Requirement Analysis Document
Holy Crushade Project

Introduction

1. **Purpose of the system:**
   The system was created to meet the needs of Munich residents and provide them a convenient way to finding restaurants in the city and book tables at any time.
2. **Scope of the system:**
   The system only works in Munich, but it can be easily modified to expand coverage area.
3. **Objectives and success criteria of the project:**
   The system should provide at least functionality to display list of restaurant and show their information. In addition, basic functionality for booking must be supported such as creation, modification, confirmation and deletion.
4. **References and overview:**
   The system displays the restaurants in a table as list, while in problem statement they were on a map. Information of restaurants is displayed on new scene. Reservation process is different, the system displays timetable for selected date from calendar and a field to write the amount of people. Calendar event isn’t supported by the system.

Current System

The system allows user to select six types of restaurants: Italian, Indian, French, Chinese, Japanese and Turkish. It can display up to 20 restaurants for each type in a table. Table contains restaurant name, location and rating, each parameter can be sorted in ascending or descending orders. User can select one restaurant and the system will display its detailed information. From details window user is able to click on booking button and make a reservation. Only basic actions are supported for booking, calendar event and remainders are not implemented.

Proposed System

1. Functional Requirements:

1.1 **Search for restaurants:** The user can search for restaurants on a list and on a map that displays up to 50 restaurants.

1.2 **See restaurant details:** The user can see pictures, ratings and comments of the restaurant as well as opening times and a link to website.

1.3 **Filter search results:** He can filter the results by the restaurant type, the prize category, by distance around a certain location, by the average rating and by free time slots for reservations for specified dates and number of visitors.

1.4 **Reserve table:** A user can see the times when he can reserve a table in the chosen restaurant. After clicking on the time, the user sees an overview of all tables in the restaurant. He can choose the exact table

the free one in the overview and thus reserve the table for the specified number of visitors.

1.5 **Save calendar event:** When the user reserves a table, an event in the local calendar is created for the reservation.

1.6 **Confirm reservation:** A user is reminded about a reservation one day before the actual date of the reservation and must confirm it until latest 12 hours before the actual date. If the user does not confirm, his reservation is cancelled automatically.

1.7 **Cancel reservation:** A user can cancel his reservation at any time up to two twelve hours before the actual date of the reservation. After the confirmation (see 2.5), the user cannot cancel the reservation anymore.

2. Nonfunctional requirements:

2.1 **Usability:** The system should be intuitive to use, and the user interface should be easy to understand. Simple interactions should be completed in less than three clicks. Complex interactions should be completed in less than six clicks.

2.2 **Conformance to guidelines:** The design of the system should conform to the typical usability guidelines such as Nielsen’s usability heuristics.

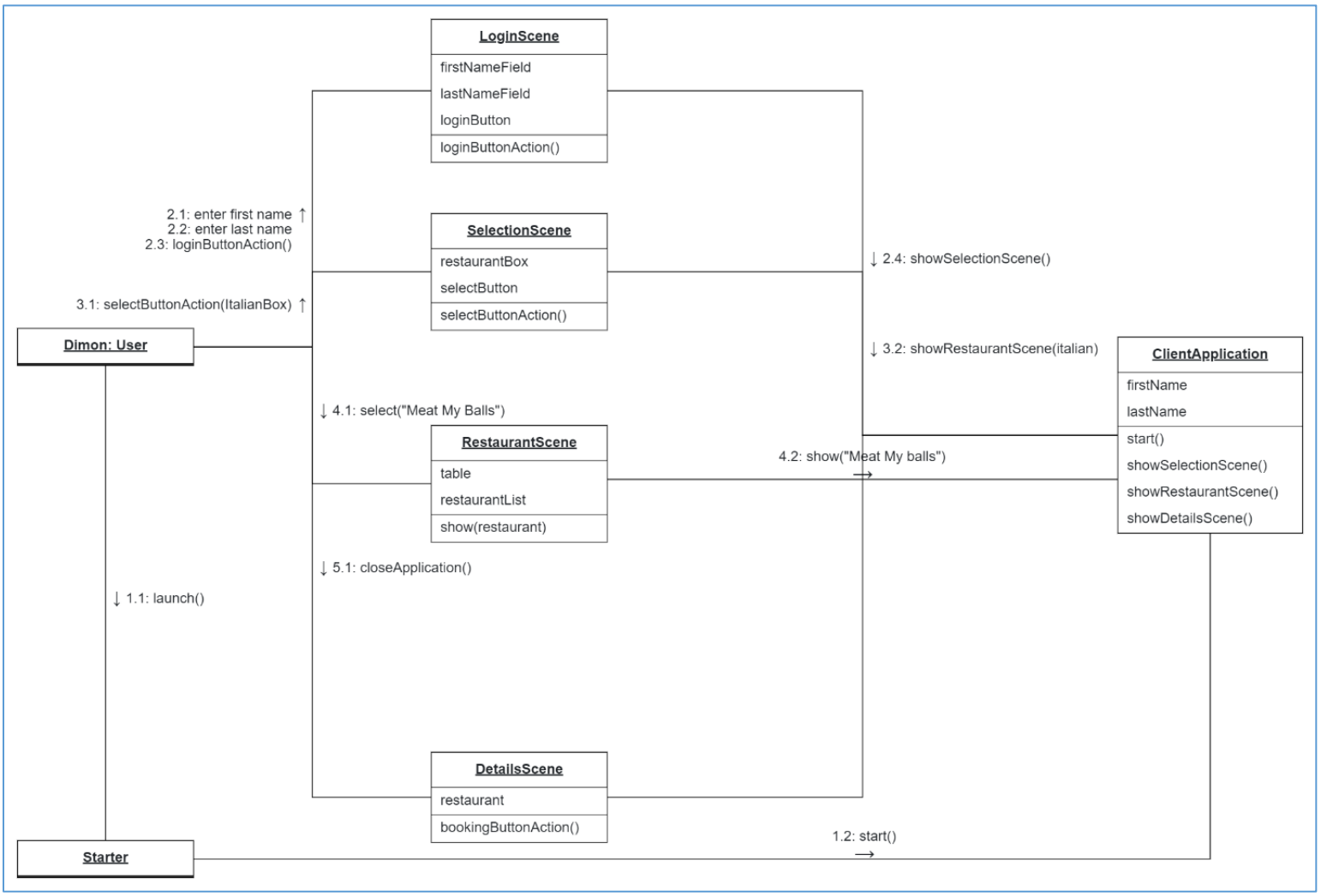
2.3 **Server system:** A server subsystem with a couple of services must be used in the system.

3. System Models:

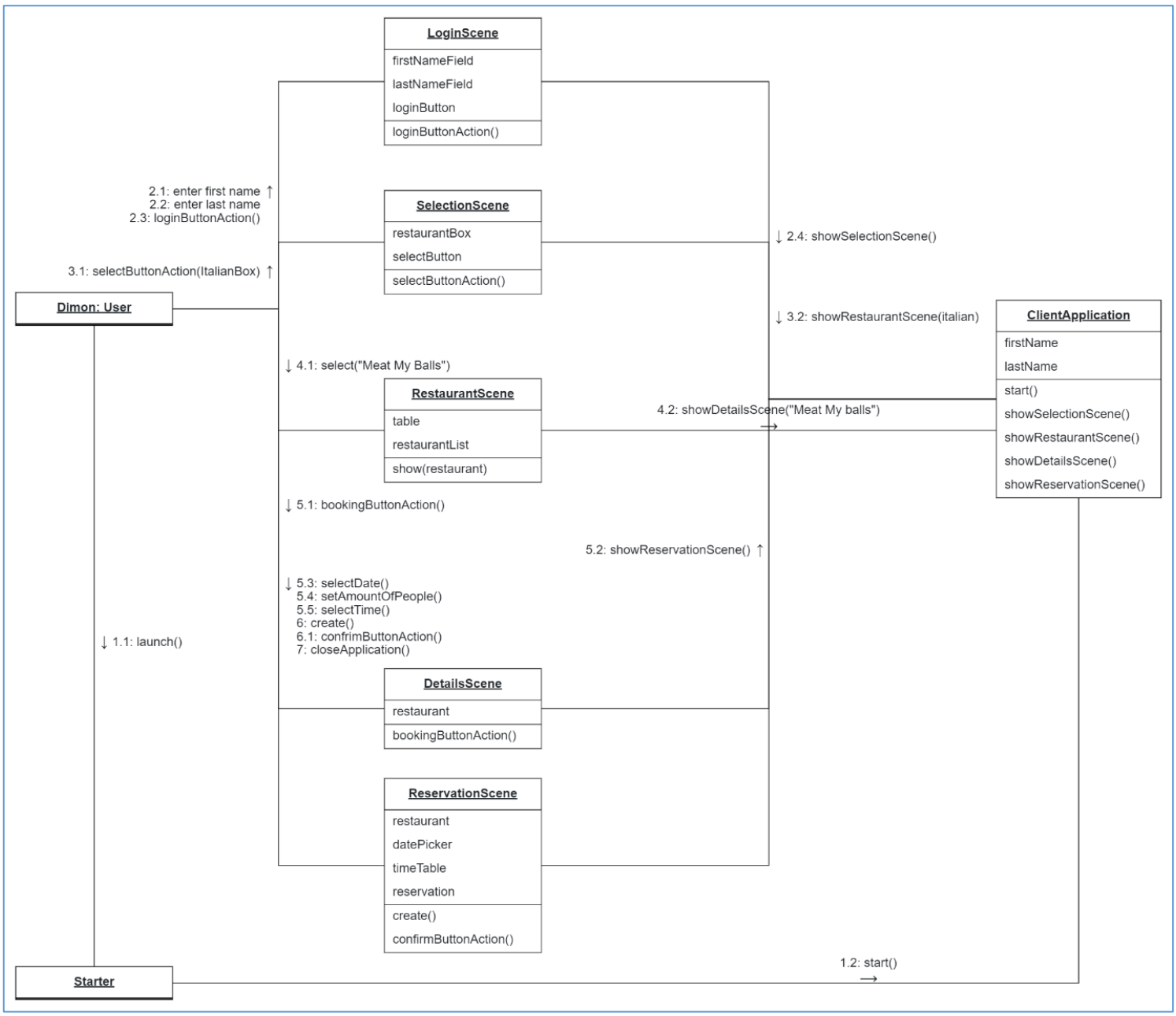
3.1 Scenarios and Communicational Diagrams:

3.1.1 **First scenario:** A user named Dimon opens the Restaurant Advisor application. He

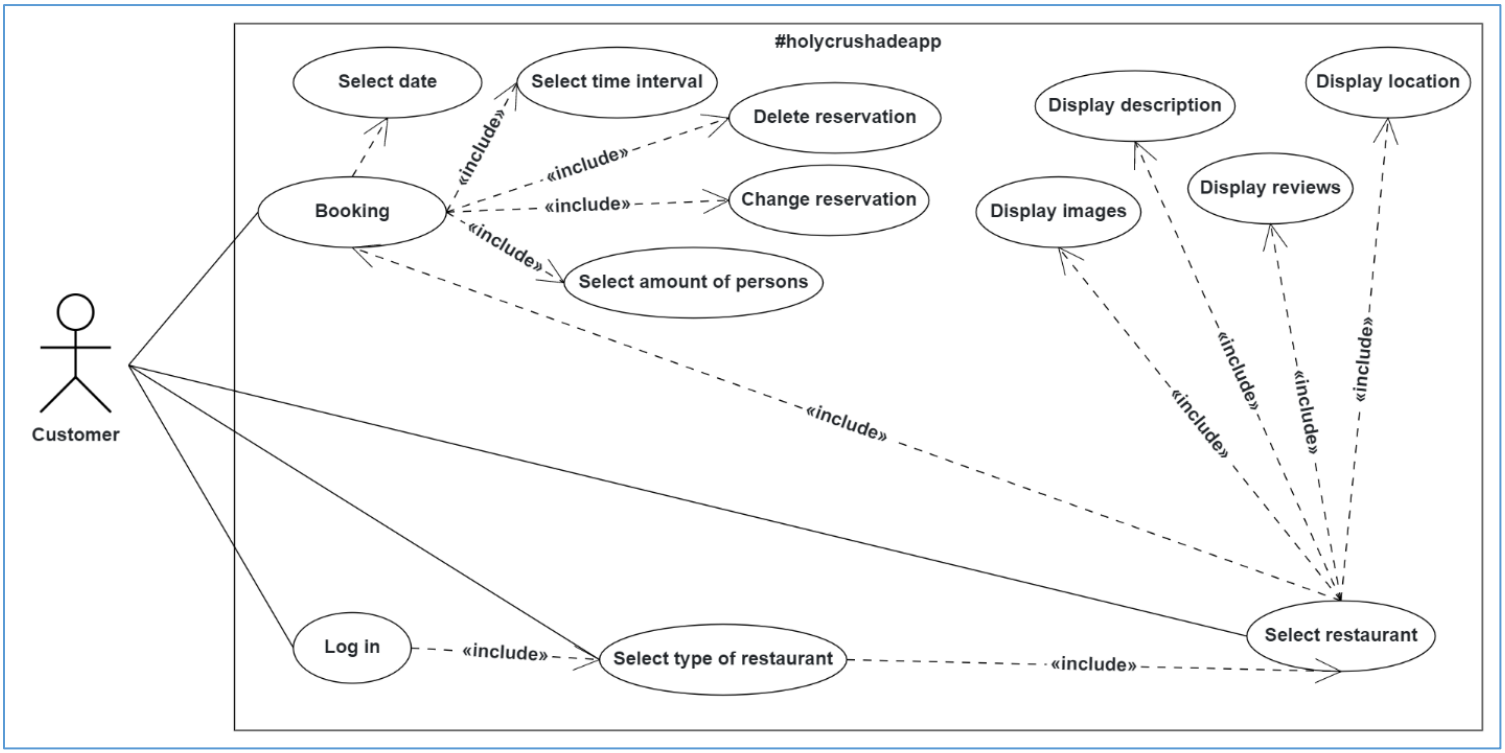
writes his full name in the login window and clicks the login button. Next, he sees 6 tiles which represent different restaurant types. He decides to select Italian restaurants and clicks the appropriate button. Then a table with different Italian restaurants is displayed. There is one with an interesting name: "Meat My Balls". He immediately selects this one, to see its information. After a short loading, a new window with all the restaurants' information is displayed. Dimon sees on the map that he is living close to this restaurant, so he can go in "Meat My Balls" with his girlfriend. He plans to call her later and closes the application.



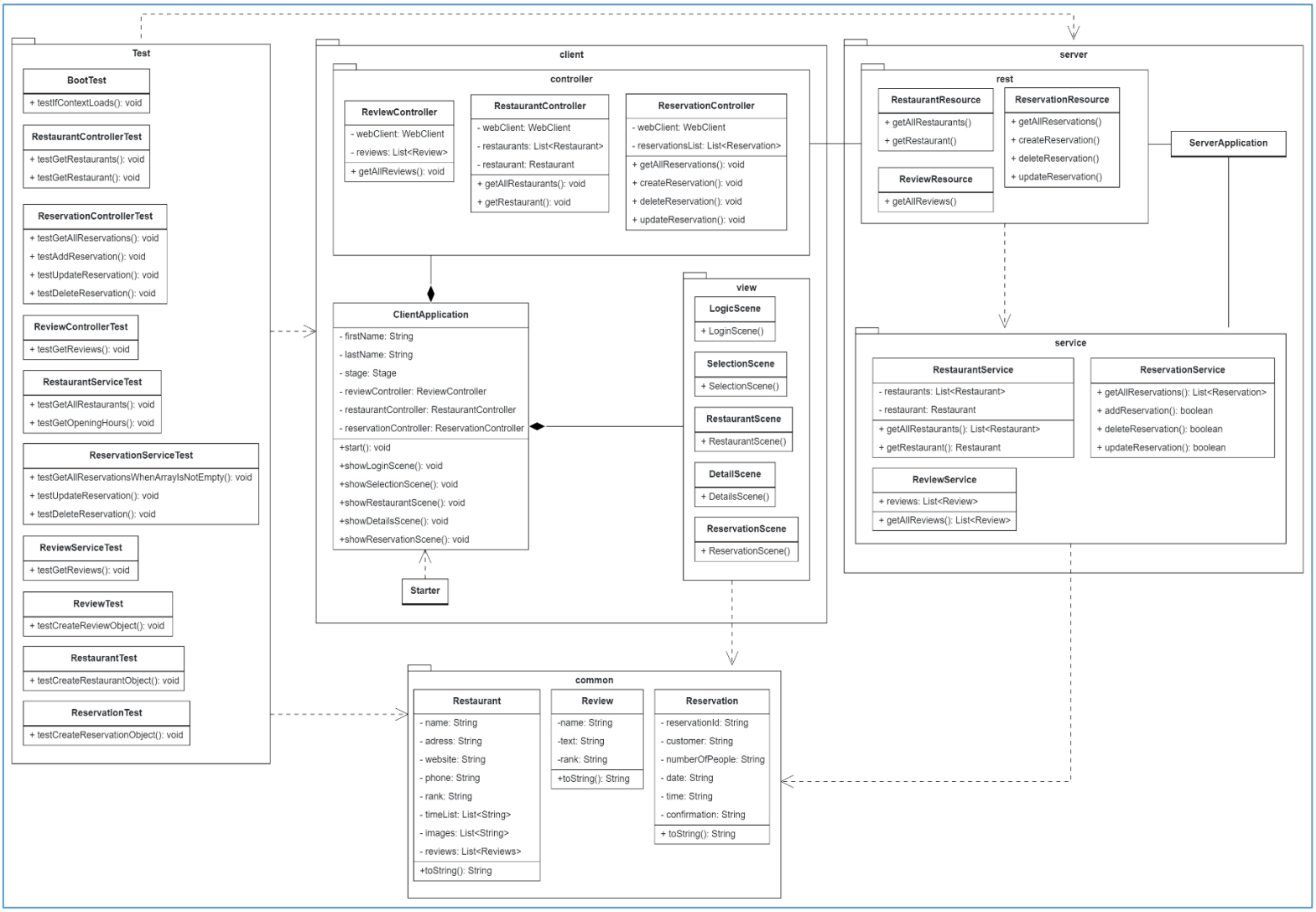
3.1.2 **Second scenario:** A user named Timor opens the Restaurant Advisor application. He writes his full name in the login window and clicks the login button. Next, he sees 6 tiles which represent different restaurant types. He decides to select Italian restaurants and clicks the appropriate button. Then a table with different Italian restaurants is displayed. There is one with an interesting name: "Meat My Balls". He immediately selects this one, to see its information. After some time spent loading, a new window with all the restaurants' information is displayed. Timor decides to make a booking; he clicks on the respective button and a reservation window is opened. He selects the desired date, writes amount of people, and selects a time from a time table for his reservation. Timor clicks the create the reservation. It is displayed in the right window of the application. Then he wants to confirm his reservation, so he clicks "Confirm" button. After confirmation button becomes inactive. Since the reservation process is complete, Timor closes the application.


LoginScene
firstNameField
lastNameField
loginButton
loginButtonAction()
2.1: enter first name ↑
2.2: enter last name
2.3: loginButtonAction()
SelectionScene
restaurantBox
selectButton
selectButtonAction()
↓ 2.4: showSelectionScene()
3.1: selectButtonAction(ItalianBox) ↑
Dimon: User
↓ 3.2: showRestaurantScene(italian)
ClientApplication
firstName
lastName
start()
showSelectionScene()
showRestaurantScene()
showDetailsScene()
showReservationScene()
↓ 4.1: select("Meat My Balls")
RestaurantScene
table
restaurantList
show(restaurant)
4.2: showDetailsScene("Meat My balls") →
↓ 5.1: bookingButtonAction()
5.2: showReservationScene() ↑
↓ 5.3: selectDate()
5.4: setAmountOfPeople()
5.5: selectTime()
6: create()
6.1: confrimButtonAction()
7: closeApplication()
↓ 1.1: launch()
DetailsScene
restaurant
bookingButtonAction()
ReservationScene
restaurant
datePicker
timeTable
reservation
create()
confirmButtonAction()
1.2: start() →
Starter

3.2 Use Case Diagram:



3.3 Class Diagram:



3.4 Component Diagram: