SRS

Railways Reservation

Table of Contents

4. DIAGRAM

5. GUI

6. REFERENCES

Introduction

1.Introduction

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS purpose ,scope, definitions, acronyms, abbreviations, references and overview of SRS.A **Software Requirements Specification** (**SRS**) - a requirements specification for a software system - is a complete description of the behaviour of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. Use cases are also known as functional requirements. In addition to use cases, the SRS also contains non-functional (or supplementary) requirements. Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints). The aim of this document is to gather and analyse and give an in-depth insight of the complete Marvel Electronics and Home Entertainment software system by defining the problem statement in detail. This is a documentation of the project **Railways Reservation System** done sincerely and satisfactorily by my group members. A Software has to be developed for automating the manual Railway Reservation System.

- RESERVE SEATS – Reservation form has to be filled by passenger. If seats are available entries like train name, number, destination are made.

- CANCEL RESERVATION- The clerk deletes the entry in the System and changes in the Reservation Status.

- VIEW RESERVATION STATUS-The user need to enter the PIN number printed on ticket.

1.1 Objective:

The purpose of this source is to describe the railway reservation system which provides the train timing details, reservation, billing and cancellation on various types of reservation namely,
• Confirm Reservation for confirm Seat.
• Reservation against Cancellation.
• Waiting list Reservation.
• Online Reservation.
• Tatkal Reservation.

The origin of most software systems is in the need of a client, who either wants to automate the existing manual system or desires a new software system. The software system is itself created by the developer. Finally, the end user will use the completed system. Thus, there are three major parties interested in a new system: the client, the user, and the developer. Somehow the requirements for the system that will satisfy the needs of the clients and the concerns of the users have to be communicated to the developer. The problem is that the client doesn't usually design the software or the software development process and the

developer does not understand the client's problem and the application area. This causes a communication gap between the parties involved in the development of the project.

The basic purpose of Software Requirement Specification (SRS) is to bridge this communication gap. SRS is the medium through which the client's and the user's needs are accurately specified; indeed SRS forms the basis of software development.

Another important purpose of developing an SRS is helping the clients understanding their own needs. An SRS establishes the basis for agreement between the client and the supplier on what the software product will do.

An SRS provides a reference for validation of the final product.A high quality SRS is a prerequisite to high quality software and it also reduces the development cost.

A few factors that direct us to develop a new system are given below -:

1. Faster System
2. Accuracy
3. Reliability
4. Informative
5. Reservations and cancellations from anywhere to any place

1.2 Scop:

"Railways Reservation System" is an attempt to simulate the basic concepts of an online Reservation system. The system enables to perform the following functions:

- SEARCH FOR TRAIN

- BOOKING OF A SELECTED FLIGHT

- PAYMENT

- CANCELLATION
- Freight Revenue enhancement
- Passenger Revenue enhancement
- Improved & optimized service

1.3 Glossary:

This should define all technical terms and abbreviations used in the document

- ➢ NTES – National Train Enquiry System
- ➢ IVRS – Interactive Voice Response system

- ➢ PRS – passenger reservation system

> ➢ <u>DFD</u> :- Data Flow Diagram
> ➢ <u>ERD</u> :- Entity Relationship Diagram
> ➢ <u>SRS</u> :- Software Requirements Specification
> ➢ <u>STD</u> :- State Transition Diagram

-

1.4 Overview:

The remaining sections of this document provide a general description, including characteristics of the users of this project, the product's hardware, and the functional and data requirements of the product. General description of the project is discussed in section 2 of this document. Section 3 gives the functional requirements, data requirements and constraints and assumptions made while designing the E-Store. It also gives the user viewpoint of product. Section 3 also gives the specific requirements of the product. Section 3 also discusses the external interface requirements and gives detailed description of functional requirements. Section 4 is for supporting information.

Overall Description

2.Overall Description

This document contains the problem statement that the current system is facing which is hampering the growth opportunities of the company. It further contains a list of the stakeholders and users of the proposed solution. It also illustrates the needs and wants of the stakeholders that were identified in the brainstorming exercise as part of the requirements workshop. It further lists and briefly describes the major features and a brief description of each of the proposed system.

2.1 Product Perspective:

Before the automation, the system suffered from the following **DRAWBACKS:**

- Ø The existing system is highly manual involving a lot of paper work and calculation and therefore may be erroneous. This has lead to inconsistency and inaccuracy in the maintenance of data.

- Ø The data, which is stored on the paper only, may be lost, stolen or destroyed due to natural calamity like fire and water.

- Ø The existing system is sluggish and consumes a lot of time causing inconvenience to customers and the airlines staff.

- Ø Due to manual nature, it is difficult to update, delete, add or view the data.

- Ø Since the number of passengers have drastically increased therefore maintaining and retrieving detailed record of passenger is extremely difficult.

- Ø An railways has many offices around the world, an absence of a link between these offices lead to lack of coordination and communication.

Hence the railways reservation system is proposed with the following

- Ø The computerization of the reservation system will reduce a lot of paperwork and hence the load on the airline administrative staff.

- Ø The machine performs all calculations. Hence chances of error are nil.

- Ø The passenger, reservation, cancellation list can easily be retrieved and any required addition, deletion or updation can be performed.

- Ø The system provides for user-ID validation, hence unauthorized access is prevented.

2.2 Project Functions:

Booking agents with varying levels of familiarity with computers will mostly use this system. With this in mind, an important feature of this software is that it be relatively simple to use. The scope of this project encompasses: -

¨ **Search:** This function allows the booking agent to search for train that are available between the two travel cities, namely the "Departure city" and "Arrival city" as desired by the traveller. The system initially prompts the agent for the departure and arrival city, the date of departure, preferred time slot and the number of passengers. It then displays a list of train available with different airlines between the designated cities on the specified date and time.

¨ **Selection**: This function allows a particular train to be selected from the displayed list. All the details of the train are shown :-

1. train Number
2. Date, time and place of departure
3. Date, time and place of arrival
4. TRAIN Duration
5. Fare per head
6. Number of stoppages – 0, 1, 2…

¨ **Review**: If the seats are available, then the software prompts for the booking of train. The train information is shown. The total fare including taxes is shown and flight details are reviewed.

¨ **Traveller Information**: It asks for the details of all the passengers supposed to travel including name, address, telephone number and e-mail id.

¨ **Payment**: It asks the agent to enter the various credit card details of the person making the reservation.

1. Credit card type
2. Credit card number
3. CVC number of the card
4. Expiration date of the card
5. The name on the card

¨ **Cancellation :** The system also allows the passenger to cancel an existing reservation. This function registers the information regarding a passenger who has requested for a cancellation of his/her ticket. It includes entries pertaining to the train No., Confirmation No., Name, Date of Journey, Fare deducted.

2.3 User Characteristics:

- Ø **EDUCATIONAL LEVEL:-**At least user of the system should be comfortable with English language.

- Ø **TECHNICAL EXPERTISE: -** User should be comfortable using general purpose applications on the computer system.

2.4 Constrains:

Software constraints:

- Ø The system will run under windows98 or higher platforms of operating system.

2.5 Assumptions and Dependencies:

- Ø Booking Agents will be having a valid user name an password to access the software

- Ø The software needs booking agent to have complete knowledge of railways reservation system.

- Ø Software is dependent on access to internet.

Requirement Specification

3.1 Function Requirements

3.1.1 performance requirements:

- **<u>User Satisfaction</u>: -** The system is such that it stands up to the user expectations.

- **<u>Response Time</u>: -** The response of all the operation is good. This has been made possible by careful programming.

- **<u>Error Handling</u>: -** Response to user errors and undesired situations has been taken care of to ensure that the system operates without halting.

- **<u>Safety and Robustness</u>: -** The system is able to avoid or tackle disastrous action. In other words, it should be foul proof. The system safeguards against undesired events, without human intervention.

- **<u>Portable</u>: -** The software should not be architecture specific. It should be easily transferable to other platforms if needed.
- **<u>User friendliness</u>: -** The system is easy to learn and understand. A native user can also use the system effectively, without any difficulties.

3.1.2 Design constrian:

There are a number of factors in the client's environment that may restrict the choices of a designer. Such factors include standards that must be followed, resource limits, operating environment, reliability and security requirements and policies that may have an impact on the design of the system. An SRS (Software Requirements Analysis and Specification) should identify and specify all such constraints.

Ø **<u>Standard Compliance</u>: -** This specifies the requirements for the standards the system must follow. The standards may include the report format and accounting properties.

Ø **<u>Hardware Limitations</u> :-** The software may have to operate on some existing or predetermined hardware, thus imposing restrictions on the design. Hardware limitations can include the types of machines to be used, operating system available on the system, languages supported and limits on primary and secondary storage.

Ø **<u>Reliability and Fault Tolerance</u>: -** Fault tolerance requirements can place a major constraint on how the system is to be designed. Fault tolerance requirements often make the system more complex and expensive. Requirements about system behavior in the face of certain kinds of faults are specified. Recovery requirements are often an integral part here, detailing what the system should do I some failure occurs to ensure certain properties. Reliability requirements are very important for critical applications.

Ø **<u>Security</u>: -** Security requirements are particularly significant in defence systems and database systems. They place restrictions on the use of certain commands, control access to

data, provide different kinds of access requirements for different people, require the use of passwords and cryptography techniques and maintain a log of activities in the system.

3.1.3 Hardware requirements:

For the hardware requirements the SRS specifies the logical characteristics of each interface b/w the software product and the hardware components. It specifies the hardware requirements like memory restrictions, cache size, the processor, RAM size etc... those are required for the software to run.

<u>Minimum Hardware Requirements</u>

Processor Pentium III

Hard disk drive 40 GB

RAM 128 MB

Cache 512 kb

<u>Preferred Hardware Requirements</u>

Processor Pentium IV

Hard disk drive 80 GB

RAM 256 MB

Cache 512 kb

3.1.4 Software requirements:

- Any window based operating system with DOS support are primary requirements for software development. Windows XP, FrontPage and dumps are required. The systems must be connected via LAN and connection to internet is mandatory.

3.1.5 other requirements:

Software should satisfy following requirements as well:-

- SECURITY
- Ø PORTABILITY
- Ø CORRECTNESS
- Ø EFFICIENCY
- Ø FLEXIBILTY

- Ø TESTABILTY
- Ø REUSABILTY

3.2 Non-Function Requirements

3.2.1 Security:

The system use SSL (secured socket layer) in all transactions that include any confidential customer information. The system must automatically log out all customers after a period of inactivity. The system should not leave any cookies on the customer's computer containing the user's password. The system's back-end servers shall only be accessible to authenticated management.

3.2.2 Reliability:

The reliability of the overall project depends on the reliability of the separate components. The main pillar of reliability of the system is the backup of the database which is continuously maintained and updated to reflect the most recent changes. Also the system will be functioning inside a container. Thus the overall stability of the system depends on the stability of container and its underlying operating system.

3.2.3 Availability:

The system should be available at all times, meaning the user can access it using a web browser, only restricted by the down time of the server on which the system runs. A customer friendly system which is in access of people around the world should work 24 hours. In case of a of a hardware failure or database corruption, a replacement page will be shown. Also in case of a hardware failure or database corruption, backups of the database should be retrieved from the server and saved by the Organizer. Then the service will be restarted. It means 24 x 7 availability.

3.2.4 Maintainability:

A commercial database is used for maintaining the database and the application server takes care of the site. In case of a failure, a re-initialization of the project will be done. Also the software design is being done with modularity in mind so that maintainability can be done efficiently.

3.2.5 Supportability:

The code and supporting modules of the system will be well documented and easy to understand. Online User Documentation and Help System Requirements.

Diagram

A **use case diagram** in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

Interaction among actors is not shown on the use case diagram. If this interaction is essential to a coherent description of the desired behavior, perhaps the system or use case boundaries should be re-examined. Alternatively, interaction among actors can be part of the assumptions used in the use case.

- Use cases
  A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse.
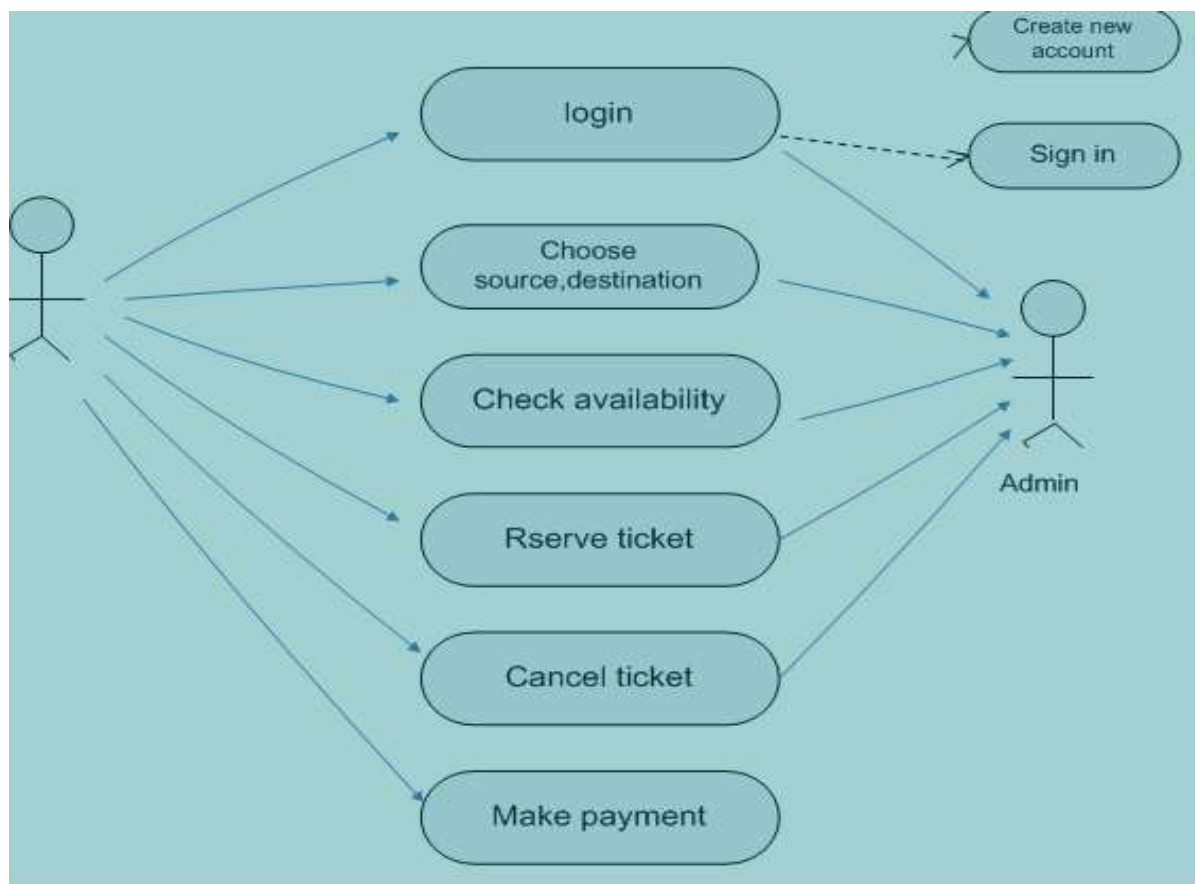- Actors
  An actor is a person, organization, or external system that plays a role in one or more interactions with the system.
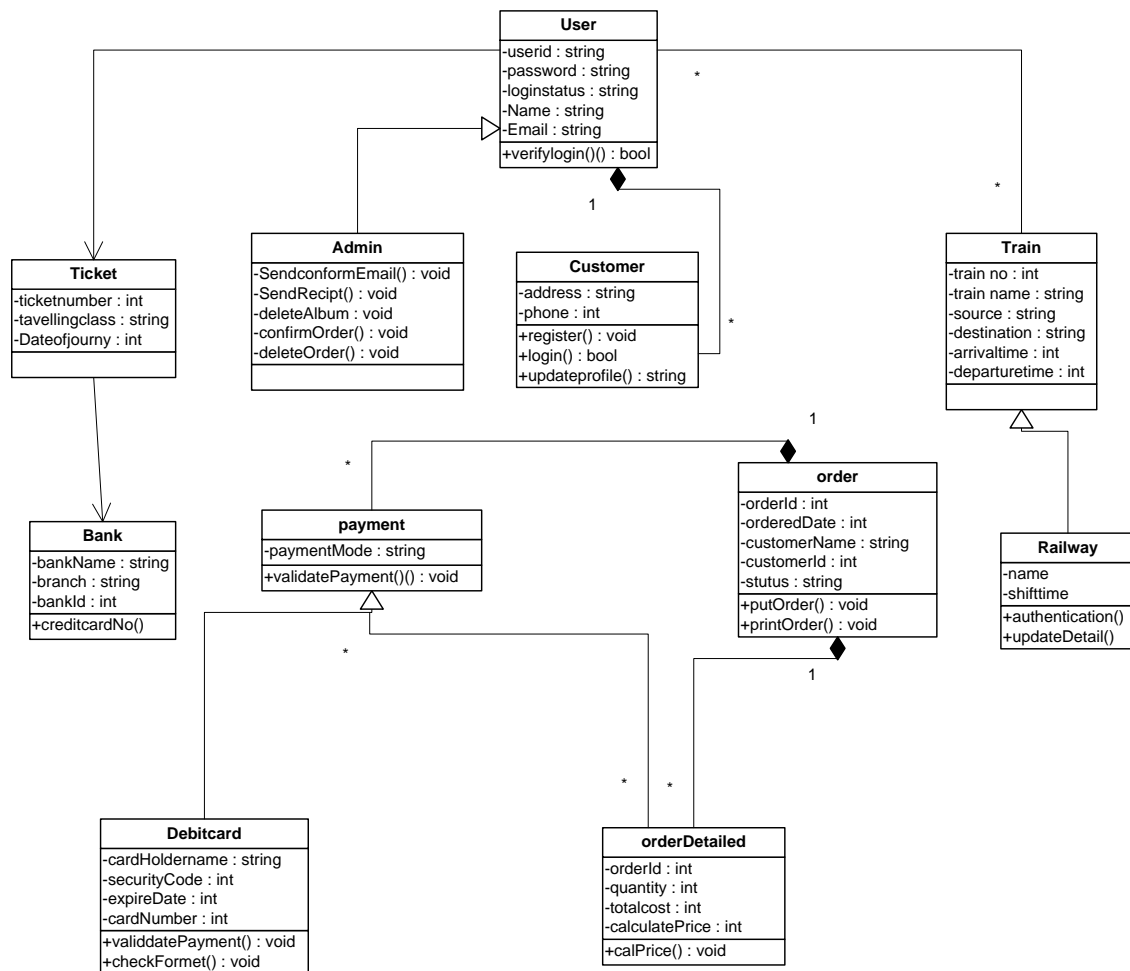- System boundary boxes(optional)
  A rectangle is drawn around the use cases, called the system boundary box, to indicate its scope of system. Anything within the box represents functionality that is in scope and anything outside the box is not.
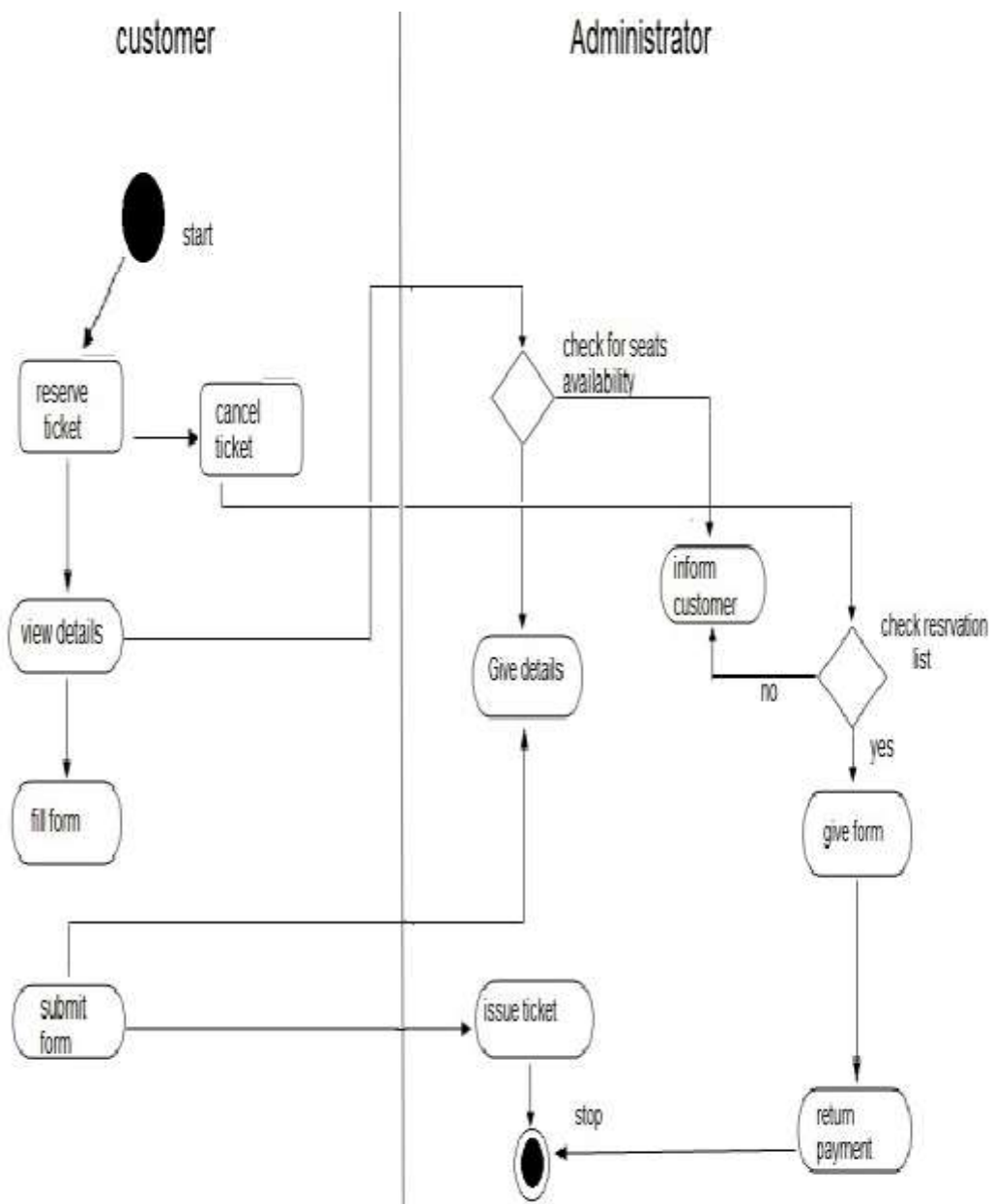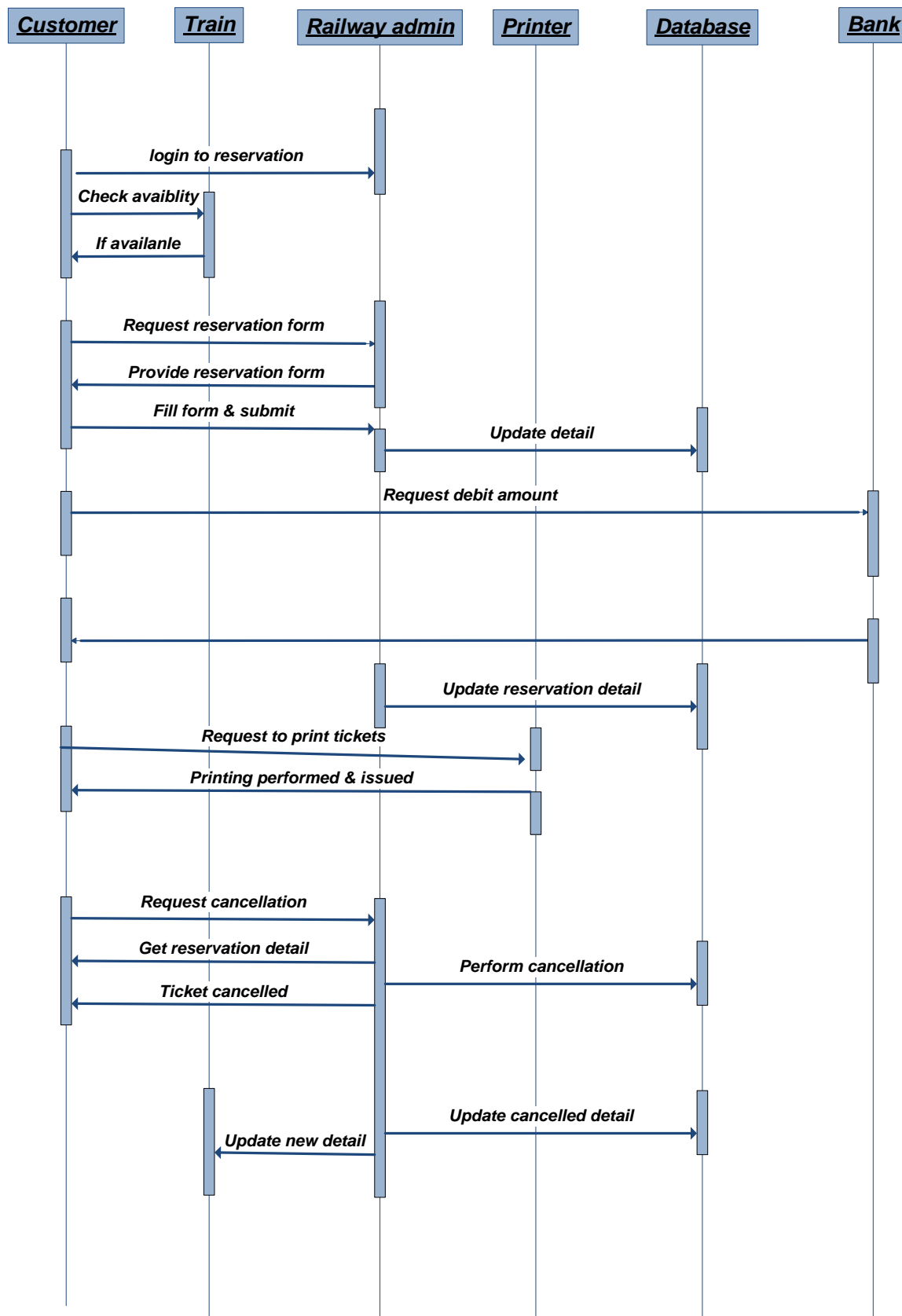
4.1 Use-case Diagram

4.2 Class Diagram

User
-userid : string
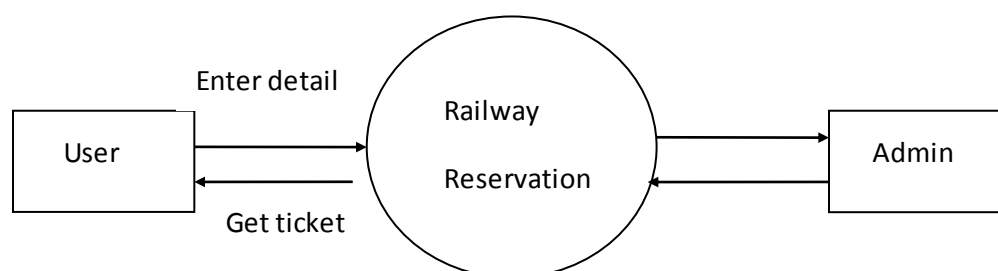-password : string
-loginstatus : string
-Name : string
-Email : string
+verifylogin()() : bool

Ticket
-ticketnumber : int
-tavellingclass : string
-Dateofjourny : int

Admin
-SendconformEmail() : void
-SendRecipt() : void
-deleteAlbum : void
-confirmOrder() : void
-deleteOrder() : void

Customer
-address : string
-phone : int
+register() : void
+login() : bool
+updateprofile() : string

Train
-train no : int
-train name : string
-source : string
-destination : string
-arrivaltime : int
-departuretime : int

Bank
-bankName : string
-branch : string
-bankId : int
+creditcardNo()

payment
-paymentMode : string
+validatePayment()() : void

order
-orderId : int
-orderedDate : int
-customerName : string
-customerId : int
-stutus : string
+putOrder() : void
+printOrder() : void

Railway
-name
-shifttime
+authentication()
+updateDetail()

Debitcard
-cardHoldername : string
-securityCode : int
-expireDate : int
-cardNumber : int
+validdatePayment() : void
+checkFormet() : void

orderDetailed
-orderId : int
-quantity : int
-totalcost : int
-calculatePrice : int
+calPrice() : void

4.3 State Diagram

4.4 Sequence Diagram

Customer	Train	Railway admin	Printer	Database	Bank

login to reservation

Check avaiblity

If availanle

Request reservation form

Provide reservation form

Fill form & submit

Update detail

Request debit amount

Update reservation detail

Request to print tickets

Printing performed & issued

Request cancellation

Get reservation detail

Perform cancellation

Ticket cancelled

Update cancelled detail

Update new detail

4.5 Data Flow Diagram

A **data flow diagram** (**DFD**) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design).On a DFD, data items flow from an external data source or an internal data store to an internal data store or an external data sink, via an internal process. A DFD provides no information about the timing of processes, or about whether processes will operate in sequence or in parallel. It is therefore quite different from a flowchart, which shows the flow of control through an algorithm, allowing a reader to determine what operations will be performed, in what order, and under what circumstances, but not what kinds of data will be input to and output from the system, nor where the data will come from and go to, nor where the data will be stored (all of which are shown on a DFD).

It is common practice to draw a context-level data flow diagram first, which shows the interaction between the system and external agents which act as data sources and data sinks. On the context diagram (also known as the 'Level 0 DFD') the system's interactions with the outside world are modelled purely in terms of data flows across the *system boundary*. The context diagram shows the entire system as a single process, and gives no clues as to its internal organization.

This context-level DFD is next "exploded", to produce a Level 1 DFD that shows some of the detail of the system being modelled. The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.
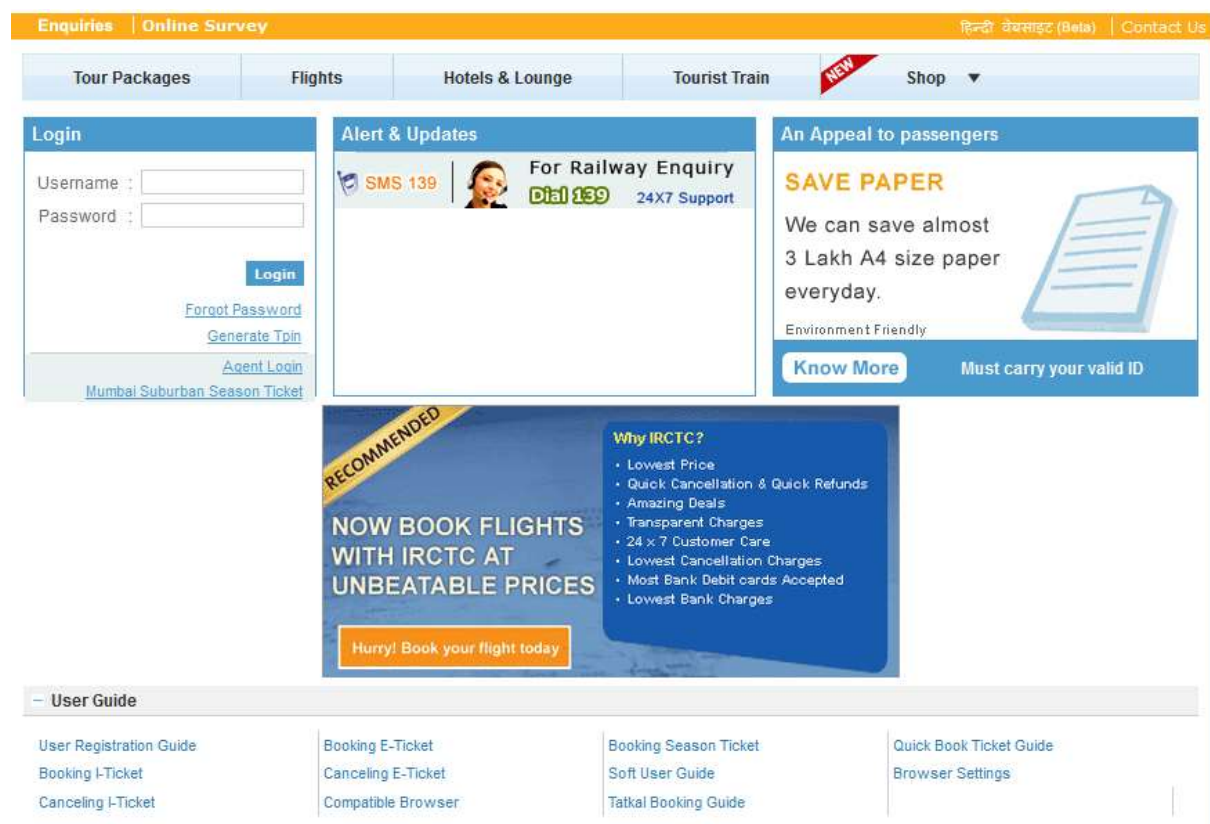
Level 0:

Level 1:

User

Railway database

Username, Route,
date, no of Seats

Seat no

Checked status

Check

Availability

Bank detail

Print   Conform Ticket   Payment

Payment Bill

Conform Payment

Print Ticket   Seat no

Ticket Database
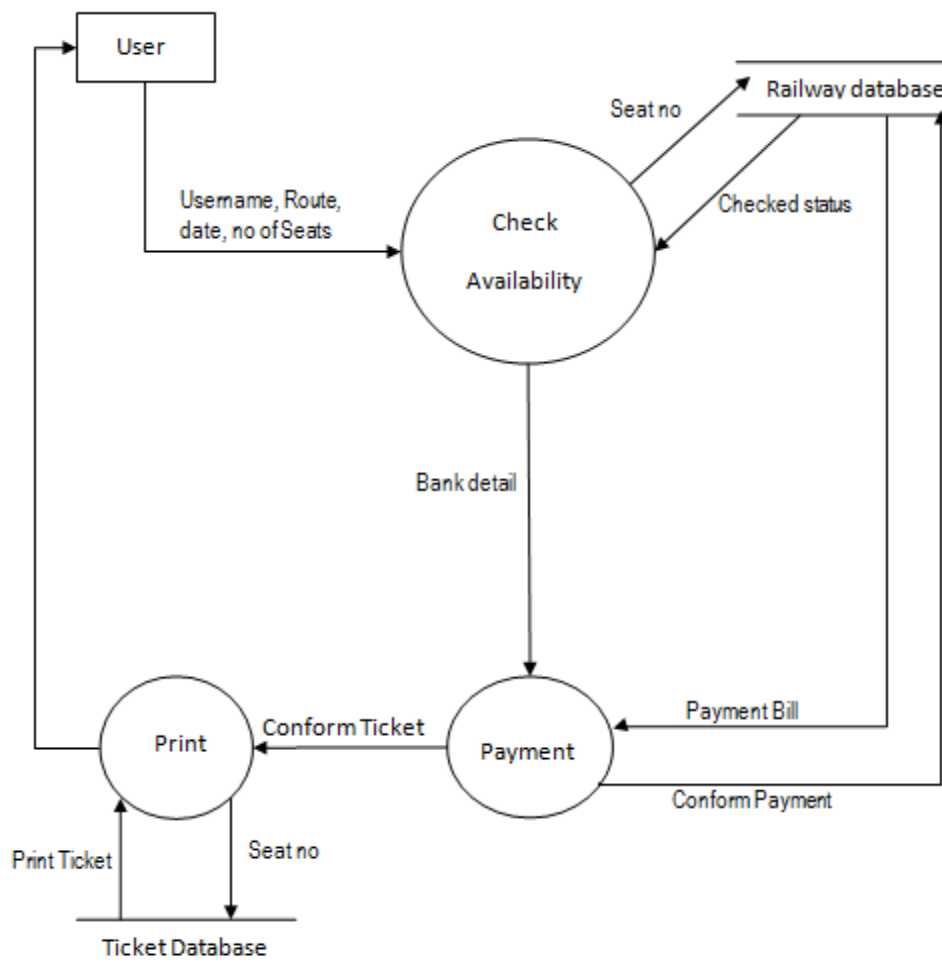
Level 2:
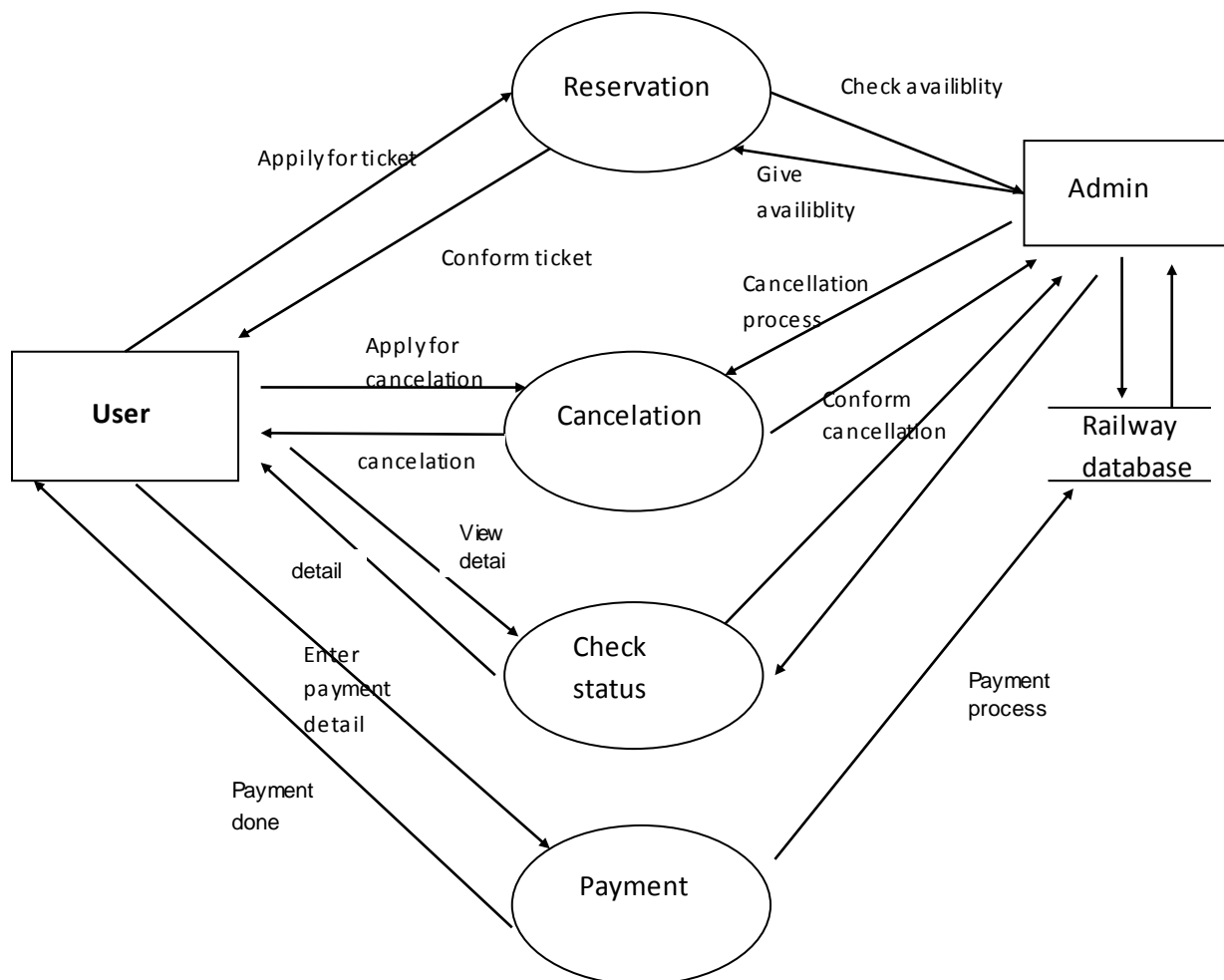
Graphical User Interface

5.1 Screen Short

The system shall provide a uniform look and feel between all the web pages.

Home Page:

Login:

Registration:

Search Train:

Other Screen:

Example:

Indian Railways Official Website:

6 . References

1. IEEE SRS Format
2. Yatra.com
3. Irctc.co.in
4. Indianrail.gov.in
5. www.google.com