**SOFTWARE**

**REQUIREMENTS**

**SPECIFICATION**

**DOCUMENT**

A Software Requirements Specification (SRS) is a document that describes the nature of a project, software or application. In simple words, SRS document is a manual of a project provided it is prepared before you kick-start a project/application. This document is also known by the names SRS report, software document. A software document is primarily prepared for a project, software or any kind of application.

There are a set of guidelines to be followed while preparing the software requirement specification document. This includes the purpose, scope, functional and non-functional requirements, software and hardware requirements of the project. In addition to this, it also contains the information about environmental conditions required, safety and security requirements, software quality attributes of the project etc.

**TABLE OF CONTENTS**

INTRODUCTION

* Purpose
* Document Conventions
* Intended Audience and Reading Suggestions
* Project Scope
* References

**OVERALL DESCRIPTION**

* Product Perspective
* Product Features
* User Classes and Characteristics
* Operating Environment
* Designing and Implementation Constraints
* Assumptions and Dependencies

**SYSTEM FEATURES**

* Functional Requirements

**EXTERNAL INTERFACE REQUIREMENTS**

* User Interfaces
* Hardware Interfaces
* Software Interfaces
* Communication Interface

**NON-FUNCTIONAL REQUIREMENTS**

* Performance Requirements
* Safety Requirements
* Security Requirements
* Software Quality Attributes

**PURPOSE**

The purpose of this document is to build an online system to manage trains and passengers to ease the train management.

**DOCUMENT CONVENTIONS**

This document uses the following conventions.

|  |  |
| --- | --- |
| DB | Database |
| DDB | Distributed Database |
| ER | Entity Relationship |

**INTENDED AUDIENCE AND READING SUGGESTIONS**

This project is a prototype for the train management system and it is restricted within the college premises. This has been implemented under the guidance of college professors. This project is useful for the train management team and as well as to the passengers.

**PROJECT SCOPE**

The purpose of the online train management system is to ease train management and to create a convenient and easy-to-use application for passengers, trying to buy train tickets. The system is based on a relational database with its train management and reservation functions. We will have a database server supporting hundreds of major cities around the country as well as thousands of trains by various companies. Above all, we hope to provide a comfortable user experience along with the best pricing available.

**REFERENCES**

* <https://krazytech.com/projects>
* Fundamentals of database systems by ramez elmarsi and shamkant b.navathe

**2. OVERALL DESCRIPTION**

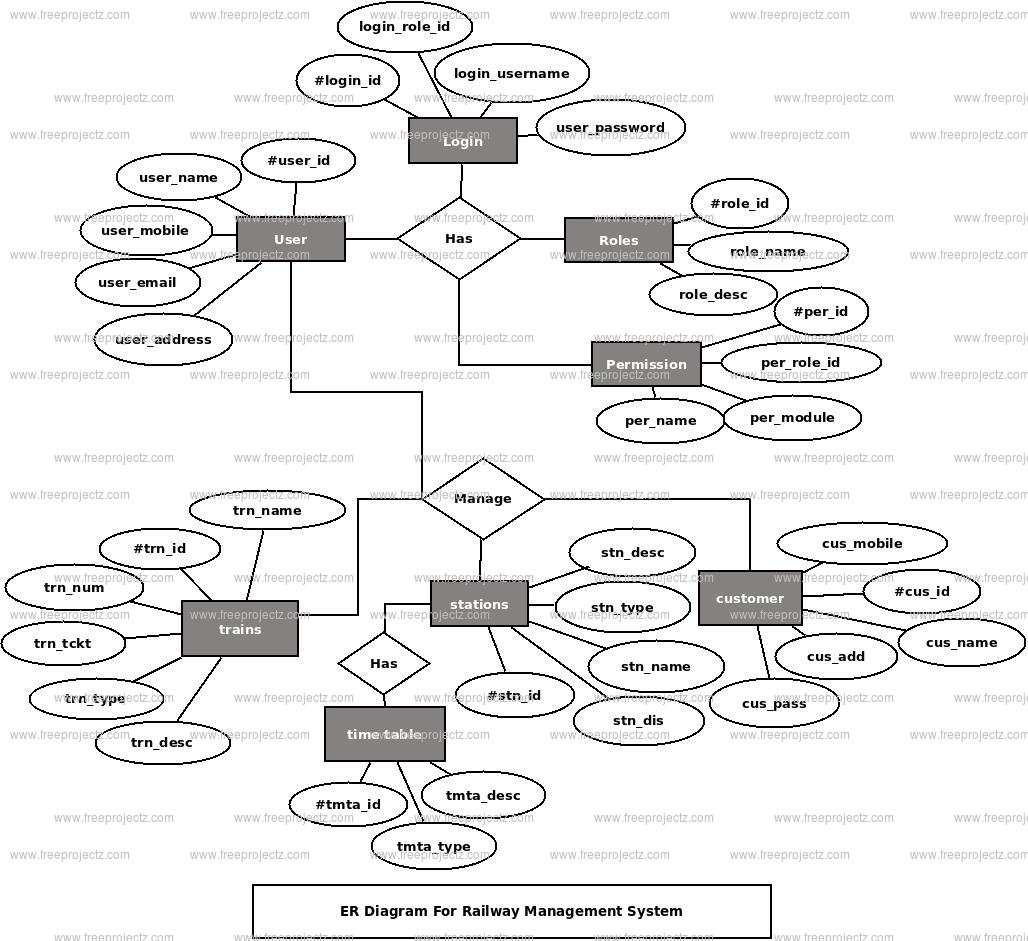
**PRODUCT PERSPECTIVE**

A distributed airline database system stores the following information.

* **TRAIN DETAILS:**  
  It includes the originating train terminal and destination terminal, along with the stops in between, the number of seats booked/available seats between two destinations etc
* **CUSTOMER DESCRIPTION:**  
  It includes customer code, name, address and phone number. This information may be used for keeping the records of the customer for any emergency or for any other kind of information.
* **RESERVATION DESCRIPTION:**   
  It includes customer details, code number, train number, date of booking, date of travel.

**PRODUCT FEATURES:**

The major features of railway database system as shown in below [**entity–relationship model**](https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model) (**ER model**)



**USER CLASS and CHARACTERISTICS**

Users of the system should be able to retrieve train information between two given cities with the given date/time of travel from the database. A route from city A to city B is a sequence of connecting trains from A to B such that: a) there are at most two connecting stops, excluding the starting city and destination city of the trip, b) the connecting time is between one to two hours. The system will support two types of user privileges, Customer, and Employee. Customers will have access to customer functions, and the employees will have access to both customer and railway management functions. The customer should be able to do the following functions:

* Make a new reservation
* One-Way
* Round-Trip
* Multi-City
* Flexible Date/Time
* Confirmation
* Cancel an existing reservation
* View his itinerary

The Employee should have following management functionalities:

* CUSTOMER
* Get all customers who have seats reserved on a given train
* Get all trains for a given station
* View trains schedule
* Get all trains whose arrival and departure times are on time/delayed
* Calculate total sales for a given train.
* ADMINISTRATIVE
* Add/Delete a train
* Add a new station
* Update fare for trains
* Add a new train leg instance
* Update departure/arrival times for train leg instances

Each train has a number of available seats. There are a number trains which depart from or arrive at different cities on different dates and time.

**OPERATING ENVIRONMENT**

Operating environment for the railway management system is as listed below.

* distributed database
* client/server system
* Operating system: Windows.
* database: sql+ database
* platform: vb.net/Java/PHP

**DESIGN and IMPLEMENTATION CONSTRAINTS**

1. The global schema, fragmentation schema, and allocation schema.
2. SQL commands for above queries/applications
3. How the response for application 1 and 2 will be generated. Assuming these are global queries. Explain how various fragments will be combined to do so.
4. Implement the database at least using a centralized database management system.

**ASSUMPTION DEPENDENCIES**

Let us assume that this is a distributed railway management system and it is used in the following application:

* A request for booking/cancellation of a railway from any source to any destination, giving connected trains in case no direct train between the specified Source-Destination pair exist.

Assuming both the transactions are single transactions, we have designed a distributed database that is geographically dispersed at four cities Delhi, Mumbai, Chennai, and Kolkata

**3. SYSTEM FEATURES**

* **DESCRIPTION and PRIORITY**

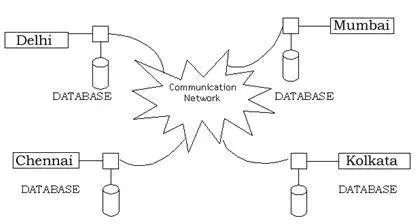
The railway reservation system maintains information on trains, classes of seats, personal preferences, prices, and bookings. Of course, this project has a high priority because it is very difficult to travel across cities without prior reservations.

* **STIMULUS/RESPONSE SEQUENCES**
  + Search for railway train tickets for two Travel cities
  + Displays a detailed list of available trains and make a “Reservation” or Book a ticket on a particular train.
  + Cancel an existing Reservation.
* **FUNCTIONAL REQUIREMENTS**

Other system features include:

**DISTRIBUTED DATABASE:**

Distributed database implies that a single application should be able to operate transparently on data that is spread across a variety of different databases and connected by a communication network as shown in below figure.

[](https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database/attachment/ad-distributed-database-located-at-four-different-cities)

*Distributed database located in four different cities*

**CLIENT/SERVER SYSTEM**

The term client/server refers primarily to an architecture or logical division of responsibilities, the client is the application (also known as the front-end), and the server is the DBMS (also known as the back-end).

A client/server system is a distributed system in which,

* Some sites are client sites and others are server sites.
* All the data resides at the server sites.
* All applications execute at the client sites.

**4. EXTERNAL INTERFACE REQUIREMENTS**

**USER INTERFACES**

* Front-end software: Vb.net version
* Back-end software: SQL+

**HARDWARE INTERFACES**

* Windows.
* A browser which supports CGI, HTML & Javascript.

**SOFTWARE INTERFACES**

Following are the software used for the railway management online application.

|  |  |
| --- | --- |
| **Software used** | **Description** |
| Operating system | We have chosen Windows operating system for its best support and user-friendliness. |
| Database | To save the train records, passengers records we have chosen SQL+ database. |
| VB.Net | To implement the project we have chosen Vb.Net language for its more interactive support. |

**COMMUNICATION INTERFACES**

This project supports all types of web browsers. We are using simple electronic forms for the reservation forms, ticket booking etc.

**5. NONFUNCTIONAL REQUIREMENTS**

**PERFORMANCE REQUIREMENTS**

The steps involved to perform the implementation of railway database are as listed below.

**E-R DIAGRAM**

The E-R Diagram constitutes a technique for representing the logical structure of a database in a pictorial manner. This analysis is then used to organize data as a relation, normalizing relation and finally obtaining a relation database.

* **ENTITIES:**Which specify distinct real-world items in an application.
* **PROPERTIES/ATTRIBUTES:** Which specify properties of an entity and relationships.
* **RELATIONSHIPS:** Which connect entities and represent meaningful dependencies between them.

passenger

enquire

confirm

Train details

Journey details

availability

**NORMALIZATION:**

The basic objective of normalization is to reduce redundancy which means that information is to be stored only once. Storing information several times leads to wastage of storage space and increase in the total size of the data stored.

If a database is not properly designed it can give rise to modification anomalies. Modification anomalies arise when data is added to, changed or deleted from a database table. Similarly, in traditional databases as well as improperly designed relational databases, data redundancy can be a problem. These can be eliminated by normalizing a database.

Normalization is the process of breaking down a table into smaller tables. So that each table deals with a single theme. There are three different kinds of modifications of anomalies and formulated the first, second and third normal forms (3NF) is considered sufficient for most practical purposes. It should be considered only after a thorough analysis and complete understanding of its implications.

**SAFETY REQUIREMENTS**

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed-up log, up to the time of failure.

**SECURITY REQUIREMENTS**

Security systems need database storage just like many other applications. However, the special requirements of the security market mean that vendors must choose their database partner carefully.

**SOFTWARE QUALITY ATTRIBUTES**

* **AVAILABILITY:** The train should be available on the specified date and specified time as many customers are doing

advance reservations.

* **CORRECTNESS:** The train should reach start from correct start terminal and should reach the correct destination.
* **MAINTAINABILITY:** The administrators and train in chargers should maintain correct schedules of trains.
* **USABILITY:** The train schedules should satisfy a maximum number of customer needs.