POWER ENJOY

DD Presentation
Politecnico di Milano
Software Engineering 2

Simone Bruzzechesse

Luca Franceschetti
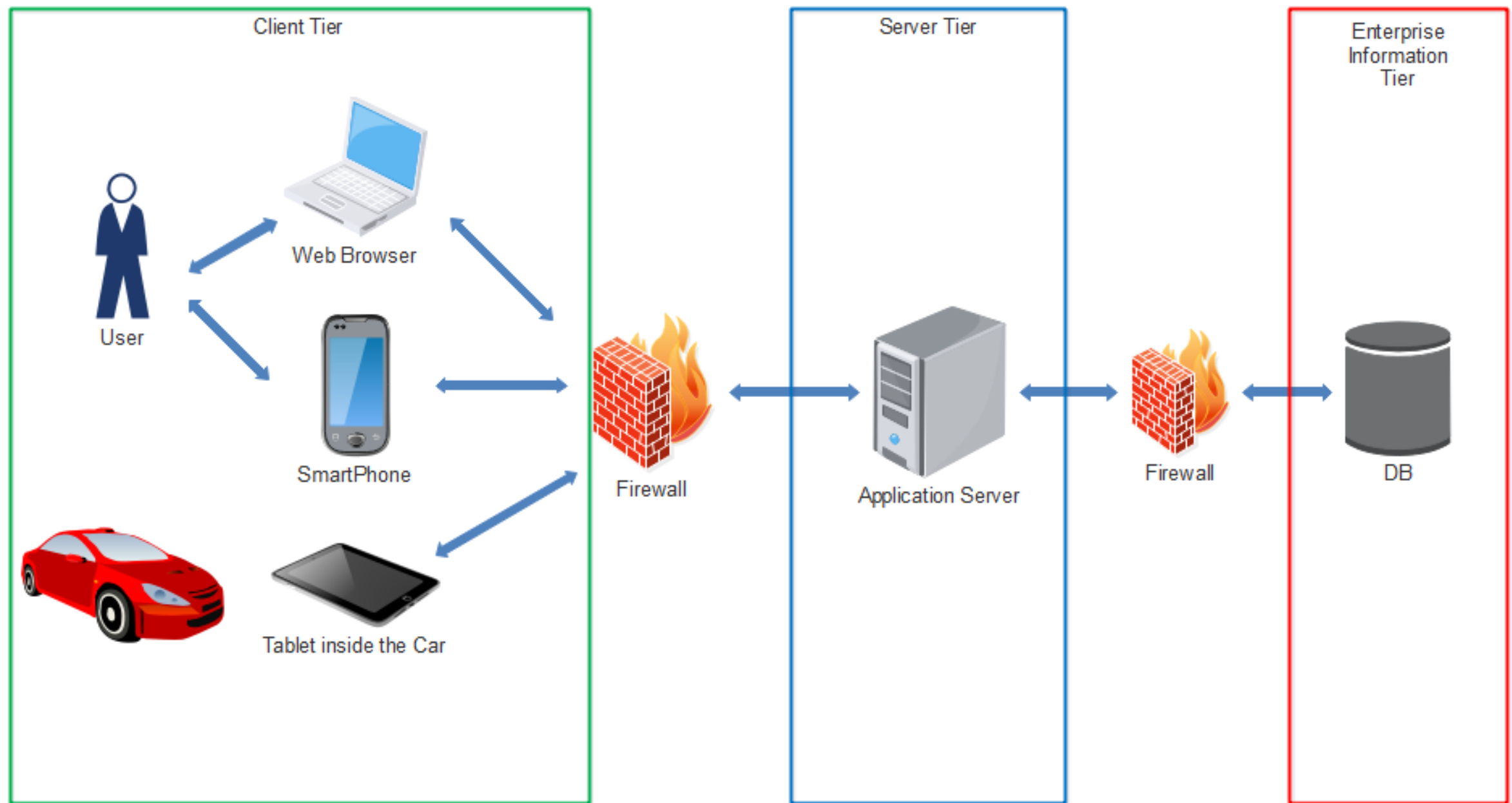
Gian Giacomo Gatti

INDEX

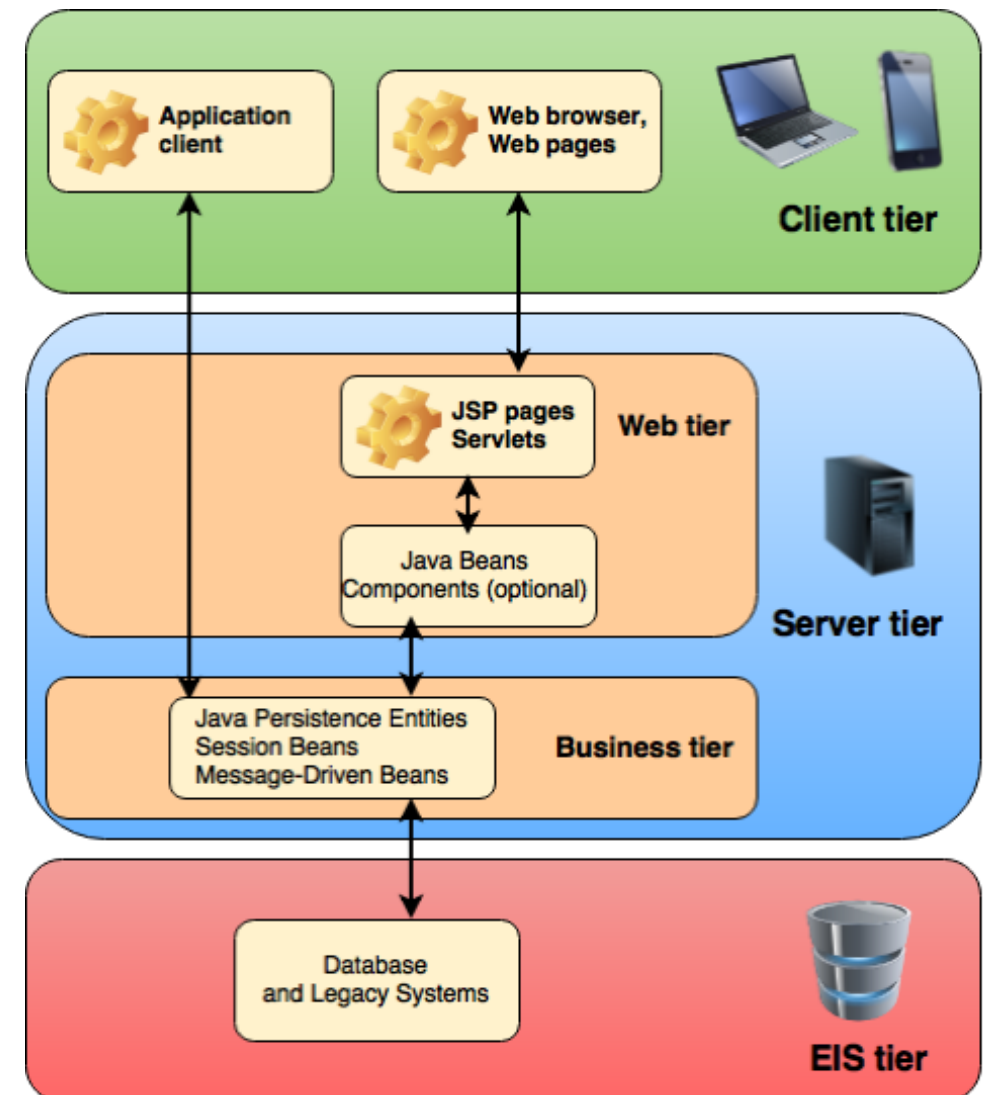The purpose of Design Document is to identify:

- ➤ High level infrastructure

- ➤ Main components and their interfaces:

  - ➤ Component Diagram

  - ➤ Data Structure

  - ➤ Deployment View

- ➤ Runtime View

  - ➤ Sequence Diagram

  - ➤ Algorithm Design

- ➤ User Interface Design
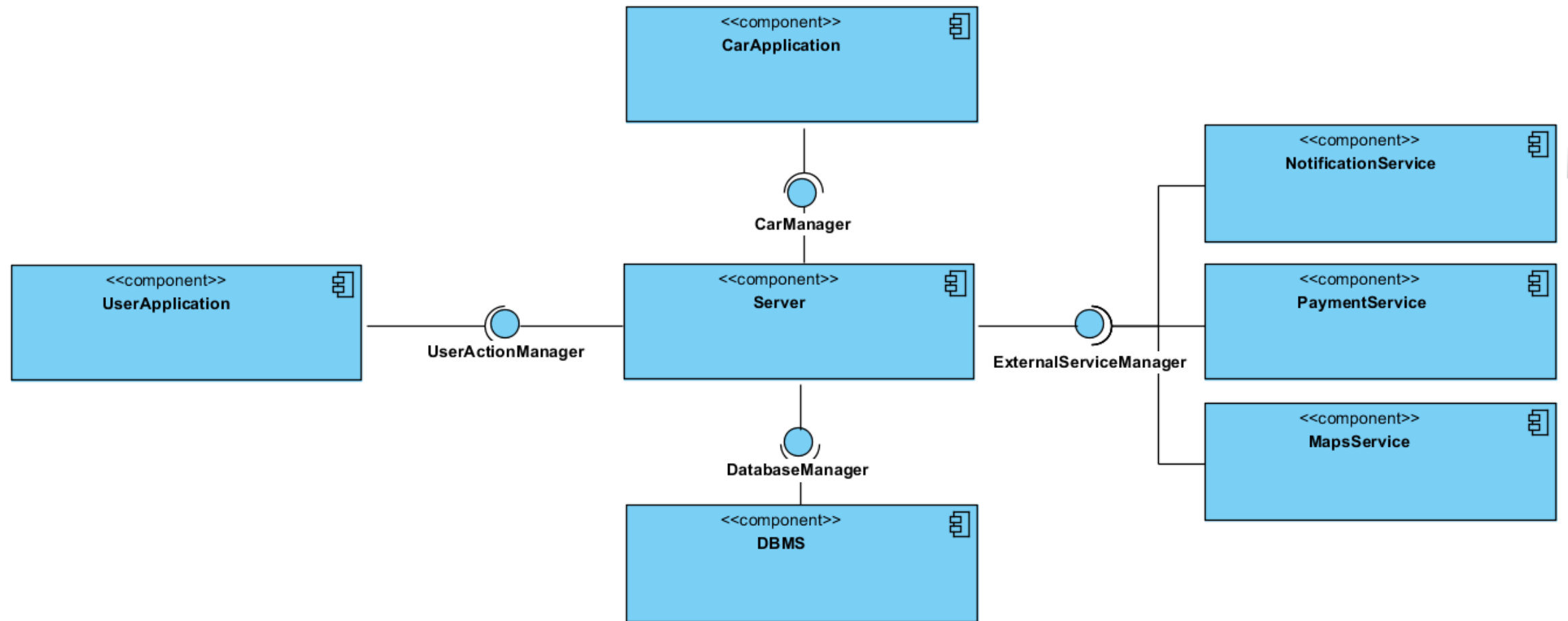
HIGH LEVEL INFRASTRUCTURE

J2EE INFRASTRUCTURE

➤ **Client-tier**: run on the client machine, both on the web browser and mobile application and also on the machine's tablet

➤ **Server-tier**:

  ➤ **Web-tier:** Servlets and JSP that are used to manage the interaction with the web application

  ➤ **Business-tier**: these components manage the internal logic of the system

➤ **Enterprise Information System (EIS)-tie**: handle enterprise infrastructure systems, database systems and legacy systems

MAIN COMPONENT DIAGRAM

COMPONENT DIAGRAM

DATA STRUCTURE

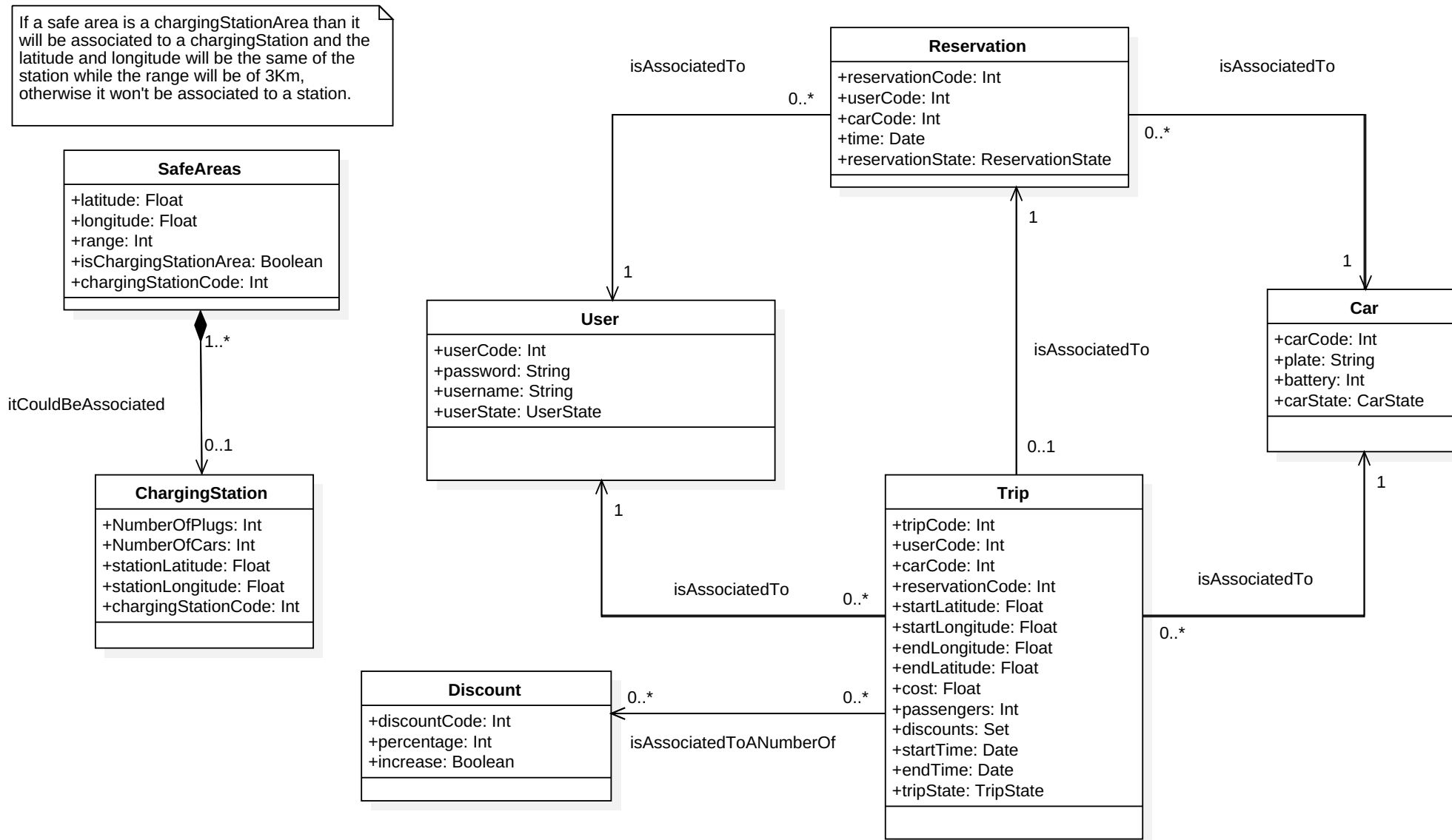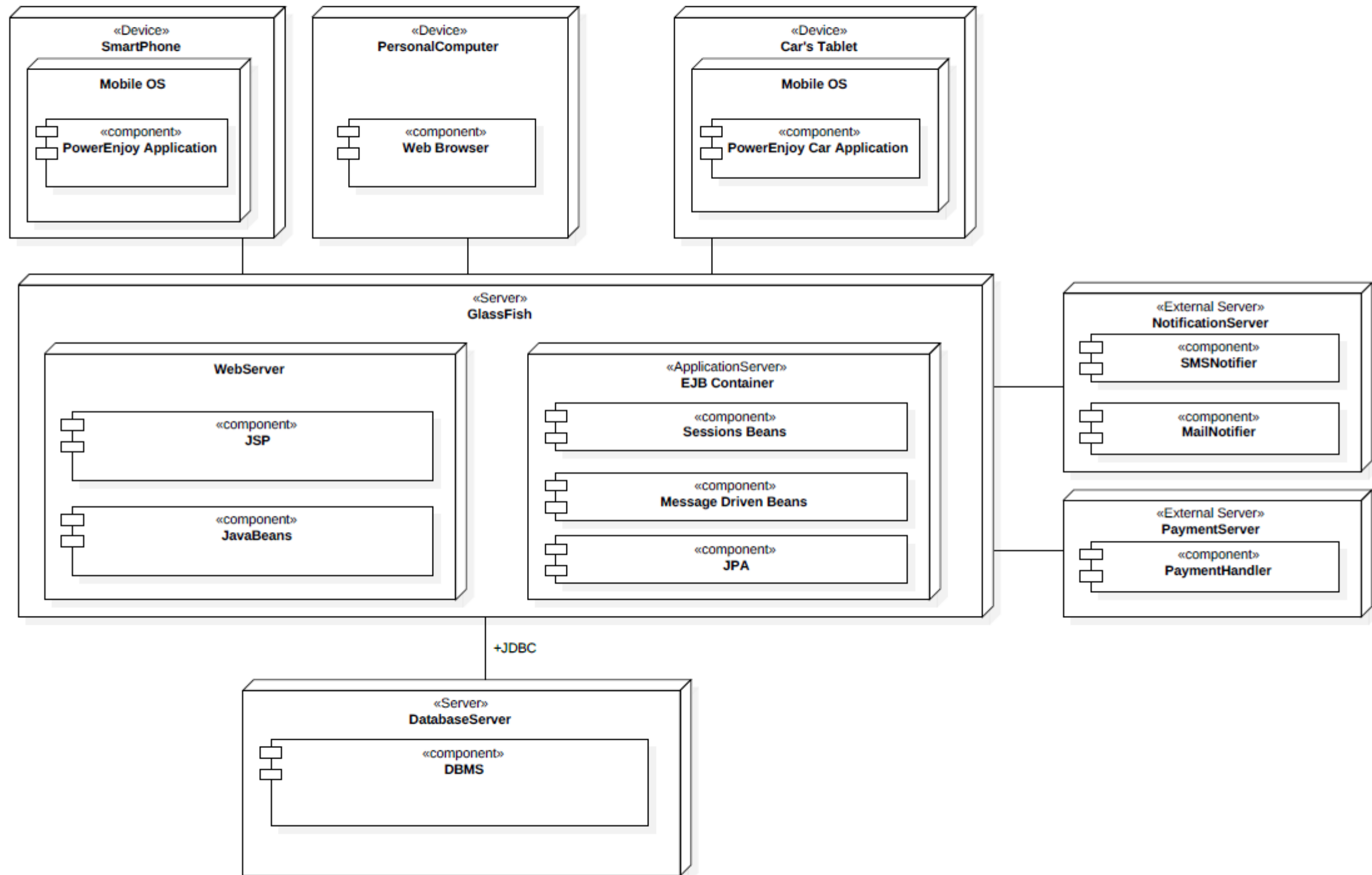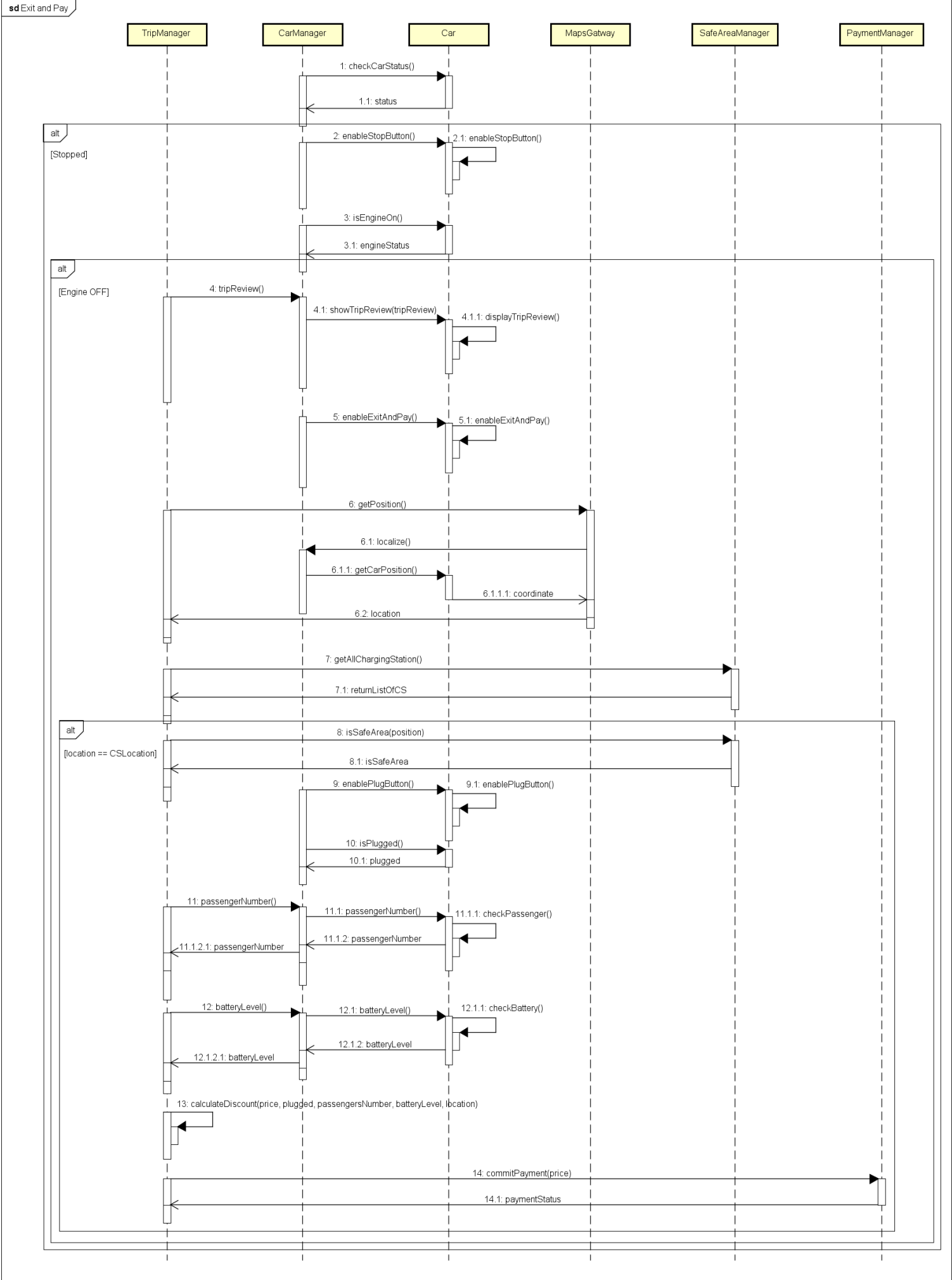If a safe area is a chargingStationArea than it will be associated to a chargingStation and the latitude and longitude will be the same of the station while the range will be of 3Km, otherwise it won't be associated to a station.

SafeAreas

+latitude: Float
+longitude: Float
+range: Int
+isChargingStationArea: Boolean
+chargingStationCode: Int

1..*

itCouldBeAssociated

0..1

ChargingStation

+NumberOfPlugs: Int
+NumberOfCars: Int
+stationLatitude: Float
+stationLongitude: Float
+chargingStationCode: Int

isAssociatedTo

Reservation

+reservationCode: Int
+userCode: Int
+carCode: Int
+time: Date
+reservationState: ReservationState

0..*

isAssociatedTo

0..*

1

1

User

+userCode: Int
+password: String
+username: String
+userState: UserState

1

isAssociatedTo

0..1

isAssociatedTo

0..*

Car

+carCode: Int
+plate: String
+battery: Int
+carState: CarState

1

1

Trip

+tripCode: Int
+userCode: Int
+carCode: Int
+reservationCode: Int
+startLatitude: Float
+startLongitude: Float
+endLongitude: Float
+endLatitude: Float
+cost: Float
+passengers: Int
+discounts: Set
+startTime: Date
+endTime: Date
+tripState: TripState

isAssociatedTo

0..*

Discount

+discountCode: Int
+percentage: Int
+increase: Boolean

0..*   0..*

isAssociatedToANumberOf

DEPLOYMENT VIEW

sd Money Saving Option

| Trip Manager | MapsGateway | CarManager | Car | SafeAreaManager | User |

1: localize(address)

2: addressConversion(address)

3: returnLocation

alt

[Wrong Location]

4: showError(text)

5: showOnDisplay(text)

6: displayNotification(text)

[Right Location]

7: getAllChargingStation()

8: returnListofCS

9: CheckCSinRange(listOfCS)

10: checkCSinRange(cs, location, range)

alt

[No charging station in range]

11: showError(text)

12: showOnDisplay(text)

13: displayNotification(text)

[Possible safe area founded]

14: calculateBestSafeArea()

15: getDestination(address)

16: getDestination(coordinate)

17: showOnDisplay(destination)

18: displayNotification(destination)

alt

[User Accept]

19: acceptButton

20: accemptanceAknoledge

21: initiateTrip

[User Decline]

22: declineButton

23: declineAknlodge

24: retryWithNewSolution

ALGORITHM DESIGN – EXIT AND PAY

This second algorithm has the aim to explain the conclusion of the rent.

- ✓ There are a lot of possible **alternatives** at the end of the rent

- ✓ We should check more details in order to guarantee coherence between the **trip** and the **payment**

- ✓ The algorithm have to check the **status** of the car (if it stopped or not)

- ✓ The algorithm have to check if the user asks to **stop** its trip

- ✓ As soon as the engine is **stopped**, the system stops **charging** the user

- ✓ Car's tablet shows the **review** and the details of the trip

- ✓ The user can now touch the **"Exit and Pay"** button on the display

- ✓ The algorithm saves **current location** of the car thanks to an interaction between CarManager and MapsGateway

- ✓ The system **downloads** all the charging station (justified as update)

- ✓ **CarManager** enables the possibility to plug the car as well as enables "plug" button only if the users stops near a charging station

- ✓ **TripManager** calculates right discount of the trip

- ✓ **PaymentManager** tries to commit the payment, and the system must check if it successful

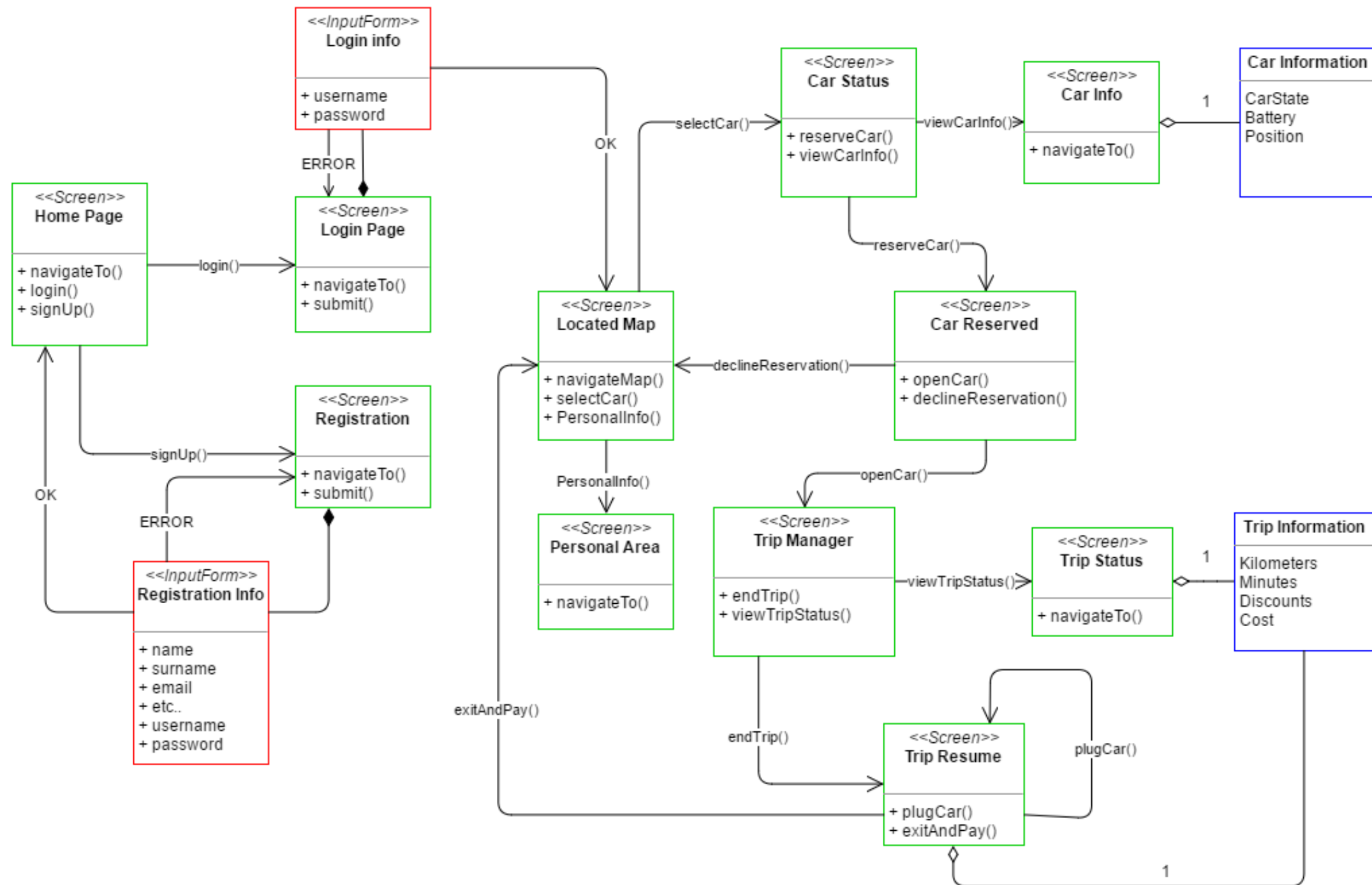# ALGORITHM DESIGN – MONEY SAVING OPTION

This first algorithm manages the "Money Saving" option:

✓ We created a **specific class** for the **location**, to provide coherent and consistent information about car's and charging station's position.

✓ This function receives an **address**, the one that user has typed on the car's display, and a range, also typed on display.

✓ **CarManager** permits to interact with the display and consequentially with the user in the car.

✓ We created a specific class for the charging station, which are provided by **SafeAreaManager** during the computation.

✓ **MapsGateway** is a third part service and can calculate if a specific charging station is or not in range, from the location desired by the user.

✓ Before the user can see the result on display, the system **checks** which charging station **has less car** and advices its position to the user.

USER INTERFACE DESIGN
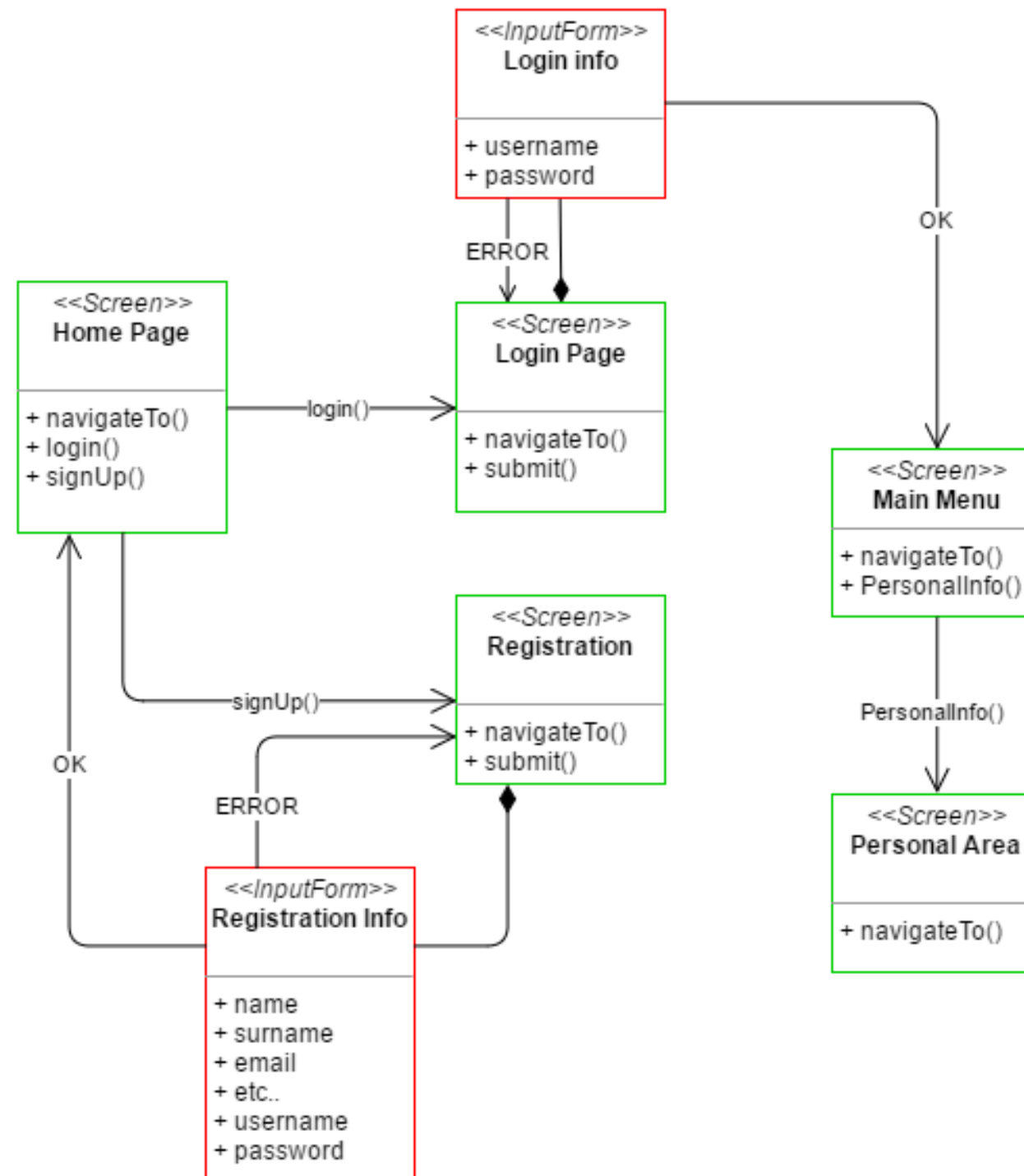
USER INTERFACE DESIGN

THANKS FOR YOUR ATTENTION