Software Requirements Specification

for

<Railway Reservation System>

Version 1.0 approved

Prepared by <Ali Haider & Hooria Khan>

<IQRA National University>

<May 10,2020>

Application Development- I

SOFTWARE REQUIREMENTS

Table of Contents

1. Introduction:

1.1. PURPOSE

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. In addition, This Railway reservation service will not only enhance the reservation but will also help the commuters in getting support, refunds and other real time fixes.

- Confirm Reservation for confirm Seat.
- Reservation against Cancellation.
- Waiting list Reservation.
- Online Reservation.
- Tatkal Reservation

1.2. SCOPE

Technology has transformed many aspects of life in the 21st century, including the way many of us make train reservations. For example, to make ticketing more convenient for travelers, an online reservation system helps us in booking tickets from the comfort of our homes or offices. While this is convenient for most people, it has made things particularly easier for people residing in remote locations.

The various advantages of using the online reservation system are as follows:

- Convenient – You can book or cancel your tickets sitting in the comfort of your home or office.

- Saves Time and Effort - You can save the time needed to travel to the railway reservation office and waiting in the queue for your turn.
- Towards a greener planet – Instead of printing your ticket you can also choose to travel with the soft copy of your booked ticket in your laptop or even on your mobiles
- Freight Revenue enhancement.
- Passenger Revenue enhancement.
- Improved & optimized service

1.3. DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

Some of the few ACRONYMS, ABBREVIATIONS used in Ticket Reservation System of our Railways are as follows:

- NTES – National Train Enquiry System
- IVRS – Interactive Voice Response system
- PRS – passenger reservation system
- RAC- Reservation Against Cancellation

1.4. REFERENCES

As Internet is an Ocean of knowledge, we, too, have been helped by the same inter network of system, We've referenced from many a sites to get Information/ for Knowledge Gathering to understand the current scenario of the market, below are the references we have got helped from, and we acknowledge the same:

- www.scribd.com
- downloads.intel.com
- cs.swt.edu
- Wikipedia.org
- ques10.com

1.5. OVERVIEW

The remaining sections of this document provide a general description, including characteristics of the users of this project, the product's hardware, the functional requirements, data requirements and constraints and assumptions made while designing or development of a Railway Reservation System. It also gives the user's viewpoint of the product. Section 3 gives the detail of all type of specific requirements of this product. It also includes the external, internal requirements and gives detailed description of functional requirements. However, Section 4 of this Document is of completely supporting information, same is the case with Last Section (Appendices). In Other words, below are the details of all the sections respectively:

Chapter 2 of the SRS is a brief description of the characteristics of the software to be built, its functions, its users, its constraints and its dependencies.

Chapter 3 is about specific requirements, such as functional requirements, external interface requirements, performance requirements, and also design constraints and quality characteristics.

Finally, chapter 4 includes all the supporting information, such as the Table of Contents, the Appendices, and the Index.

2. General Description:

This section describes the general factors that affect the product and its requirements. This section consists of five subsections that follow. This section does not state specific requirements. Each of the subsections makes those requirements easier to understand; it does not specify design or express specific requirements. Such all details are provided in section 3.

2.1. PRODUCT PERSPECTIVE:

Before making this a real time running online reservation system, old system suffered from many of the DRAWBACKS, such as:

- The existing system is highly manual involving a lot of paper work and calculation and therefore may be erroneous. This has led to inconsistency and inaccuracy in the maintenance of data.
- The data, which is stored on the paper only, may be lost, stolen or destroyed due to any natural calamity of fire or water.
- Existing system is sluggish and consumes a lot of time, resource etc. causing inconvenience to customers and staff.
- Due to manual working, it is difficult to add, delete, update, or view the data.
- Since number of passengers has increased to an uncertain multiple, it is very difficult to maintain or retrieve detailed record of passengers.
- A Railway has many offices around the world, an absence of link between them all causes to a lack of miscommunication and discoordination.

Hence, this Railway reservation system is proposed, with following benefits:

- Computerization of reservation system will reduce a lot of daily paperwork and hence load on the staff of admin department.
- Machine does all the calculations. Hence, chances of error arelow.

- Reservation, Cancellation or updation lists of Ticket's can easily be maintained and retrieved and any required additions, deletion or updation can easily be performed.
- This system provides User Name-Password validation, hence unauthorized access is prevented.

2.2. PRODUCT FUNCTIONS:

Users 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 can be relatively simple to use. The scope of this project encompasses:

- **Search**: This function allows the user to search for train that is available between two travel cities namely "Departure City" and "Arrival City" as desired by the traveler. The System initially prompts the agent for these two column values, the date of the journey, preferred time slot and the number of passengers. It then displays a list of trains available with different classes.

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

  - Train Number.
  - Date, Time and place of departure.
  - Train Duration
  - Fair per head
  - Number of stoppages – 0, 1, 2…

- **Review:** if the seats are available, the software prompts for the booking of train. The train information is shown, The total fare including taxes is shown and train details are reviewed before final payment

- **Travel Information:** This system asks for details of all the passenger before the booking confirmation. Hence, lesser cases of seat issues.

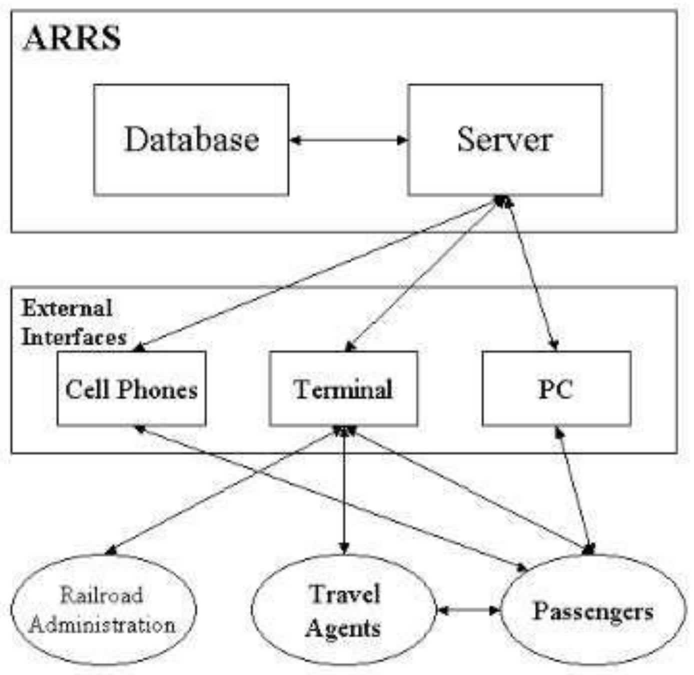
- **Payment:** It requires details of credit/ debit card of the person to make payment and reserve the required seat, Details such as:

    - Card Number
    - Card Type
    - CVV Number
    - Expiry Date
    - Name on card

- **Cancellation:** System also allow cancellation of existing reservation done, making seats unreserved for others to book and refunding the money back to the accounts of users cancelling the tickets.

(The Automated Railway Reservation System diagram showing the overview of the system's modules and the relationship of the system to external interface is presented in Figure 2.1.)

Figure 2.1 Overview Diagram of the ARRS



2.3. USER CHARACTERISTICS:

As working on this website will require some basic computer knowledge, we classify knowledge required by the users in two basic categories:

- EDUCATIONAL LEVEL: At least the user of this system should be comfortable with English Language.
- TECHNICAL EXPERTISE: User should be comfortable using general-purpose applications on the computer system.

2.4. GENERAL CONSTRAINTS:

- Software Constraints:

  - This System will run on windows Xp and and higher platform/ operating system with At least Internet Explorer 8 installed.

2.5. ASSUMPTIONS AND DEPENDENCIES:

Every system requires some certain parameters to work, to work as per the requirement, our system also requires some parameters, and we assume them as fulfilled before using this system, which are as:

- Booking agent/ user will be having his/ her own username registered before booking of any ticket, else, they'll have to register themselves on our website.
- This software needs booking agent/ user to have somplete knowledge of railway reservation system and its working.
- Software is dependent on access of Internet, as it is a remote application, it is necessary to have internet access.

3. Specific Requirements:

Requirements refers to the needs of fabricated software to work efficiently and effectively, some of the requirements of this software are as follows:

3.1. EXTERNAL INTERFACE REQUIREMENTS:

External Interface requirements refers to needs of this software's front end to work efficiently, the requirements are further classified into certain topics, which are as:

i. User Interfaces:

For the efficient working of the User Interface, i.e. the Front End of the system, the OS must be having at least Internet Explorer 8 installed. To log into the website.

ii. Hardware Interfaces

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 restriction, cache size, processor, RAM etc. those are required for software to run.

Minimum Hardware Requirements

Processor Pentium IV

HDD 40 GB

RAM 128 MB

Cache 512 kb

<u>Preferred Hardware Requirements</u>

Processor Core 2 Duo E7300

HDD 80 GB

RAM 512 MB

Cache 1 MB L1

Cache 512kb L2

iii. *Software Interfaces:*

For Hosting: Any Windows Operations System with DOS Support and Visual Studio for development. Primarily Windows 8, having Dream Weaver Installed with a working LAN connection to be mandatory.

For Using: Any type of Operating System with at Least Internet Explorer Installed and having minimum of 512 kbps working LAN compulsorily.

iv. *Communication Interfaces:*

Our Railway's web-site, www.Pakrail.gov.in offers PRS enquiries on the internet Berth/Seat availability, Passenger Status, Fare, Train Schedule etc,.

National Train Enquiry System (NTES) website, www.trainenquiry.comgives dynamic information about the running status of any train and its expected arrival/departure at any given station.

Mobile telephone based SMS enquiry service. A new mobile phone based facility for rail users’ which is. Country wide extension of Universal Rail Enquiry number “139”through setting up of Interactive Voice Response System (IVRS).

3.2. FUNCTIONAL REQUIREMENTS:

Functional requirements refer to the Functions, which were required before and covered in this system/ software we have developed. Mentioned below are the functions/ features of our newly fabricated software system:

i. ***Feature #1 - TRAIN DETAILS:***

Customers may view the train timing at a date their name and number of tickets.

ii. ***Feature #2 – RESERVATION:***

After checking the number of seats available the customers reserve the tickets.

iii. ***Feature #3 – BILLING:***

After reserving the required amount of tickets, the customer paid the amount.

iv. ***Feature #4 – CANCELLATION:***

If the customers want to cancel the ticket, then half of the amount paid by the customer will be refunded to him.

v. ***Feature #5 – PERFORMANCE REQUIREMENTS:***

It is available during all 24 hours.

vi. ***Feature #6 – SOFTWARE SYSTEM ATTRIBUTES:***

- Reliable
- Available
- Secure

3.3. USE CASES:

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.

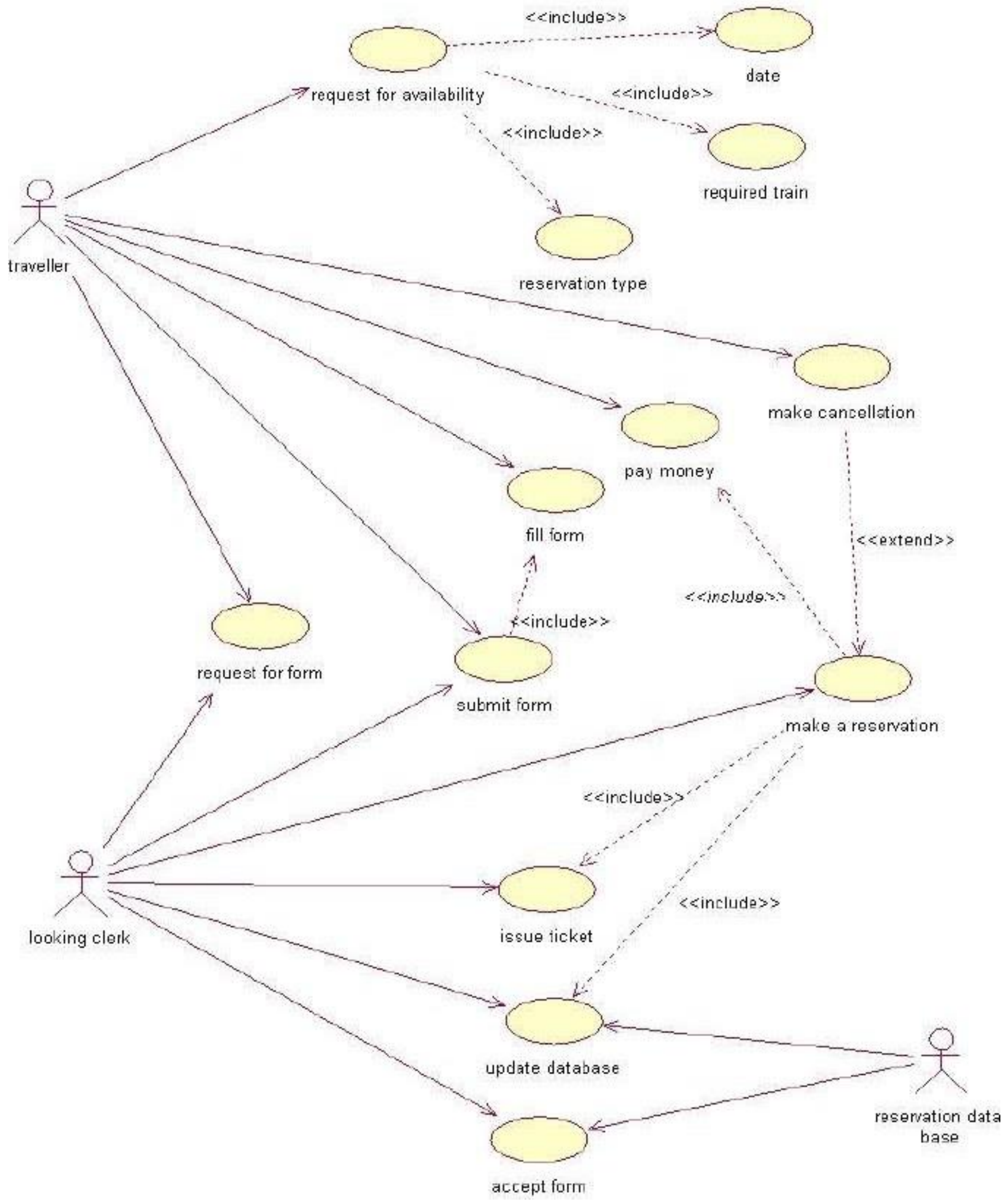
Use cases: A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse.

Actors: An actor is a person, organization, or external system that plays a role in one or more interactions with the system.

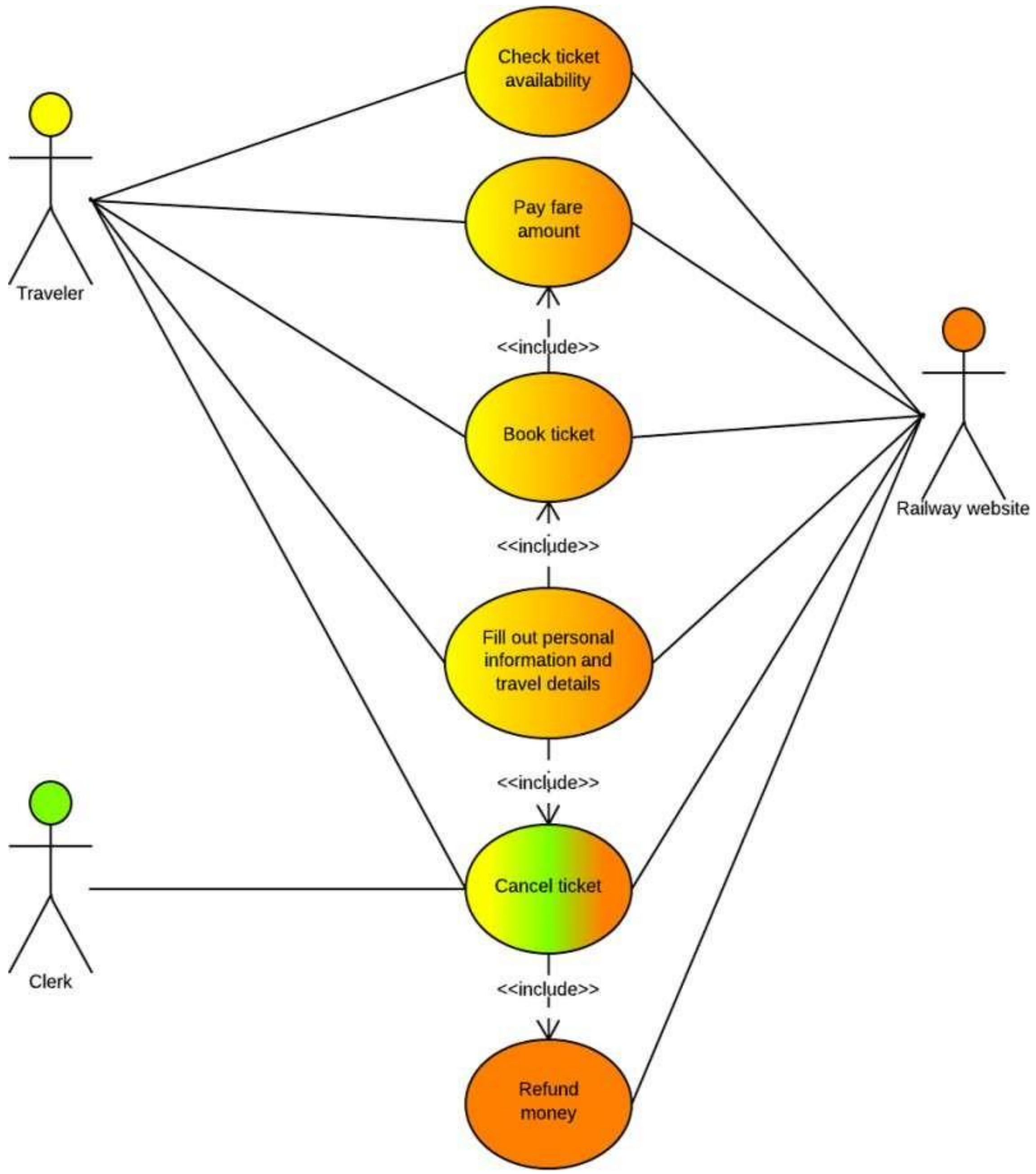
System boundary boxes (optional): A rectangle is drawn around the use cases, called the system boundarybox, to6 indicate the scope of system. Anything within the box represents functionality that is in scope and anything outside the box is not

(Use Cases for our System are on the next page)

i. ***Use case #1 –Ticket Availability, Reservation type, Booking and Cancellation:***



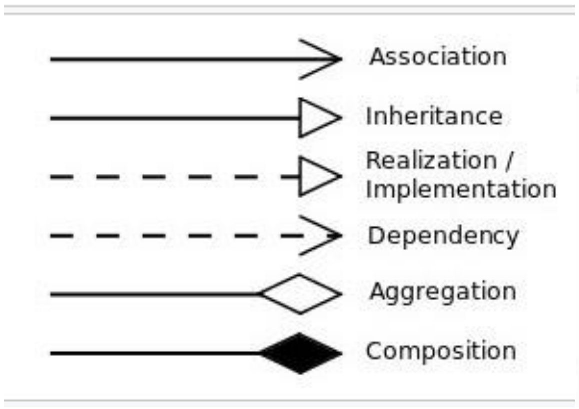
ii. Use case #2 – Ticket Booking and Cancellation and money refunding:

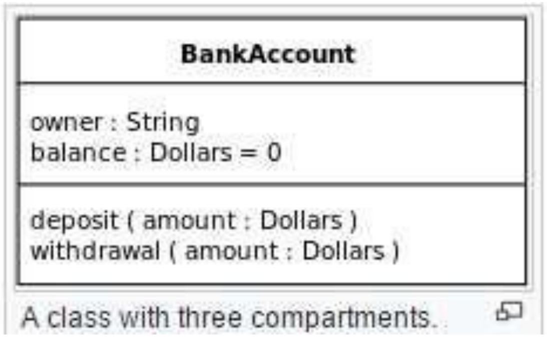


3.4. OBJECTS/ CLASSES:

In software engineering, a **class diagram** in the Unified Modeling Language(UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

The class diagram is the main building block of object-oriented modelling. It is used both for general conceptual modelling of the systematics of the application, and for detailed modelling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.



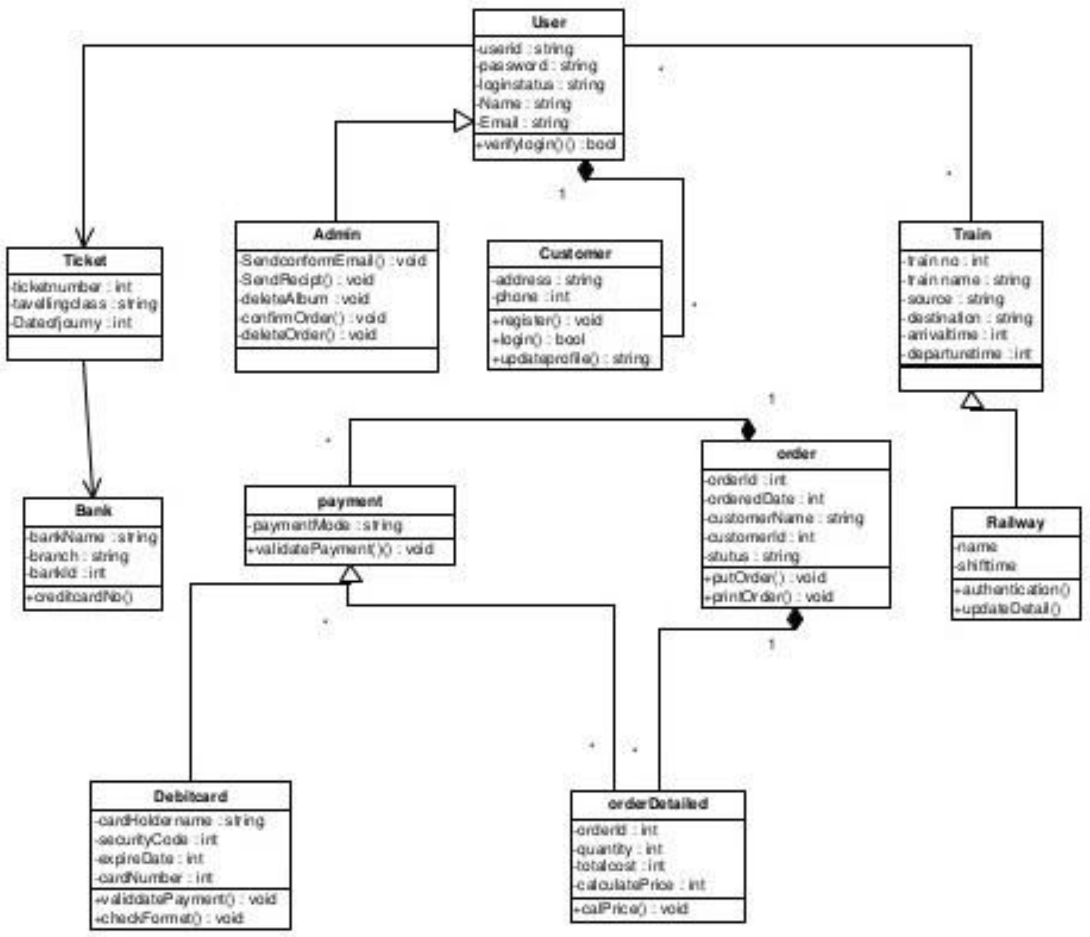

BankAccount
owner : String
balance : Dollars = 0
deposit (amount : Dollars)
withdrawal (amount : Dollars)
A class with three compartments.

Classes are represented with boxes that contain three compartments:

- The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
- 6The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.
- The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

In the design of a system, a number of classes are identified and grouped together in a class diagram that helps to determine the static relations between them. With detailed modelling, the classes of the conceptual design are often split into a number of subclasses.

In order to further describe the behavior of systems, these class diagrams can be complemented by a state diagram or UML state machine.



- **Classes:**
  - Train
  - Passenger
  - RailwayAdministration
  - Ticket

3.5. NON-FUNCTIONAL REQUIREMENTS:

Nonfunctional requirements make up a significant part of the specification. They are important as the client and user may well judge the product on its non-functional properties. Provided the product meets its required amount of functionality, the nonfunctional properties -- how usable, convenient, inviting and secure it is -- may be the difference between an accepted, well-liked product, and an unused one

1. ***Performance:***

   This system helps in increasing the overall performance of the Railway Reservation functionality by shifting a large chunk of load online causing in less hassle in ticket booking, cancellation or querying. This System is 22 hours Live per day giving us greater availability time as compared to that of 9 hours offline activity.

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. ***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 hardware failure or database corruption, a replacement page will be shown. Also, in case of a hardware failure or database corruption, backup of the database should be retrieved from the server and saved by the Organizer. Then the service will be restarted. It means 24x7 availability.

4. ***Security:***

   This system should work under 3-Level Architecture combining DB-Class-Front end with different security facilities and encryption. The System use SSL in all transactions that include any confidential customer information. The system must automatically log out all customer after a period of inactivity of those users respectively. 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.

5. ***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.

6. ***Supportability:***

The code and supporting modules of the system will be well documented and easy to understand. Online user Documentation and Help system requirements.

3.6. INVERSE REQUIREMENT:

A requirement stated as a negative proposition (shall not). An inverse requirement is untestable. Everything outside the system is what the system does not do. Testing would have to continue forever to prove the system does not do something. State what the system does. It isn't that difficult to correct a negative requirement. Substitute an active constraint that expresses what the system must do. For example change "the system shall not allow X," to "the system shall prevent Y." Another method is to use the prefix "un," such as: The system shall reject unauthorized users.

Problem requirement: The system shall produce the ABC report in a timely manner.

Discussion: "Timely" needs to be defined in accordance with the needs of the railway management

Problem requirement: The system shall require users to select only one of the following...

Discussion: The placement of booking can cause problems. In this requirement, "booking" can mean the one action available to users is "select," or it can mean users are limited to one item from the list. The correction is not adding more words ("one and only one"). The correction is deleting "booking," which is an extraneous word.

Problem requirement: The system shall automatically do month selection when numbers are entered."

Discussion: If there is an event that triggers this system action, it may need to be stated. Otherwise, automated systems are, by nature, automated, so the numbers are unnecessary.

3.7. DESIGN CONSTRAINT:

Design constraints can be imposed by other standards, hardware limitations, etc.

- **Standards Compliance:**

Specify the requirements derived from existing standards or regulations. They might include:

(1) Report format

(2) Data naming

(3) Accounting procedures

(4) Audit Tracing. For example, this could specify the requirement for software to trace processing activity.

Such traces are needed for some applications to meet minimum government or financial standards. An audit trace requirement might, for example, state that all changes to a payroll data base must be recorded in a trace file with before and after values.

- **Hardware Limitations**:

Identify the requirements for the software to operate inside various hardware constraints.

- **Quality Characteristics:**

There are a number of quality characteristics that can apply to software. Pick the ones most important to this product and develop a section for each one. Definitions of the quality characteristics follow.

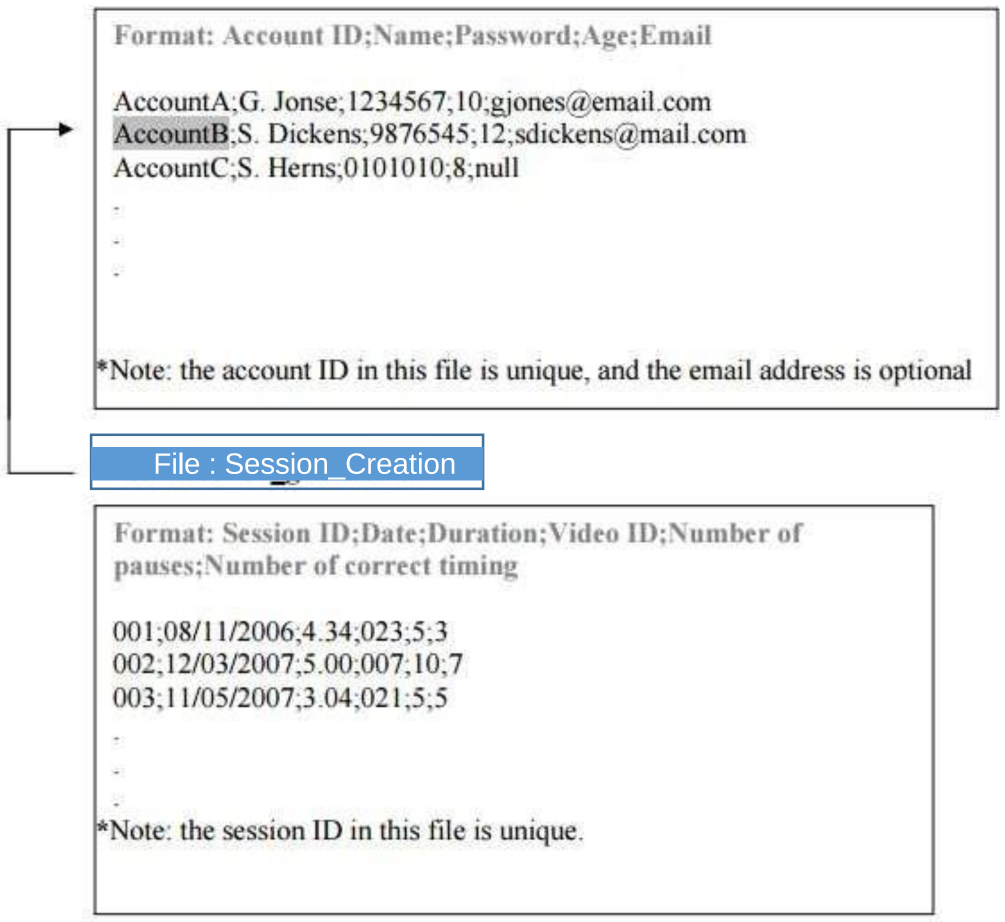
- **Correctness** - extent to which program satisfies specifications, fulfills user's mission objectives
- **Efficiency** - amount of computing resources and code required to perform function
- **Flexibility** - effort needed to modify operational program
- **Integrity/security** - extent to which access to software or data by unauthorized people can be controlled
- **Interoperability** - effort needed to couple one system with another
- **Maintainability** - effort required to locate and fix an error during operation
- **Portability** - effort needed to transfer from one h/w or s/w environment to another
- **Reliability** - extent to which program performs with required precision
- **Reusability** - extent to which it can be reused in another application
- **Testability** - effort needed to test to ensure performs as intended
- **Usability** - effort required to learn, operate, prepare input, interpret output

3.8. LOGICAL DATABASE REQUIREMENTS

The system must store all the user account information as well as the sessional grade records. All the data shall be stored in text-based flat files. For each user account, the login ID, name, password, age, email address (optional) shall be stored in one file. The email address gives the user option to receive any further information or update about the software. Each attribute shall be delimited by a semicolon, and all the entries shall be sorted alphabetically by the login ID. Furthermore, for each user account, there shall be a grade report file which contains every sessional grade result for a certain user. A grade report file shall contain the following attributes: session ID, date (dd/mm/yyyy), duration (minutes), video ID, number of pauses, and number of correct timings. Each entry shall also be delimited by a semicolon and sorted alphabetically by the session ID. Here is an example which shows how the data shall be stored and presented.

*Note: the file format and type may vary when the system is being developed.

File: UserAccounts.txt



```
Format: Account ID;Name;Password;Age;Email

AccountA;G. Jonse;1234567;10;gjones@email.com
AccountB;S. Dickens;9876545;12;sdickens@mail.com
AccountC;S. Herns;0101010;8;null
.
.
.

*Note: the account ID in this file is unique, and the email address is optional
```

File : Session_Creation

```
Format: Session ID;Date;Duration;Video ID;Number of
pauses;Number of correct timing

001;08/11/2006;4.34;023;5;3
002;12/03/2007;5.00;007;10;7
003;11/05/2007;3.04;021;5;5
.
.
.
*Note: the session ID in this file is unique.
```

Accessibility and Security : Only the user who has access to a certain account can access the grade report file belongs to that account. None of the users has access to modify any of the data files.

3.9. OTHER REQUIREMENTS

Certain requirements may, due to the nature of the software, the user organization, etc., be placed in separate
Categories such as those below.

- **Data Base.**

This could specify the requirements for any data base that is to be developed as part of the product. This might
include:

(1) Types of information
(2) Frequency of use
(3) Accessing capabilities
(4) Data element and file descriptions
(5) Relationship of data elements, records and files
(6) Static and dynamic organization
(7) Retention requirements for data

Note: If an existing data base package is to be used, this package should be named under Interfaces to Software and details of using it specified there.

- **Operations.**

- This could specify the normal and special operations required by the user such as:
- The various modes of operations in the user organization; for example, user-initiated operations
- Periods of interactive operations and periods of unattended operations
- Data processing support functions
- Backup and recovery operations

- Note: This is sometimes specified as part of the User Interfaces section.

- **Site Adaptation Requirements.**

This could:

1. Define the requirements for any data or initialization sequences that are specific to a given site, mission, or operational mode, for example, safety limits.
2. Specify features that should be modified to adapt the software to an installation.

4. Analysis Models:

The Unified Modeling Language (UML) is a graphical language for OOAD that gives a standard way to write a software system's blueprint. It helps to visualize, specify, construct, and document the artifacts of an object-oriented system. It is used to depict the structures and the relationships in a complex system.

- **System** : A set of elements organized to achieve certain objectives form a system. Systems are often divided into subsystems and described by a set of models.
- **Model** : Model is a simplified, complete, and consistent abstraction of a system, created for better understanding of the system.
- **View** : A view is a projection of a system's model from a specific perspective.

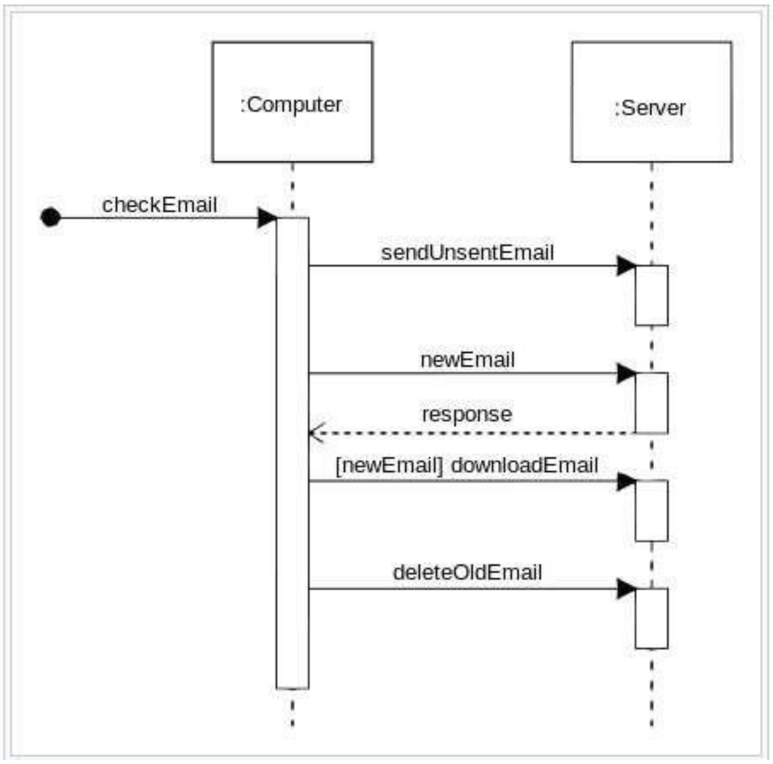
Conceptual Model of UML

The Conceptual Model of UML encompasses three major elements:

- Basic building blocks
- Rules
- Common mechanisms

4.1. SEQUENCE DIAGRAM:

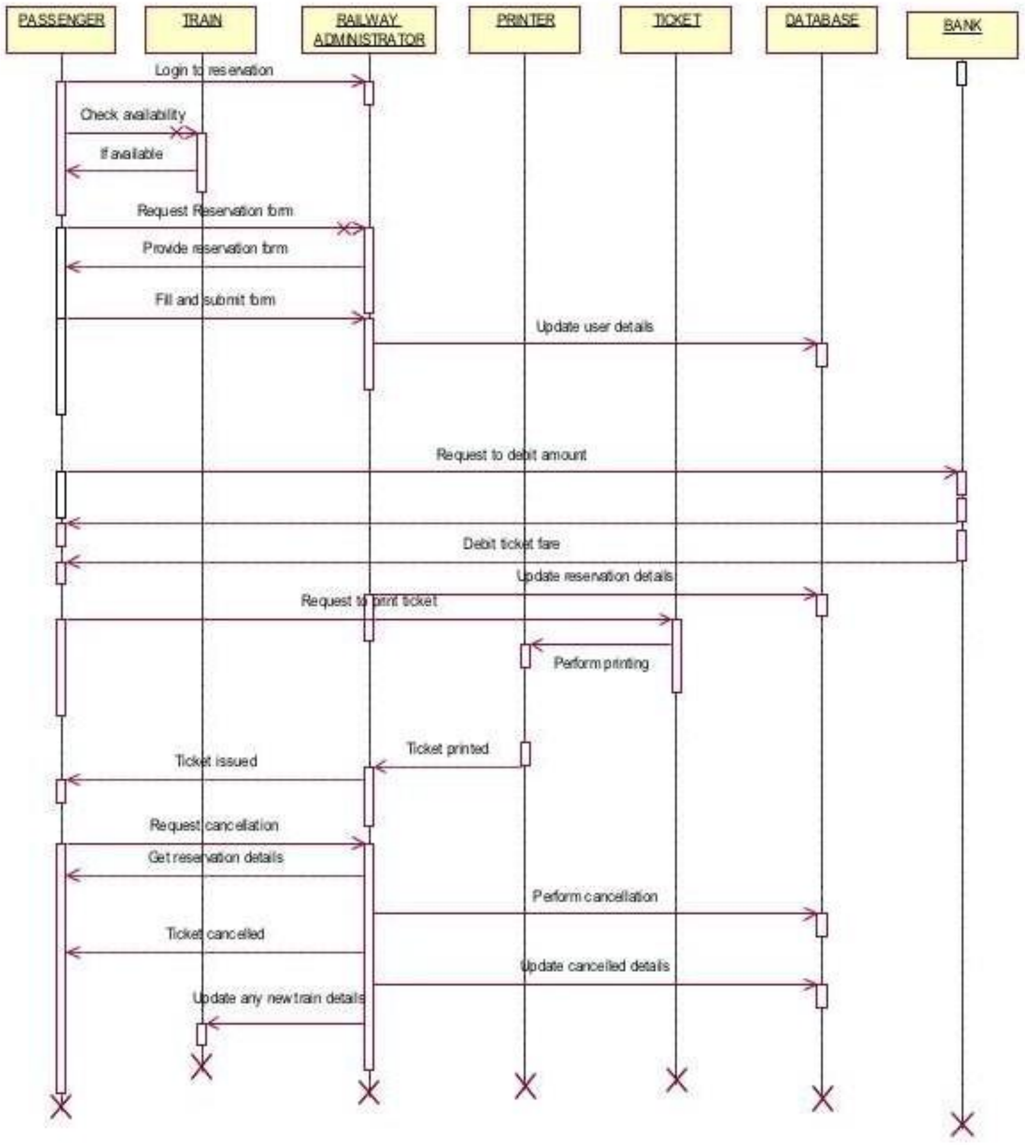
A Sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart.



A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams orevent scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

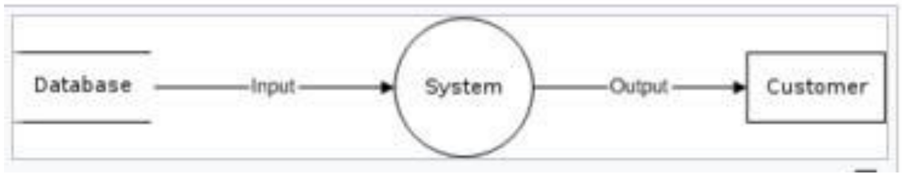
SEQUENCE DIAGRAM - ONLINE RAILWAY RESERVATION SYSTEM



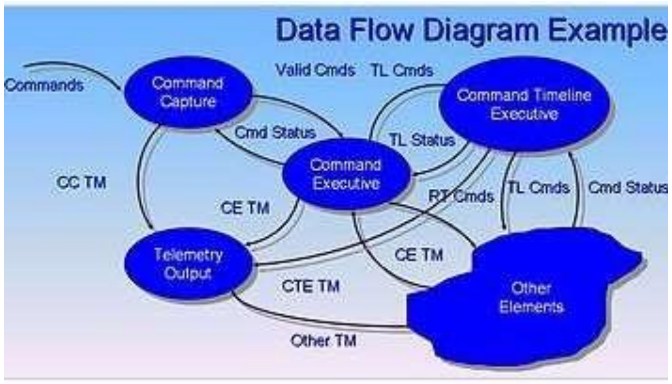
Sequence diagram describes the sequence of steps from login to ticket booking/ cancellation and refund.

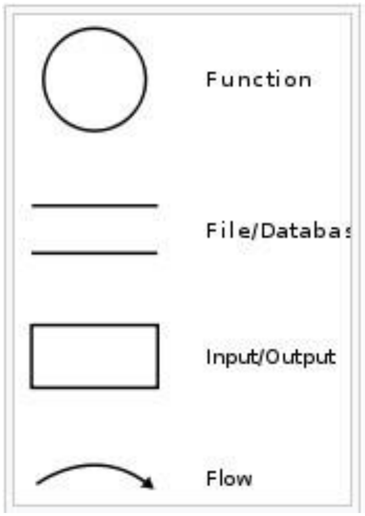
4.2. DATA FLOW DIAGRAM:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its *process* aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).



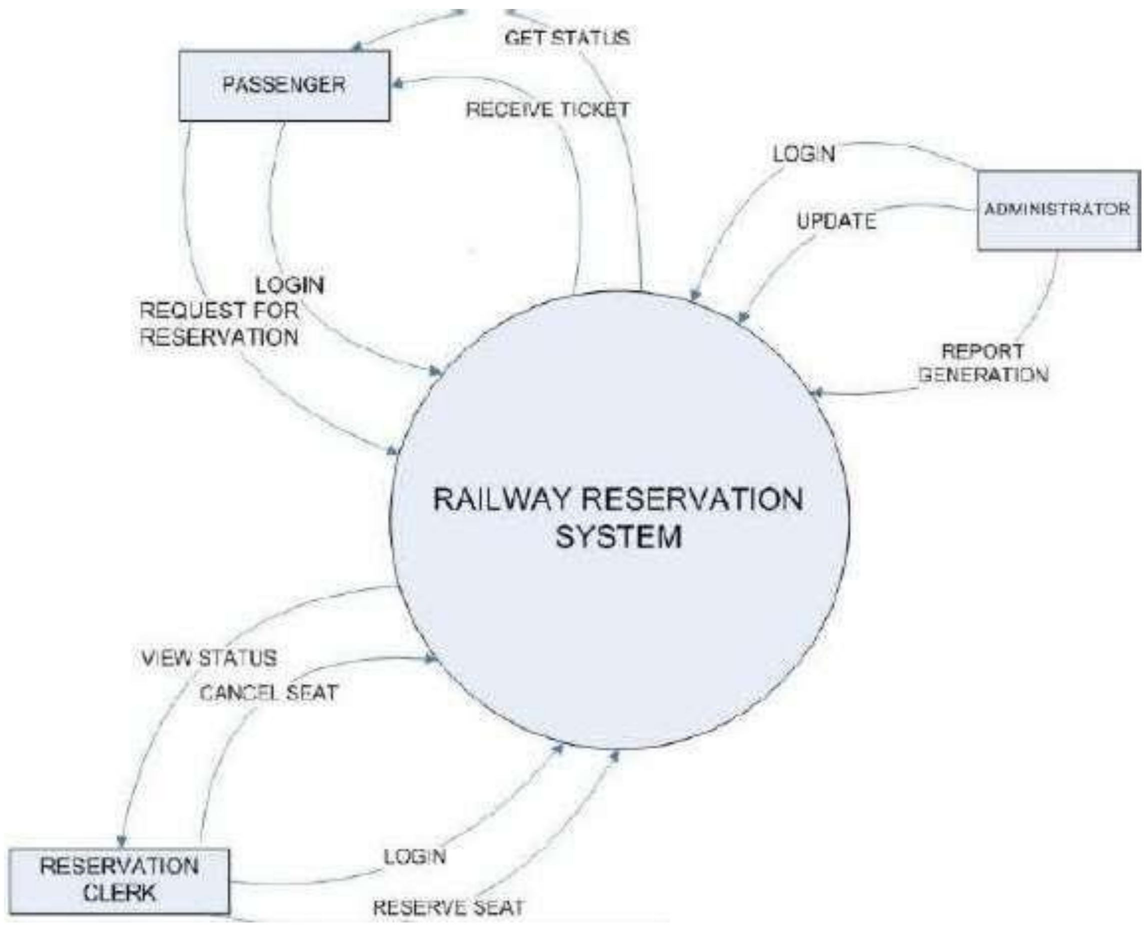
A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart).



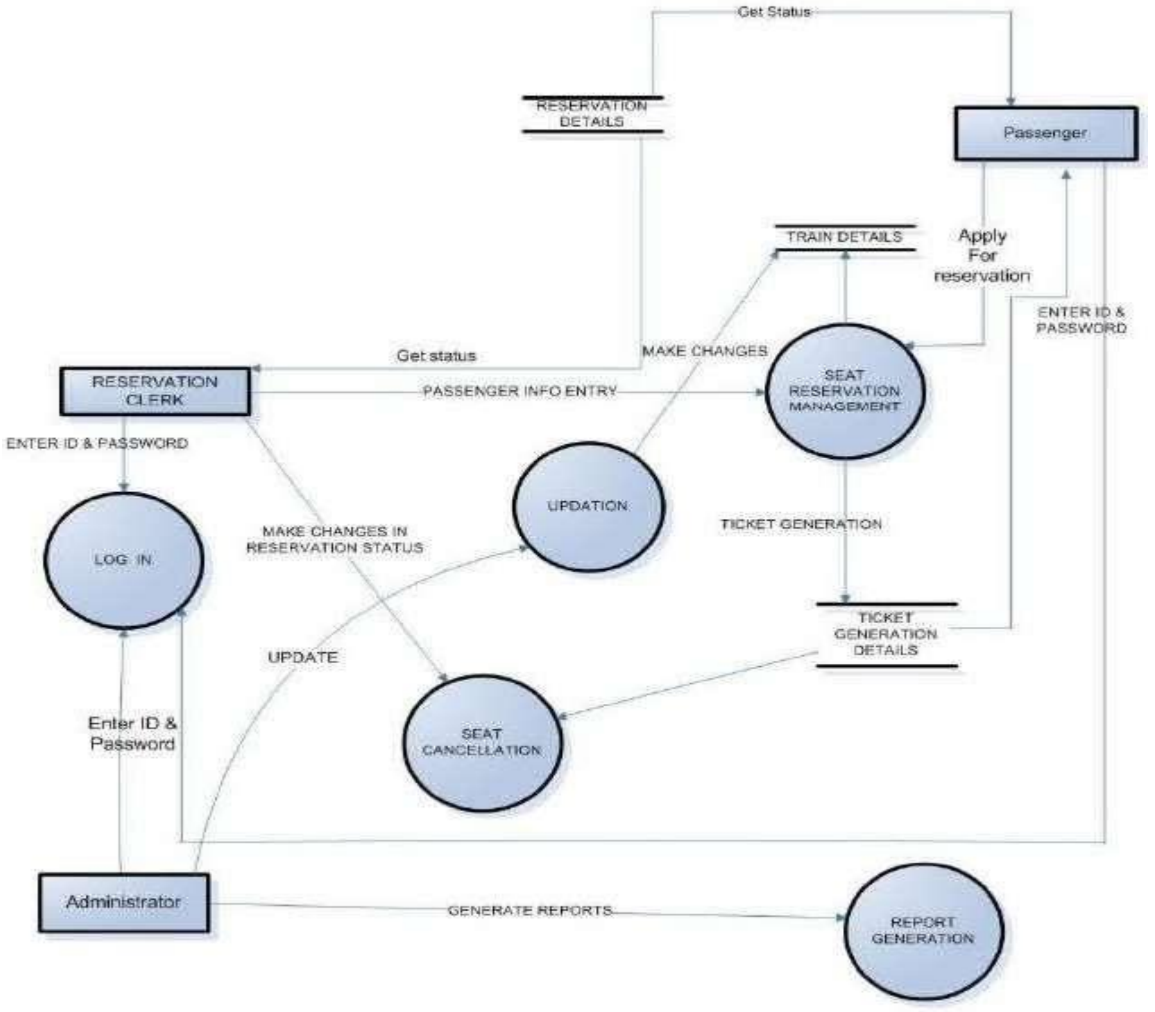


Data flow diagrams are also known as bubble charts.[5] DFD is a designing tool used in the top-down approach to Systems Design. This context-level DFD is next "exploded", to produce a Level 1 DFD that shows some of the detail of the system being modeled. The Level 1 DFD shows how the system is divided 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 of RAILWAY RESERVATION SYSTEM



➢ LEVEL- 1 of RAILWAY RESERVATION SYSTEM



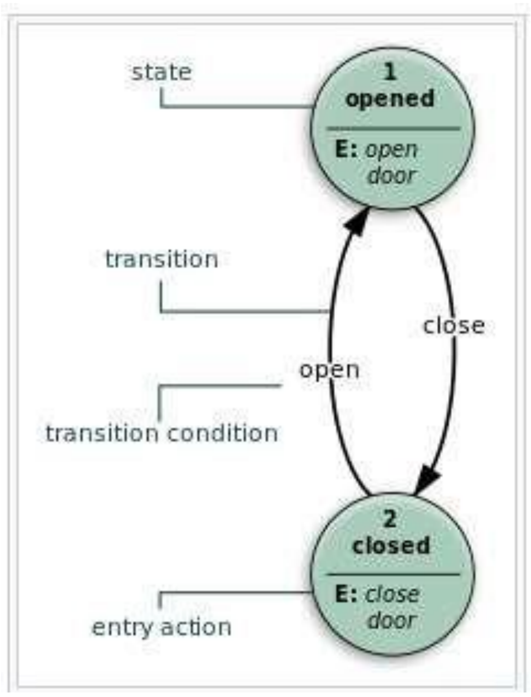
LEVEL 1 DFD: RAILWAY RESERVATION SYSTEM

Level 0: Describes the Dataflow with Login and Logout having processes of reservation and cancellation in between.

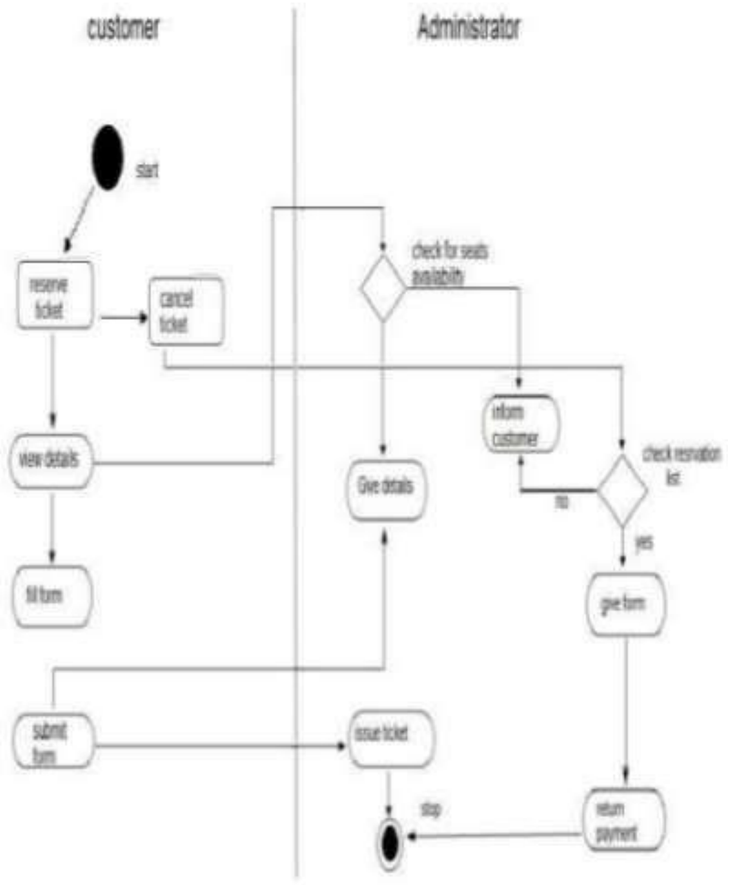
Level 1: Describes the data flow from Login to Logout consisting of Seat Reservation/ Cancellation management, Admin’s Report Generation Tasks etc.

4.3. STATE TRANSITION DIAGRAMS (STD):

A state diagram is a type of diagram used in computer science and related fields to describe the behavior of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.



State diagrams are used to give an abstract description of the behavior of a system. This behavior is analyzed and represented as a series of events that can occur in one or more possible states. Hereby "each diagram usually represents objects of a single class and track the different states of its objects through the system".


customer
Administrator
start
reserve ticket
cancel ticket
view details
fill form
submit form
check for seats availability
Give details
issue ticket
inform customer
check resrvation list
no
yes
give form
return payment
stop

5. Change Management Process:

Change is an event that results in a new status of one or more configuration items (CI's). Request for change is requested, by the initiator on a CI or a Service asset or on a service as whole.

Request for change (RFC) or a Change Ticket shall be created in the e-helpline by the person who wants the change to be implemented or has a request from customer to implement a change or raise RFC. In the absence of tool, manual RFC form shall be created.

RFC are triggered for a wide variety of reasons and from a wide variety of sources. Some of the possible reasons include:

- The RFC's shall be generated mainly by domain leads. The change requestor shall raise an RFC in the e-helpline tool.
- Changes must be submitted with the appropriate lead-time to ensure that appropriate individuals receive adequate notice of changes.
- The RFC is registered and a unique identification number is associated to the registered change to avoid duplication, tracking and to help monitoring purposes of the registered RFC.
- The unique identification number of the registered RFC is communicated to the Change Initiator.
- RFC can be raised from any other triggers also for e.g.:

    a. Business requirement (any cost factor related change)
    b. Planning for new or changed services (e.g. implementation of new AD/exchange version)
    c. New customer implementation (e.g. enterprise tools)
    d. Change required for resolution of an Incident or Problem
    e. Proposal to introduce or remove a service component
    f. Proposal to upgrade some component of the infrastructure

g. Changed business requirements or new direction or changed legislation/statutory and regulatory requirements

h. Change of location

i. Product or service change.

I. INITIAL ASSESSMENT AND RFC CHECK

Each Change request logged is verified for the completeness, accuracy of the information like:

a. Change Implementation Plan.

b. Change Implementation window.

c. If the CI or the Service Asset or the service would be up during the change implementation duration or not and so on.

d. Repetition of already raised RFC that are accepted rejected or still under consideration.

e. Impractical, inadequate RFCs without necessary budgetary approval.

f. Rollback Plan.

g. Executor and evaluator details.

Post Review of RFC with appropriate details would be further processed and details been recorded in the CSI portal and RFC with inappropriate details will be rejected and the outcome of verification should be informed to the initiator. Any such rejected RFC can be raised again after gathering the complete information required for proceeding with the change request.

II. REVIEW PRIORITY, TYPE

Pre-Approved Changes **(C0 & C0-Approval)**

The Pre-approved changes are the change which do not have any business impact / outage and can be executed with/without standard operational procedure. The preapproved changes need not mandatory to have POA, Build, and Testing etc. The final outcome of the change executed needs to be evidenced to declare successful closure.

C0 -Change will be approved by customer in mail and no tool approval will be triggered for the same

C0-Approval - Trigger to customer approval in tool

Emergency Change (C2)

If the type of change is an emergency change, then invoke Emergency change procedure.

Emergency or urgent changes have over-riding priority over scheduled changes. Emergency changes should follow similar steps as for Major, Minor Change, but needs to be executed in the shortest possible time.

An authorization body will be defined for an emergency change, called Emergency CAB (E-CAB) as it may not be possible to convene a full CAB Meeting in an emergency situation.

In Emergency changes, while the build process may not be very different, the testing may not be as exhaustive as other changes.

Essential testing should be done in case of emergency changes if there is no time to do complete testing.

Implementing untested system should be avoided to the maximum extent.

Emergency changes shall have Back-out planning. It is critical to have the relevant technical team on hand while deploying emergency changes.

Change Impact Analysis

Change Impact Analysis shall be recorded in the RFC considering the effects of change on the infrastructure, business applications, current service(s), related service(s), service portfolio, organizational resources, etc.

Analysis of benefits to be gained from the implementation of RFC as well as the expected costs shall be updated, wherever required.

The issue of risk to business has to be considered prior to authorization of the Change Request. Details of risks associated with the Change should be updated in RFC.

RFC originator can consider or seek inputs related to configuration items such as attributes, relationship etc. from the respective process owners or subject matter experts as appropriate. It is recommended to consider the following questions while evaluating the change.

- Who RAISED the Change?
- What is the REASON for the Change?
- What is the RETURN required from the Change?
- What are the RISKS involved in the Change?>
- What RESOURCES are required to deliver the Change?
- Who is RESPONSIBLE for the build, test and implementation of the Change?
- What is the RELATIONSHIP between this change and other changes

III. CONVENE CAB MEETING FOR CHANGE APPROVAL

All Changes recorded as Minor (C1) and Major (C3) changes shall be approved by CAB.

CAB meetings should be held as per CAB Meeting Guidelines.

List of Change Categories discussed:

Major (C3) and Minor (C1) changes.

<u>Participants:</u>

- Change Manager (Chair Person)
- Change Coordinator
- Problem Manager(Optional)
- Technical Manager/Domain Leads
- Requestor – OM/PM (On behalf of Customer)
- Executors - Domain Engineers.
- Customer & Onsite Teams (Optional)

<u>Frequency of CAB meeting & PIR discussion:</u>

PIR Discussion: Every Monday, Wednesday and Friday between 3:00 PM to 4:00 PM.

Registration of Changes:- Every Monday, Wednesday and Friday before 4:30 PM

CAB Meeting: Every Monday, Wednesday and Friday 4:00 PM to 6:00 PM.

The issue of risk to business has to be considered prior to approval of the Change Request. Details of risks associated with the Change should be updated in RFC.

In the CAB or CAB/ECAB, the Change Requests should be reviewed and the eligible ones should be selected for implementation. A formal approval must be obtained from the Change Authority for each change approved by the CAB or CAB/ECAB. The Change Authority could be the customer engagement manager.

The Change Requests should typically go through the following approvals:

Technical approval: The technical approval is required from the technical team. It is required to ensure that the change is feasible and does not affect the business or processes negatively.

Business approval: The approval from business is needed to ensure that the customer business team is convinced and are in line with the business benefit the change is supposed to bring.

The CAB should approve the change upon ascertaining that the approval is available from all the above two functions.

Emergency changes may not have time for multiple levels of approval. Based on factors such as impact assessment, risks and costs, CAB shall make a decision as to whether or not change should be pursued. CAB meetings should follow CAB Guidelines.

RFC status shall be communicated to the all the stakeholders before Implementation along with the time-frame, risk, impact, cost and other parameters which may have an business impact in future.

In case of cancellation of the activity for certain reason, Ticket needs to be cancelled though tools team or Change Team post, delivery manager approval and same must be communicated to all relevant stakeholders with proper reason and justification.

- **Emergency Change (C2) Management Procedure**

Activity	Description
Inputs	When a major incident happens, sometimes an emergency change needs to be deployed into production. The trigger for Emergency Change Management is always via the Major Impact Incident or the Business Urgency. The meaning of Emergency Change is that, this is a change that needs to go into production very quickly and hence may not have the necessary lead time required for the normal RFC process. While the scheduled change approvals are bypassed for an Emergency Change, it is how ever mandatory to complete the RFC from all aspects before the Change request is closed.

Activity	Description
	The Emergency Change management process triggers therefore by logging an Emergency Change in the Service Desk with the appropriate categorization.
1. Emergency Change needed?	The Change Manager will be notified of the Emergency Change. The Change Manager and the change owner will ascertain if the change is an emergency change or not. If the change is not an Emergency Change, the Service Desk Tool ticket is closed with the information updated.
2. Functional and technical requirements gathered	The change owner will be ready with all the functional, technical requirements, will also document impact of the change if launched and back out readiness. The costs of launching such a change along with the effort estimates are also kept ready for the emergency CAB meeting
3. Convene Emergency CAB meeting	An Emergency CAB (ECAB) meeting will be called for by the Change Manager. The ECAB meetings are not scheduled meetings, but will be convened based on the urgency of the situation. These meetings maybe via a bridge line opened by the Change manager. The ECAB members will be aware of the urgency of participating in the meetings. The following (but not limited to) are the members of the ECAB • Change Manager • Customer – Optional • Wipro Onsite/Offshore Service Delivery Managers • Technical Manager/Tower Lead • Major Incident Manager – Optional

Activity	Description
4. Approve?	The ECAB will consider the change from an impact/cost perspective and either approve or do not give approval. While a verbal approval may be the trigger to get the change to be built, it is mandatory to attach an email approval to the RFC. • Change Manager • Respective Client • Respective SDM or Head SNXT • Tower Lead or Head - Technical or Head SNXT
5. Change coordination and implementation	The change is scheduled for implementation, all relevant tower leads are informed via a broad cast mail by the Change Manager. The Change Owner coordinates the change with the Release Manager. Minimal tests are done via the Service Testing and Validation process and not extensive tests..
6. Implementation Successful?	The Change Manager validates the changes are implemented successfully and the objective were achieved
7. Trigger Roll back	In case the change is not successful, the change is rolled back and it will be scheduled again.
8. Complete RFC and associated documentation	Once the service is restored, the ticket is updated with all the information and the RFC is completed in all aspects. Related process is triggered to conduct a root cause analysis and problem elimination in future
9. Update the ticket information and Close the ticket	The ticket is updated by Engineer / Service Desk and change Ticket will be closed post validation of PIR by Change manager.

- **Roles and Responsibilities**

Change Manager	• Receive approved changes from the CAB and Organization Level process. • Coordinate the envisioning, planning, development, stabilizing, and deployment phases of the change • Reports status and present findings to the CAB. • Decide on CAB meeting attendees based on the RFC to be assessed, attendee expertise & decision making authority • Liaise with all necessary parties to coordinate Change building, testing and implementation in accordance with schedules • Review all implemented Changes to ensure that they have met their objectives. Refer back any that have been backed out or have failed • Analyse Change records to determine any trends or apparent problems that occur. Seek rectification with relevant parties • Close RFCs • Produce regular and accurate management reports.
Change Advisory Board (CAB)	• Review all submitted RFCs. As appropriate, determine and provide details of their likely impact, the implementation resources and the on-going costs of all Changes • Attend all relevant CAB meetings. Consider all Changes on the agenda and give an opinion on which Changes should be authorized. Participate in the scheduling of all Changes

	• Provide advice to Change Management on aspects of proposed Changes • Communicate changes to the Change Management process, stakeholders and promote the use of the changed process
Change Staff / Change Implementer	• Coordinate activities with cross functional teams • Identify specific needs of the cross functional teams and inform these needs to the Change Manager • Follow the Change Management process, procedures, work instructions, policies and required documentation • To be informed pro-actively of current and past changes • Document RFC for PIR, minutes of meetings (for e.g. CAB meetings) • Identify possible bottlenecks and issues regarding scheduled changes and communicate them to the Change Manager

A. Appendices

A.1 APPENDIX 1

Term	**Definition**
CMDB	Configuration Management Database
CAB	Change Advisory Board
RFC	Request for Change
FSC	Forward Schedule of Change
PIR	Post Implementation Review
DB	Database
UML	Unified Modeling Language
CI	Configuration Item
SD	Sequence Diagram
CM	Change Management
KPI	Key Performance Indicator
DFD	Data Flow Diagram
STD	State Transition Diagram

References:

Wikipedia.org

Slideshare.net

Cse.iitb.ac.in