ANDROID SUB-URBAN RAILWAY TICKETING USING GPS AS TICKET CHECKER

Sana Khoja, Maithilee Kadam
Department of Computer Engineering
STES. Smt. Kashibai Navale College Of Engg,
Vadgoan(BK), Pune, India

***Abstract*— One of the major challenges in present ticketing provision is "QUEUE" in buying suburban railway tickets. In this rapid growing world of technology user buy with oyster and octopus cards for suburban tickets, this is frustrating at times to stand in the queue or if user forgot his cards. Android Suburban Railway(ASR) ticketing is to buy the suburban tickets which is the most difficult when compared to booking the long journey tickets through 'M-ticket'. ASR ticket can be bought with smart phone application, where user can carry his railway tickets in his smart phone as QR-code.**

It uses the smart phone's "GPS" facility to authenticate and erase ticket automatically after a specific interval of time when user reaches to the destination. User's ticket information is stored in a CLOUD database for security, which is absent in present suburban system. Also the ticket checker is provided with a checker application to look for user's ticket with the ticket number in the cloud database for checking purposes.

***Index Terms*—Android, railway ticketing, qr- code cloud etc**

I. Introduction

In the past years, it has been rapid growth in the field of technology. If we consider railway department, e-ticket facility was introduced, where users could browse through an official website and book their long journey tickets which can be printed out after confirmation to show it to the checker when needed. Few months later a new technology called M-ticketing (mobile-ticketing) was introduced where customers messaged to a web- portal using their mobile phones after which a complete web page was downloaded to their phones where user could do the same booking process as in the e-ticketing. The use of oyster and octopus cards has become mandatory in foreign countries. The problem arises when the user forgot to carry the travel cards along with and stand in the queue for local suburban tickets, the user could suffer.

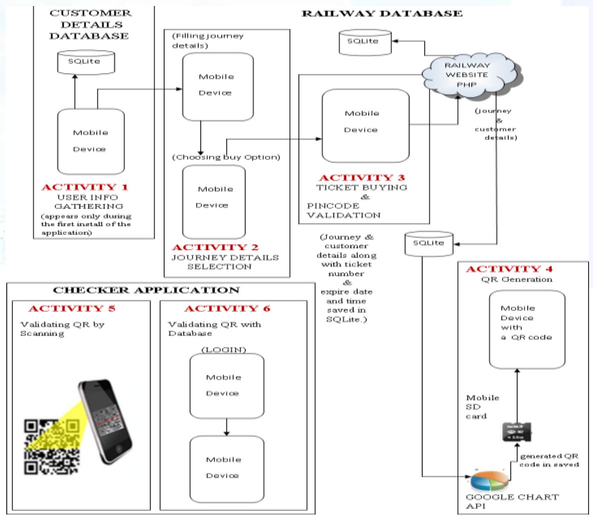
Android Suburban Railway (ASR) ticketing is mainly to buy suburban tickets which just a smart phone application, In which you can carry your ASR ticket in your smart phone as QR-code(Quick-Response). It uses the smartphone's "GPS"

Facility to validate. The ticket gets automatically deleted after specific interval of time once the user reaches the destination. It calculates the timing of jouney and then the ticket gets invalid after exceeding that time interval. Ticket information is stored in a cloud database for security purpose which is missing in the present suburban system. On the other side, the ticket checker has a checker application to search and validate the user's ticket information which is been stored in the cloud. The ticket checker scans the given ticket and compare it with the database provided.

II. Need of the this Application Software

Time is the major concern and threat in the today's world. In order to make minimum use of the time in the field of travelling this android suburban software was introduced. By which travelling becomes very simple. Booking of the tickets now doesn't need standing in along queue. By this tickets can be booked through phone only. It provides all the information of the trains to be enquired. User needs to enter their login id and confirm the ticket. Ticket is then stored in the phone itself. Carrying the tickets or missing the ticket problem is been solved here. This makes booking and handling easier. The main aim of this software is providing the user ease.

III. System Design:


CUSTOMER DETAILS DATABASE
SQLite
Mobile Device
ACTIVITY 1
USER INFO GATHERING
(appears only during the first install of the application)
(Filling journey details)
Mobile Device
(Choosing buy Option)
Mobile Device
ACTIVITY 2
JOURNEY DETAILS SELECTION
RAILWAY DATABASE
SQLite
RAILWAY WEBSITE PHP
Mobile Device
ACTIVITY 3
TICKET BUYING & PINCODE VALIDATION
(journey & customer details)
(Journey & customer details along with ticket number & expire date and time saved in SQLite.)
SQLite
ACTIVITY 4
QR Generation
Mobile Device with a QR code
Mobile SD card
generated QR code in saved
GOOGLE CHART API
CHECKER APPLICATION
ACTIVITY 5
Validating QR by Scanning
ACTIVITY 6
Validating QR with Database
(LOGIN)
Mobile Device
Mobile Device
Fig 1:System Architecture

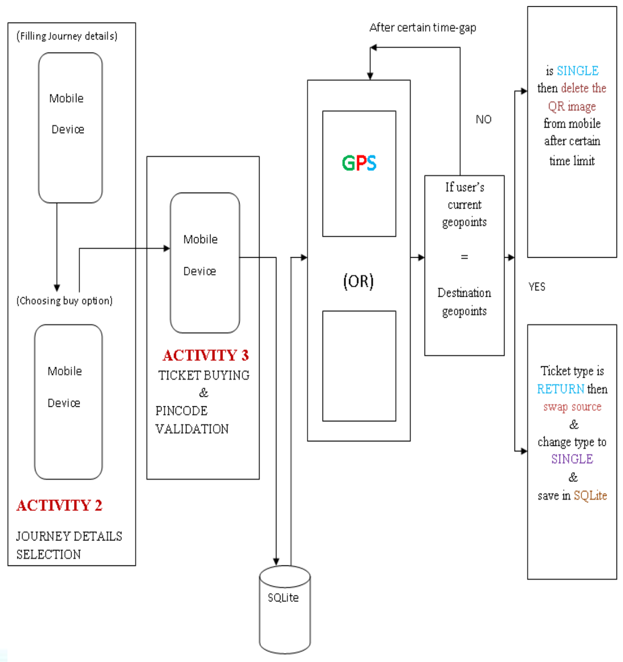

(Filling Journey details)
Mobile
Device
(Choosing buy option)
Mobile
Device
ACTIVITY 2
JOURNEY DETAILS SELECTION
Mobile
Device
ACTIVITY 3
TICKET BUYING
&
PINCODE VALIDATION
SQLite
After certain time-gap
GPS
(OR)
NO
YES
If user's current geopoints = Destination geopoints
is SINGLE then delete the QR image from mobile after certain time limit
Ticket type is RETURN then swap source & change type to SINGLE & save in SQLite

Fig 2: Ticket Validation Process

1. CLIENT SIDE:- LOGIN USER:

The user install this software in their respective android phone. The user does this by adding all his personal information required for the login. The information includes his name, address, telephone number, city etc. Once the information is filled the user gets his login id by which he can access his account and use the application. The login id is used each time the user wants to buy the ticket. Al the personal information is stored in the cloud database.

2. TICKET BOOKING:

In this process after getting the login id the user can access the application to book his tickets. For this the user has to choose the source and destination address of his journey. The type of ticket depending upon personal expenses is done. The user can also select the ac and other reservation techniques. Available tickets are shown and the booking is done accordingly. And all the information is stored in the SQL server database.

3. PIN CODE JUSTIFICATION:

After booking the tickets and filling the required information. When the customer press the buy button a PHP code in the railway server validates the pin number and passwords, it saves both journey details and customer info in the server's MYSQL database. After this ticket number and time of buying is generated by PHP code and balance credit value is displayed.

4. QR CODE FORMATION:

Once the PHP code generates the ticket number and time of buy details saved in the MySQL database are sent to Google Chart API engine in order to generate the QR code. All the individual information and the ticket details is then stored in the QR code and then used to validate by the ticket checker.

5. *CHECKER: - GPS TICKET ASSESMENT:*

GPS is an important part of the software. With the help of the GPS the current location of the user is been found out. The ticket assessment is done with the help of the GPS. The info that is stored in the database is compared with the user's geo points and the validation is carried out. If the geo points of the user exceeds the destination address then the ticket is turned invalid. The time is been calculated with the help of GPS.

6. VALIDATING QR CODE AND QR READER:

This task is handled by the ticket checker. The ticket checker is given the QR validation software by which he can check and scan the QR code. The ticket checker clicks the image of the code and then scans. The information that is stored in the database is then compared with the QR code and the ticket is validate. Once the ticket is validate and reached the destination the ticket info is been deleted from the SQL server database.

7. *IN CASE OF SYSTEM FAILURE:*

In case of failure like display damage or battery problem the QR code is not visible to the user and checker then the hassle is been solved with the help of the ticket number and pin code number. The ticket number is used by the ticket checker and then the ticket information is validated through dadabase.

IV. HARDWARE AND SOFTWARE REQUIRMENTS

- Total ram
- Hard disk
- Android 2.1pk handset
- Atleast p4 machine with 2 gb ram
- Android SDK 2.1
- JDK 1.6
- Tomcat 6.0
- MYSQL 6.0
- Eclipse

V. MATHEMATICAL MODEL

1. **Set Theory Analysis**

1. Let 'S' be the 'Android Suburban Railway Ticketing'.

S= {…………….}

Set S is divided into 6 modules

S= {S1, S2, S3, S4, S5, S6}

S1= GUI Handler (GH)

S2= Location Manager (LM)

S3= Railway Ticket Logic (RTL)

S4= QR Code Generator (QRG)
S5= QR Code Detector (QRD)
S6= Ticket Checker (TC)

Identify the inputs.
Inputs = {X1, X2, X3, ……..Xn}
X1= Location Information
X2= Railway Ticket Information
2. Identify the output as O.
Outputs = {Y1, Y2, Y3, ……..Yn}
Y1= Railway Ticket
Y2= QRCode Image
Y3= Ticket Updater

SET THEORY

Sr. No.	**Description**	**UML Design Observations.**
1.	**Problem Description**	
	Let S be a system which do Android Suburban Railway Ticketing; suchthat S = {S1,S2,S3,S4,S5, S6} where S1 represent GUI Handler (GH); S2 represent Location Manager (LM); S3 represent Railway Ticket Logic (RTL); S4 represent QRCode Generator (QRG); S5 represent QRCode Detector (QRD); S6 represent Ticket Checker (TC)	S holds list of modules in the system.
2.	**Activities**	
	2.1 Activity I User Login Process. Let S1 be a set of User's parameters for login. S1= {userid, password} Where, userid – User id of the user password – password of the user	If userid/password of the user is valid then proceed Else discard the user

Condition/Parameter	*Operation/Function*
If user== valid user	f1:Proceed()
Else..	Discard user

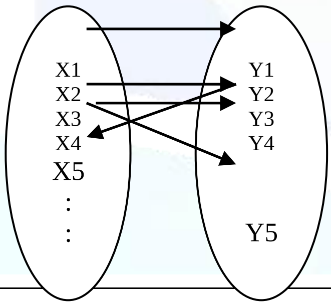
	2.2 Activity II **Railway Ticket Booking Process** Lets S2 be a set of ticket booking parameters: S2={userid, src_station, destination_station, no_of_seats} Where, userid – userid of the user src_station – source station name destination_station – destination station name no_of_seats– no of seats to be book	Search the required seats in the train running from source to destination train. If seats available the book those seats for user Else throw error

Condition /Parameters	*Operation / Function*
no_of_seats	f1:Search();
If (no_of_seats are available) Book seats Else Throw error	f2:findSeats() ; f3:error()

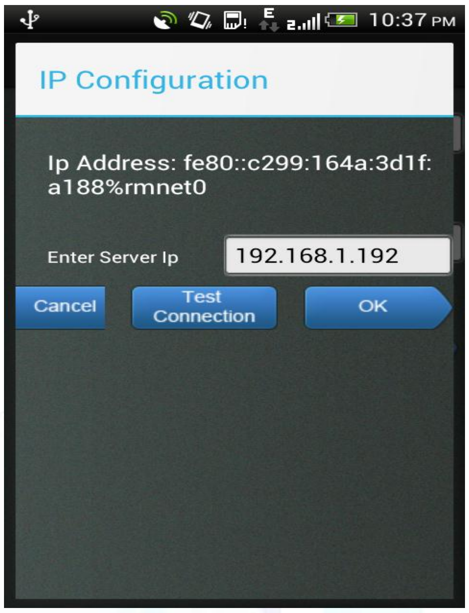
	2.3 Activity III **Ticket Checking Process** Let S3 be the set of parameters to validate ticket. S3:{user_id, ticket_id, return_ticket}	If the user's location

	Where, user_id – userid of the user ticket_id – ticket id of the user ticket return_ticket - is return ticket	is equals to destination station location, invalidate ticket. Else Try for next geo location

Condition/ Parameter	*Operation /Function*
ticket_id	F1:Validate()
If(geo location of user is same as destination location) Invalidate Ticket Else Try for next geo locations	F2:CheckLocation() F3: Invalidate() F4: CheckLocation()

3.	**Venn Diagrams** As described above in entire Process: Railway Ticketing using GPS Input(Ticket & Location Info) Output(Ticket) 	

VI. IMPLEMENTATION RESULTS

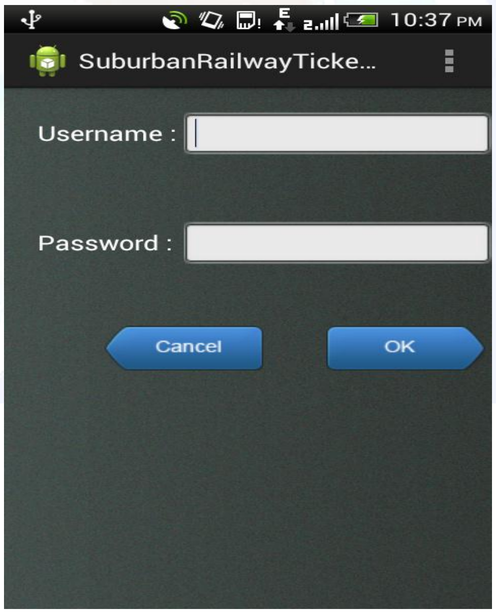


IP Configuration

Ip Address: fe80::c299:164a:3d1f:a188%rmnet0

Enter Server Ip 192.168.1.192

- Before connecting to the software we need to gain the IP address. The IP address is the medium for the connection to the server. After getting the IP configuration the user then can proceed further for the login details.
- The user has to have the IP address for connecting to the server. Also the user has to first sign up on the server and then use the app to login and do the latter transactions.

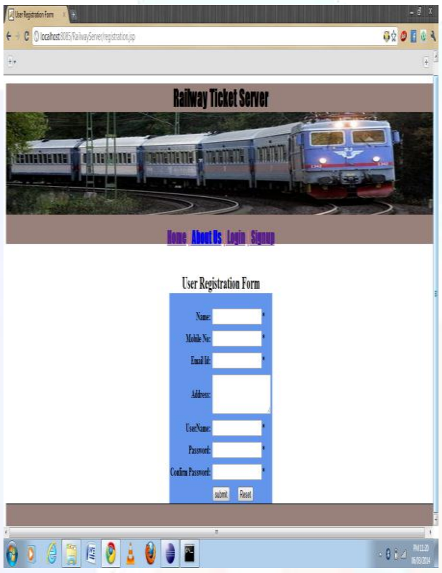


SuburbanRailwayTicke...

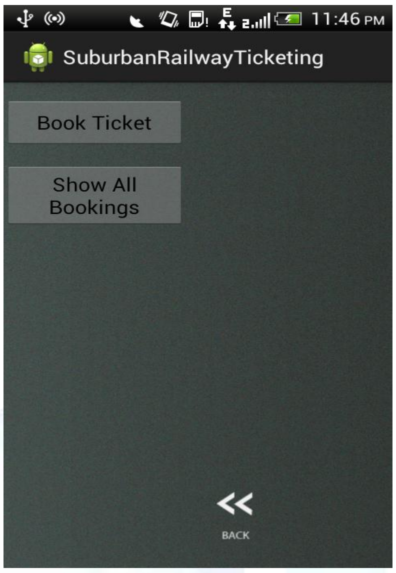
Username :

Password :

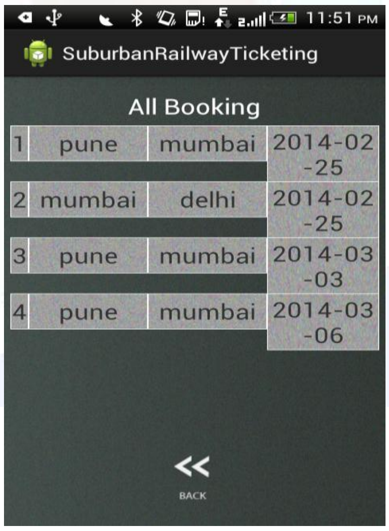
- After collecting the IP address the user can then proceed for the details of login.
- It is compulsory that the user first signs up on the server.
- User can submit the username and password for booking the tickets.
- This information is stored in the database and then can be used whenever the user wants to book the tickets.
- The user won't be able to login if the user doesn't of the either two; i.e. gain the wrong IP address or submit the incorrect password.



- For booking the ticket the user has to give his complete information like his name, address, phone number, email address etc.
- After giving all the information demanded, the user can then book his ticket.
- If the information isn't valid, the user won't be able to book his ticket.
- Every textbox has a limit of its appropriate specification. Exceeding the limit, the user will be notified about it and asked to correct it.
- The info is stored in the database and then user can directly access the software.



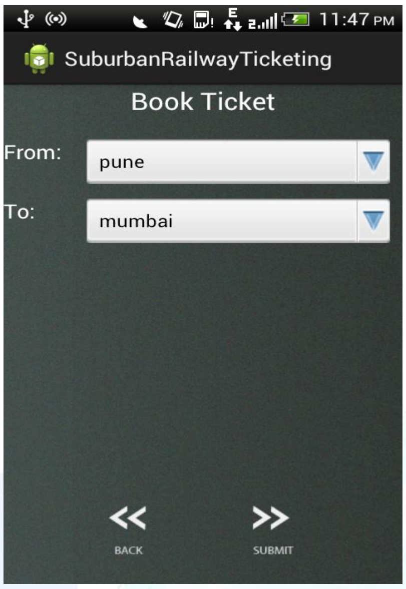
- User can then book his ticket accordingly.
- The "Book Ticket" button will direct the user to the page where he is supposed to fill in all the required information. If valid, this ticket as booked.
- The "Show All Bookings" button will direct the user to a page where all his booking history will be displayed.
- All the booking details are been displayed.



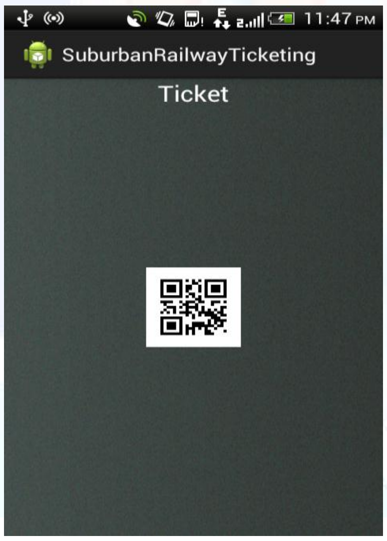
All Booking

1	pune	mumbai	2014-02-25
2	mumbai	delhi	2014-02-25
3	pune	mumbai	2014-03-03
4	pune	mumbai	2014-03-06

- All the booking details are then displayed.
- With this clear idea is obtained for the user.



- Here the user enters the source information and the destination address he wants to travel.
- And book the ticket likewise
- The user can only book tickets for sub-urban travel.



- After booking the tickets and paying, the QR code is generated.
- The QR code contains all the information related to the user and the ticket he has booked.
- This QR code is stored in the database of the user, i.e. gallery. The QR code, after travelling from source to destination, is later deleted from the server database as well as from the user's phone.
- The ticket checker then can scan this with the help of ticket checker application.

VII. CONCLUSION

Mobile ticket application developed for Android 1.5 using Java, SQLite, MySQL, and PHP on the server side which can change the way people buy their tickets in future. This kind of ticketing application can be applied to any kind of transport system. Our android app is one of its kinds and finds huge application to buy sub-urban railway tickets through android mobile. Also our app saves a huge work for our ticket checkers by GPS validation of tickets and also moving from manual ticket checking process to digital ticket checking process by just scanning with his own android mobile to validate the ticket.

Knowing at what time trains will be available will also ease the user to allot his time accordingly to reach the station, so in our project we will be using GPS here to find the location of the user and nearby train station to display the train arrival timings.

REFERENCES

[1] Damon Oehlman and Sebastien Blanc (20 II)" Pro Android Web Apps develop for Android using HTML5,CSS3 &JavaScript "-Apress Publications.

[2] Dave Smith and Jeff Friesen's (2011)" Android Recipes A Problem Solution Approach" - Apress Publications.

[3] Jeff" JavaJeff" Friesen's (2010) "LearnJavafor Android Development" - Apress Publications.

[4] Lauren Darcey and Shane Conder (2010)" Sams Teach Yourself Android Application Development" - Sams Publications.

[5] Mark Murphy's (2011)" Beginning Android 3" - Apress Publications.

[6] Reto Meier (2009)" Professional Android Application Development" - Wiley Publishing Inc.

[7] Satya Komatineni (2009) " Pro Android" - Apress Publications.

[8] Shawn Van Every's (2009) " Pro Android Media developing Graphics, Music , Video and Rich Media Apps for Smartphones and Tablets" - Apress Publications.

[9] Wallace Jackson's (2011) "Android Appsfor Absolute Beginners"Apress Publications.

[10] Wei - Meng Lee (20 II)" Beginning Android Application Development" - Wiley Publishing Inc..