SOFTWARE ENGINEERING 2 PROJECT

PRESENTATION

RUARO NICOLA

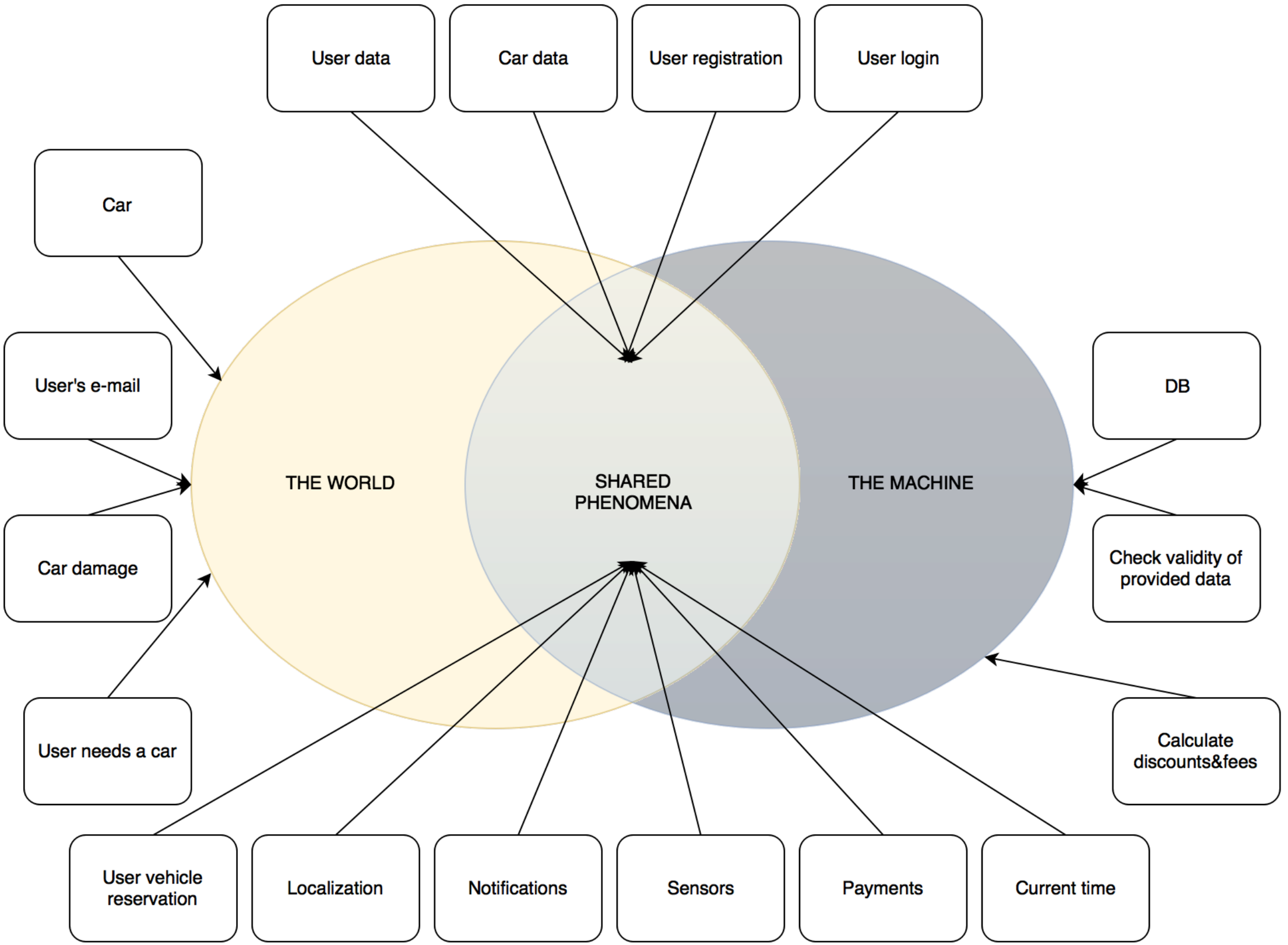
GREGORI GIACOMO

POWERENJOY

- The purpose of this project is to develop a digital management system called PowerEnJoy.
  PowerEnJoy is a car-sharing service which uses only electric-cars and allows the users to easily find a car and to use it.

REQUIREMENTS ANALYSIS AND SPECIFICATION DOCUMENT

RASD


User data
Car data
User registration
User login
Car
User's e-mail
Car damage
User needs a car
DB
Check validity of provided data
Calculate discounts&fees
THE WORLD
SHARED PHENOMENA
THE MACHINE
User vehicle reservation
Localization
Notifications
Sensors
Payments
Current time

IMPORTANT ASSUMPTIONS

- The system knows the battery level, location and number of passengers in each car
- Users driving licenses are checked
- The vehicle is driven by the user that reserved it
- If a vehicle is charging it’s in a charging station
- There are no time or distance limits for a single ride
- During a ride the number of passengers in a vehicle doesn’t change
- The user starts using an unlocked vehicle after a short amount of time
- There are precedence rules regarding the fee/discount application

BRIEF INTERFACES EXPLANATION

- The PowerEnJoy system interacts with an external system offering **maintenance** services: when a problem with a vehicle is detected (eg. engine fault) a PowerEnJoy employee working for the customer care department signals the problem to the external system which notifies back when the problem is solved.
- The notifications email are handled using the Open-Source **JavaMail API**.
- The payments are handled using the **Stripe API**, that gives us a secure and fast way to transfer money.
- To provide location services we use the **W3C Geolocation API** combined with the **GeoNames API** for reverse geocoding. As well described on the producer's website, this combination gives us an accurate way to retrieve the user position for any location on Earth.

SYSTEM GOALS → FUNCTIONS

[G1] Registration

[G2] Log-in and session management

[G3] Account management

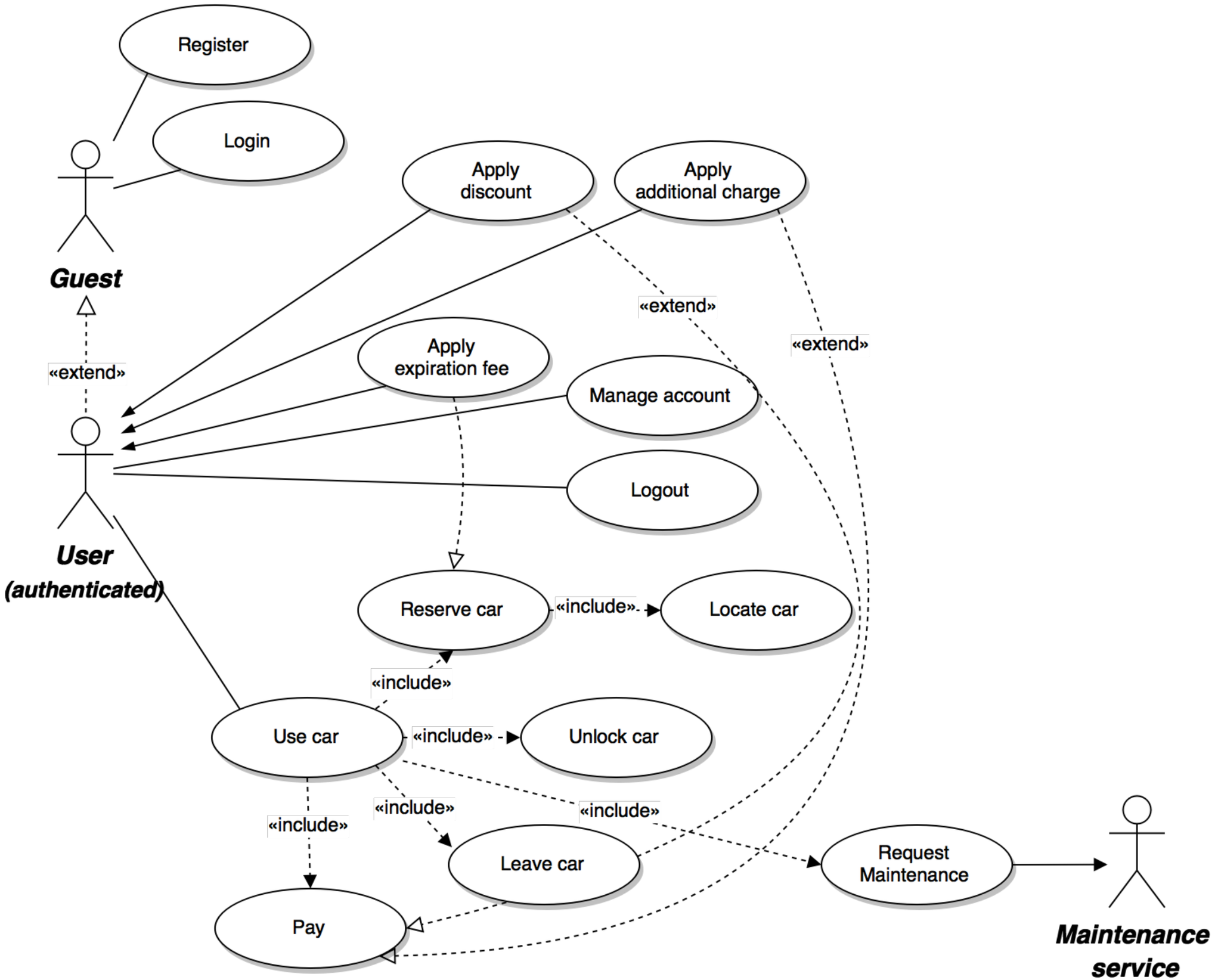
[G4] Car localisation

[G5] Reservation and reservation management

[G6] Car un-locking and end ride

[G7] Payments and charges application

Use-case	Actor	Brief description	Goal
1	Guest	Registration	G1
2	Guest	Log-in	G2
3	User	Manage account	G3
4	User	Create reservation	G4,G5
5	User	Delete reservation	G5
6	User	Reservation expiration	G5
7	User	Use car	G4,G6
8	User	Charge ride	G7
9	User	Discounts & fees	G7

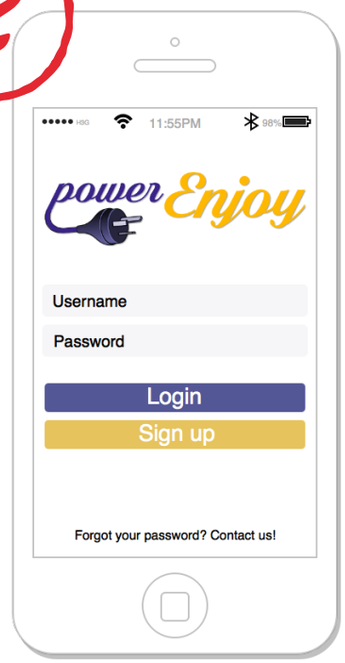

Register
Login
Guest
Apply discount
Apply additional charge
«extend»
«extend»
Apply expiration fee
«extend»
Manage account
Logout
User (authenticated)
Reserve car
«include»
Locate car
«include»
Use car
«include»
Unlock car
«include»
«include»
«include»
Leave car
Request Maintenance
Pay
Maintenance service

DESIGN DOCUMENT

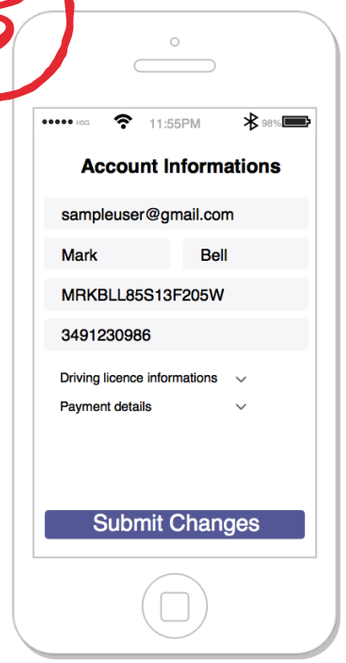
DD

MOST RELEVANT MOCKUPS

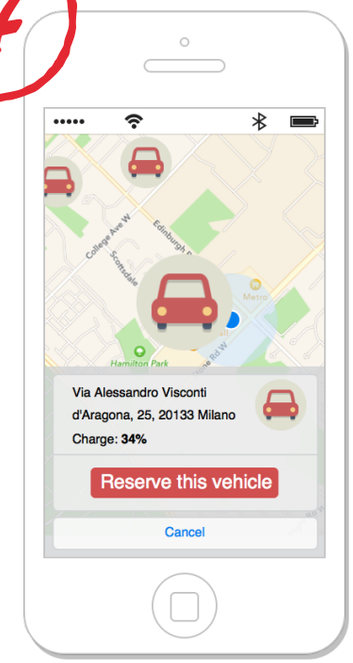
2



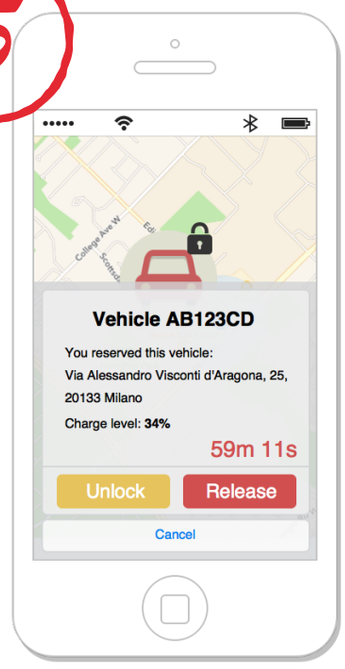
3



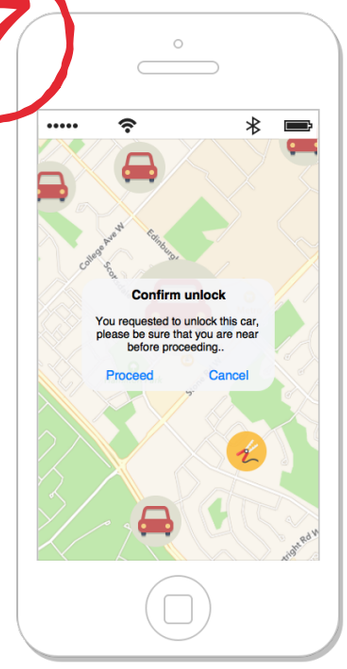
4



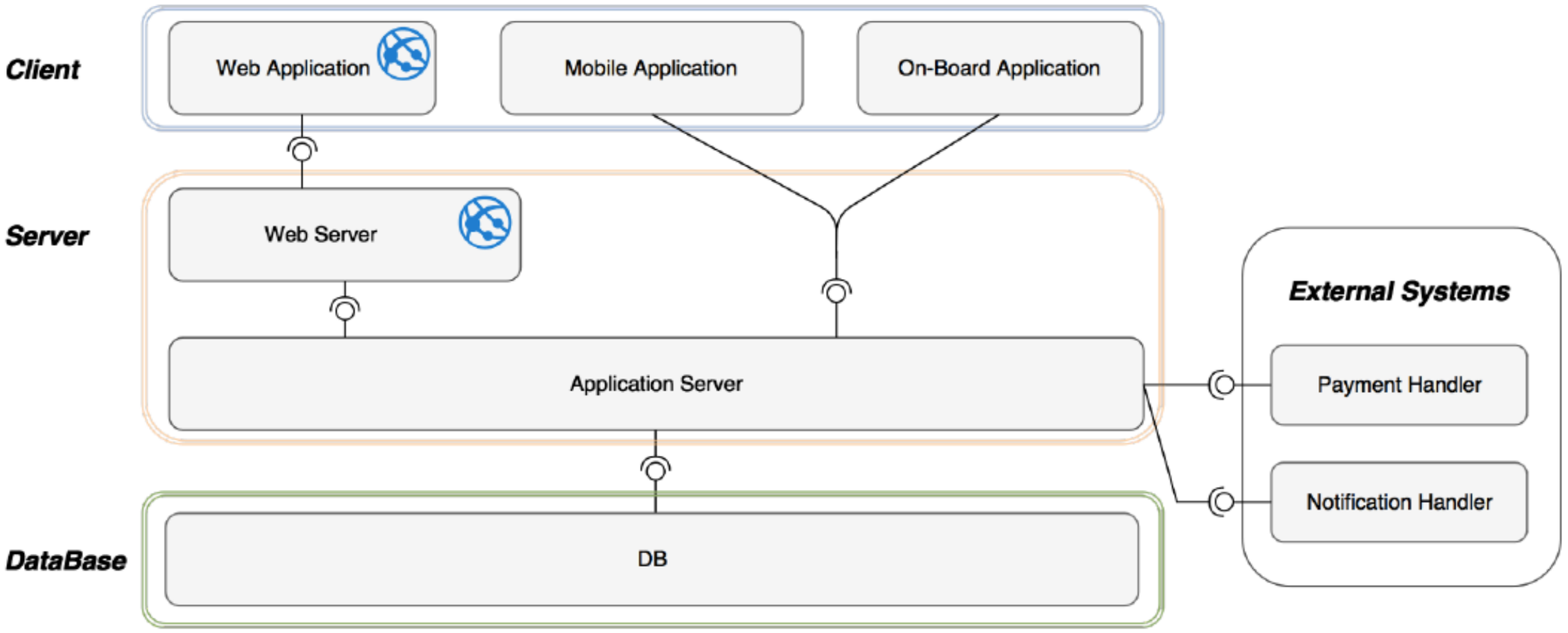
5



7

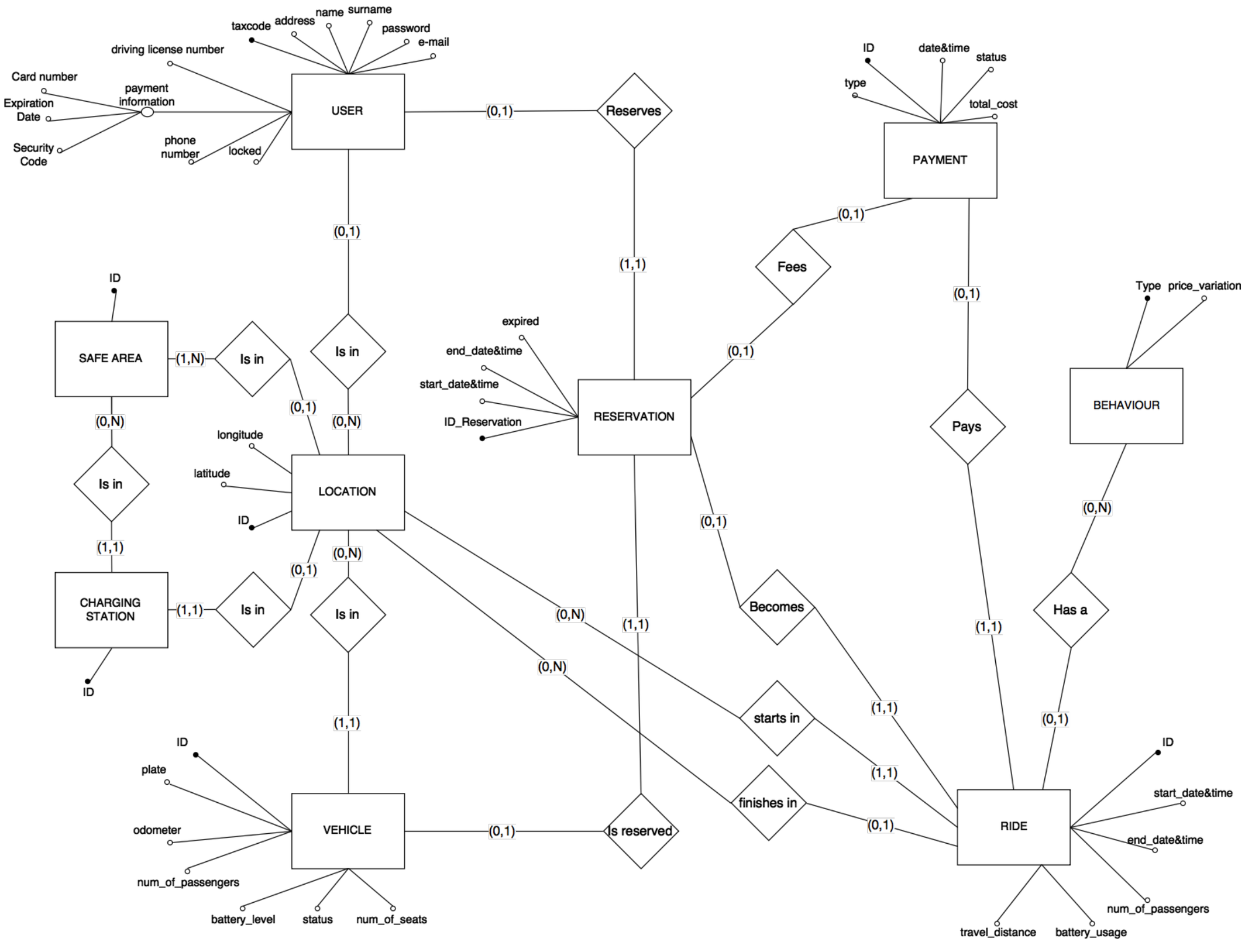


SYSTEM ARCHITECTURE

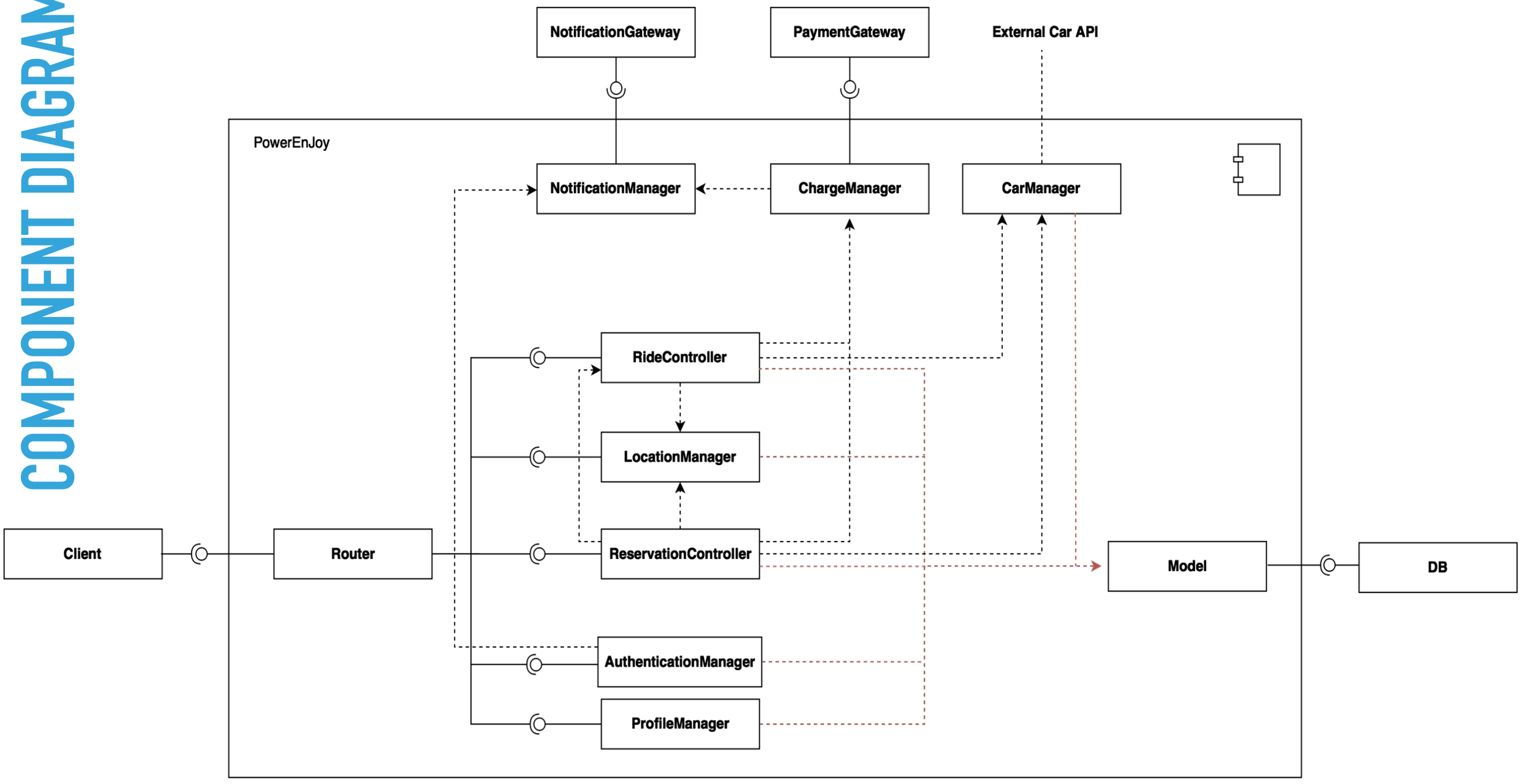


DESIGN DOCUMENT

ER DIAGRAM



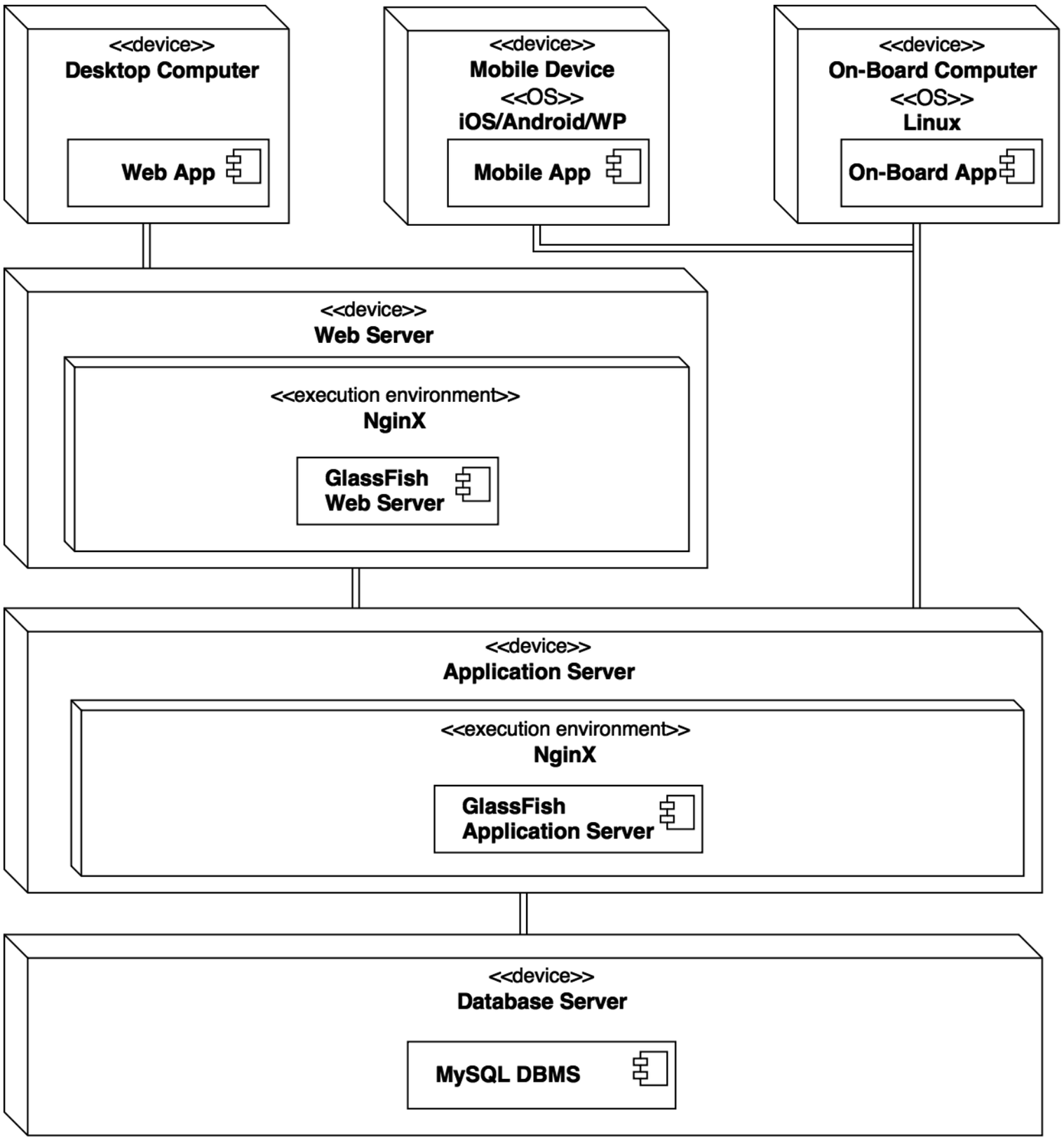
COMPONENT DIAGRAM



REQUIREMENTS TRACEABILITY

RASD Goals	RASD Functions	DD Component
G1, G2	Registration Login	**AuthenticationManager**
G3	Account Management	**ProfileManager**
G4	Create reservation	**LocationManager**
	Use car	
G5	Create reservation Delete reservation Reservation expiration	**ReservationController**
G6	Use car	**RideController**
G5, G6	Use car	**CarManager**
	Create reservation Delete reservation Reservation expiration	
G1, G7	Charge ride Registration	**NotificationManager**, **NotificationGateway**
G7	Charge ride Discounts & fees	**ChargeManager**, **PaymentGateway**

DEPLOYMENT DIAGRAM



INTEGRATION TESTING PLAN DOCUMENT

ITPD

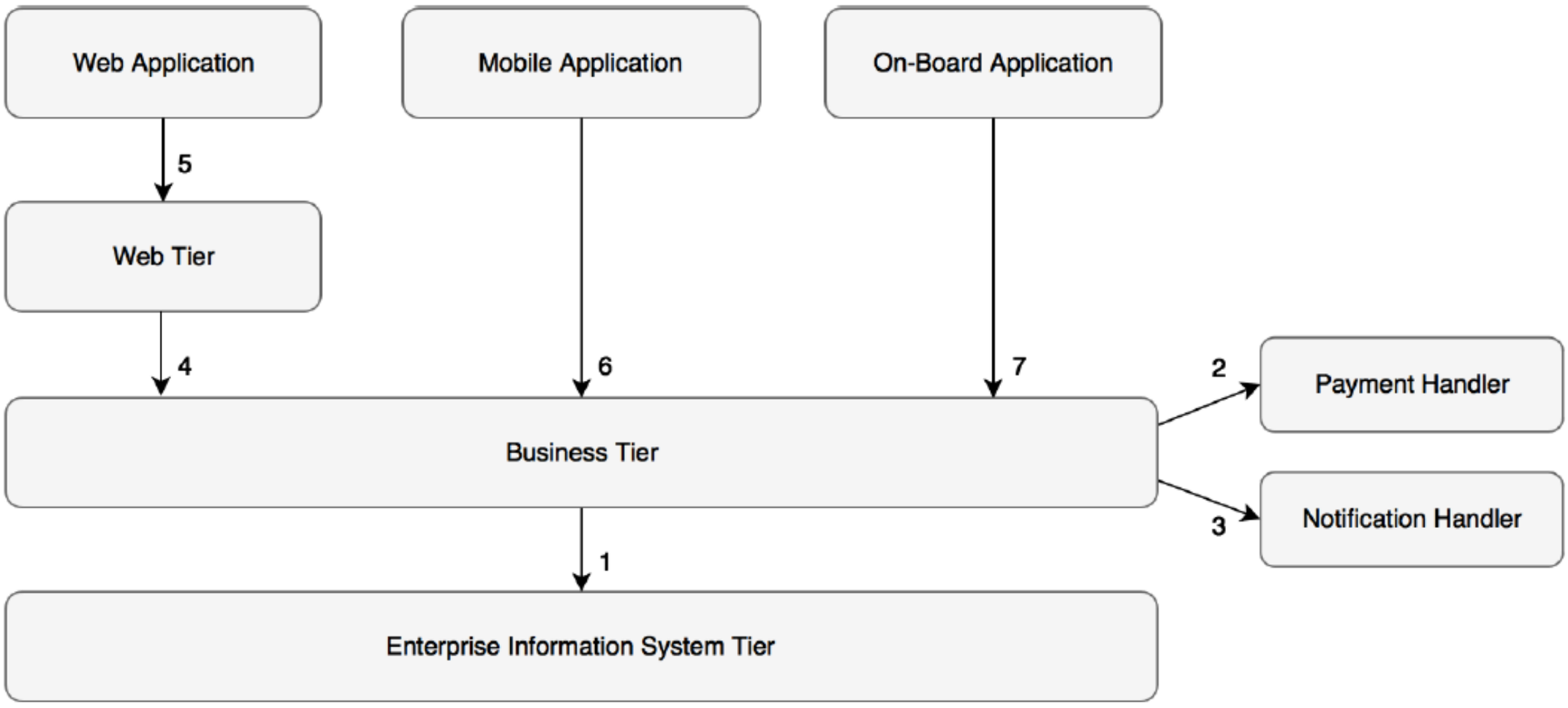
INTEGRATION STRATEGY DESCRIPTION

- We choose for our integration testing strategy to adopt a **bottom-up approach**. In this way, we test the subsystems from the lower level to the top level, where all the modules are integrated. In the testing phase the order of the subsystems to analyse is selected, not randomly but **privileging the critical ones**.

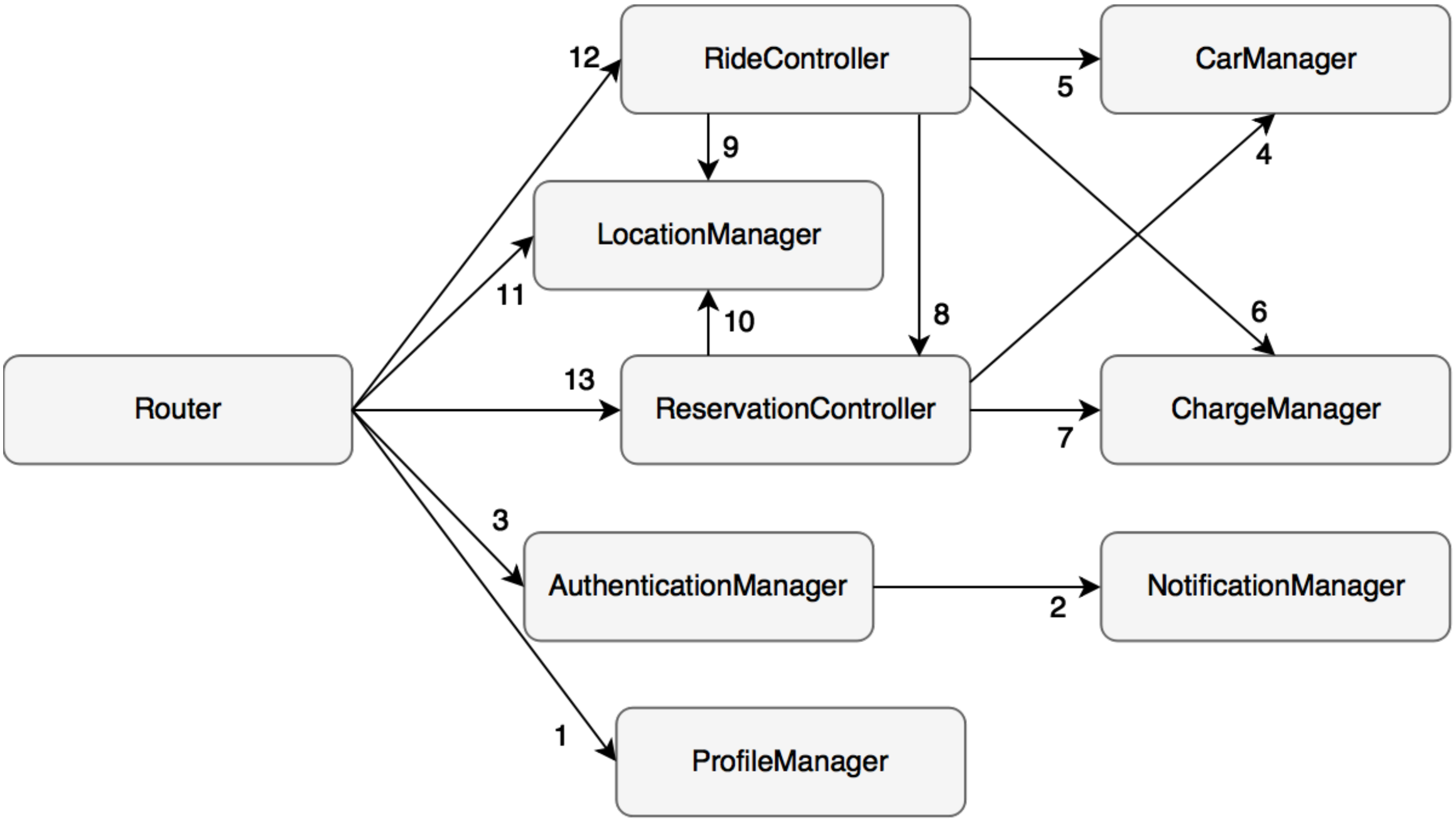
- **Advantages**: The test conditions for each module are easier to create and the test results can be analysed in a simpler way. Then it's easier to localise problems and faults. In the end we can proceed with the test phase of our subsystems alongside their implementation and a last evidence is the fact that probably almost all the faults occurs toward the bottom of the system.

- **Disadvantages**: On the other side the bottom-up approach brings some disadvantages. The main one is the need of driver programs in order to simulate the missing modules while they aren't already deployed. Another disadvantage is the fact that we can't test the whole program until the last module has been developed.

SUBSYSTEM INTEGRATION SEQUENCE



PRECEDENCE DIAGRAM



TOOLS USED

- **Arquillian** (*arquillian.org*) is a flexible and portable integration testing framework designed specifically for JEE. It will be used to test the **correct behaviour** of the containers and their **interaction with the system**.
- **JMeter** (*jmeter.apache.org*) is an open-source application developed and maintained by the Apache foundation. It is designed to load test functional behaviour and measure performance: in this project it will be used to achieve system and **non-functional requirements** testing.
- **JUnit** (*junit.org*) is probably the most common framework for Java unit testing. In this project, though, it will be used in combination with Arquillian and Mockito to achieve **High-level and integration testing**.
- **Mockito** (*site.mockito.org*) is a clean, simple and well supported Java mocking framework. It will be used to **verify the interactions** between objects and to **create stubs** when a component cannot be tested in isolation.

PROJECT PLANNING DOCUMENT

PPD

EFFORT ESTIMATION

- The Function Point Analysis and COCOMO II methods are used for the project complexity and effort estimation.
- The Effort Equation:

Effort = A x EAF x KSLOC^E where A = 2.94

- From the Function Points we obtain an estimation of the project complexity and total number of lines of code, SLOC.
- From the **Scale Drivers** we will obtain the value of **E**, while from the **Cost Drivers** we will obtain **EAF**.

FUNCTION POINTS

- We can proceed estimating the total number of lines of code: for JEE the **upper and lower-bound** conversion factors are, respectively, 46 and 67.

Type	Value
ILF	98
EIF	20
EI	34
EO	28
EQ	13
Total	193

- The computed lower-bound is:
  - SLOC = 193 x 46 = 8.878
- The computed lower-bound is:
  - SLOC = 193 x 67 = 12.931

COCOMO II

- For our project E = 0.91 + 0.01 x 14.18 = 1.052
- With the computed lower-bound we have:
  - Effort = 2.94 x 1.356 x 8.8781.052 = 40 PM
- with the upper:
  - Effort = 2.94 1.356 12.9311.052 = 59 PM

Cost Driver	Factor	Value
RELY	high	1.15
DATA	nominal	1.08
CPLX	high	1.15
RUSE	high	1.15
DOCU	nominal	1.00
TIME	nominal	1.00
STOR	nominal	1.00
PVOL	nominal	1.00
ACAP	high	0.86
PCAP	nominal	1.00
APEX	low	1.10
PLEX	low	1.10
LTEX	nominal	1.00
PCON	very high	0.65
TOOL	high	0.91
SITE	very high	0.86
SCED	nominal	1.00

EAF	1.356

Scale Factor	Factor	Value
Precedentedness	low	4.96
Development Flexibility	high	2.03
Architecture / Risk Resolution	high	1.41
Team Cohesion	very high	1.10
Process Maturity	nominal	4.68

TOTAL	14.18

SCHEDULE ESTIMATION

- To give an estimation of the duration of the project we used the following formula:
    - **Duration** = 3.67 x Effort^F
- where: F = 0.28 + 0.2(E − B) = 0.308
- Then, with the lower-bound we estimate a duration between 11.5 and 13 months.
- Finally we can estimate the number of team's components needed to complete the project:
    - **Members** = Effort/Duration = 3
- in the lower-bound case, while with the upper-bound the number is 5.

RISKS

- Project Risks

Risk	Probability	Impact
Milestone Delay	Average	Negligible
Knowledge Overestimation	Average	Critical
Code Loss	Low	Catastrophic
Lack of Communication	Average	Marginal

- Technical Risks

Risk	Probability	Effects
External Services	Low	Critical
Hardware Solutions	High	Critical

- Business Risks

Risk	Probability	Effects
Competitors Companies	High	Marginal
Intended Users	High	Critical
City Administration	Low	Marginal
National Legislation	Low	Marginal

THANK YOU

FOR YOUR ATTENTION