POLITECNICO DI MILANO

myTaxiService

Prof. Raffaela Mirandola
February 29, 2016

CHRISTIAN ZICHICHI (840565)
LUIGI MARROCCO (854884)

THE PROJECT → Big_City Taxi Service Optimization

- **MAKE THE ACCESS TO THE SERVICE EASIER FOR PASSENGERS**
- **FAIRNESS OF TAXI QUEUES**
- **WEB APPLICATION – MOBILE APPLICATIONS**
  - User's Side – Ride Requests/Reservations
  - Taxi driver's Side – Availability and Call Confirmations

GOALS

REQUIRED SYSTEM FUNCTIONALITIES

- **FAIR MANAGEMENT OF QUEUES OF AVAILABLE TAXIS**
  - Zones ←→ Queues
  - Availability Updates and Call Confirmations
  - Ride Rejection → Taxi moved to Last Position

- **CREATE AND MANAGE TAXI RIDE REQUESTS**
  - User: Form Filling (Web) / GPS Position (Mobile)
  - System: Pick a Taxi (How?)

- **CREATE AND MANAGE TAXI RIDE RESERVATIONS**
  - User: Form Filling
  - System: Pick a Taxi 10 Minutes before the Ride (How?)

- **POSSIBILITY TO EXTEND BASIC SERVICES AND ADD FUNCTIONS**
  - Application Programming Interfaces (APIs)

MAIN ASSUMPTIONS

- **BIG_CITY**
  - Division in fixed-size zones (2 km^2 each)
  - Origin and Destination of Taxi Ride

- **TAXIS**
  - Pre-registered Drivers and each Taxi has a unique ID Code
  - Embedded Android Auto OS Device
  - GPS correct
  - 10 minutes of Maximum Waiting Time
  - End of Ride

- **PASSENGERS**
  - Correct Data (GPS, Forms)
  - Honesty
  - Max N° People for a Ride
  - Cash Payments

PROPOSED SYSTEM AND ACTORS

CLIENT SIDE, managed by a BACK-END Centralized Server Application:

- **USER MOBILE APPLICATION:** Android, iOS, HTML/CSS/JS Hybrid Development
- **USER WEB APPLICATION:** Front-end similar to User Mobile App
- **TAXI MOBILE APPLICATION:** Android Auto

ACTORS

- **GUEST:** Not yet registered, signs up filling a form
- **USER:** Registered with Email and Password, has access to the service
- **TAXI DRIVER:** Employed and pre-registered by the government

FUNCTIONAL REQUIREMENTS

GUEST

- Sign up into the User Web Application or Mobile Application by filling up a form with Username and Password

USER

- Log in into the web application or mobile application with username and password
- Create a Ride Request
- Create a Ride Reservation
- Check the list of Booked Taxis
- Cancel a Ride Reservation

TAXI DRIVER

- Confirm a Ride
- Reject a Ride
- Update their Availability

SYSTEM

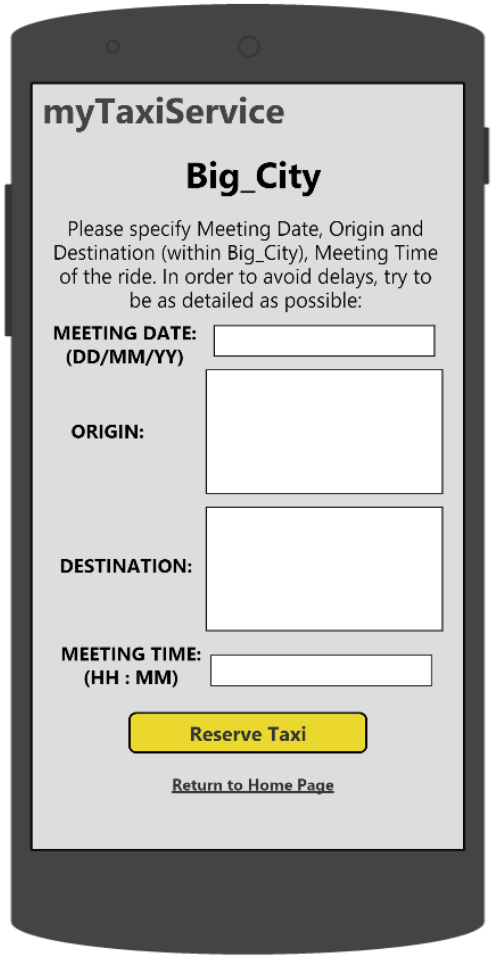
- Propose a Ride to taxi drivers
- Confirm Requests or Reservations to the User with related information
- Insert, remove or move a Taxi from a queue

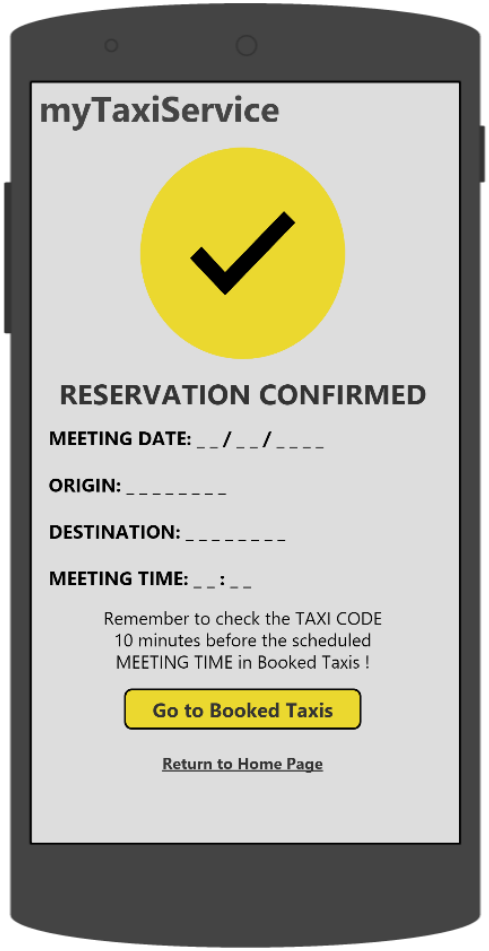
NON FUNCTIONAL REQUIREMENTS (1)

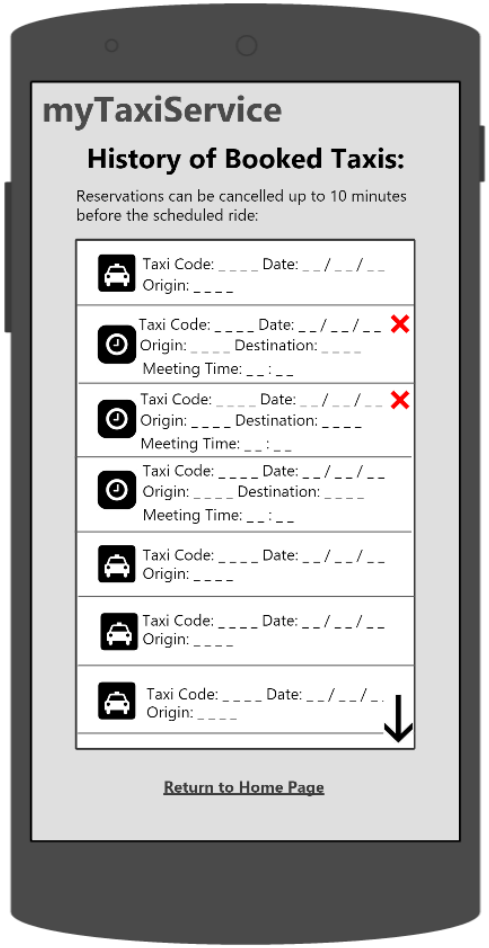
USER WEB APPLICATION INTERFACE MOCKUP

NON FUNCTIONAL REQUIREMENTS (2)

USER MOBILE APPLICATION INTERFACE MOCKUP

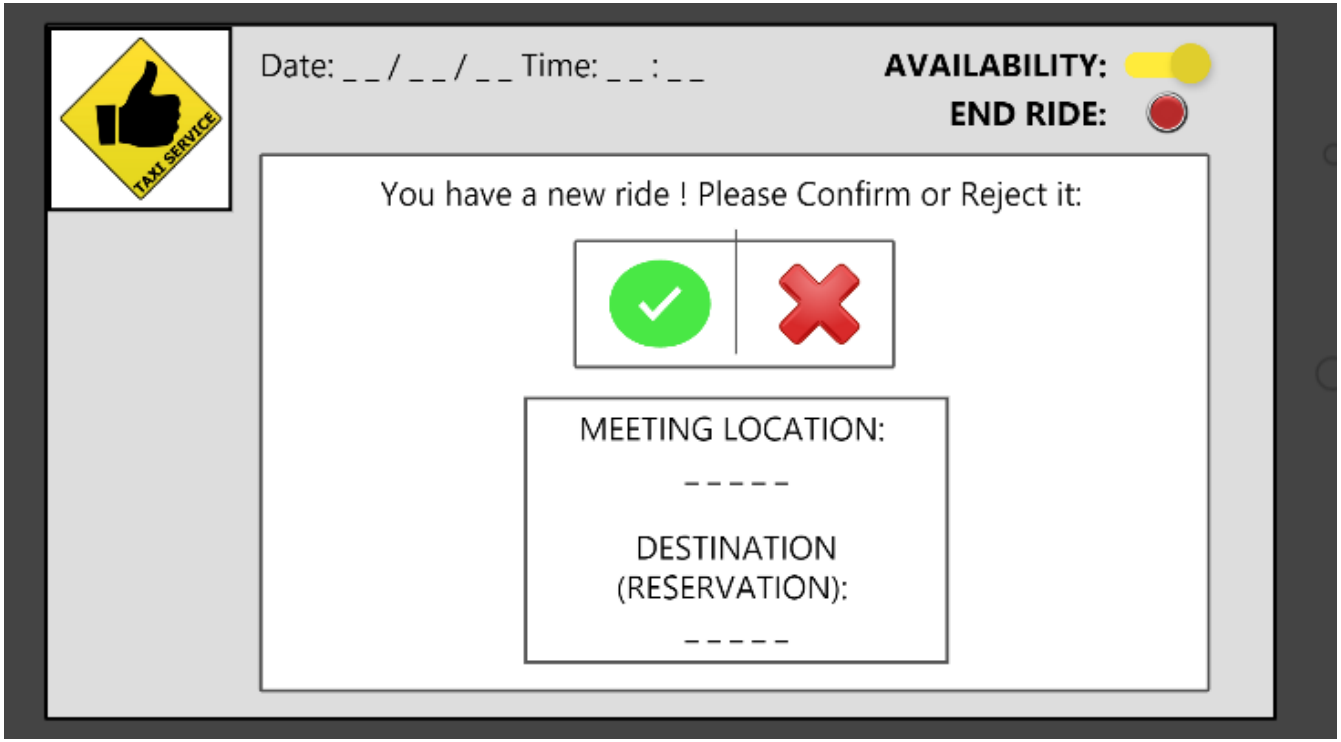






NON FUNCTIONAL REQUIREMENTS (3)

TAXI ANDROID AUTO OS APPLICATION INTERFACE MOCKUP



SOFTWARE SYSTEM ATTRIBUTES

AVAILABILITY: system online 24/7

SECURITY: SSL encryption. Password hashing and salting

RELIABILITY: 99.99 %. 52.56 minutes of downtime per year

SPECIFICATIONS (1)

- **REGISTRATION OF A GUEST**
  - Email address used only once
- **USER / TAXI DRIVER LOG-IN**
  - Correct information must be inserted
- **DYNAMIC MANAGEMENT OF TAXI QUEUES**
  - A FIFO Queue for each Zone
  - Taxi AVAILABLE ←→ Taxi inserted in the Queue
  - Taxi AVAILABLE moves to other Zone → Dynamic update of Queues (TimeStamp)
  - Taxi NOT AVAILABLE ←→ Taxi not present in the Queue
  - Ride Rejection → Taxi moved to the Last Position of the Queue
- **TAXI RIDE REQUEST**
  - Web Application → Meeting Location is required (Form)
  - Mobile Application → Meeting Location provided by built-in GPS
  - No Taxi for a Ride in a Zone → Picking the First in the nearest zone's queue
  - Taxi Code and Waiting Time Assignment
  - A request is ACTIVE → no other requests can be made

SPECIFICATIONS (2)

- **TAXI RIDE RESERVATION**
  - Form Filling (Origin, Destination, Meeting Date and Time)
  - Confirmed only if Meeting Time is at least 2 hours before
  - 10 minutes before the Meeting Time, for the system Ride Reservation = Ride Request
  - Up to 10 minutes before the Meeting Time, Cancellation is allowed

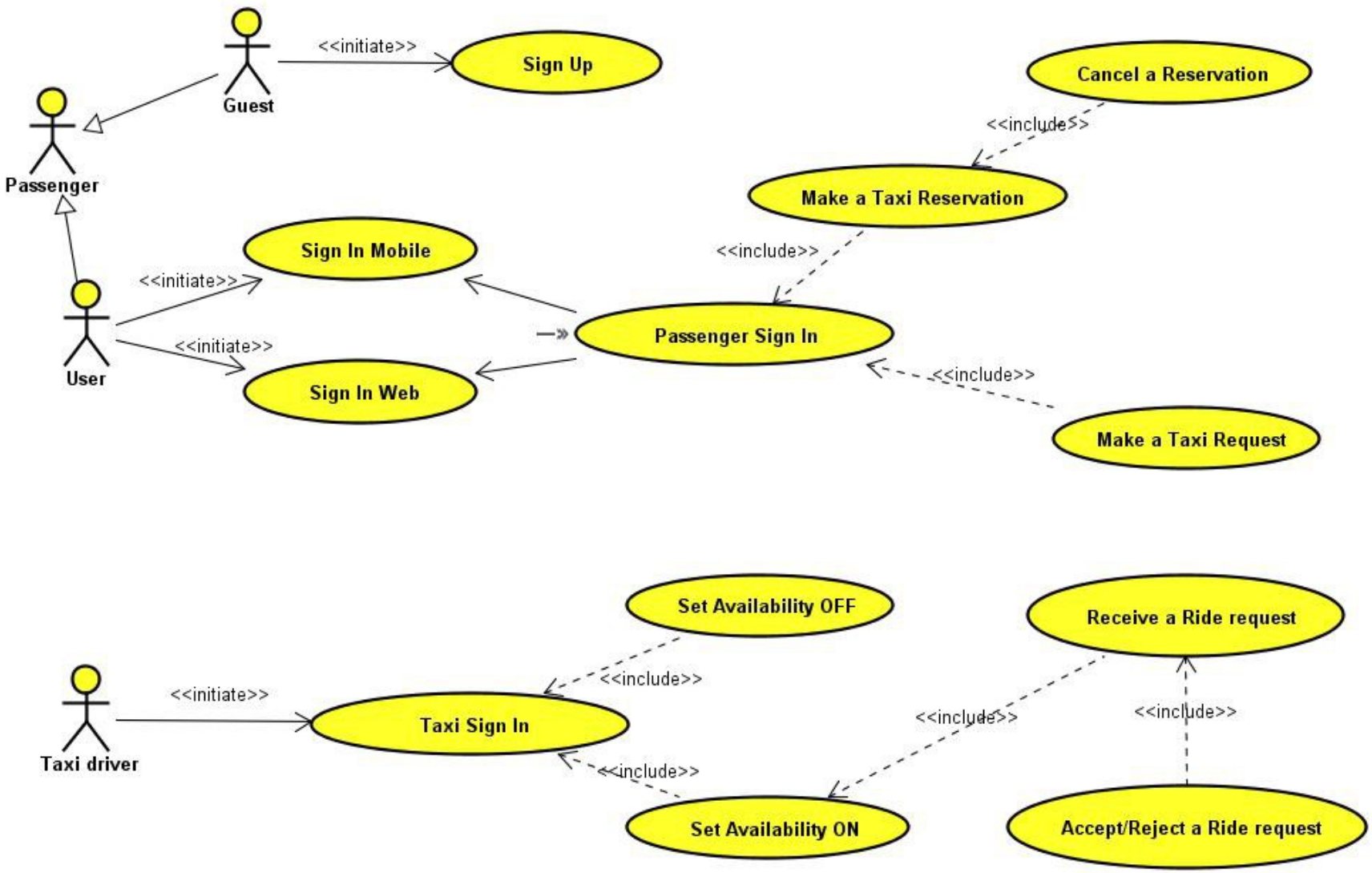
- **CONFIRMATION / REJECTION OF A RIDE**
  - Once a Ride is proposed on display, Taxi drivers have 15 seconds to answer
  - In those 15 seconds, no other Ride are proposed to the same Taxi Driver
  - At the End of a Ride, End Ride button will be pressed

- **UPDATE TAXI DRIVER AVAILABILITY**
  - Confirmation of a ride → Availability OFF → Taxi removed from Queue

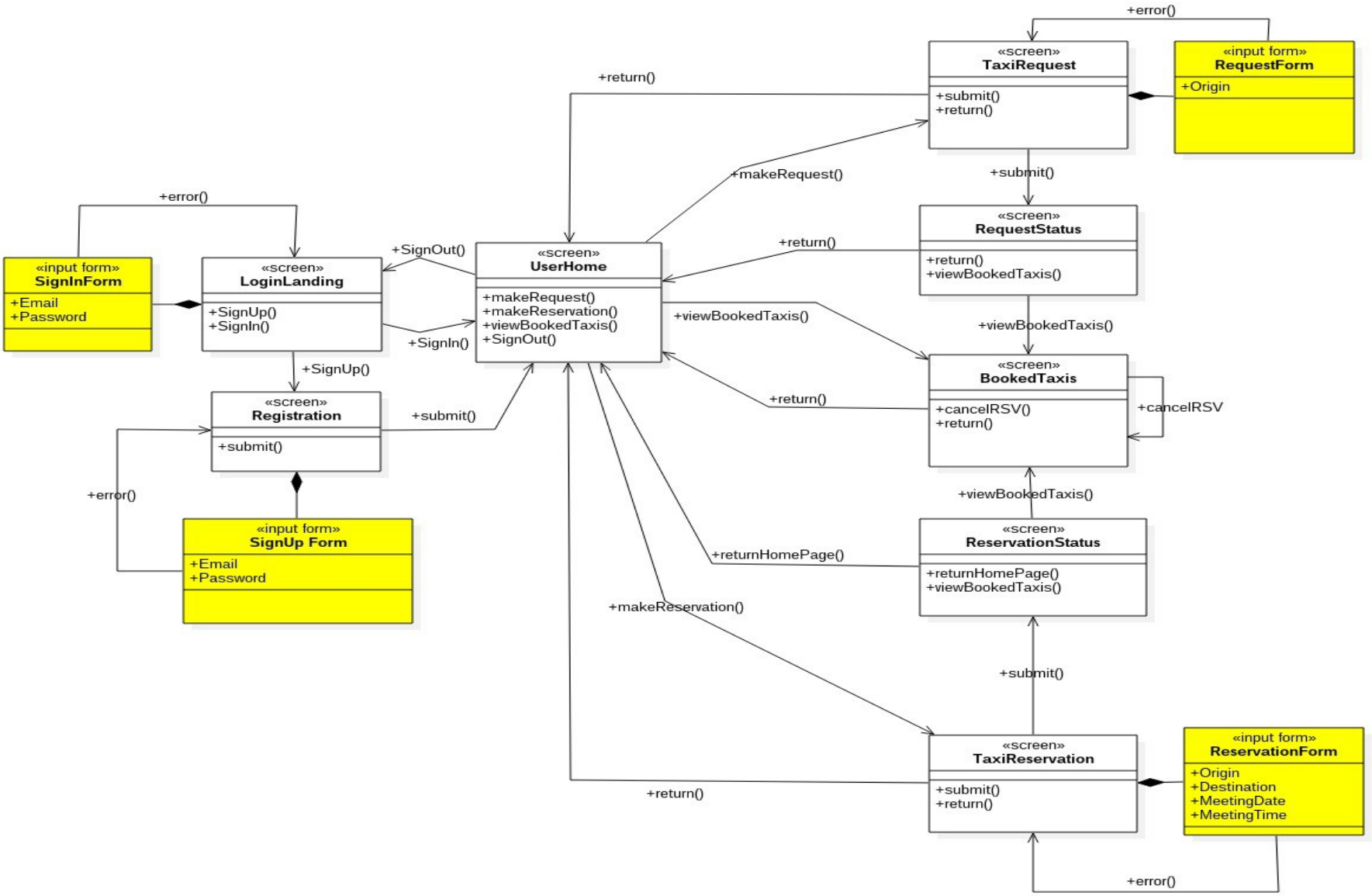
- **LIST OF BOOKED TAXIS**
  - History of concluded/pending requests/reservations

- **SERVICE EXTENSIONS AND NEW FUNCTIONS**
  - APIs to add functions (Priority to Reservations, Fake Requests Tracking, Taxi Sharing Service)

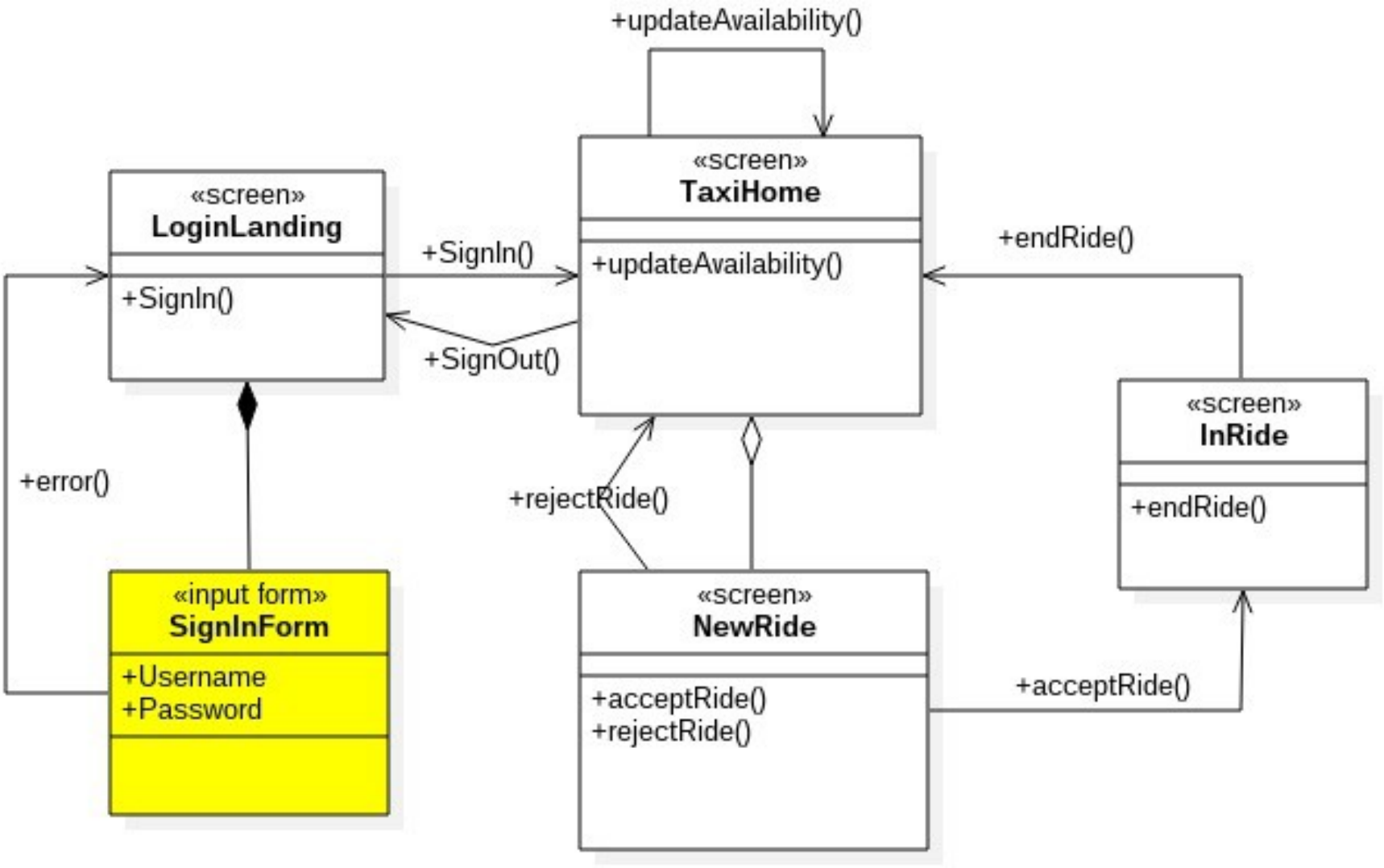
USE CASE DIAGRAM



USER EXPERIENCE DIAGRAM (PASSENGER)

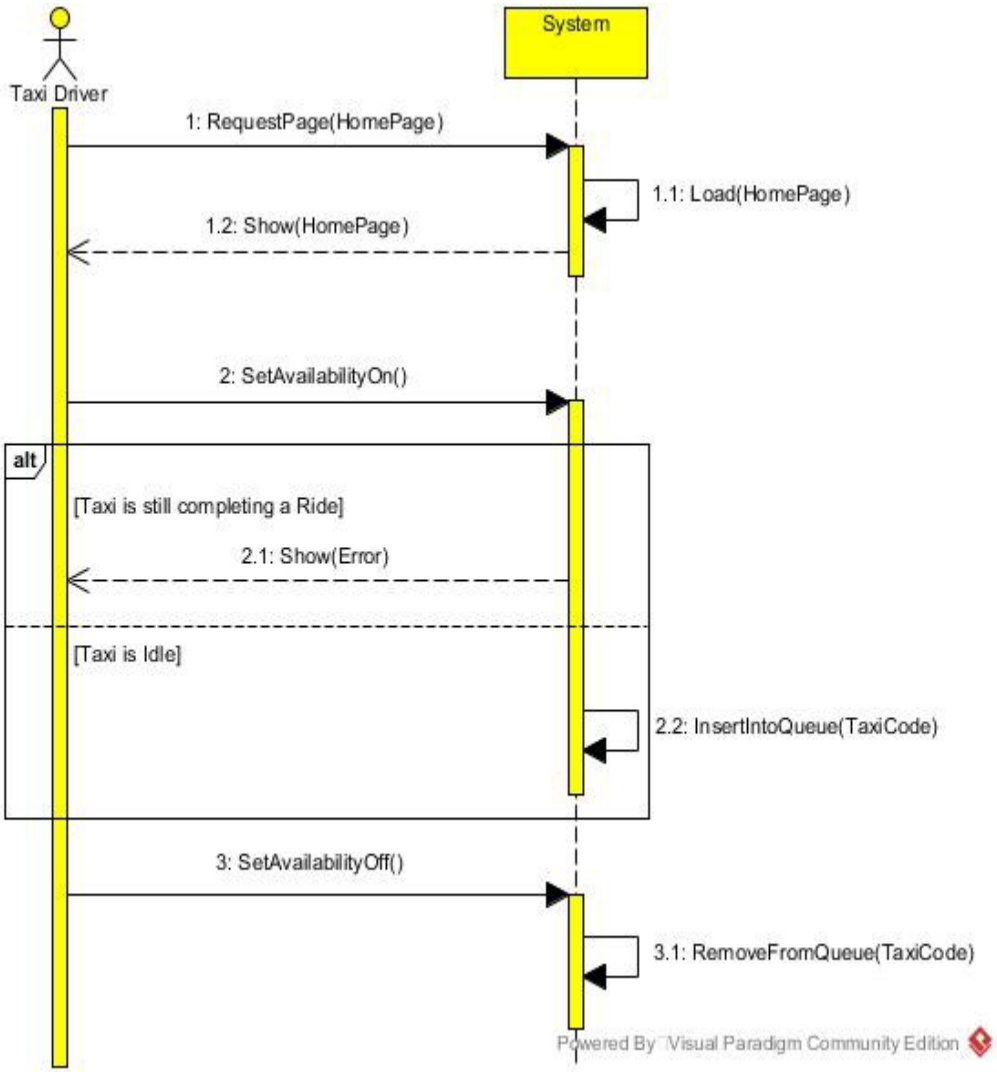


USER EXPERIENCE DIAGRAM (TAXI DRIVER)

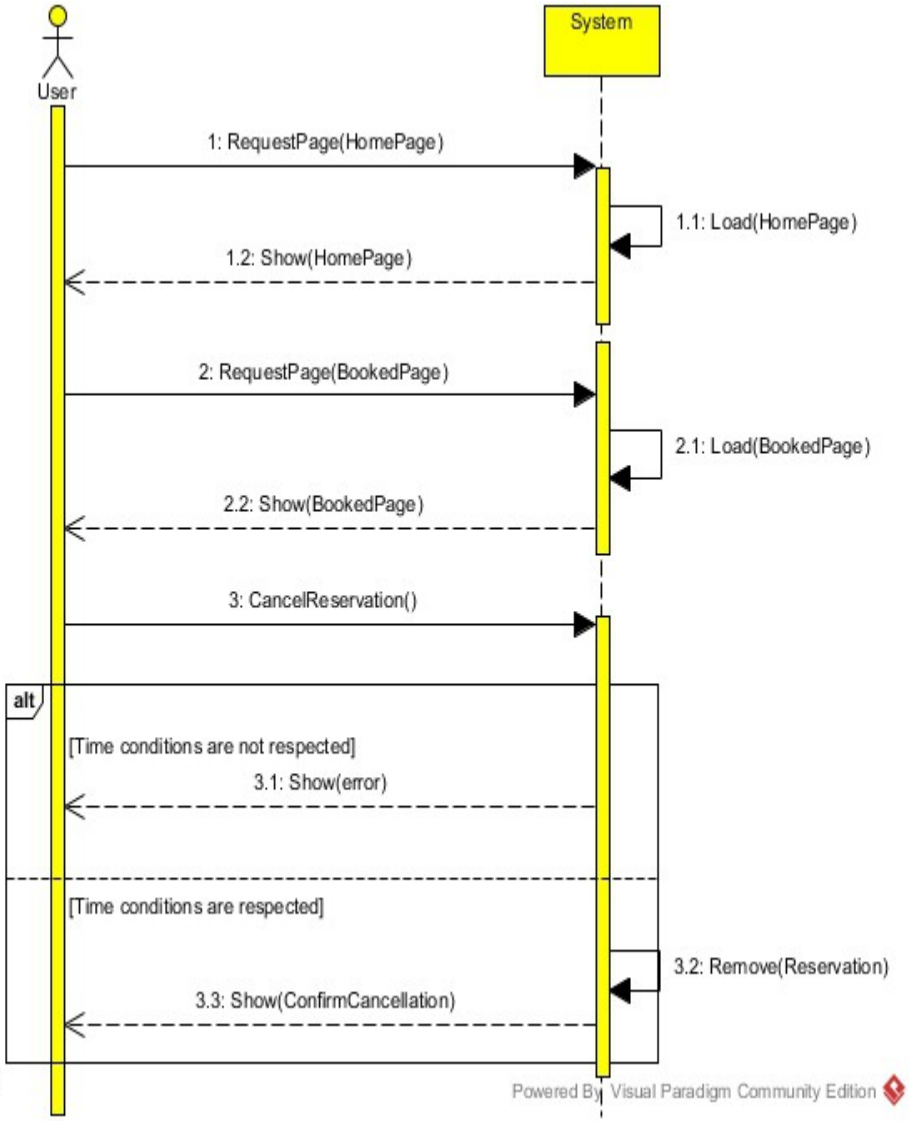


SEQUENCE DIAGRAMS

TAXI DRIVER UPDATES THEIR AVAILABILITY (ON-OFF)

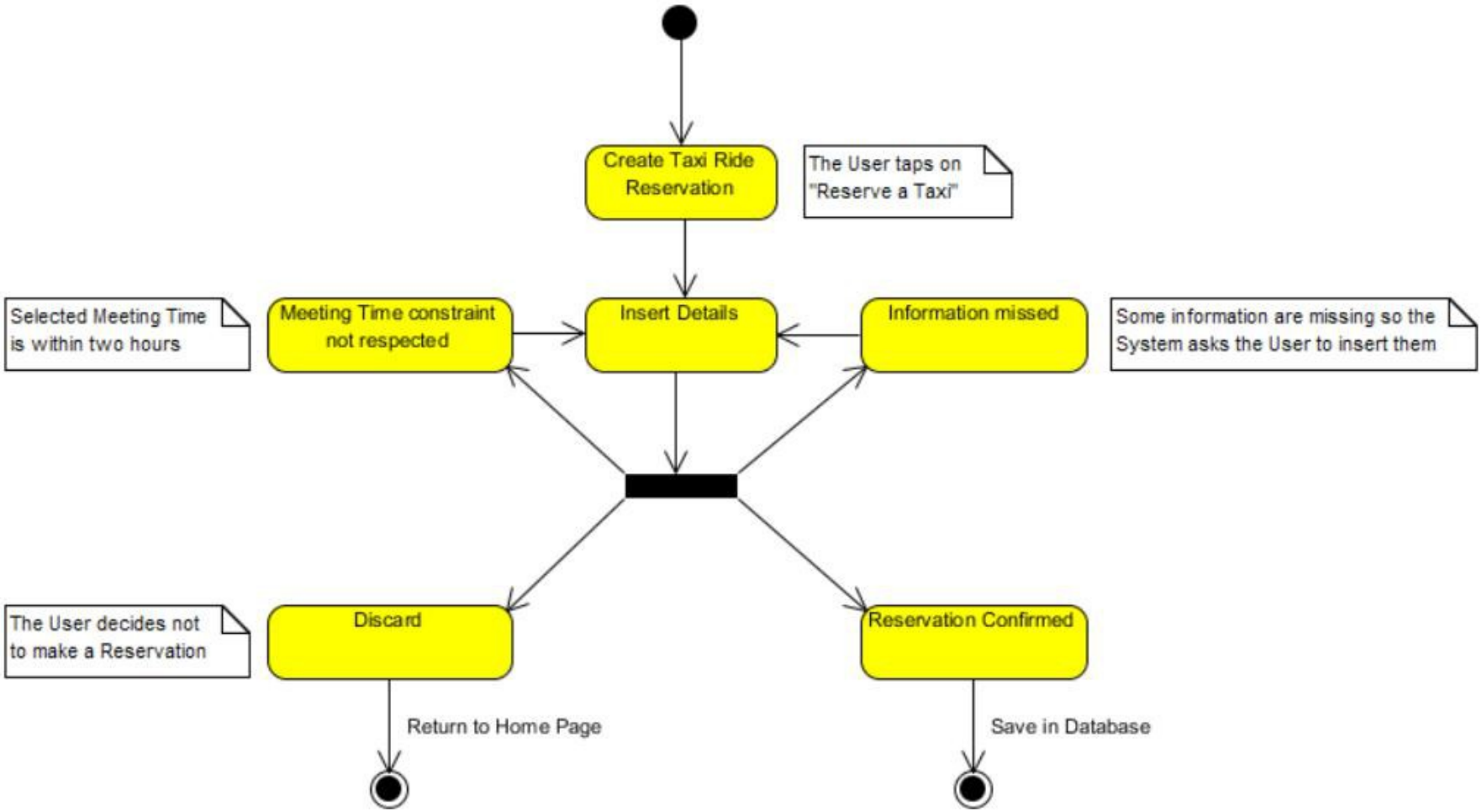


USER VIEWS LIST OF BOOKED TAXIS AND CANCEL A RESERVATION

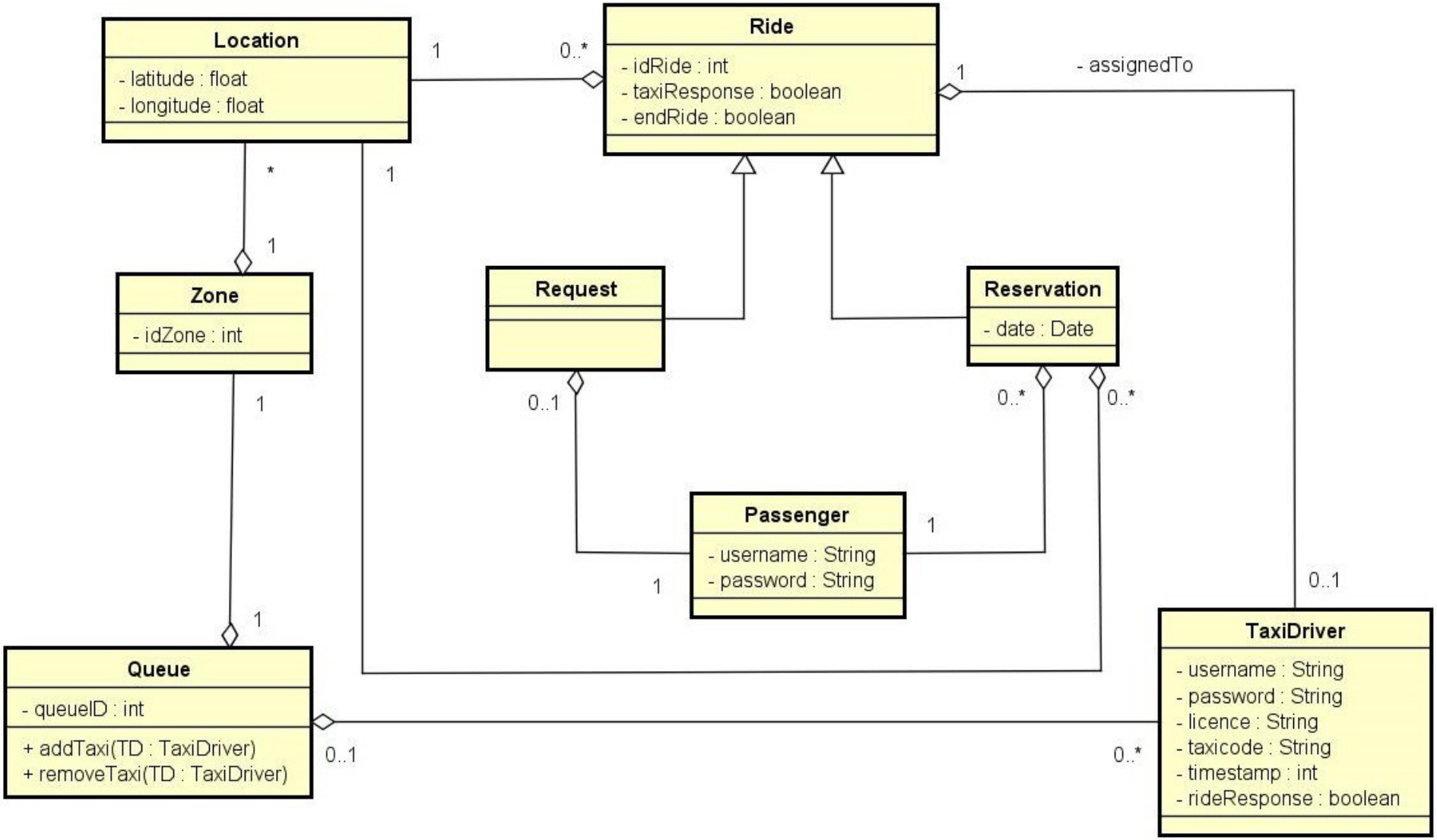


STATE CHART DIAGRAM (EXAMPLE)

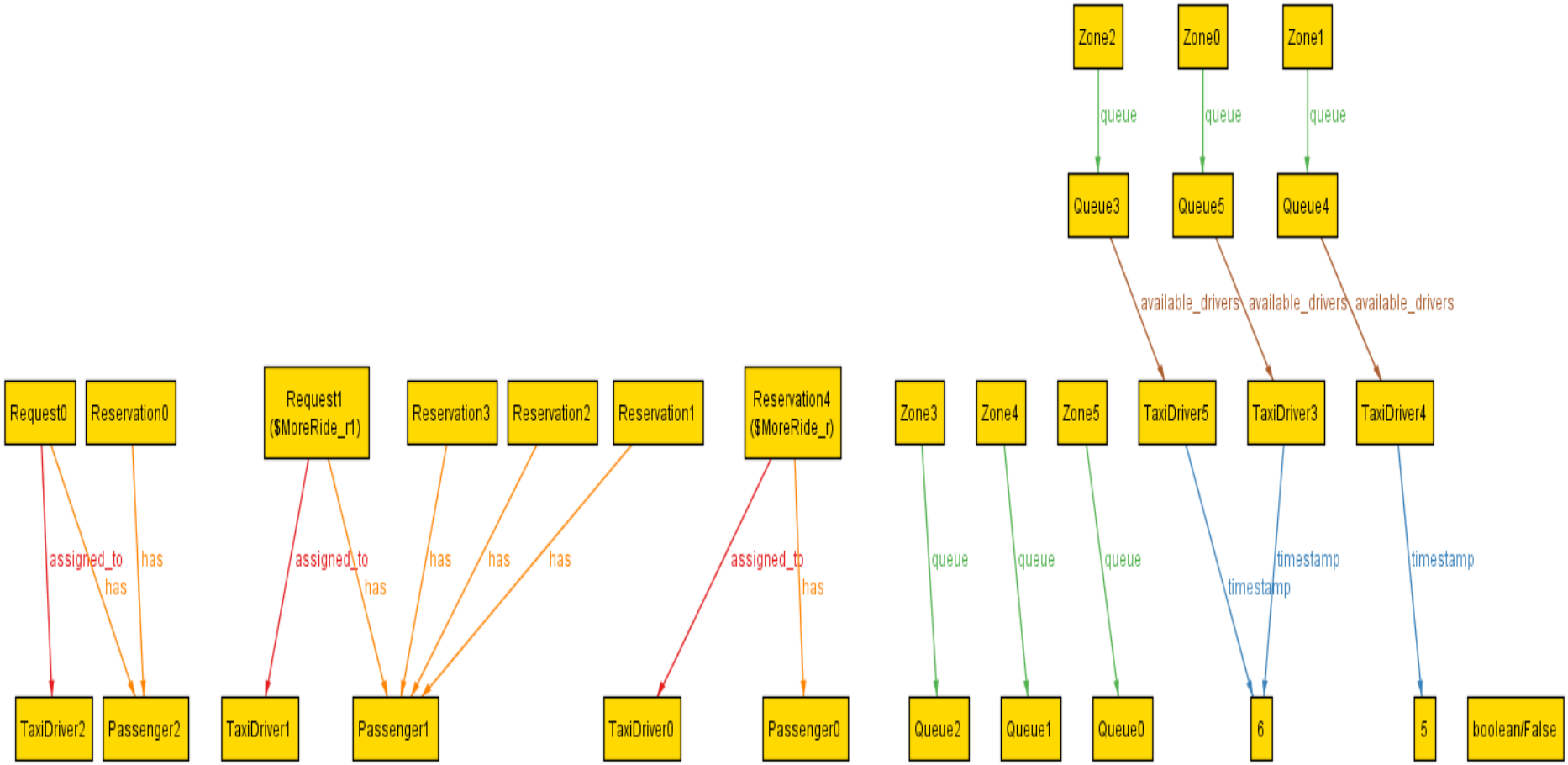
USER CREATES A TAXI RIDE RESERVATION



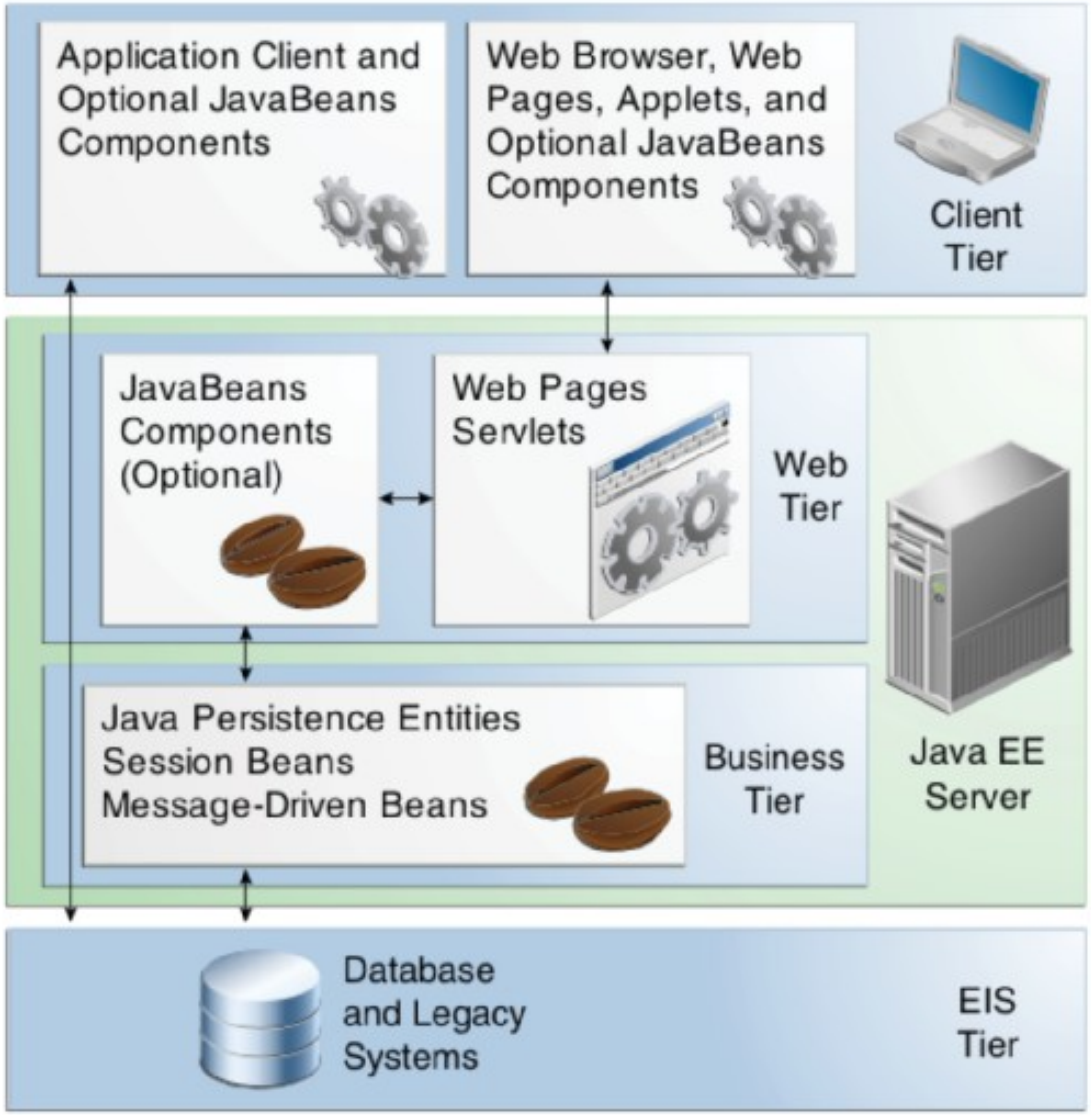
CLASS DIAGRAM



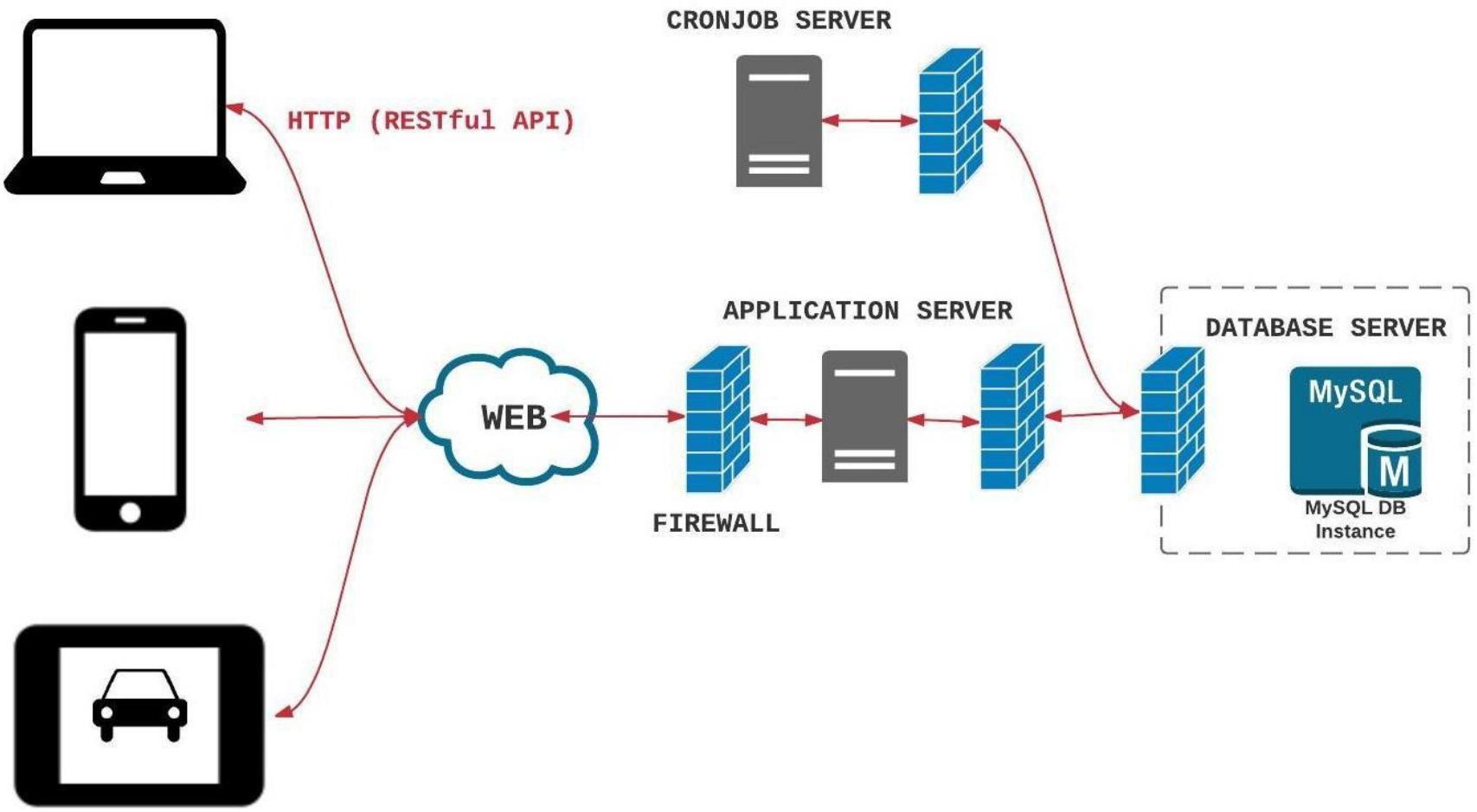
ALLOY


Zone2
Zone0
Zone1
queue
queue
queue
Queue3
Queue5
Queue4
available_drivers
available_drivers
available_drivers
Request0
Reservation0
Request1
($MoreRide_r1)
Reservation3
Reservation2
Reservation1
Reservation4
($MoreRide_r)
Zone3
Zone4
Zone5
TaxiDriver5
TaxiDriver3
TaxiDriver4
assigned_to
has
has
assigned_to
has
has
has
has
assigned_to
has
queue
queue
queue
timestamp
timestamp
timestamp
TaxiDriver2
Passenger2
TaxiDriver1
Passenger1
TaxiDriver0
Passenger0
Queue2
Queue1
Queue0
6
5
boolean/False

ARCHITECTURAL DESIGN (1)

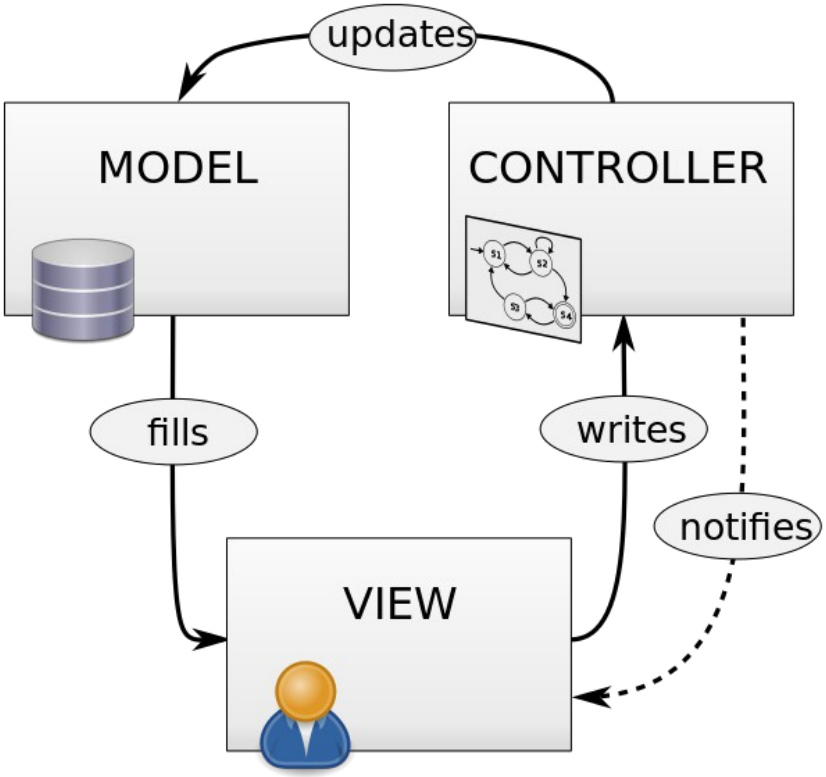


ARCHITECTURAL DESIGN (2)

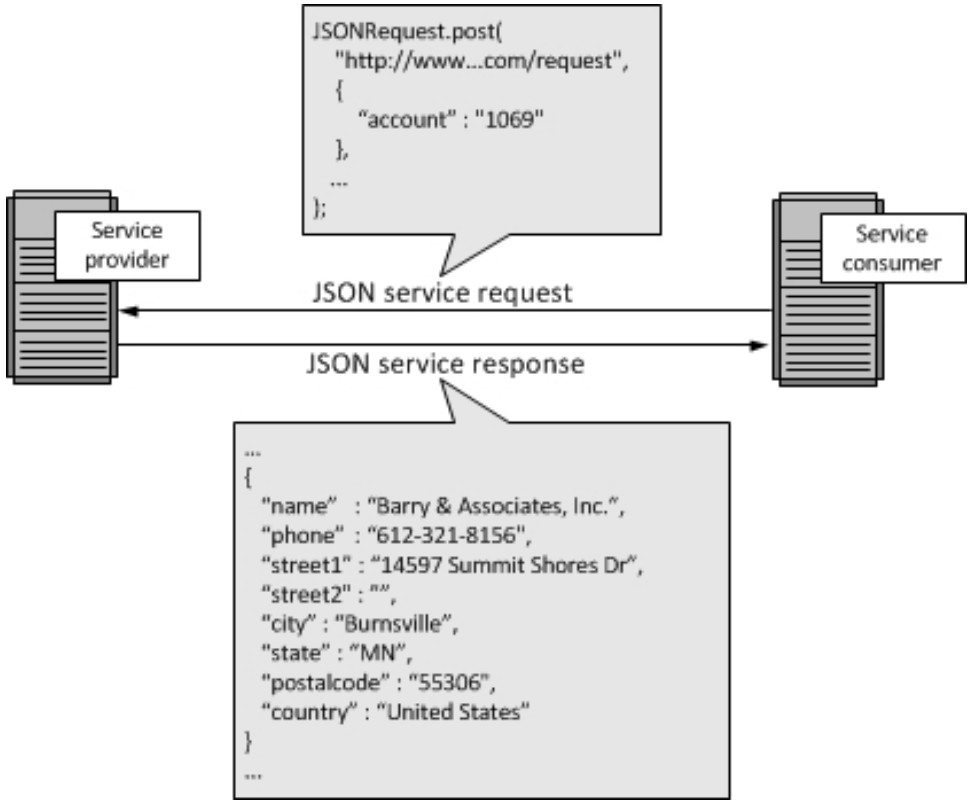


DESIGN CHOICES

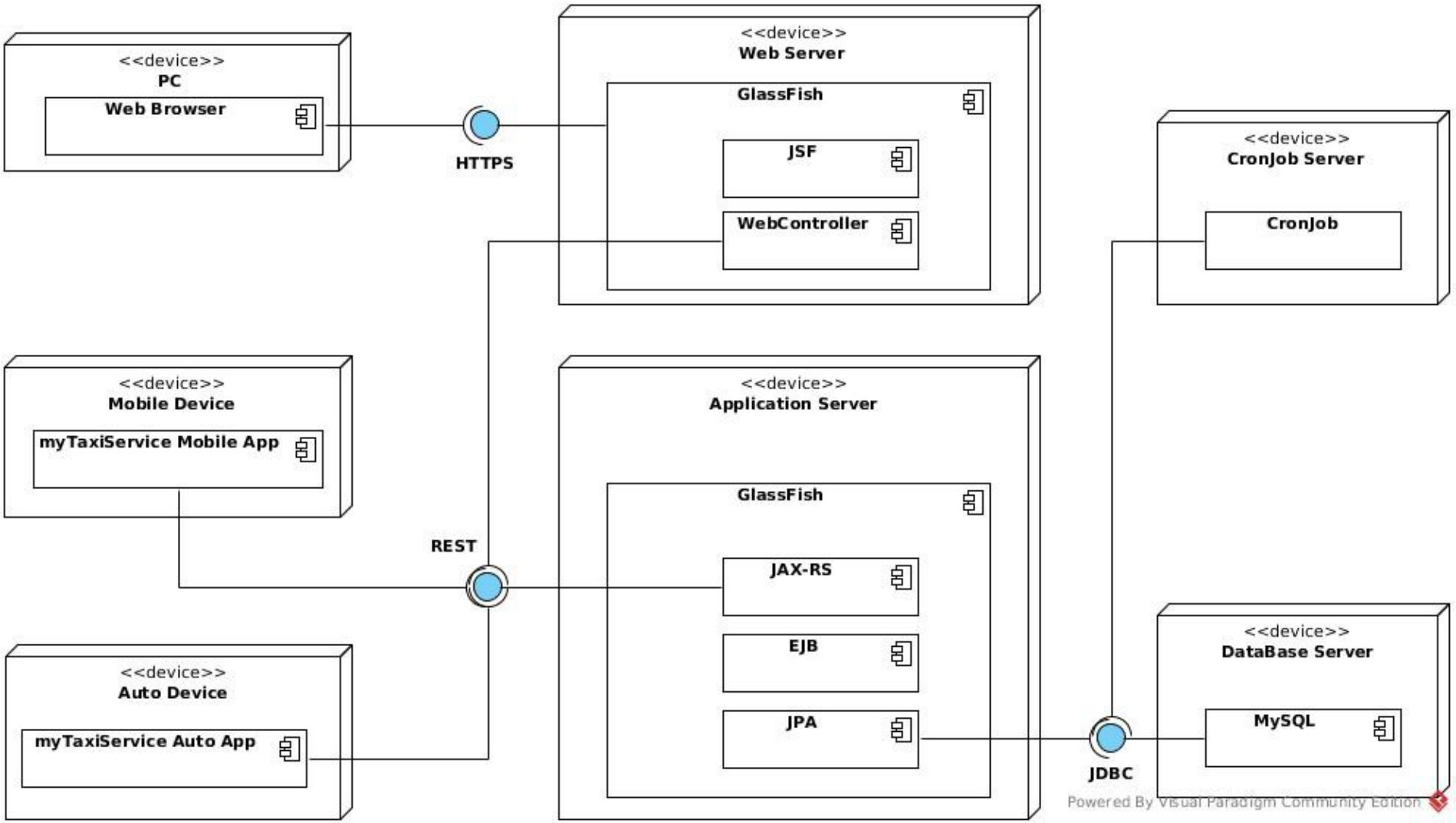
MVC DESIGN PATTERN:



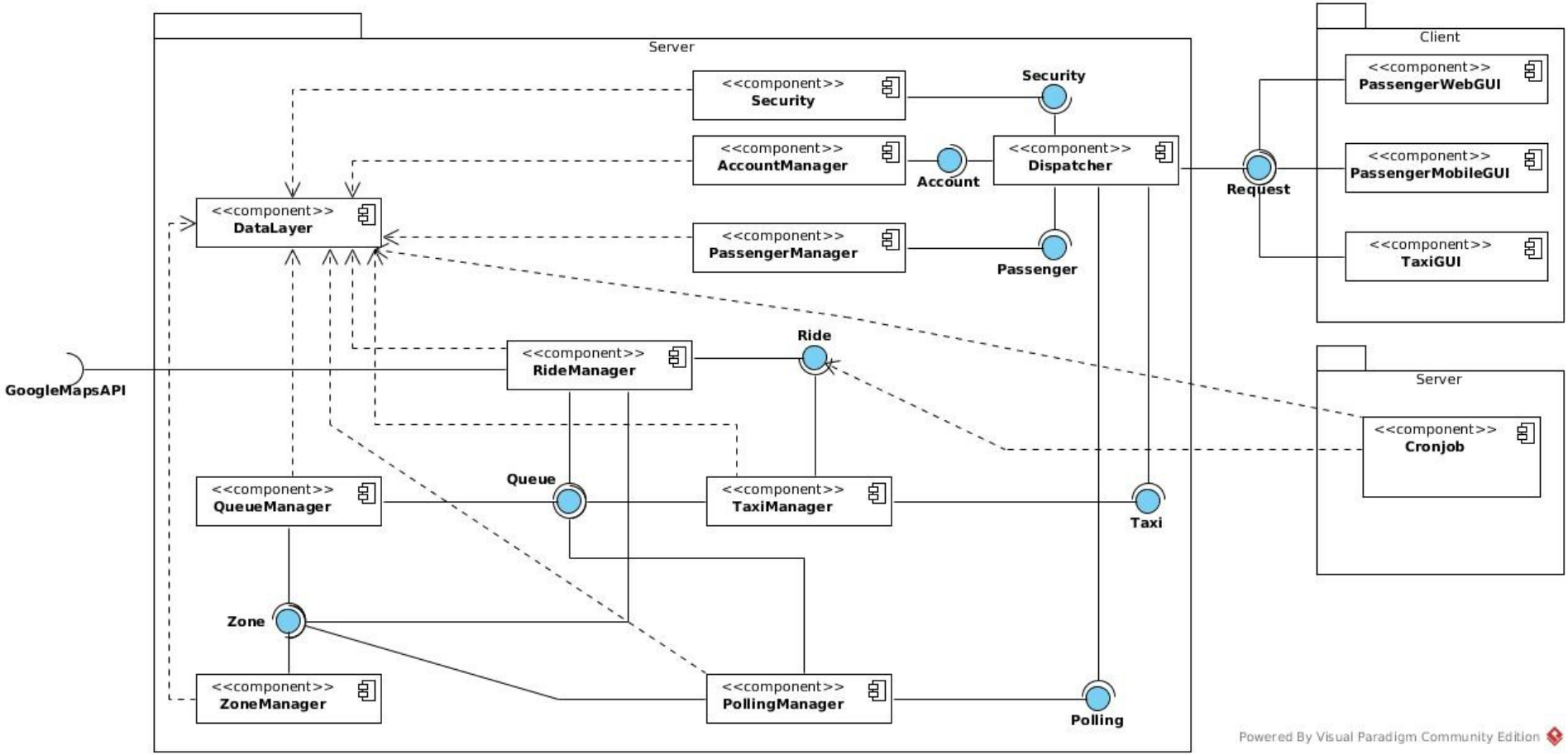
REST MESSAGES:



DEPLOYMENT DIAGRAM

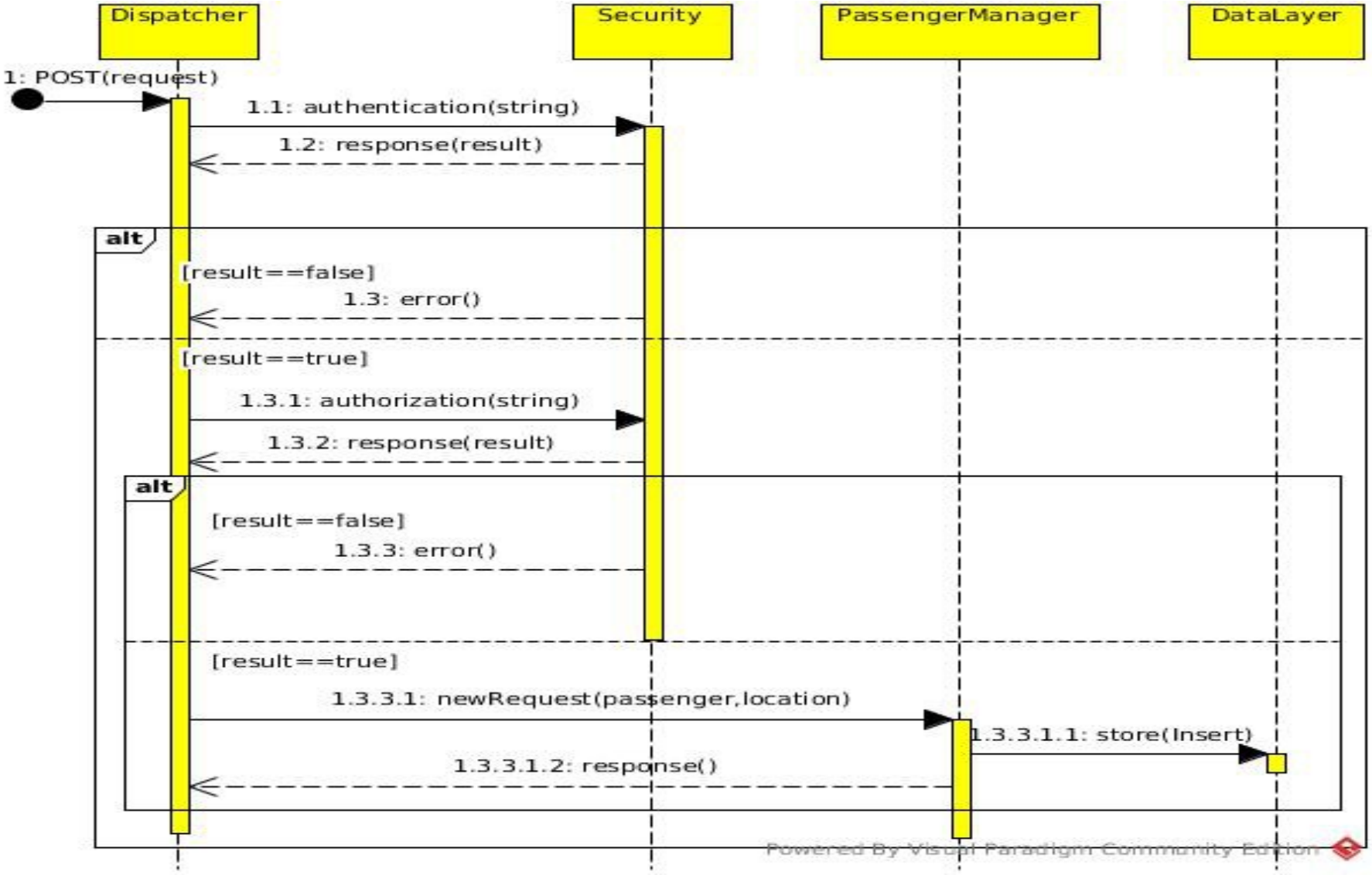


COMPONENT DIAGRAM



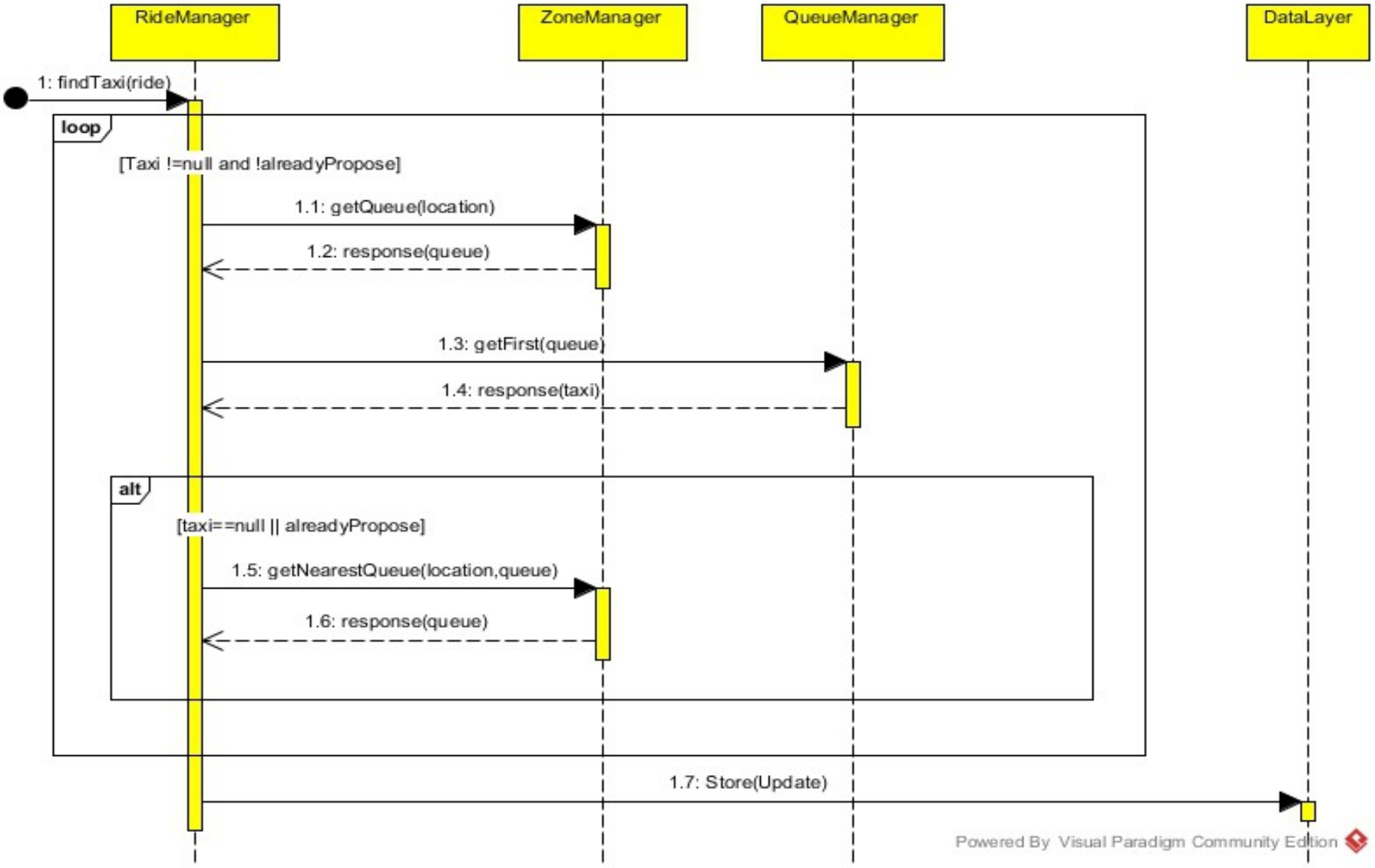
RUNTIME VIEW (1)

A USER MAKES A TAXI RIDE REQUEST FROM THE MOBILE APPLICATION



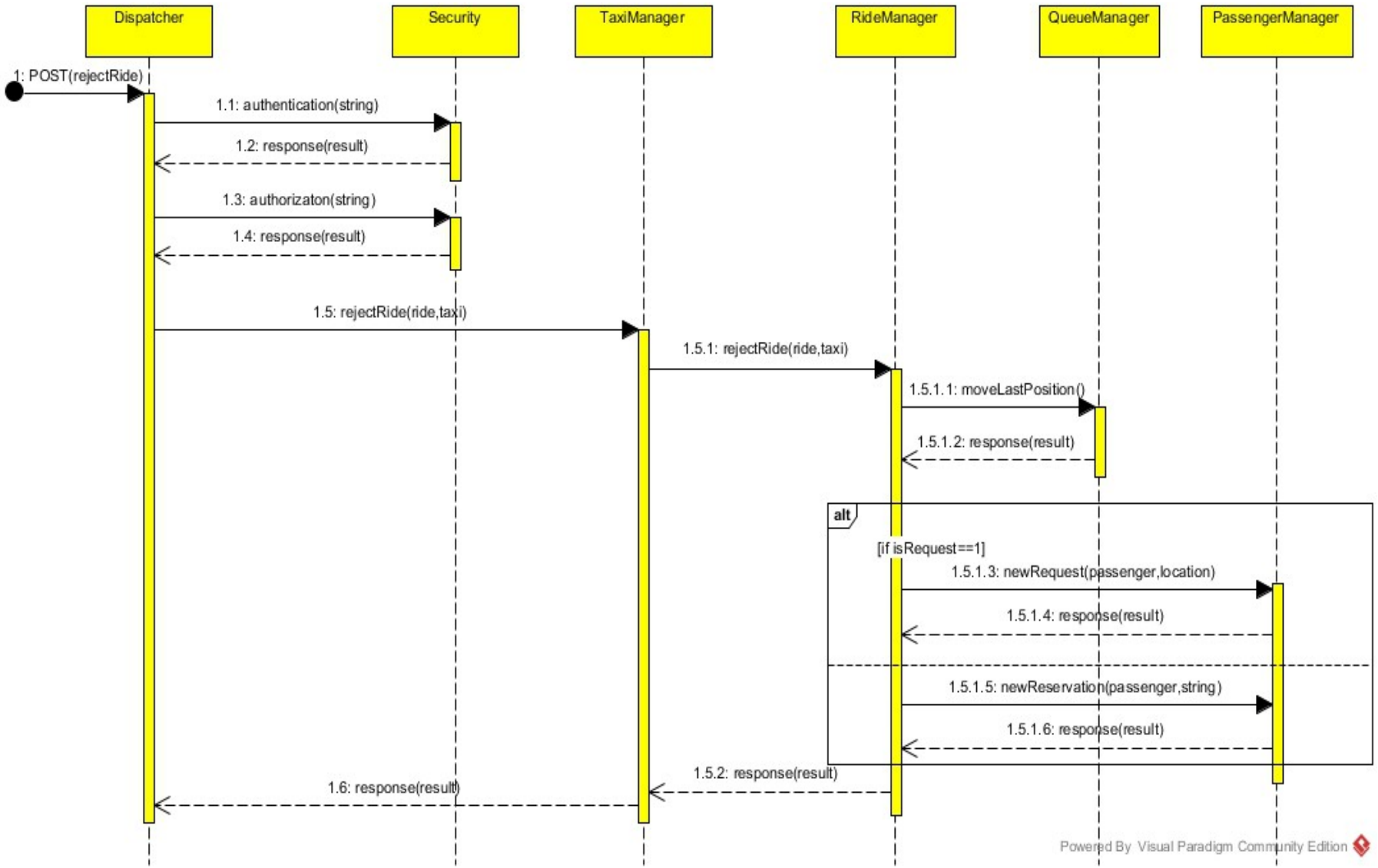
RUNTIME VIEW (2)

A RIDE IS PROPOSED TO A TAXI DRIVER



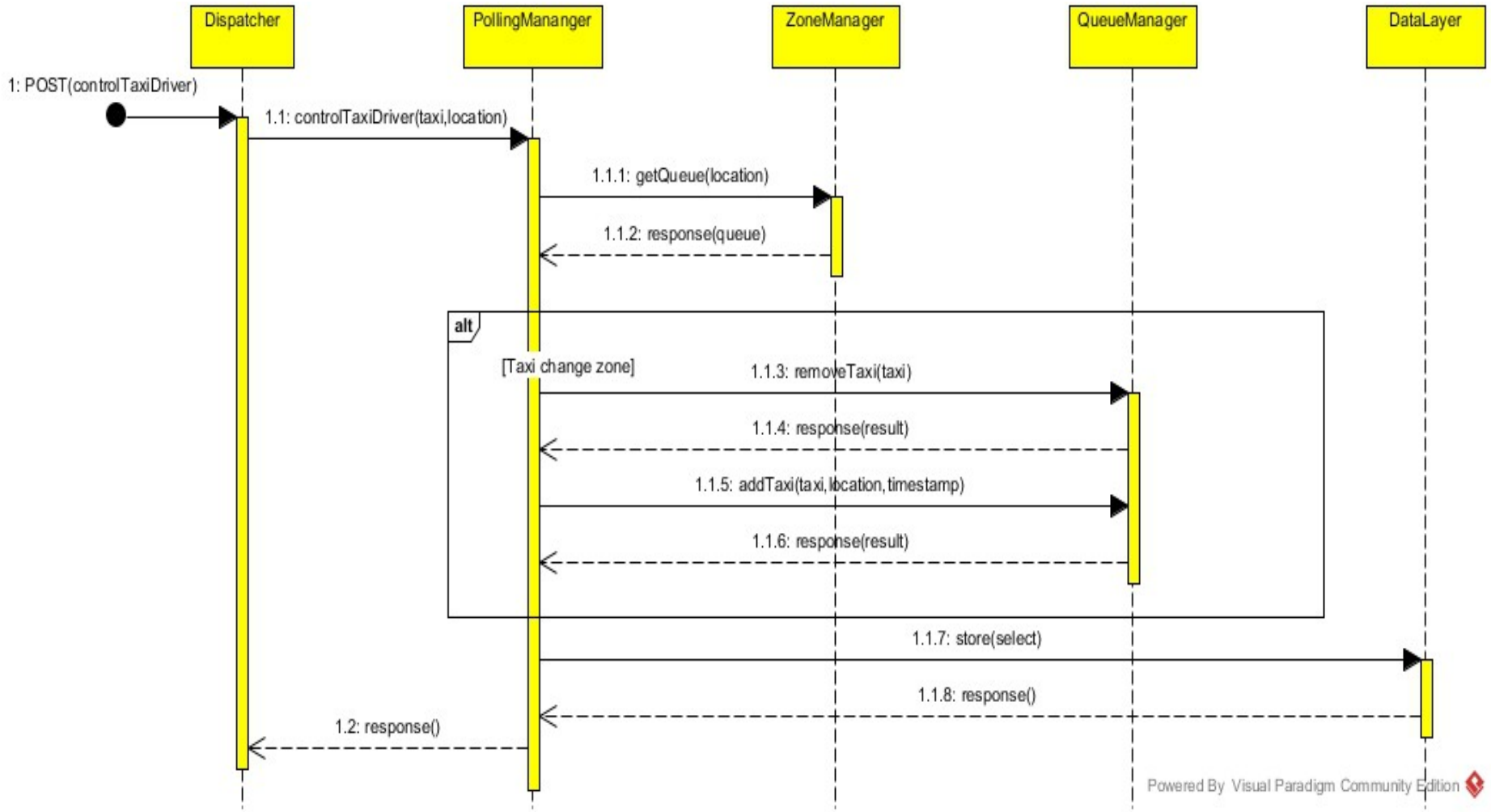
RUNTIME VIEW (3)

A TAXI DRIVER REJECTS A RIDE



RUNTIME VIEW (4)

A TAXI MOVES FROM ONE ZONE TO ANOTHER WHILE ITS AVAILABILITY IS ON

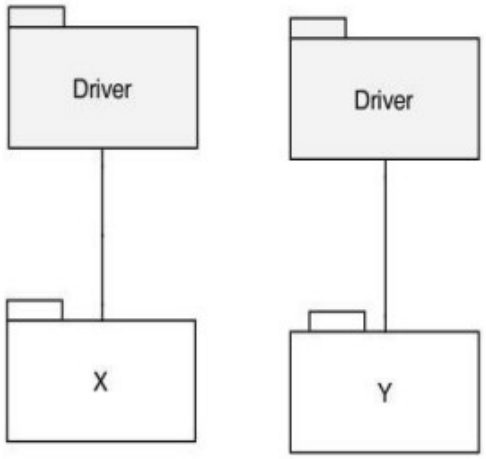


INTEGRATION TESTING STRATEGY

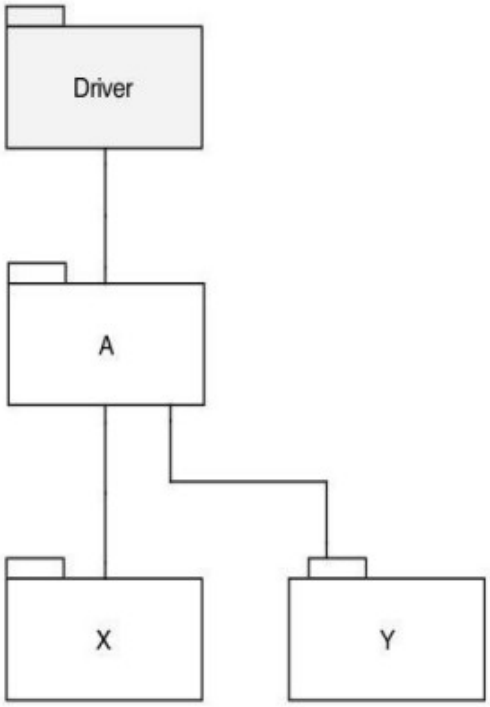
BOTTOM-UP STRATEGY

- Drivers are used to test “leaf” components
- Once leaf modules have been tested, components of the relative higher level are integrated in sequence and another driver is used to test the modules below
- This operation is repeated until the top level of the system is reached

First step

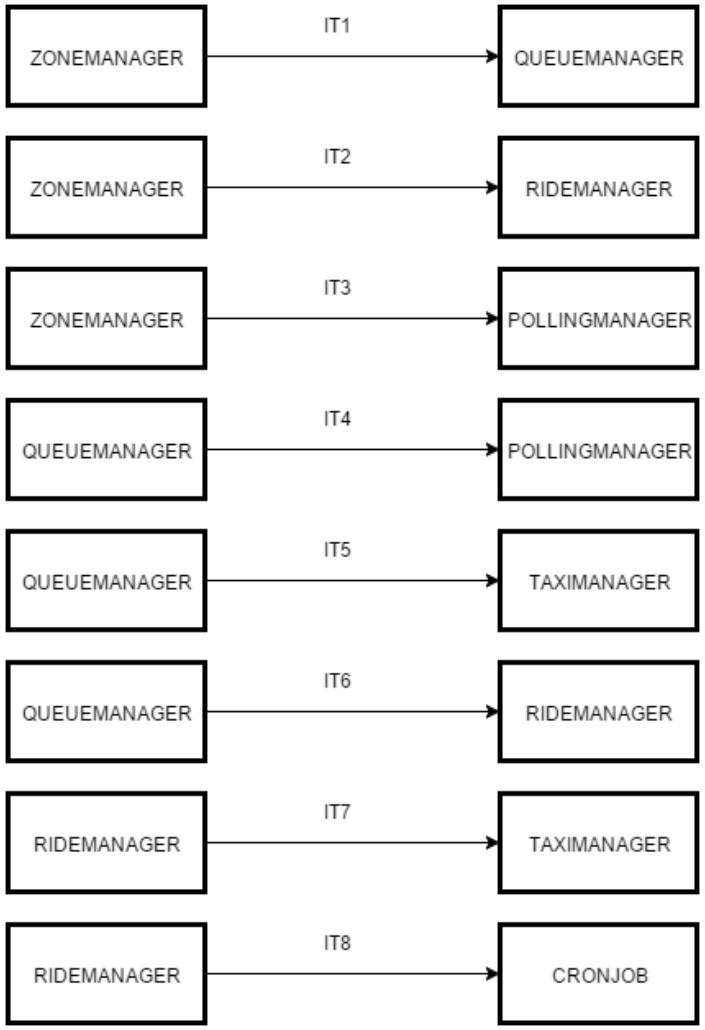


Second step

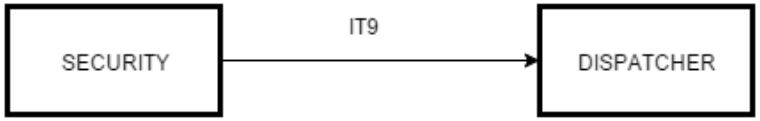


SOFTWARE & SUBSYSTEMS INTEGRATION (1)

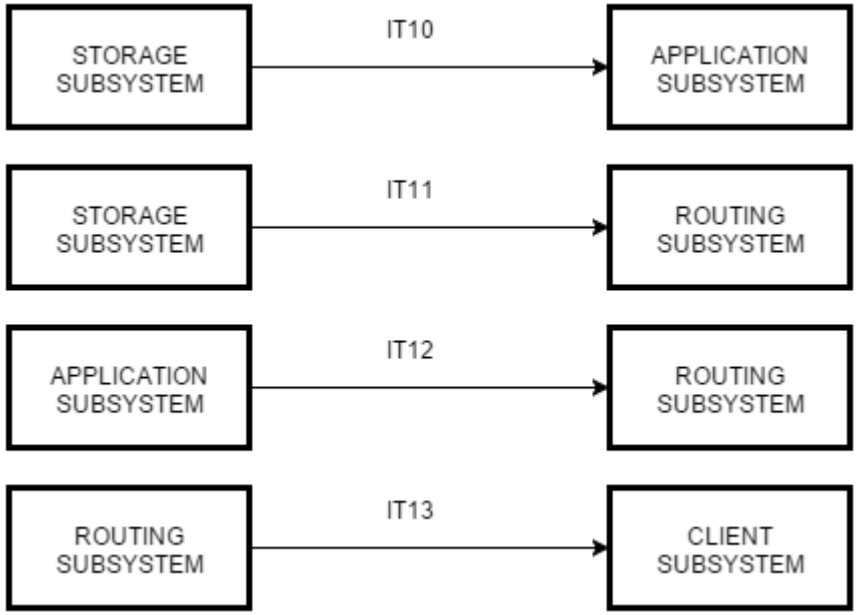
APPLICATION SUBSYSTEM



ROUTING SUBSYSTEM

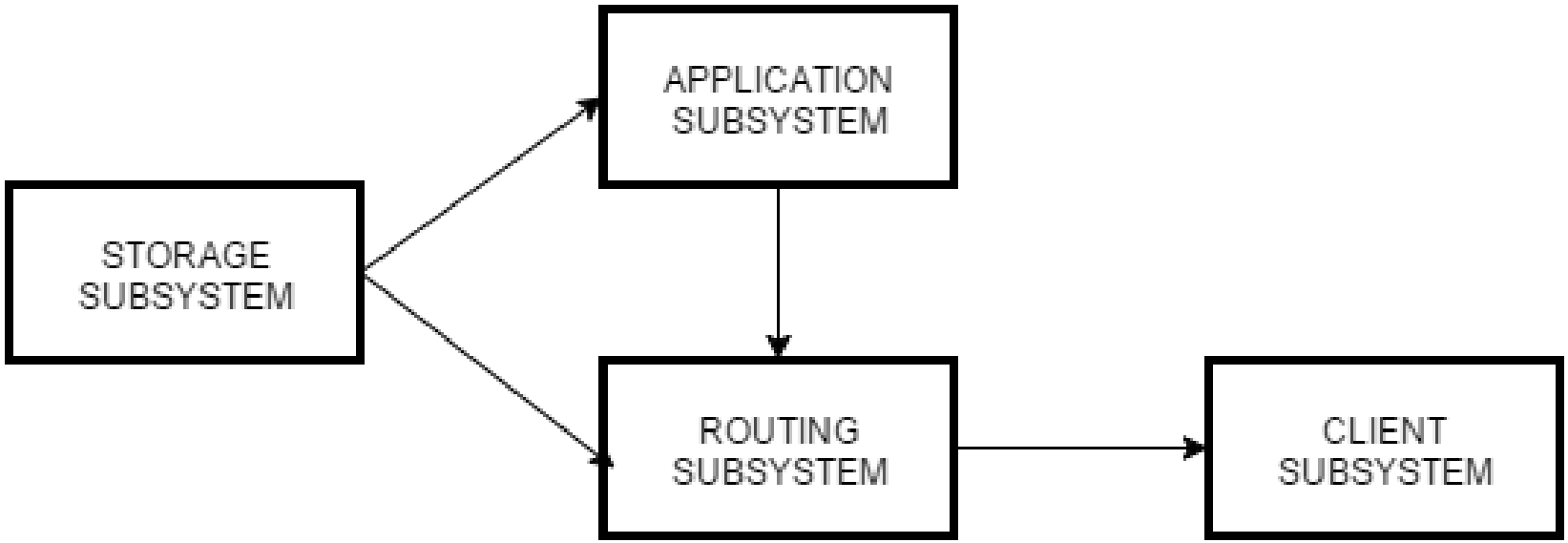


SUBSYSTEM INTEGRATION SEQUENCE



SOFTWARE & SUBSYSTEMS INTEGRATION (2)

ENTIRE INTEGRATED SYSTEM



TOOLS AND TEST EQUIPMENT REQUIRED

- **APACHE JMETER: to test the performance of subsystems**
  - Routing: simulation to test the maximum number of simultaneously connected users
  - Application: simulation of heavy load on REST API
  - Storage: performance of database queries

- **JUNIT: Framework for Unit Testing in Java**
  - Unit tests of the single components
  - Also used for Integration Testing

- **MOCKITO**
  - It generates mock objects, stubs and drivers

- **ARQUILLIAN**
  - Framework useful to perform Integration Testing of components with environment configurations and utilities

- **MANUAL TESTING**
  - Technique needed to simulate typical input data of the User (GUI Testing)

PROGRAM STUBS AND TEST DATA REQUIRED

- **TEST DATABASE**
  - Sample User data needed to perform some test cases

- **LIGHTWEIGHT API CLIENT**
  - To test the REST APIs without the actual client applications

- **EXTERNAL GPS STUB**
  - Needed to replace the external GPS system (Google Maps). It provides sample data

- **DRIVERS**
  - Needed to replace components not yet available for the integration test phase to test other lower-level components

FUNCTION POINTS TECHNIQUE (1)

- **INTERNAL LOGIC FILES (ILF)**
  - Passengers, Taxi Drivers, Queues, Zones, Locations, Ride Requests/Reservation
  - Entities with a simple structure and small number of fields
  - SIMPLE weight of ILF for all of them (7)
  - ILF FP = 6 x 7 = 42

- **EXTERNAL LOGIC FILES (ELF)**
  - External GPS Data
  - Entity with a structure of average complexity
  - MEDIUM weight of ELF for it (7)
  - ELF FP = 7

- **EXTERNAL INPUTS (EI)**
  - User Login/Logout, User Registration, Ride Request creation, Ride Reservation creation, Ride Reservation Cancellation, Taxi Driver Ride Response, Taxi Driver Availability Switch.
    All of them are SIMPLE operations (SIMPLE Weight of EI, 3)
  - Polling Request, simple operation but periodic and frequent (MEDIUM Weight of EI, 4)
  - EI FP = 7 x 3 + 1 x 4 = 25

FUNCTION POINTS TECHNIQUE (2)

- **EXTERNAL OUTPUT (EO)**
  - Polling Output
  - COMPLEX operation due to the use of more entities (cronjob)
  - COMPLEX weight of EO for it (7)
  - EO FP = 7

- **EXTERNAL INQUIRY (EI)**
  - Passenger Notification View, Taxi Driver Notification View
    They are SIMPLE operations (SIMPLE weight of EI, 3 each)
  - Passenger List of Booked Taxis View (Retrieval of all rides and Cancellation possibility)
    Taxi Driver In-Ride View (Google Maps API invocations)
    They are operations of Average Complexity (MEDIUM weight of EI, 4 each)
  - EI FP = 2 x 3 + 2 x 4 = 14

TOTAL FUNCTION POINTS (UFP) = 42 + 7 + 25 + 7 + 14 = 144

COCOMO ANALYSIS (1)

COCOMO II - Constructive Cost Model

Software Size Sizing Method: Source Lines of Code

	SLOC	% Design Modified	% Code Modified	% Integration Required	Assessment and Assimilation (0% - 8%)	Software Understanding (0% - 50%)	Unfamiliarity (0-1)
New	6624						
Reused		0	0				
Modified							

Software Scale Drivers

Precedentedness	Nominal	Architecture / Risk Resolution	Low	Process Maturity	Nominal
Development Flexibility	Nominal	Team Cohesion	High		

Software Cost Drivers

Product		Personnel		Platform	
Required Software Reliability	High	Analyst Capability	Nominal	Time Constraint	Very High
Data Base Size	High	Programmer Capability	High	Storage Constraint	Nominal
Product Complexity	Nominal	Personnel Continuity	Nominal	Platform Volatility	Low
Developed for Reusability	Nominal	Application Experience	Low	**Project**	
Documentation Match to Lifecycle Needs	Nominal	Platform Experience	Low	Use of Software Tools	Nominal
		Language and Toolset Experience	Nominal	Multisite Development	Low
				Required Development Schedule	Nominal

Maintenance Off

Software Labor Rates

Cost per Person-Month (Dollars) 2000

COCOMO ANALYSIS (2)

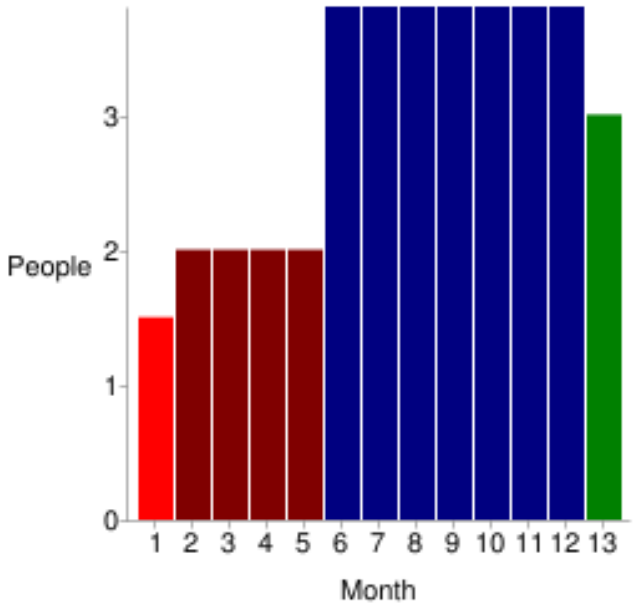
Results

Software Development (Elaboration and Construction)

Effort = 38.3 Person-months
Schedule = 12.2 Months
Cost = $76567

Total Equivalent Size = 6624 SLOC

Staffing Profile



Acquisition Phase Distribution

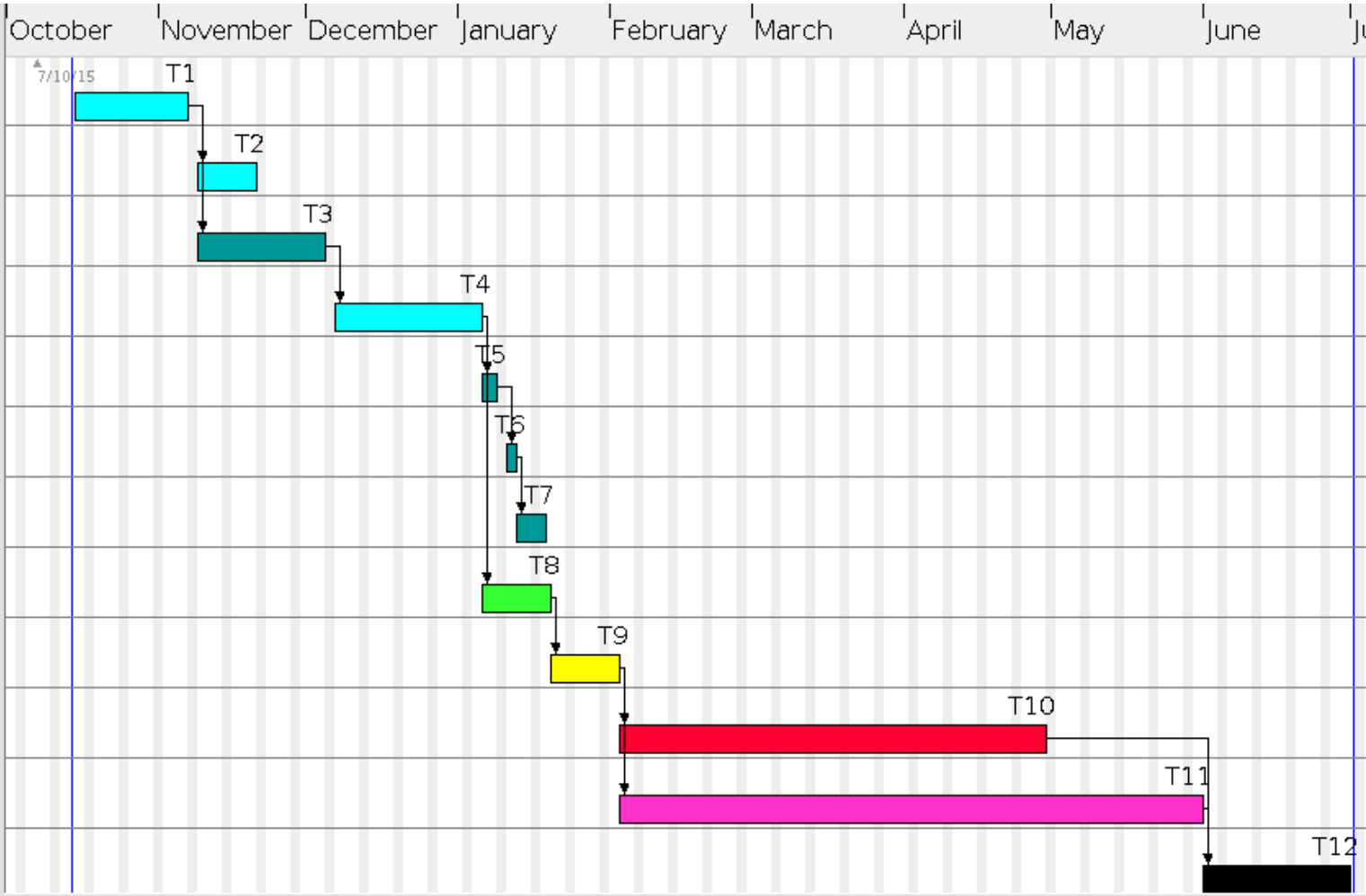
Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	2.3	1.5	1.5	$4594
Elaboration	9.2	4.6	2.0	$18376
Construction	29.1	7.6	3.8	$58191
Transition	4.6	1.5	3.0	$9188

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.3	1.1	2.9	0.6
Environment/CM	0.2	0.7	1.5	0.2
Requirements	0.9	1.7	2.3	0.2
Design	0.4	3.3	4.7	0.2
Implementation	0.2	1.2	9.9	0.9
Assessment	0.2	0.9	7.0	1.1
Deployment	0.1	0.3	0.9	1.4

TASK SCHEDULE

Name	Begin d...	End date
T1	15/10/15	6/11/15
T2	9/11/15	20/11/15
T3	9/11/15	4/12/15
T4	7/12/15	5/1/16
T5	6/1/16	8/1/16
T6	11/1/16	12/1/16
T7	13/1/16	18/1/16
T8	6/1/16	19/1/16
T9	20/1/16	2/2/16
T10	3/2/16	29/4/16
T11	3/2/16	31/5/16
T12	1/6/16	30/6/16



T1: RASD v1.0, **T2:** RASD v2.0, **T3:** DD v1.0, **T4:** RASD v3.0, **T5:** DD v2.0, **T6:** DD v3.0, **T7:** ITPD v1.0, **T8:** DD v4.0, **T9:** PPD v1.0, **T10:** Front-end development, **T11:** Back-end development, **T12:** Acceptance Testing

RISK MANAGEMENT

RISK	PROBABILITY OF OCCURRENCE	EFFECT (IMPACT ON THE SYSTEM)	RECOVERY STRATEGY
Geolocation System (Google Maps) stops working	Very Low	Catastrophic	If the problem is temporary, use another external service to let the system work again.
Database cannot process as many transactions per second as expected	Moderate	Serious	Investigate the possibility of buying a higher performance database.
Temporary unavailability of personnel involved in critical tasks	Moderate	Serious	Reorganize the resource allocation so that there is more overlap of work and people therefore understand each other's jobs. Another aspect to consider is to select those people that are not already executing other critical activities.
Underestimated Development Time	Moderate	Serious	Investigate buying-in components; investigate the use of a program generator.

CONCLUSION

Any Questions?