Cairo University
Faculty of Computers and Information
Department of Computer Science

Intelligent Mobile Bus Ticketing System

Supervised by

Dr. Ahmed Shawky Moussa
TA. Mahmoud Mohsen

Implemented by

20140192	Amr Saeed Hosny
20140064	Ashrakat Mokhtar
20140187	Omar Fawzy Salama
20140196	Amr Magdy Ibrahim

Academic Year 2017-2018
Midyear Documentation of Graduation Project

Abstract

The issue of fast, safe, and non-distracting payment method is very paramount in any transportation, especially if it affects the safety of passengers. Therefore, an efficient solution has to be applied to enforce the separation of driving the vehicle and issuing tickets responsibilities.

Our solution intends to replace old-fashioned methods of issuing tickets by an efficient digital system that falls under the realm of *internet of things* (*IoT*). A software application is downloaded on the passenger’s smartphone. Every passenger can have an account on that application containing his/her information. The passenger’s account can be charged with a predetermined amount of balance specified by the passenger at anytime and anyplace. Once enough balance is charged on the account, the passenger can ride any transportation that provides our digital service. The transportation contains an attached IoT device in which passenger can pass his/her smartphone over. That IoT device works as a reader to check passenger’s balance against the transportation ticket price. After the process of validation, the IoT device communicates with a back-end server which in turn sends a digital ticket to passenger’s account and deducts the price of the ticket from the current balance. At this moment, the passenger has a digital ticket on his/her smartphone containing the necessary details that work as a proof of payment.

The tools used to implement our solution are *Android*, *PHP Laravel framework*, *MySQL database*, and *Near Field Communication* technology (*NFC*). *Android* is used as a front-end technology representing the mobile application downloaded on the passenger’s smartphone. *NFC technology* is used in both smartphones and the IoT device mentioned earlier as a way of communication between them. Modern smarphones come with NFC component which is mostly attached to their back. The IoT device device consists of a board which is connected to NFC component. *PHP Laravel* is used as back-end server framework representing the implementation of our services. *PHP Laravel* communicates with *MySQL* database to store and retrieve passengers’ data.

Project Idea

- Developing a payment method for means of transportation based on cellphones. The passenger downloads an application on his cellphone and creates an account on it. The passenger’s account could be charged with balance then his/her phone could be passed over a reader on several types of public transportation to deduct the fare.

The system is an application belonging to the realm of
Internet of Things (IoT).

Problems Targeted

- Driver distraction
- Arrival delay
- Slow process of ticketing
- Non trackable transactions
- Tickets production costs
- Change problems
- Ticket loss

Why and How to Solve It

How?

- Safety of passengers
- Increase transactions speed
- Reduce arrival delay

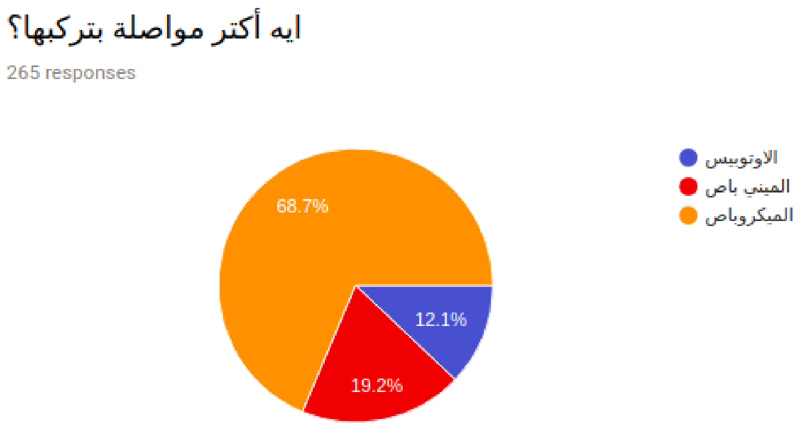
Why?

- Utilize a single compact device
- Easier usage more flexibility

Our Statistics

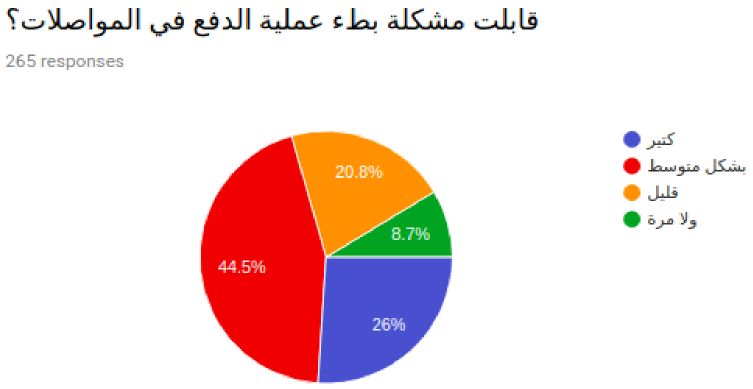
A statistics for 265 people who live at Giza and Cairo who use public transportation daily. It includes the following questions with pie-charts representing their answers.

1. **What is the transportation that you mostly take?**
   - Bus: 12.1 %
   - Mini bus: 19.2 %
   - Micro bus: 68.7 %



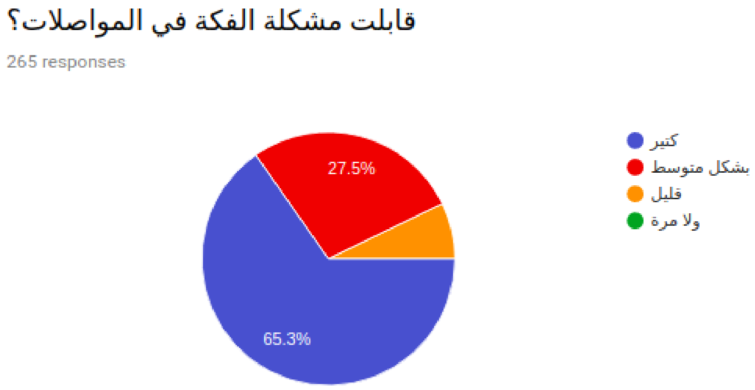
2. Did you face a payment delay of the transportation fare?

- Frequently: 26 %
- On average: 44.5 %
- Rarely: 20.8 %
- Never 8.7%



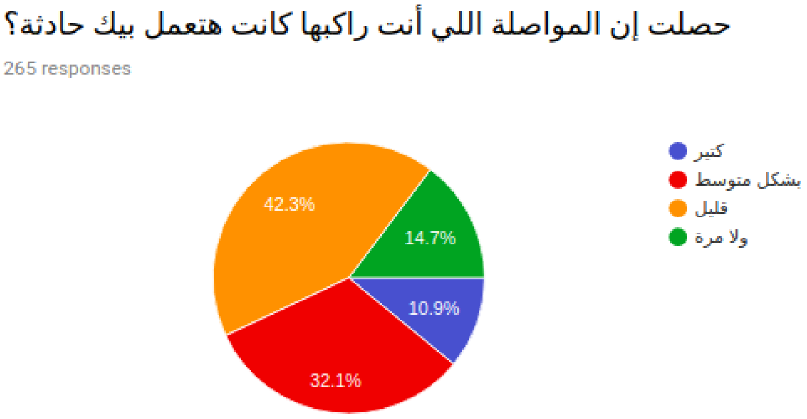
3. Did you face the problem of not finding enough change for the fare?

- Frequently: 65.3 %
- On average: 27.5 %
- Rarely: 7.2 %
- Never: 0 %



4. Have your transportation ever had an accident while you were taking it?

- Frequently: 10.9 %
- On average: 32.1 %
- Rarely: 42.3 %
- Never: 10.9 %

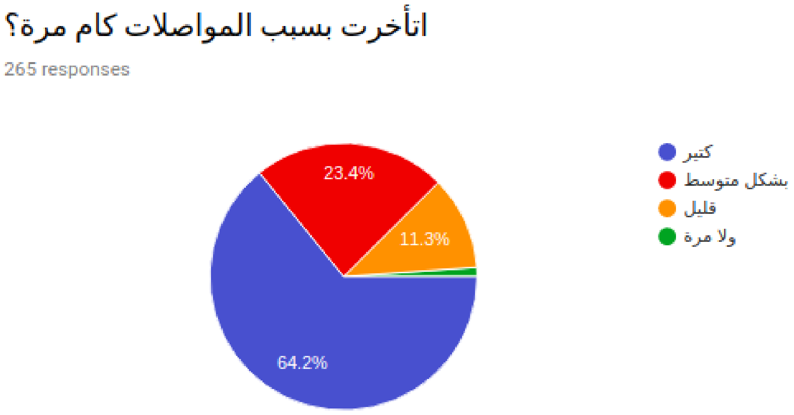


حصلت إن المواصلة اللي أنت راكبها كانت هتعمل بيك حادثة؟

265 responses

5. Did you face a delay because of your transportation?

- Frequently: 64.2 %
- On average: 23.4 %
- Rarely: 11.3 %
- Never: 1.1 %

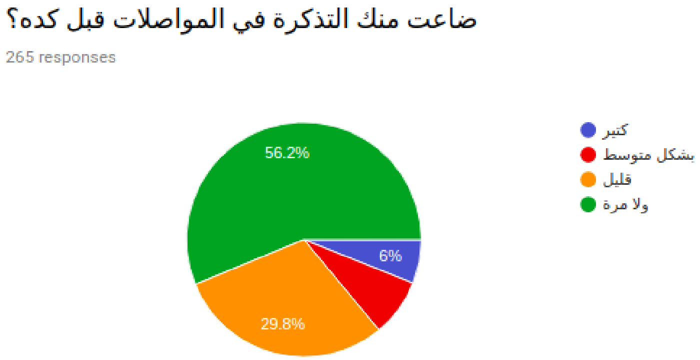


اتأخرت بسبب المواصلات كام مرة؟

265 responses

6. Did you lose your transportation ticket before?

- Frequently: 6 %
- On average: 8 %
- Rarely: 29.8 %
- Never: 56.2 %



ضاعت منك التذكرة في المواصلات قبل كده؟

265 responses

Previous Solutions

1. Existence of driver assistant

- Advantages:
  - Less distraction on the driver
  - Arrival delay is minimized
- Disadvantages:
  - Low reliability and efficiency
  - Other subproblems aren’t solved yet

2. Rechargeable and non-rechargeable transportation cards

- Advantages:
- Saves time and effort
- Disadvantages:
- Several cards distracts the holder
- Hard to track
- Still subproblems targeted aren’t solved yet

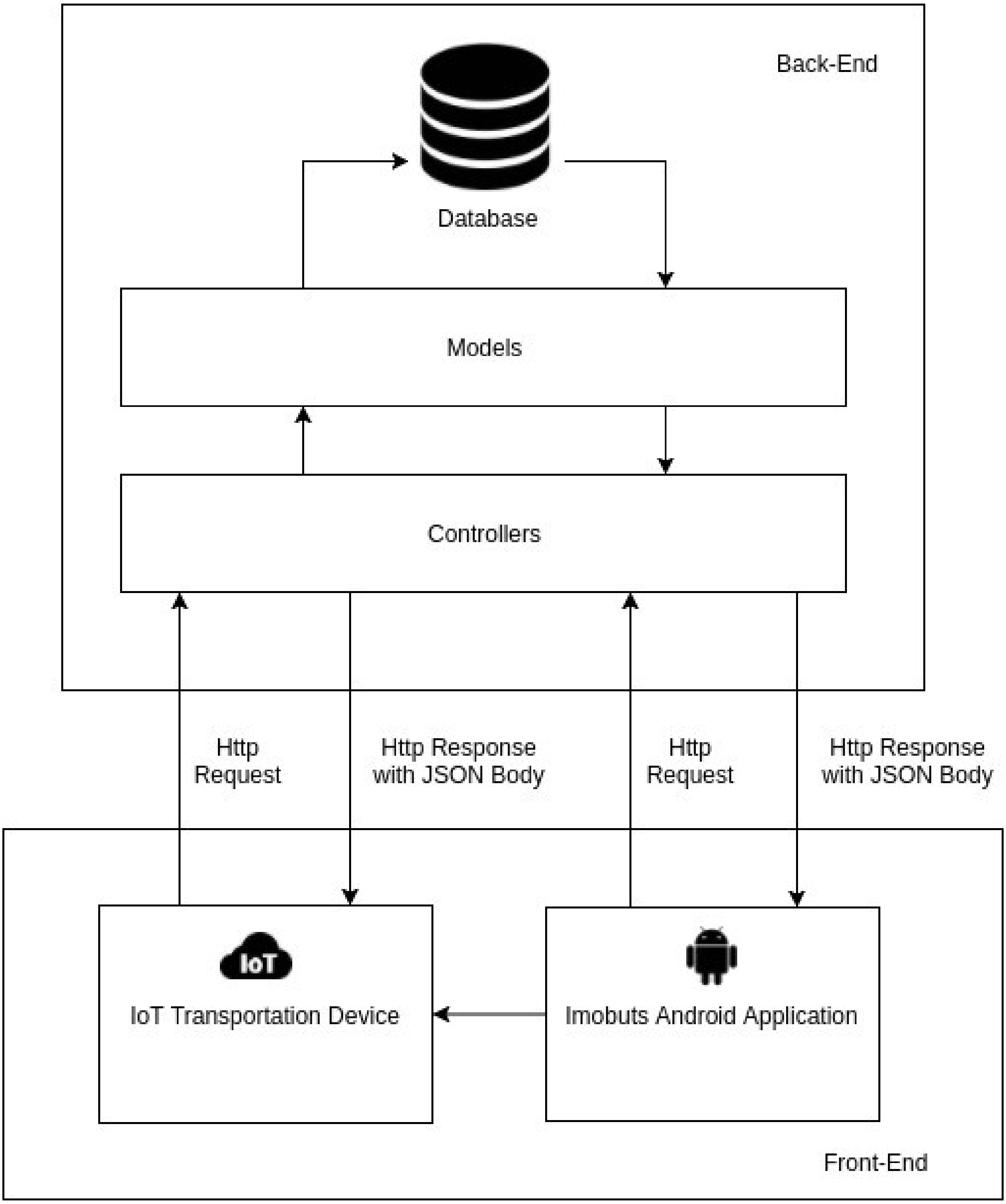
Our Digital Solution

- Advantages:
- No distraction on the driver
- No arrival delay
- No change problem
- No payment delay
- Saving ticket production

System analysis and Design

System Architecture

The system architecture used with imobuts is the well-known MVC architecture. The reasons for using it is the need of having a single back-end with multiple frond-end devices that depends on that back-end.



Back-End
Database
Models
Controllers
Http Request
Http Response with JSON Body
Http Request
Http Response with JSON Body
IoT
IoT Transportation Device
Imobuts Android Application
Front-End

Stakeholders

- Passenger (Main user)
- Vehicle driver
- System owner
- Transportation Policy Makers

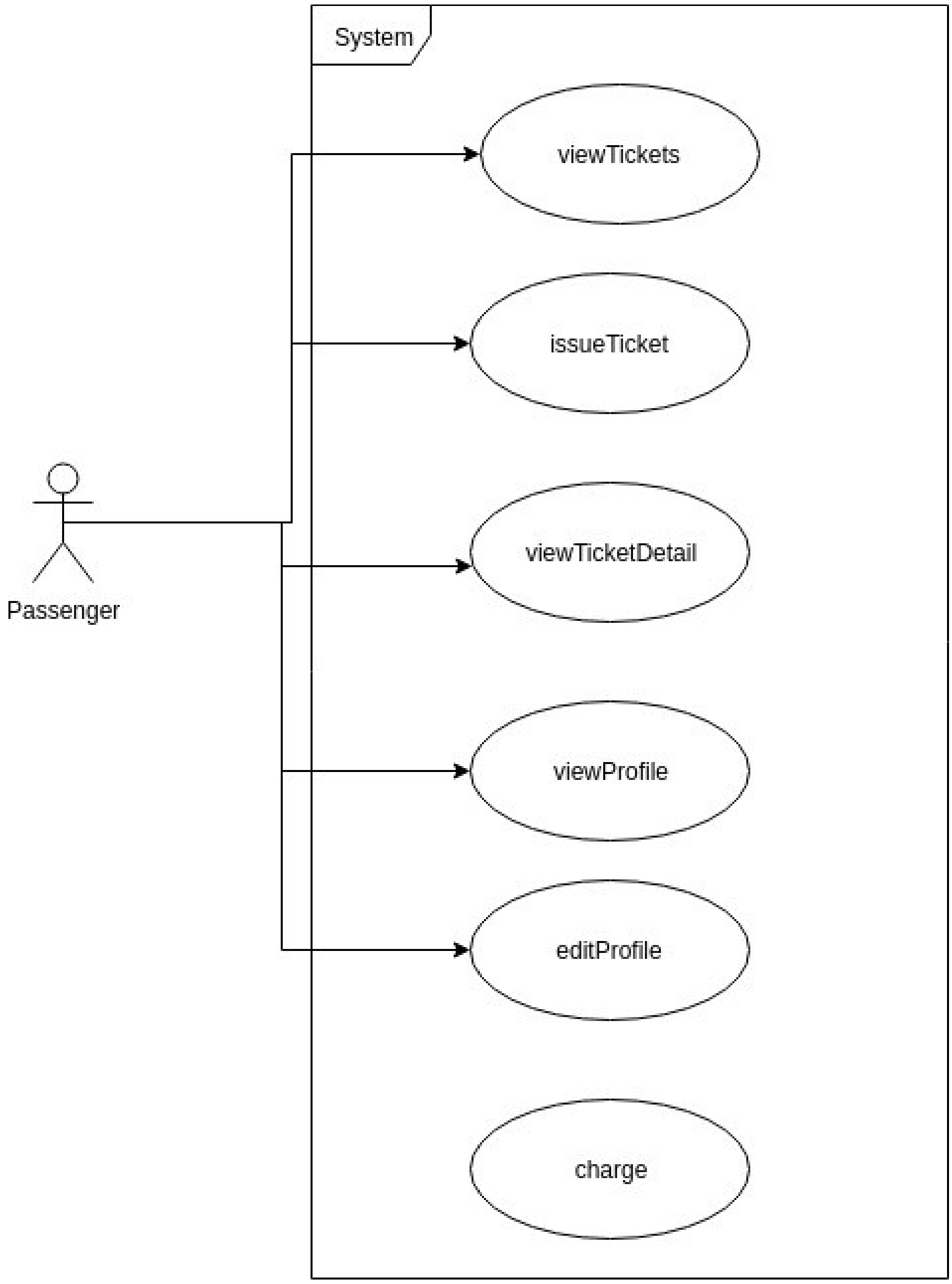
Functional Requirements

- Sign up
- Sign in
- Logout
- Forget password
- View profile
- Edit profile
- Charge
- View tickets
- View ticket details
- Issue ticket

Non-Functional Requirements

- Security
- Usability
- Performance
  - Response time
  - Accuracy
- Reliability
- Robustness

Use Case Diagram



Use Case Tables

1- viewTickets

Use Case ID:	001	
Use Case Name:	viewTickets	
Actors:	User “Passenger”	
Pre-conditions:	User already logged in	
Post-conditions:	All tickets have been cut by the user are shown up	
Flow of events:	**User Action**	**System Action**
	1- User clicks on “tickets” in the navbar	
		2- System views all tickets have been cut by this user sorted descendingly by date of cutting
Exceptions:	**User Action**	**System Action**
	-	
		-
Includes:	-	
Notes and Issues:	-	

2- issueTicket

Use Case ID:	002	
Use Case Name:	issueTicket	
Actors:	User “Passenger” - IoT Device	
Pre-conditions:	User already logged in & in tickets view & NFC is ON	
Post-conditions:	Ticket is sent to user with balance updated	
Flow of events:	**User Action**	**System Action**
	1- User clicks on “issue-ticket” button	
		2- System asks user to pass the mobile before the corresponding IoT device
	3- user passes mobile before the corresponding IoT device	
		4- System sends both balance, userId to the corresponding IoT device that sends (balance, user-Id , fees of this transportation) to the server where balance is updated (deducting the value of transportation fare) and ticket is sent to the user.
Exceptions:	**User Action**	**System Action**
	1- User has INVALID balance	
		2- IoT denies the Permission for cutting ticket
Includes:	-	
Notes and Issues:	-	

3- viewTicketDetails

Use Case ID:	003	
Use Case Name:	viewTicketDetails	
Actors:	User “Passenger”	
Pre-conditions:	User already logged in and all tickets are shown up	
Post-conditions:	Details of selected ticket are shown up	
Flow of events:	**User Action**	**System Action**
	1- User clicks on one of the tickets shown up.	
		2- Ticket-Details are shown up
Exceptions:	**User Action**	**System Action**
	-	
		-
Includes:	-	
Notes and Issues:	-	

4- viewProfile

Use Case ID:	004	
Use Case Name:	viewProfile	
Actors:	User “Passenger”	
Pre-conditions:	User already logged in	
Post-conditions:	Profile information is shown up.	
Flow of events:	**User Action**	**System Action**
	1- User clicks on “profile” in the navbar.	
		2- System views all User’s Details
Exceptions:	**User Action**	**System Action**
	-	
		-
Includes:	-	
Notes and Issues:	-	

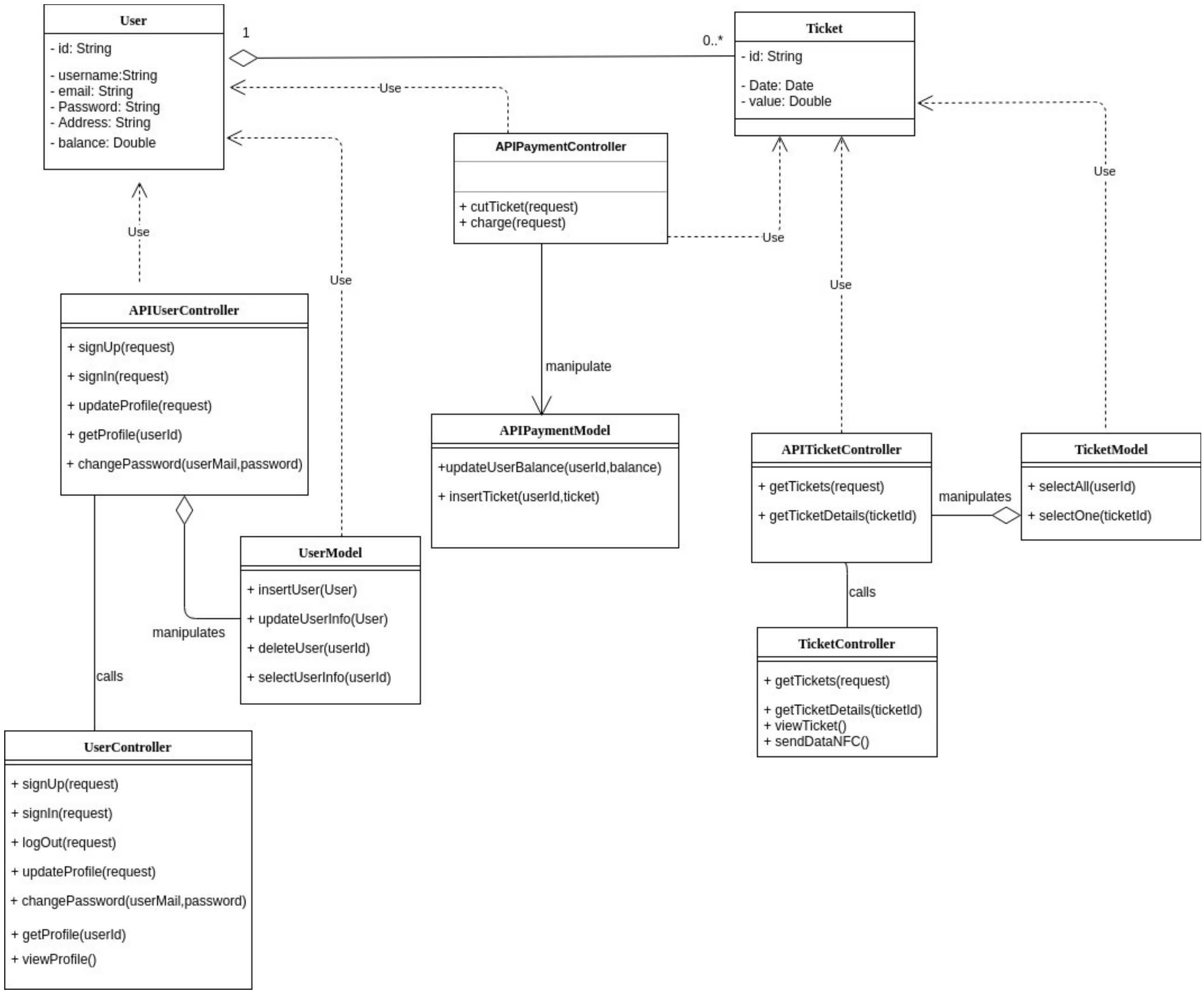
5- editProfile

Use Case ID:	005	
Use Case Name:	editProfile	
Actors:	User “Passenger”	
Pre-conditions:	User already logged and in the profile view.	
Post-conditions:	Profile information is updated.	
Flow of events:	**User Action**	**System Action**
	1- User clicks on “Edit profile” button	
		2- System views all Details to be updated.
	3- User updates details and press “save”	
		4- System validates data changed
		5.1- If INVALID data, System rejects updates, informs user and views Details again.
		5.2- If VALID data, Profile got updated.
Exceptions:	**User Action**	**System Action**
	-	
		-
Includes:	-	
Notes and Issues:	Balance won’t be included to be edited by user.	

6- charge

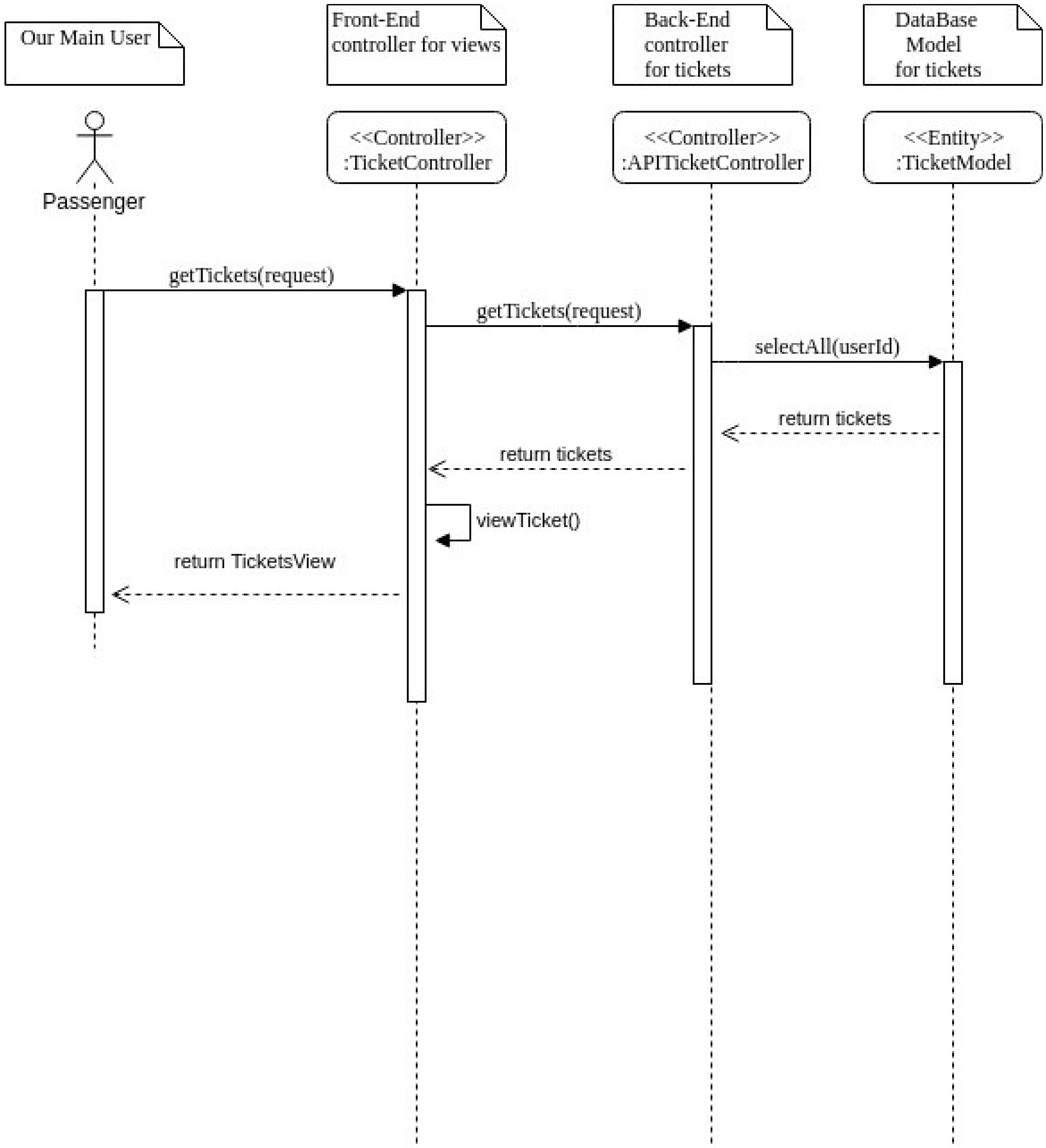
Use Case ID:	006	
Use Case Name:	charge	
Actors:	User “Passenger”	
Pre-conditions:	User has already logged in at the system and with Google account where he added a payment method and in the profile view	
Post-conditions:	Balance is updated with the given amount of money & notification message is sent to user.	
Flow of events:	**User Action**	**System Action**
	1- User clicks on “charge” button	
		2- System views text input for user to enter amount of money to be added to balance.
	3- User enters charge value and press “charge” button.	
		4.1- If user’s credit is enough for transaction, Balance is updated with the new charge value with notification message sent to user
		4.2- If user’s credit is Not enough, a failure message is sent to user.
Exceptions:	**User Action**	**System Action**
	-	
		-
Includes:	-	
Notes and Issues:	-	

Class Diagram

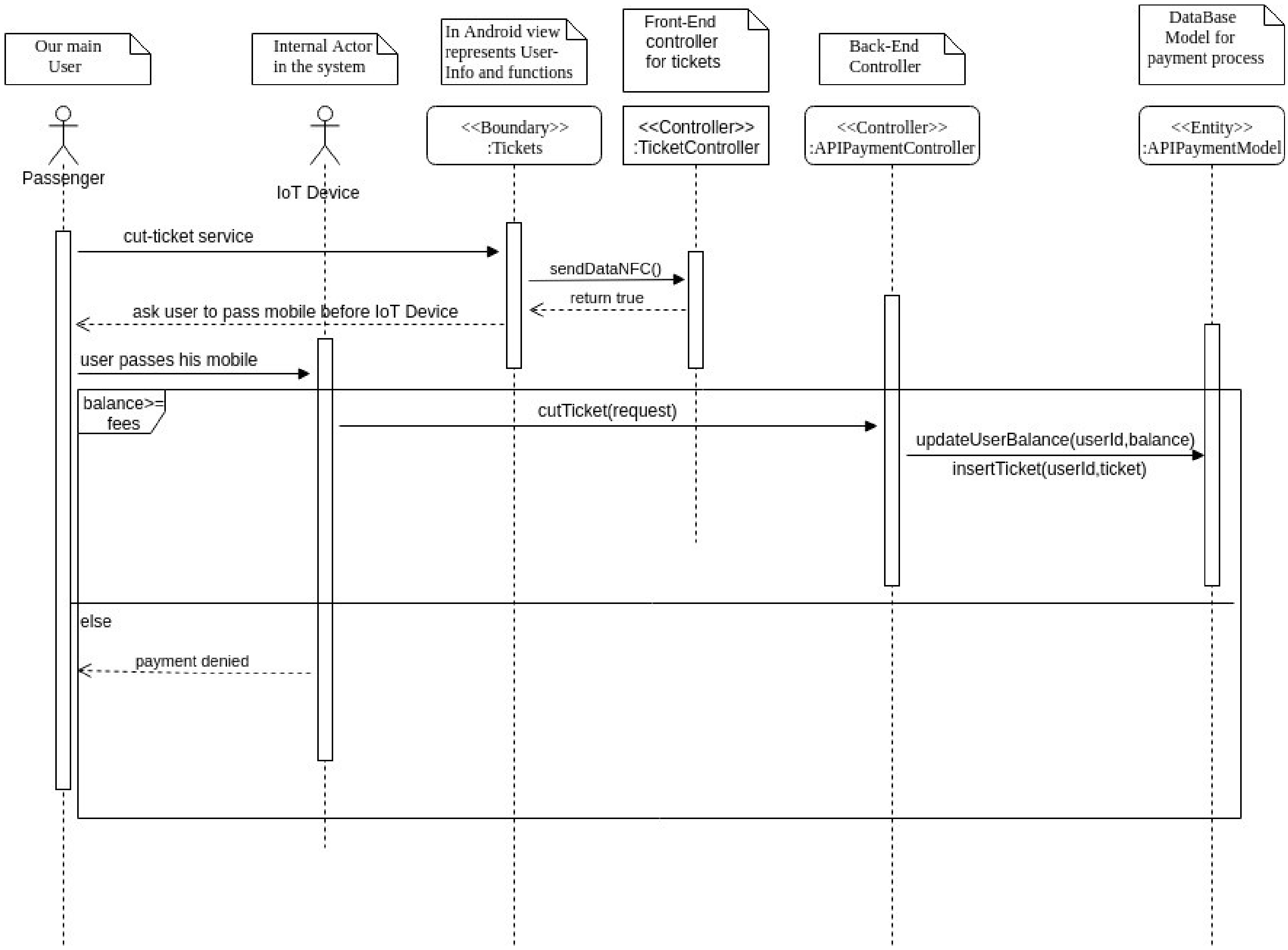


Sequence Diagrams

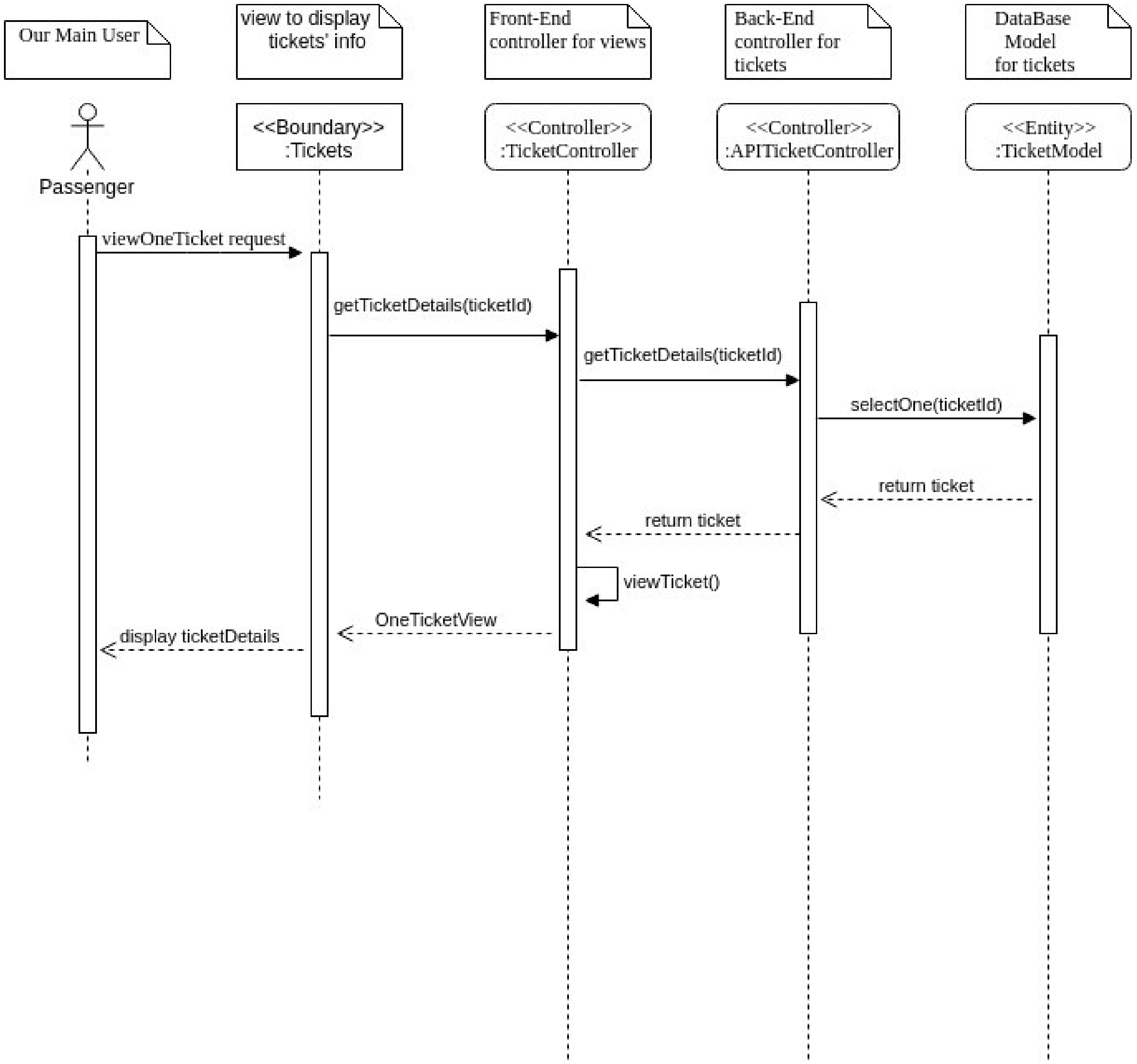
1- viewTickets



2- issue-ticket

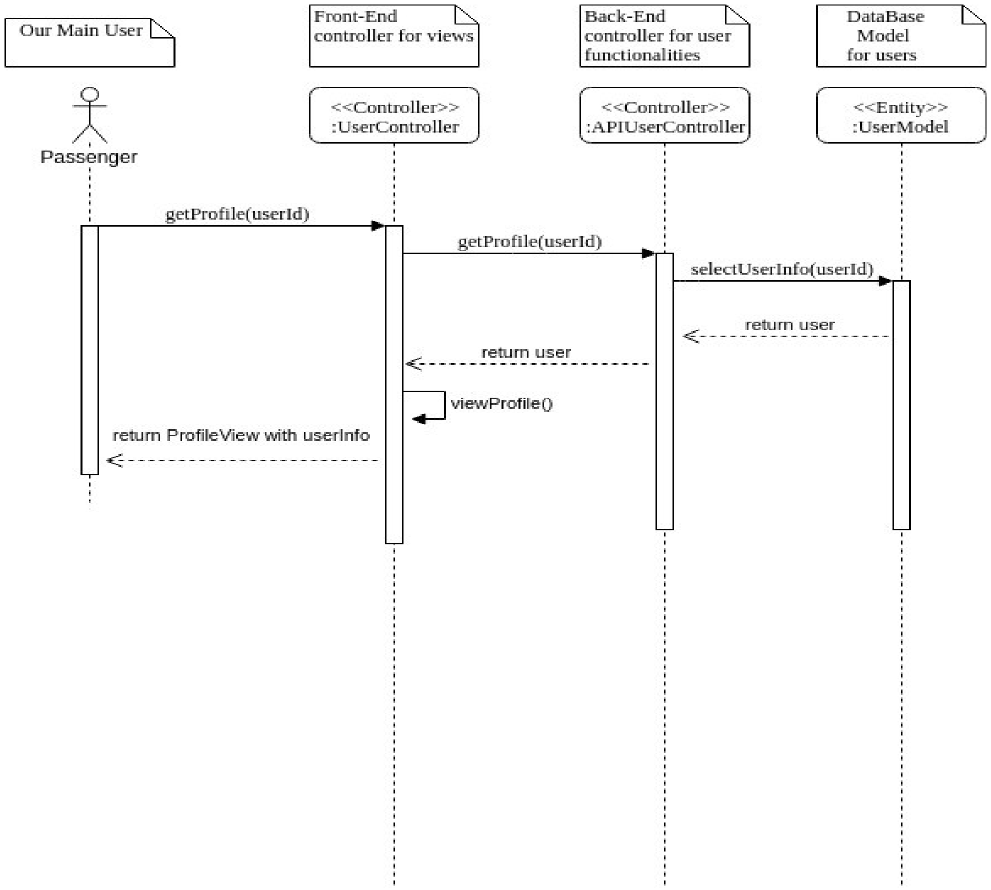

Our main User
Internal Actor in the system
In Android view represents User-Info and functions
Front-End controller for tickets
Back-End Controller
DataBase Model for payment process
Passenger
IoT Device
<<Boundary>> :Tickets
<<Controller>> :TicketController
<<Controller>> :APIPaymentController
<<Entity>> :APIPaymentModel
cut-ticket service
sendDataNFC()
return true
ask user to pass mobile before IoT Device
user passes his mobile
balance>= fees
cutTicket(request)
updateUserBalance(userId,balance)
insertTicket(userId,ticket)
else
payment denied

3- viewTicketDetails

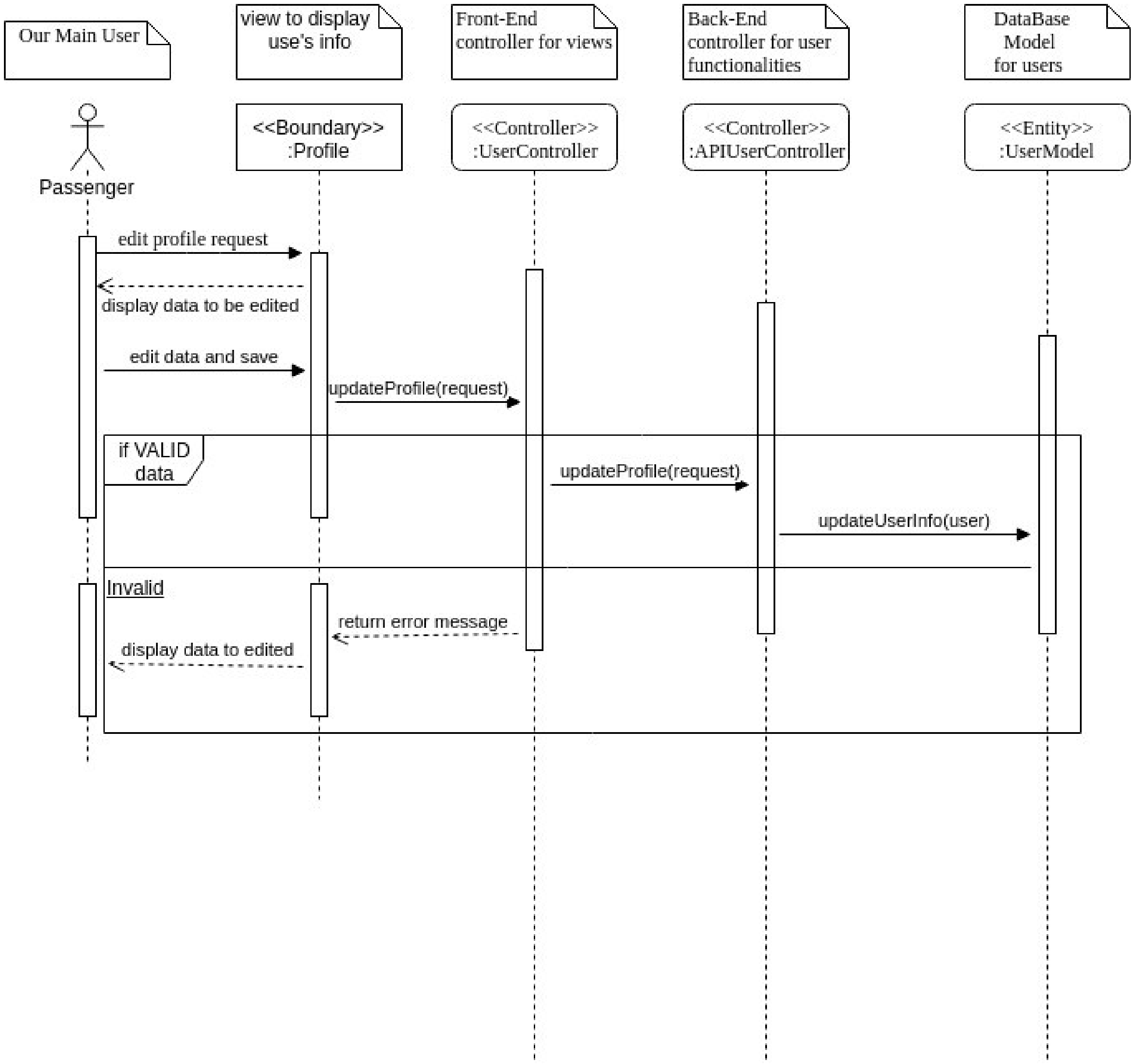


Our Main User
view to display tickets' info
Front-End controller for views
Back-End controller for tickets
DataBase Model for tickets
Passenger
<<Boundary>> :Tickets
<<Controller>> :TicketController
<<Controller>> :APITicketController
<<Entity>> :TicketModel
viewOneTicket request
getTicketDetails(ticketId)
getTicketDetails(ticketId)
selectOne(ticketId)
return ticket
return ticket
viewTicket()
OneTicketView
display ticketDetails

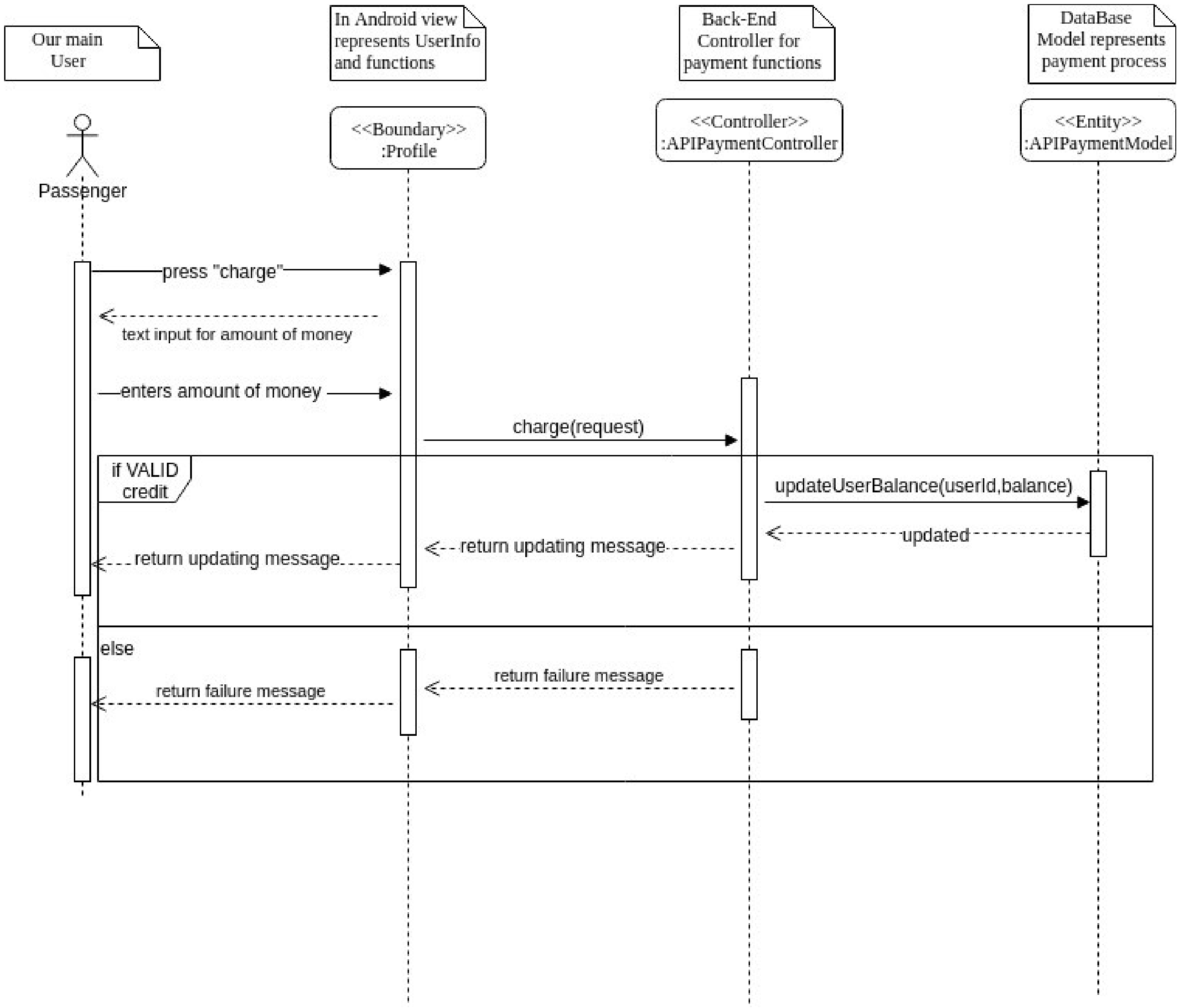
4- viewProfile


Our Main User
Front-End controller for views
Back-End controller for user functionalities
DataBase Model for users
Passenger
<<Controller>> :UserController
<<Controller>> :APIUserController
<<Entity>> :UserModel
getProfile(userId)
getProfile(userId)
selectUserInfo(userId)
return user
return user
viewProfile()
return ProfileView with userInfo

5- editProfile

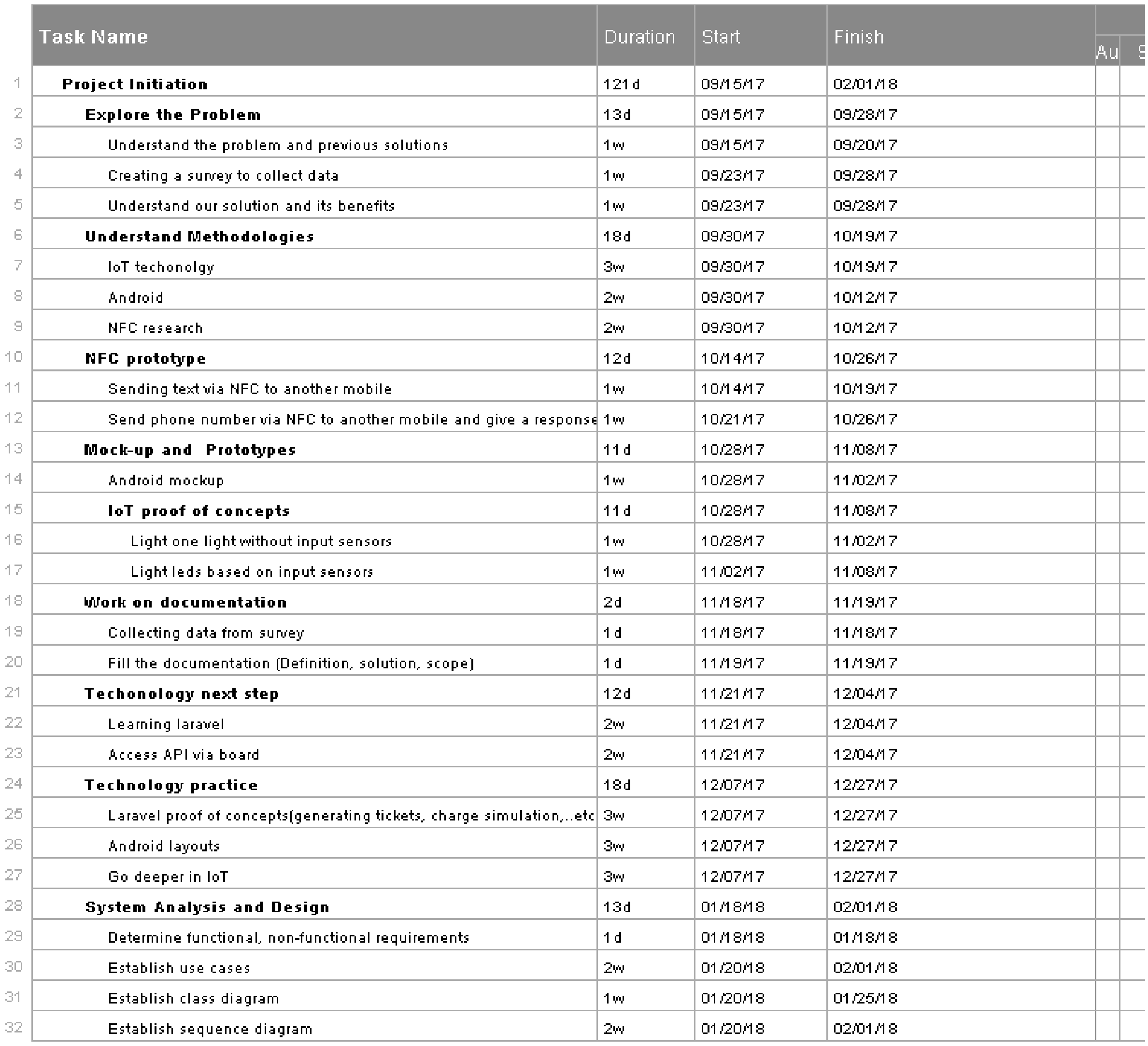


6- charge

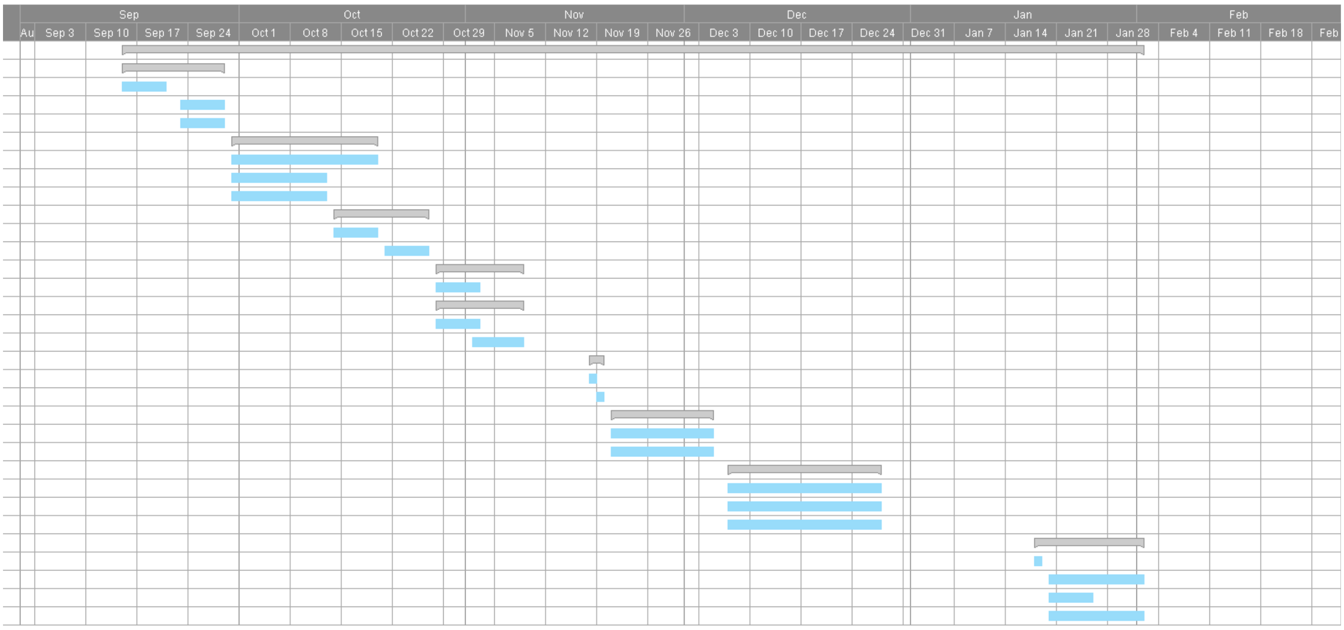

Our main User
In Android view represents UserInfo and functions
Back-End Controller for payment functions
DataBase Model represents payment process
Passenger
<<Boundary>> :Profile
<<Controller>> :APIPaymentController
<<Entity>> :APIPaymentModel
press "charge"
text input for amount of money
enters amount of money
charge(request)
if VALID credit
updateUserBalance(userId,balance)
updated
return updating message
return updating message
else
return failure message
return failure message

Work plan

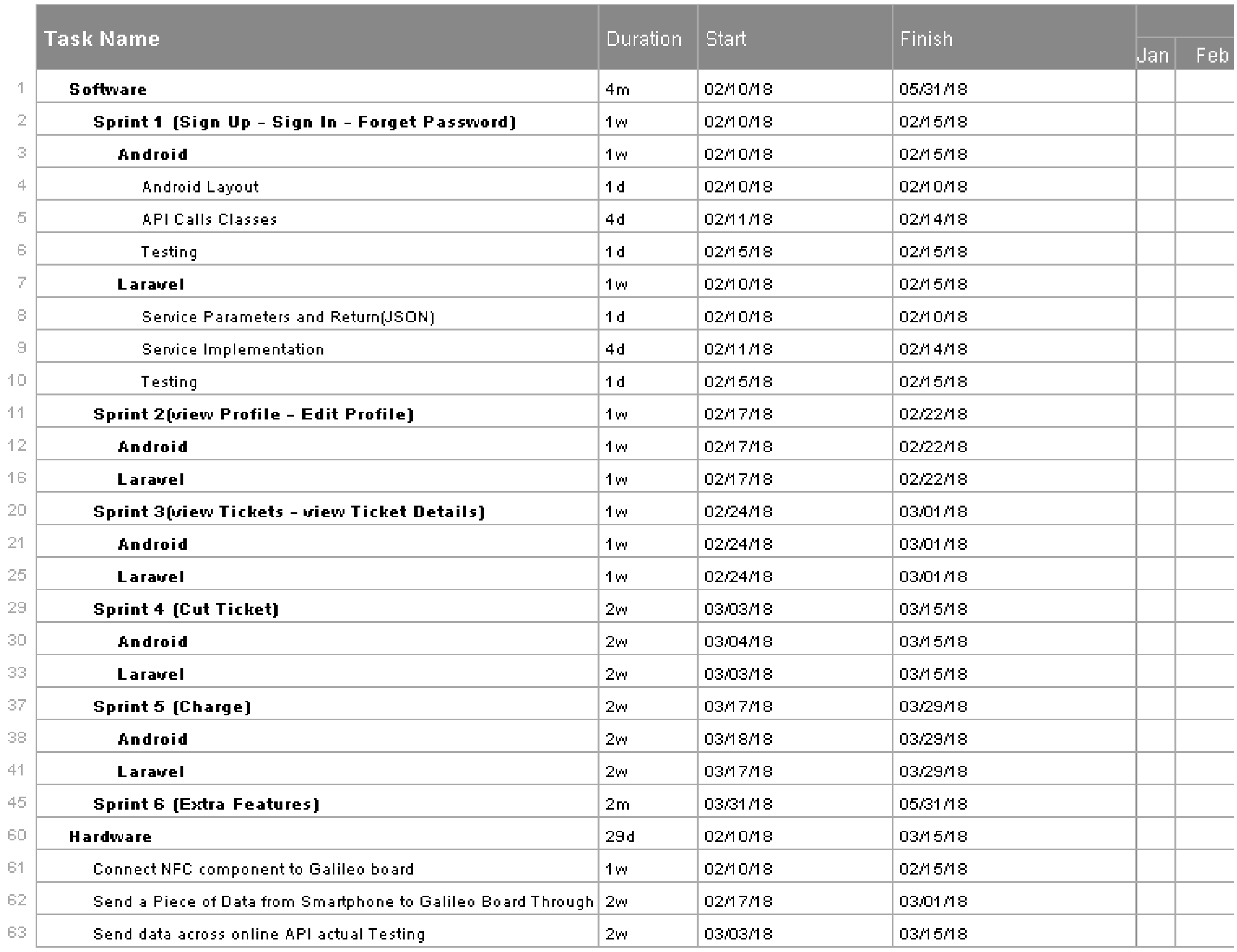
Completed



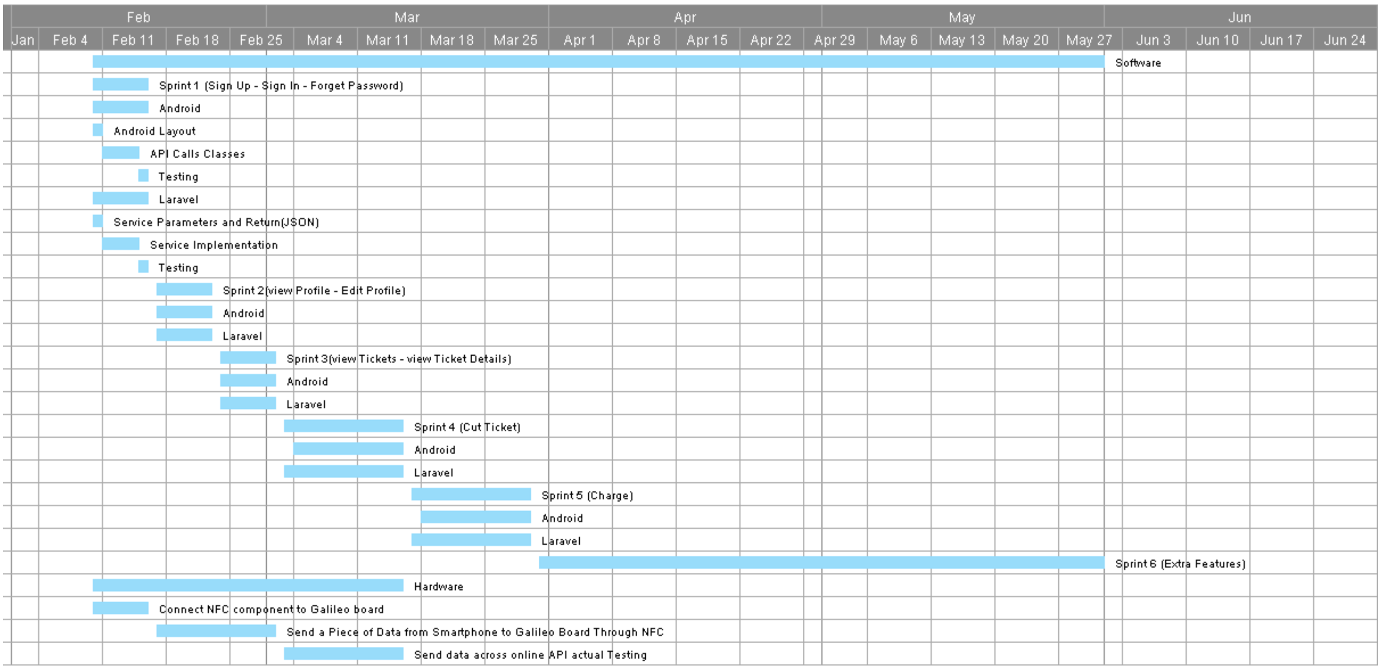
	Task Name	Duration	Start	Finish	Au S
1	**Project Initiation**	121d	09/15/17	02/01/18	
2	**Explore the Problem**	13d	09/15/17	09/28/17	
3	Understand the problem and previous solutions	1w	09/15/17	09/20/17	
4	Creating a survey to collect data	1w	09/23/17	09/28/17	
5	Understand our solution and its benefits	1w	09/23/17	09/28/17	
6	**Understand Methodologies**	18d	09/30/17	10/19/17	
7	IoT techonolgy	3w	09/30/17	10/19/17	
8	Android	2w	09/30/17	10/12/17	
9	NFC research	2w	09/30/17	10/12/17	
10	**NFC prototype**	12d	10/14/17	10/26/17	
11	Sending text via NFC to another mobile	1w	10/14/17	10/19/17	
12	Send phone number via NFC to another mobile and give a response	1w	10/21/17	10/26/17	
13	**Mock-up and Prototypes**	11d	10/28/17	11/08/17	
14	Android mockup	1w	10/28/17	11/02/17	
15	**IoT proof of concepts**	11d	10/28/17	11/08/17	
16	Light one light without input sensors	1w	10/28/17	11/02/17	
17	Light leds based on input sensors	1w	11/02/17	11/08/17	
18	**Work on documentation**	2d	11/18/17	11/19/17	
19	Collecting data from survey	1d	11/18/17	11/18/17	
20	Fill the documentation (Definition, solution, scope)	1d	11/19/17	11/19/17	
21	**Techonology next step**	12d	11/21/17	12/04/17	
22	Learning laravel	2w	11/21/17	12/04/17	
23	Access API via board	2w	11/21/17	12/04/17	
24	**Technology practice**	18d	12/07/17	12/27/17	
25	Laravel proof of concepts(generating tickets, charge simulation,..etc	3w	12/07/17	12/27/17	
26	Android layouts	3w	12/07/17	12/27/17	
27	Go deeper in IoT	3w	12/07/17	12/27/17	
28	**System Analysis and Design**	13d	01/18/18	02/01/18	
29	Determine functional, non-functional requirements	1d	01/18/18	01/18/18	
30	Establish use cases	2w	01/20/18	02/01/18	
31	Establish class diagram	1w	01/20/18	01/25/18	
32	Establish sequence diagram	2w	01/20/18	02/01/18	



Expected



	Task Name	Duration	Start	Finish	Jan	Feb
1	**Software**	4m	02/10/18	05/31/18		
2	**Sprint 1 (Sign Up - Sign In - Forget Password)**	1w	02/10/18	02/15/18		
3	**Android**	1w	02/10/18	02/15/18		
4	Android Layout	1d	02/10/18	02/10/18		
5	API Calls Classes	4d	02/11/18	02/14/18		
6	Testing	1d	02/15/18	02/15/18		
7	**Laravel**	1w	02/10/18	02/15/18		
8	Service Parameters and Return(JSON)	1d	02/10/18	02/10/18		
9	Service Implementation	4d	02/11/18	02/14/18		
10	Testing	1d	02/15/18	02/15/18		
11	**Sprint 2(view Profile - Edit Profile)**	1w	02/17/18	02/22/18		
12	**Android**	1w	02/17/18	02/22/18		
16	**Laravel**	1w	02/17/18	02/22/18		
20	**Sprint 3(view Tickets - view Ticket Details)**	1w	02/24/18	03/01/18		
21	**Android**	1w	02/24/18	03/01/18		
25	**Laravel**	1w	02/24/18	03/01/18		
29	**Sprint 4 (Cut Ticket)**	2w	03/03/18	03/15/18		
30	**Android**	2w	03/04/18	03/15/18		
33	**Laravel**	2w	03/03/18	03/15/18		
37	**Sprint 5 (Charge)**	2w	03/17/18	03/29/18		
38	**Android**	2w	03/18/18	03/29/18		
41	**Laravel**	2w	03/17/18	03/29/18		
45	**Sprint 6 (Extra Features)**	2m	03/31/18	05/31/18		
60	**Hardware**	29d	02/10/18	03/15/18		
61	Connect NFC component to Galileo board	1w	02/10/18	02/15/18		
62	Send a Piece of Data from Smartphone to Galileo Board Through	2w	02/17/18	03/01/18		
63	Send data across online API actual Testing	2w	03/03/18	03/15/18		


Feb
Mar
Apr
May
Jun
Jan
Feb 4
Feb 11
Feb 18
Feb 25
Mar 4
Mar 11
Mar 18
Mar 25
Apr 1
Apr 8
Apr 15
Apr 22
Apr 29
May 6
May 13
May 20
May 27
Jun 3
Jun 10
Jun 17
Jun 24
Software
Sprint 1 (Sign Up - Sign In - Forget Password)
Android
Android Layout
API Calls Classes
Testing
Laravel
Service Parameters and Return(JSON)
Service Implementation
Testing
Sprint 2(view Profile - Edit Profile)
Android
Laravel
Sprint 3(view Tickets - view Ticket Details)
Android
Laravel
Sprint 4 (Cut Ticket)
Android
Laravel
Sprint 5 (Charge)
Android
Laravel
Sprint 6 (Extra Features)
Hardware
Connect NFC component to Galileo board
Send a Piece of Data from Smartphone to Galileo Board Through NFC
Send data across online API actual Testing