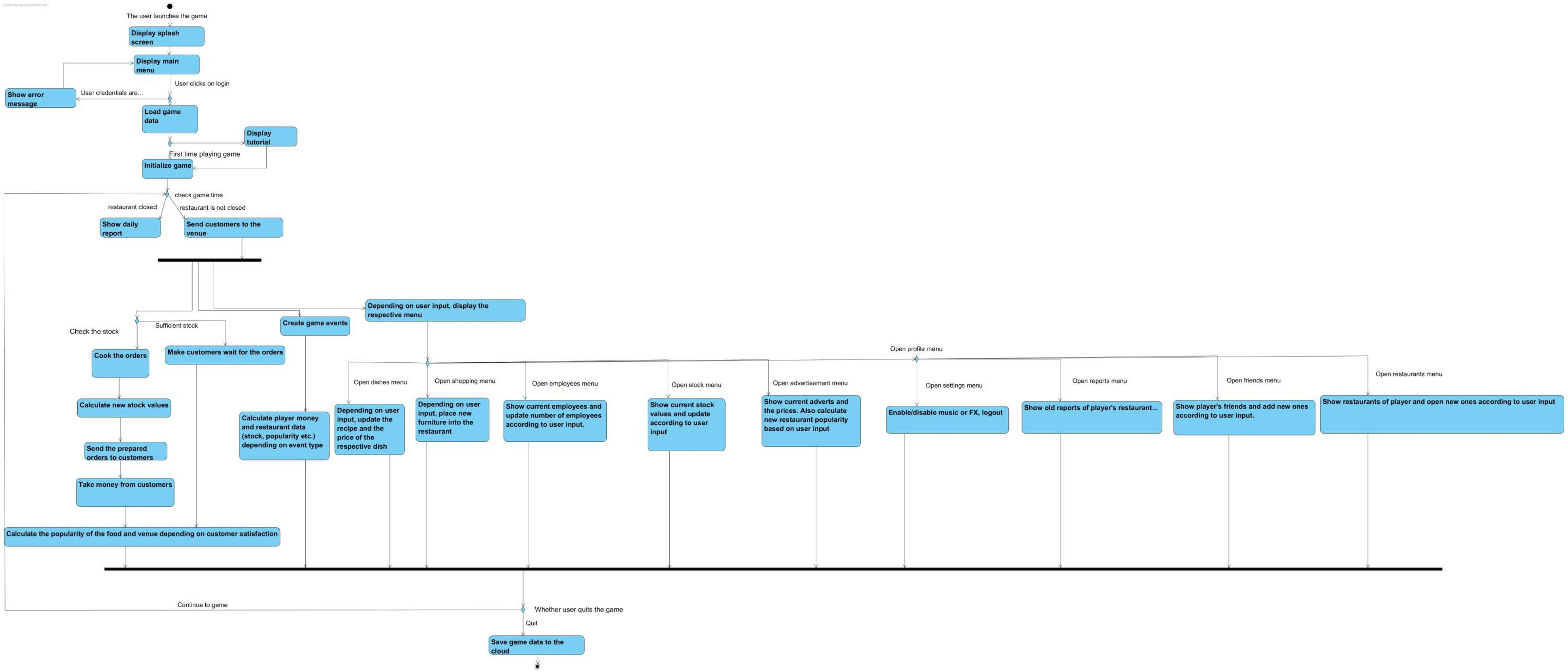
For the simplicity reason; FriendsMenu, AdvertMenu, StockMenu, ReportsMenu, VenuesMenu, SettingsMenu, their respective controllers and menu-controller pairs of the sub-menus are not shown in this diagram.

All classes have getters and setters but for the sake of simplicity they are not shown in this diagram.

All the screen types and the controller types are represented by the same values in relation to each other. For instance the type value for SplashController is the same as the type value of the SplashScreen.

BaseScreen implements Screen interface of LibGDX and Controller implements GestureListener interface of LibGDX.

7. Appendix B



The user launches the game
Display splash screen
Display main menu
User clicks on login
User credentials are...
Show error message
Load game data
Display tutorial
First time playing game
Initialize game
check game time
restaurant closed
restaurant is not closed
Show daily report
Send customers to the venue
Depending on user input, display the respective menu
Check the stock
Sufficient stock
Create game events
Cook the orders
Make customers wait for the orders
Calculate new stock values
Send the prepared orders to customers
Take money from customers
Calculate the popularity of the food and venue depending on customer satisfaction
Calculate player money and restaurant data (stock, popularity etc.) depending on event type
Open dishes menu
Depending on user input, update the recipe and the price of the respective dish
Open shopping menu
Depending on user input, place new furniture into the restaurant
Open employees menu
Show current employees and update number of employees according to user input.
Open stock menu
Show current stock values and update according to user input
Open advertisement menu
Show current adverts and the prices. Also calculate new restaurant popularity based on user input
Open profile menu
Open settings menu
Enable/disable music or FX, logout
Open reports menu
Show old reports of player's restaurant...
Open friends menu
Show player's friends and add new ones according to user input.
Open restaurants menu
Show restaurants of player and open new ones according to user input
Continue to game
Whether user quits the game
Quit
Save game data to the cloud