Railway Ticket Reservation System

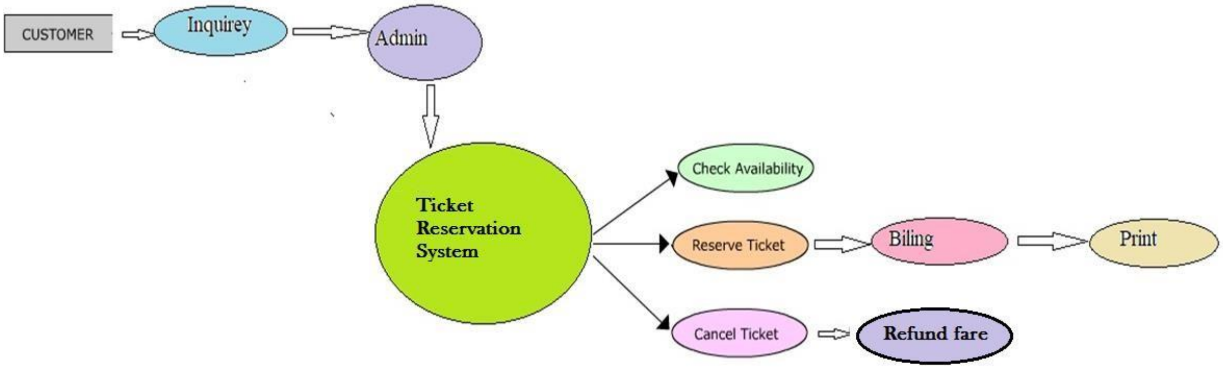
Requirement Engineering

Group # 6

Arsalan Ali Daim	**BSCS14068**
Hafiz M. Abdul Wajid	**BSCS14054**
Azhar Ali	**BSCS14058**
Danish Javed	**BSCS14028**

Introduction

Railway Ticket Reservation System is a ticketing system which provides facility to the railway administration of booking the tickets for passengers digitally, using our system. Passengers who want to travel will come to railway station for booking and the operator at the desk there will use our system to check available trains and seats according to the passenger's need like destination, comfort zone etc. The passenger will be given a ticket for a particular seat. And the fare for the acquired ticket will be taken by the operator. The record of each train, each coach of train, each seat of coach and all the passengers who have travelled, who are travelling and who will travel (reserved seats) will be stored in our system. It is fast system. It is easy to use and digital system of ticket reservation.



Useful Techniques for Requirements Elicitation

We are using two main techniques for requirement elicitation of our project. There techniques are as follows

- Use Cases
- Ethnography

The reasons behind the use of mentioned techniques in our project are, passengers come to the railway station for ticket reservation according to their destination, available train, and desired seat in the different coaches of the train their cases vary each individual passenger. Now not all passengers come to the station to reserve a ticket but some of them are there to cancel their tickets and get the given amount for the seat in return. Therefore, we need use-cases that will cover all of the different scenarios. When we talk about the technique of ethnography then we need to know, what is the procedure that is being followed by the operator? When people come and ask for reservation? So, these are the reasons why we are going to use above mentioned techniques for requirements elicitation in our project accordingly.

Categorizing the System Requirements into Functional and Non-Functional Requirements

Functional Requirements:

- A Desktop Application

- System will provide “Operator Login” functionality
- System should generate a unique identification No. for each passenger that is to differentiate between passengers with similar bio-data.
- System should generate daily report of ticket booking and cancelation
- Operator will be able to:
    - See available trains and their arrival and departure time.
    - Number of seats, coaches, berths, either available or not, in each train.
    - Select the train according to the destination of the passenger.
    - Search the vacancy of passengers in the train according to the coaches.
    - Differentiate the coaches of the train.
    - Issue the ticket to passengers according to the ticket’s category or passenger’s requirement.
    - Bill the passenger for the issued ticket and balance the residual amount with return.
    - Cancel the ticket according to railway rules, if the passenger asks to do so.
    - Fare refund in case of cancellation.
- Monthly report of tickets booking and cancelation

Non-Functional Requirements:

Product Requirements

- Response Time of system Transactions and Searching should be less because it’s a real-time application and its response time depends upon performance and space. So, in short system should be efficient.
- Also, System should be easy to use and there should be no data redundancy.
- System shall give a good User Interface to easily see the output.

Organizational Requirements

- User or operator should authenticate himself to access the software by login procedure.

External Requirements

- Passenger’s information should be secure in the software. The ways to access information should be secure and the information shall only be accessed through the system.

Metrics for Specifying Non-Functional Requirements

Speed

Of course, if we want a good and high performing software that will definitely require the speed. In order to measure the speed of this system we are going to analyze the response time of the software when it is asking for transactions of data, which is the passenger’s information entered by the operator. Because the passenger is standing in front of the operator and communicating the information for travelling on runtime. So, software should save the information ASAP.

- Maximum Time for a transaction should be less than 2 seconds.
- This would require use of best algorithms and efficient coding

Size

When thousands of passengers(per day), come for travelling then amount of data will rapidly go up and size of the information to be stored will be increased, which in turn can affect the software performance especially while performing searches on the stored data. So we will analyze that how much data units e.g. KBs, MBs, it takes to store each passenger’s data by knowing how much data units have been occupied by one passenger’s information. And try to reduce the redundancy if exists.

Ease of use

User Interface is the first and basic requirement in a software. It defines sometimes the software quality for the users. If we achieve a good interface then the use of the software will be easier and understanding of the operator will be more.

Reliability

Every software should be reliable because it is one of the basic demands of the user. Also when the system is to be deployed on a scale like this reliability is a must have because if it fails lots and lots of people will face difficulties. To know how much our system is reliable we will analyze the rate of failures when requested to save information.

Other Requirements

Hardware Requirements

- Minimum Pentium 4 processor
- Ram 1 GB

Development Environment

- Desktop
- Microsoft Visual studio 2015 (IDE)
- Programming language C#

Requirements Validations

Consistency

After building the software we will validate our system that either it has any confliction with Software Requirements or not, which will define the consistency of our railway ticket reservation system. There could be some design conflicts or technical issues that can due arise due to difference of preferences between us and users or due to some technical functionality.

Realism

All current features of our system can be implemented in real. A Desktop Application with all requirements can be created using C#. In future, we want our system to be a web portal that can be accessed by multiple stations and users. To fulfill that we need a web developer in our team, this will be finalized at the end of the project either web portal can be implemented or not using current tools.

Verifiability Check

- Prototypes
- Test Case Generation

Our system can be verified manually by **prototyping** and using **test case generation** techniques.

As, we are following incremental approach so we will add different modules and functionality to our system at every step and check all new requirements manually. It consists of systematic analysis of requirements on each level that will ensure every increment is functional properly. We can use prototyping to check verifiability at each increment and at the end when we have fully functional system, we will make test cases to test our system so that we can verify all the requirements and see if there is any conflict. We can take help of the system admin who will be the user of the system to make effective test cases.