Android Suburban Railway Ticketing with GPS as Ticket Checker

Sanaan Refai
Associate Professor
Department of Computer Engineering
Padmabhushan Vasantdada Patil Prathisthan’s College of Engineering
Mumbai University

Akash Kumhar
P.G. Scholar
Department of Computer Engineering
Padmabhushan Vasantdada Patil Prathisthan’s College of Engineering
Mumbai University

Sarthak Kothari
Associate Professor
Department of Computer Engineering
Padmabhushan Vasantdada Patil Prathisthan’s College of Engineering
Mumbai University

Arpita Raut
P.G. Scholar
Department of Computer Engineering
Padmabhushan Vasantdada Patil Prathisthan’s College of Engineering
Mumbai University

Vinod Alone
P.G. Scholar
Department of Computer Engineering
Padmabhushan Vasantdada Patil Prathisthan’s College of Engineering
Mumbai University

Abstract

One of the biggest challenges in the current ticketing facility is "QUEUE" in buying our suburban railway tickets. In this fast growing world of technology we still stand in the queue or buy with ATVM smart cards and coupons for our suburban tickets, which is more frustrating at times to stand in the queue or if we forget our cards. This paper Android Suburban Railway (ASR) ticketing is mainly to buy the suburban tickets which is the most challenging when compared to booking the long journey tickets through 'M- ticket' which fails with suburban(local travel) tickets. Our ASR ticket can be bought with just a smart phone application, where you can carry your suburban railway tickets in your smart phone as a QR (Quick Response) code. It uses the smart phones "GPS" facility to validate your ticket. User's ticket information is stored in a Cloud database for security purpose which is missing in the present suburban system. Also the ticket checker is provided with a checker application to search for the user's ticket with the ticket number in the cloud database for checking purposes.

Keywords: Android; SQLite; Cloud Database; ASR; QR code

I. INTRODUCTION

In the past few years there were more advancement in the field of technology. Considering railway department, e-ticket facility was introduced where users browse through a governmental website and book their long journey railway tickets which can be printed out after confirmation to show it to the checker when needed. After which months before a new technology called M-ticketing (Mobile Ticketing) was introduced where customers messaged to the web portal through mobile phones after which a complete web page download to the mobile phone where users can do the same booking process as it was in the e-ticketing facility. Also in foreign countries the use of Oyster cards & Octopus card has become mandatory during travel. But we suffer if we forget our travel cards and we stand in the Queue for our local suburban tickets, which is a place where e-ticketing; m-ticketing was unable lay their foot prints.

Android Suburban Railway (ASR) ticketing is mainly to buy the suburban tickets which are the most challenging. Our ASR ticket can be bought with just a smart phone application, where you can carry your suburban railway tickets in your smart phone as a QR (Quick Response) code. It uses the smart phones "GPS" facility to validate your ticket. User's ticket information is stored in a cloud database for security purpose which is missing in the present suburban system. Also the ticket checker is provided with a checker application to search for the user's ticket with the ticket number in the cloud database for checking purposes.
II. THE GROWING IMPORTANCE OF ANDROID MOBILE

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. It is a Linux-based operating system for mobile devices such as Smartphone's and tablet computers. It is developed by the Open Handset Alliance led by Google.

Android has a large community of developers writing applications ("apps") that extend the functionality of the devices. Developers write primarily in a customized version of Java. Apps can be downloaded from third-party sites or through online stores such as Android Market, the app store run by Google. As of October 2011 there were more than 400,000 apps available for Android, and the estimated number of applications downloaded from the Android Market as of December 2011 exceeded 10 billion.

A. **SQLite**:

SQLite is an ACID-compliant embedded Relational Database Management contained in a small C programming library. SQLite implements most of the SQL standard, using a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity. In contrast to other database management systems, SQLite is not a separate process that is accessed from the client application, but an integral part of it. SQLite read operations can be multitasked, though writes can only be performed sequentially.

The source code for SQLite is in the public domain. SQLite is a popular choice for local/client storage on web browsers. It has many bindings to programming languages. It is arguably the most widely deployed database engine, as it is used today by several widespread browsers, operating systems, and embedded systems, among others.

B. **Android Cloud to Device Messaging Framework**:

Android Cloud to Device Messaging (C2DM) is a service that helps developers sends data from servers to their applications on Android devices. The service provides a simple, lightweight mechanism that servers can use to tell mobile applications to contact the server directly, to fetch updated application or user data. The C2DM service handles all aspects of queuing of messages and delivery to the target application running on the target device.

C. **QR Code**:

A QR code (abbreviated from Quick Response code) is a type of matrix-barcode (or two-dimensional code) first designed for the automotive industry. More recently, the system has become popular outside of the industry due to its fast readability and comparatively large storage capacity. The code consists of black modules arranged in a square pattern on a white background. The information encoded can be made up of four standardized kinds (*modes*) of data (numeric, alphanumeric, byte/binary, Kanji), or by supported extensions virtually any kind of data.

The amount of data that can be stored in the QR code symbol depends on the data type (mode, or input character set), version (1,...,40, indicating the overall dimensions of the symbol), and error correction level (L[low], M[medium], Q[quality], H[high]).

The maximum storage capacities occur for 40-L symbols

- Numeric only: Max. 7,089 characters (0,1,2,3,4,5,6,9)
- Max. 4,296 characters (0-9, A-Z [upper•Alph anumen], space, $ 01 *on y./0, +, -,.../.. )
- Binary/byte Max. 2,953 characters (8-bit bytes)
- Kanji/Kana Max. 1,817 characters

(version 40, error correction level L), and are as follows (where characterrefers to individual values of the input mode/data type, as indicated):

1) **Encryption**:

Although encrypted QR codes are not very common, there are a few implementations. An Android app, for example, manages encryption and decryption of QR codes using DES Algorithm (56 bits). Japanese immigration use encrypted QR codes when placing visas in passports.

2) **Error Correction**:

Codeword's are 8-bits long and use the Reed-Solomon Error Correction algorithm with four error correction levels. The higher the error correction level, the less storage capacity. The following table lists the approximate error correction capability at each of the four levels:

- **Level L** 7% of codeword's can be restored.
- **Level M** 15% of codeword's can be restored.
- **Level Q** 25% of codeword's can be restored.
- **Level H** 30% of codeword's can be restored.

Due to the design of Reed-Solomon codes and the use of 8-bit codeword's, an individual code block cannot be more than 255 codeword's in length. Since the larger QR symbols contain much more data than that, it is necessary to break the message up into multiple blocks. The largest possible block size is never used, though. The QR specification defines the block sizes so that no
more than 15 errors can be corrected within each block. This limits the complexity of certain steps in the decoding algorithm. The code blocks are then interleaved together, making it less likely that localized damage to a QR symbol will overwhelm the capacity of any single block.

Thanks to error correction, it is possible to create artistic QR codes that still scan correctly, but contain intentional errors to make them more readable or attractive to the human eye, as well as to incorporate colors, logos and other features into the QR code block.

### III. System Design

#### A. Explanation:

1) **Personal Information Gathering:**
The work here starts during the first time installation of our application. It gathers the basic customer information like first name, last name, date of birth, city, state etc., and it will be stored into user mobile's, SQLite database. So every time when the user buys the ticket this customer information is also sent to the database for security purpose and used also in the QR generation.

2) **Ticket Buying:**
The user selects source, destination, class, no. of Adult and child tickets, ticket type like return or single etc. then the user browse through the menu option to choose either credit buy option or token buy which simplifies the buy process by remembering the credit card details. Once the user chooses any of these options the application moves on to the pin code validation module.

3) **Pin Code Validation:**
Once the customer hits the buy button a PHP code in the railway server validates the pin number and passwords, if it is successful it saves both the journey details and customer info in the server's MySQL database. After which ticket number and time of buying is generated by the PHP code and the balance credit value is displayed.

4) **Generating QR Code:**
Once the php code generates the ticket number and time of buy the details saved in the MySQL database are sent to Google Chart API engine in order to generate the QR code. here all the personal and ticket information are converted into QR codes and sent back to the user mobile as HTTP response and saved in the application memory.

5) **GPS Ticket Validation:**
In this module (fig 1.0) the GPS plays the role of the checker, where when the user buys the ticket, the source geo points, destination geo points, ticket type, expiry time & date are stored in a mobile SQLite database. This service checks the user's current location in accordance with the destination geo points, after which the ticket type is checked and accordingly the ticket is deleted if two is single or updated if type is return.

6) **Checking QR Code with QR Reader:**
In this module the checker will have QR Code reader and scan the QR code with the application in order to validate QR code and verify the journey details, especially the time and date of the ticket.

7) **Checking With Database:**
If suppose the user's display is being damaged and not able to scan the QR code due to other reasons like battery failure, we have another failsafe option to check the ticket by searching the ticket database with the user's ticket number for validation purposes.

#### B. Architecture Description:
The structure of system divided into two components (fig 1.1):

1) The customer application which resides personal information gathering, buying ticket, pin code validation, generating QR code, GPS ticket validation and stored into cloud database.

2) The checker application is to validate the ticket by entering the ticket number of the user and searching in the cloud database to check whether the user has bought the ticket.
IV. FIGURE

![Diagram of Android Suburban Railway Ticketing with GPS as Ticket Checker]

Fig. 1:

V. CONCLUSION

In this paper we have presented a 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. Hence a huge problem of issuing local train tickets has been solved with our new application.

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. Still more advance modification can be a Dynamic display of Train locations by fitting GPS devices in trains to show its location in the Google map display which is available in our application. Also as a station level security we can have Hardware devices to validate the QR codes before the user enters or leaves the station, where the user can have access towards platform after being validated by the hardware device.

REFERENCES