Metro System Database Management*

Jishan Shaikh
jishanshaikh9893@gmail.com

Ankit Chouhan
ankitchouhan.dw1@gmail.com

Swapnil Dubey
swapnildubey@gmail.com

Draft — May 15, 2018

Abstract

A Metro System is one which helps us to maintain an organizational data regarding a metro system. Since it is very difficult to maintain a record of data items and this information manually or in file system, we are creating a database for the complete management of resources and tasks of metro system. A Metro System database is one which is used for operational ease in place of traditional file system or Excel worksheets for information storage and retrieval with customized user queries implemented in SQL and is a fully functionally required software prototype with proper user interface and some random software engineering standards. Organizing database for data collection and retrieval helps us to maintain data easily. We design it with the help of Entity-Relationship diagram, UML diagrams, data flow diagrams, and using tables & proper User interfaces. It consists of several steps to be followed and many web-pages are to be maintained simultaneously. It is a working prototype of a database schema which can further be enhanced. The usage of standard web-pages in HTML, CSS, JavaScript, PHP helps us to take project online without much modifications.

For main functionalities, we have to maintain many records such as train numbers, train route, departure time, arrival time, train type, fare, managements, departments, customers, etc. When we want to travel we reserve seat (Additional feature for Metro System but it is not compulsory for all passengers to reserve seats for them, they can still check in at last moment in case of availability in first in first serve order), for this we have to fill the reservation form which includes entries such as train number, train name, departure time, arrival time, route, train type. So, we should gather all the information regarding all these.

Keywords – Metro System, Railway, Database, Prototype, Software Engineering principles.

1 Introduction

A Metro System database is one which is used for operational ease in place of traditional file system, or Excel worksheets for information storage and retrieval with customized user queries implemented in SQL, and is a fully functionally required software prototype with proper user interface and incorporating random software engineering standards.

Output of the Project:

1. Working prototype of a sample metro system with user interfaces and some queries.
2. Documentation and/or Project report.

*This comprehensive white-paper is based on project report published simultaneously at MANIT under supervision of Prof. Sweta Jain, and Github.com/AnkitJishan/Metro-System-Database-Project

Aims and Scope of the project

Aims:

1. To build a database used for practical implementation (For Metro System).
2. To learn the concepts of databases and their management of systems.
3. To understand software construction process and its engineering.
4. To learn various Software Engineering principles and standards.
5. To enhance team work and individual responsibilities.

Scope:

1. Software can be used theoretically (Modifications for practical usage)
2. Further modification and enhancements possible in maintenance stage
3. Research oriented aspect of project may lead to better technology implementations

Is the product output of this project is a software?

Sort of Yes. The work product output of this project is a 'software'; working properly in low scalability and is properly tested on basic functionalities, which is to be incorporated in that. An engineering approach is used for building that software that includes first define problem, analyze it, design it, implement it, test it, and regularly modify it. Technically the product output of this project is a demo prototype.

Open Source CASE tools used-

Upper CASE tools used:

1. LibreOffice Writer, and LaTeX(For documentation).
2. LibreOffice Draw (For designing diagrams).

Lower CASE tools used:

1. MySQL Workbench/LibreOffice Base (For SQL implementation).

Corresponding Conceptual models for this project-

1. Relational Model (See Appendix-A for All relations and attributes)
2. Enhanced Entity-Relational Model (See Entity-Relationship diagram)
3. Object Oriented UML modelling
   (a) Use case diagrams.
   (b) Sequence diagrams, etc.

See all diagrams in section 'Design Framework'.

Project management overview:

1. Embedded Project Planning
2. Project Scheduling (Internal in team)
3. Risk management (Overview and Intuitive)
4. Project monitoring and Controlling
5. W5HH Principles (See preliminary Synopsis)

6. Basic Project estimation
    (a) Cost estimation.
    (b) Duration estimation.
    (c) Effort estimation.

Level-0 Project Synopsis: W5HH Principles

Why is the project being developed ?

1. To learn and understand the conceptual and fundamental knowledge in Database Management Systems.
2. To contribute a little to open-source software community.
3. To have a practical hands on experience on a real life database system built and maintained by us.
4. To enhance team skills of all the members of team.
5. To learn modern technologies effectively with software engineering approaches.

What will be done ?

1. Analysis and design of metro database system
2. Implementation of design into various platforms
3. Evaluation of all the implementations
4. Building a final software with proper user interface
5. Prepare documentation, project report, and other documents

When will it be done ?

1. The project is supposed to be completed by April 2018
2. First release by April 15, 2018
3. Maintenance of that will be referred then to GitHub community

Where are the organizations located ?

1. This project is a single organization, single team project located at MANIT, Bhopal

Who is responsible for the desired functionality ?

1. Each member has assigned his/her own responsibility regarding projects in sections and of risk analysis, they have to follow their rules
2. Team members are atomic elements of project and are solely responsible for their work
3. Timely supervision is responsibility of the instructor.

How will the job be done technical and management side ?

Technical issues:

1. Database configuraton and building is to be done on Oracle Database 12c, MySQL Workbench 6.3, etc.
2. UML diagrams, Entity-Relationship diagrams are to prepared on LibreOffice Draw.
3. Documentaton is to done on LibreOfce Writer and a LaTeX editon will also be available (prepared on TeXLive)

4. For user interface, either java or python or web-development languages (HTML, CSS, JS, PHP) will be used with SQL as major query language

Management issues: Software project management guidelines are to followed on following-

1. Risk analysis.
2. Cost estimation.
3. Time scheduling.
4. Project monitoring and controlling.

How much of each resource is needed ?

1. Human resource of 3 team members
2. Monetary resource of at most 1,000 INR.
3. Hardware resource of a computer with required software installed.
4. Software resources used.

2 Literature Survey

Database Management Systems

Data Access: Modern relational database management system programs use a programming language known as structured query language to access, update, and delete data within its tables. These programs, including Microsoft's SQL Server and the open-source MySQL systems, allow outside programs to access its data via SQL queries. For instance, a web site can display product data, including photos, prices and descriptions, when the web server software connects to the data contained in the relational database management system.

Data Relationships: One of the most important aspects of an relational database management system program is how it allows different data tables to relate to each other. When a database contains a table with employee data on its sales staff and another with data on its product sales, the relational database management system can manage the relationship between the two tables. This relationship can help management determine which salesperson has the highest sales totals and which product that salesperson is selling the most.

Data Updates: A fully-functional relational database management system allows users to enter new information, update current records and delete outdated data. As an example, when a salesperson sells 1,000 units, that person will enter the transaction information into the relational database management system. The data can include the salesperson's name, the customer information, the product sold and the quantity sold. The relational database management system enters a new record in the customer table, updates the salesperson's record and subtracts 1,000 units from the inventory record.

Data Searches: The relational database management system also ensures that a company can build and maintain its data over the system's lifetime. The various tables in the relational database management system allow users to search through the system using any available criteria. Customers can search a product table by name, brand, price, color or any other feature. The system stores data in a predictable, sequential format, enabling users to look up previous records with relative ease.

Structured Query Language (SQL)

SQL (Structured Query Language) is a standardized programming language used for managing relational databases and performing various operations on the data in them. Initially created in the 1970s, SQL is regularly used by database administrators, as well as by developers writing data integration scripts

and data analysts looking to set up and run analytical queries. The uses of SQL include modifying database table and index structures; adding, updating and deleting rows of data; and retrieving subsets of information from within a database for transaction processing and analytic applications. Queries and other SQL operations take the form of commands written as statements – commonly used SQL statements include select, add, insert, update, delete, create, alter and truncate. SQL became the de facto standard programming language for relational databases after they emerged in the late 1970s and early 1980s. Also known as SQL databases, relational systems comprise a set of tables containing data in rows and columns. Each column in a table corresponds to a category of data – for example, customer name or address – while each row contains a data value for the intersecting column.

SQL standard and proprietary extensions

An official SQL standard was adopted by the American National Standards Institute (ANSI) in 1986 and then by the International Organization for Standardization, known as ISO in 1987. More than a half-dozen joint updates to the standard have been released by the two standards development bodies since then; as of this writing, recent version is SQL:2011, approved that year. Both proprietary and open source RDBMS built around SQL are available for use by organizations. They include MS SQL Server, Oracle Database, IBM DB2, SAP HANA, SAP Adaptive Server, MySQL (now owned by Oracle) and PostgreSQL. However, many of these database products support SQL with proprietary extensions to the standard language for procedural programming and other functions. For example, Microsoft offers a set of extensions called Transact-SQL (T-SQL), while Oracle's extended version of the standard is PL/SQL. As a result, the different variants of SQL offered by vendors aren't fully compatible with one another.

SQL commands and syntax

SQL commands are divided into several different types, among them data manipulation language (DML) and data definition language (DDL) statements, transaction controls and security measures. The DML vocabulary is used to retrieve and manipulate data, while DDL statements are for defining and modifying database structures. The transaction controls help manage transaction processing, ensuring that transactions are either completed or rolled back if errors or problems occur. The security statements are used to control database access as well as to create user roles and permissions.

3 Problem Statement

The major task is to prepare a fully functional database system prototype (with user interfaces) for a metro network (quite similar to Indian Metro ltd./Indian Railways) which will be implemented as a working open source prototype/project (hosted at GitHub) prepared with the help of standard software engineering techniques, software project management guidelines, proper documentation & report preparation, etc. using various platforms [LibreOffice Base and MySQL Workbench for database preparation] using CASE tools such as LibreOffice Writer for documentation and diagram designing. User Interfaces are built in form of web-pages using HTML, CSS, JavaScript, PHP for proper handling of user queries and data entrance.

Problem (Task) description and major tasks

Project aspects : (P-Poor, G-Good, E-Excellent)

1. Reliable backup and recovery G
2. Distributed functions G
3. Performance G
4. Heavily used configurations P
5. Online data entry P

6. Operational ease G
7. Updates G
8. User Interface E
9. Complex processing G
10. Re-usability of Software E
11. Installation ease G
12. Multiple sites P
13. Facilitated Changes E

We've explored various aspects of a practical system (Indian Metro/Indian Railways) at overview level and seems basic attributes (basis records) to be included in the database and are listed as-

1. Train No.
2. Station No.
3. Reservation ID
4. Passenger ID
5. Departure Time
6. Train Type
7. Fare and Distances
8. Source – Destination, etc.

Most of the above attributes are taken into account for project problem and are properly incorporated in relational model design of project and in Entity-Relationship diagram. See E-R diagram.

Tasks:

1. Planning and Analysis of commercially available software on metro system
2. Database implementations
3. Standard testing of all implementations
4. Performance analysis of all implementations (to evaluate all implementations)
5. Providing a user interface for end & front users, administrators, etc.
6. Make a fully functional prototype.

4 Software and Hardware Requirements

Minimum System/Software/Hardware requirements:

1. Windows 7/Linux 3/Mac OS/Chromium OS
2. Memory requirements: 64 MB (RAM). Recommended 128 MB.
3. Secondary memory requirements (Hard disk): 10 GB (ROM). Recommended 256/512GB
4. Oracle MySQL Workbench/LibreOffice Base. Recommended MySQL Workbench/Command Line Client
5. Writer and diagram designing software such as MS Visio or LibreOffice Draw.

6. Clock Speed: 866 MHz
7. Virtual Memory: 32 bits
8. Cache Memory: 512KB, etc.

Resources Usage:

1. Operating System: Windows 10/Ubuntu 17.10
2. Memory: 8 GB (RAM)
3. Secondary memory: 1 TB (ROM)
4. Oracle MySQL Workbench 6.3 CE (Windows)/ LibreOffice Base 5 (Ubuntu 17.10/Linux 4)
5. Microsoft Word 16/ LibreOffice Writer 5.4 (Linux)

Functional requirements:

1. Customer/Passenger/Employee data storage and retrieval
2. Proper User Interface such as input forms
3. Customized user queries for complete system including proper error handling

Performance requirements:

1. Shouldn't wear out in passage of time
2. Shouldn't lag in responding to user inquires
3. Shouldn't crash abruptly
4. Must be in accordance to data consistency (Proper resemblance of foreign keys)

5 Methodology Used

The methodology adopted in the project is simple and is easy to implement. Functions and Object oriented methodology constitute a major portion of methodology. Project is divided into individual tasks. Tasks are then taken as initial problem and solved individually and have to integrate them at last. Objectives are set first and then we tried to fulfill all objectives asserted.

We've first implemented preliminary Entity-Relationship diagram in 2 platforms namely- MySQL Workbench (Windows 10) and LibreOffice Base (Ubuntu 17.10). We then intuitively analyzed which platform is more suitable for this particular schema and required functionalities. After that we enhance that schema to further more relations, entities, relationships, forms, queries, etc.

The Project development model we used in project development is closely related to hybrid association of waterfall phases, Rapid application development (Rapid delivery of project), Agile methodology (Simplicity and Swift evaluation), Some Extreme Programming practices, and 4th generation tool usage (Usage of CASE tool: LibreOffice Base/Draw). The involvement of multiple model element in single project leads to higher level of customization, better management of project, better requirements adaptability, and improved rigidity of overall management procedure and development life cycle.

We've often used many Software engineering principles in the project to make it well organized, formal, and more productive. Use of Open-Source tools helps us to contribute Free software foundation and Open-source community. We've modeled the project structure into various diagrams such as Entity-Relationship diagrams, Use-Case diagrams, Sequence diagrams, Data flow diagrams. An Architecture diagram is also designed for an overview of database procedure inside the system and the interconnection of network between Client and Server.

6 Design Framework

Checkout figures 1-6 for design diagrams.

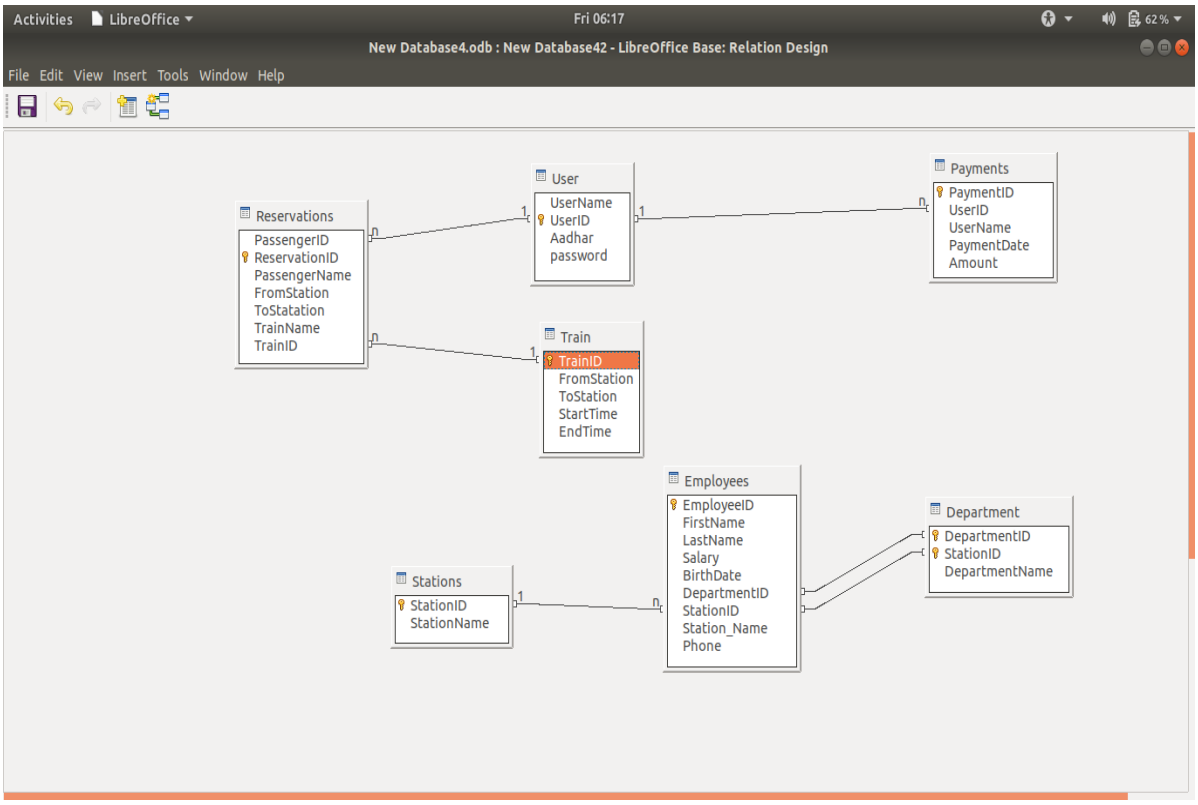

Figure 1: Entity-Relationship diagram (Generated from LibreOffice Base).

7 Implementation

Coding

Coding is mainly done in Structured Query Language (SQL) for MySQL Command Line Client for customized queries. The Codes of HTML, CSS, JS, and PHP are formatted, understandable, customization, and modifiable. We can combine other functionalities using these codes since there we kept a margin for re-usability. Check out Implementation snapshots for a view of codes.

Also Check out Appendix-B for executable codes of sample queries in SQL.

Implementation snapshots

Front-end web-pages:

Checkout figures 7-13 for front end web-pages.

Back-end codes: Checkout figures 14-21 for back-end code snapshots.

MySQL Command Line Client: Checkout figures 22-25 for MySQL CL client snapshots.

8 Testing

Procedure

The procedure of testing of SQL codes used is quite simple, standard, effective. The steps used in complete testing procedure is shown below-

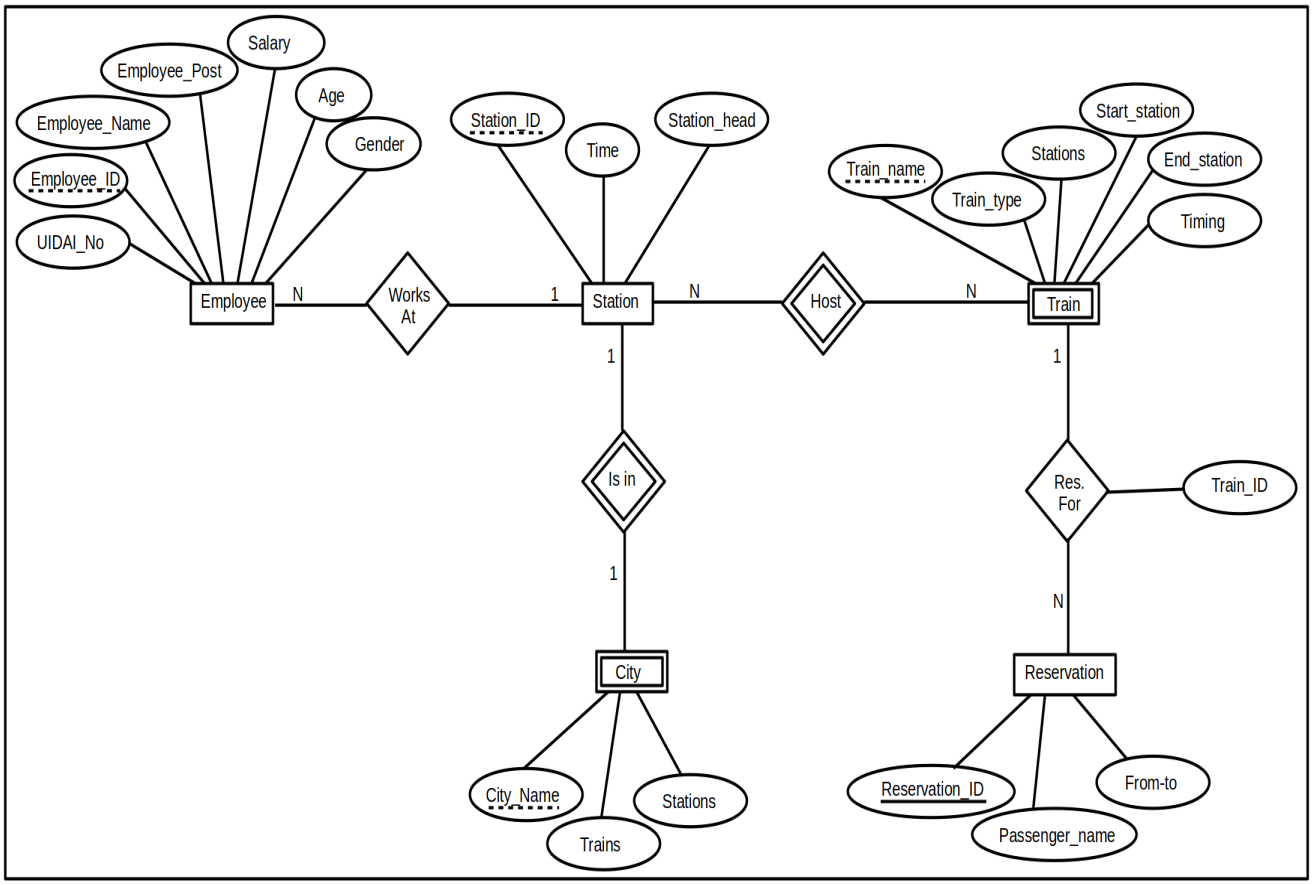

Figure 2: Entity-Relationship diagram (Preliminary).

1. Checking for SQL Syntax by directly running the queries
2. Checking for broader logic's of code: Is code working according to logic's or not?
3. White box Checking

   3.1 Structural testing which checks outer structure of logic
4. Black box checking

   4.1 Functional testing which checks if code working according to functionality or not.
5. Coding standards which is not a testing step but we check for some standard code including good formatting, alignments, procedure calls, etc.

Outcomes

Testing provides us to minimize sort of error prone functionalities as per our implementation. Here, following simple procedure for testing we get desired hassle free functionality. Project is currently working fine and since testing procedure is available, we can further apply that testing procedure in case of malfunction or bug findings.

9 Conclusion

Usage of database today is of utmost importance for an enterprise or an institution because it eliminates all the major drawbacks of file system via Programming language or Offline file saving. It helps in improving the whole procedure of data/information storage and retrieval. There are also some risks

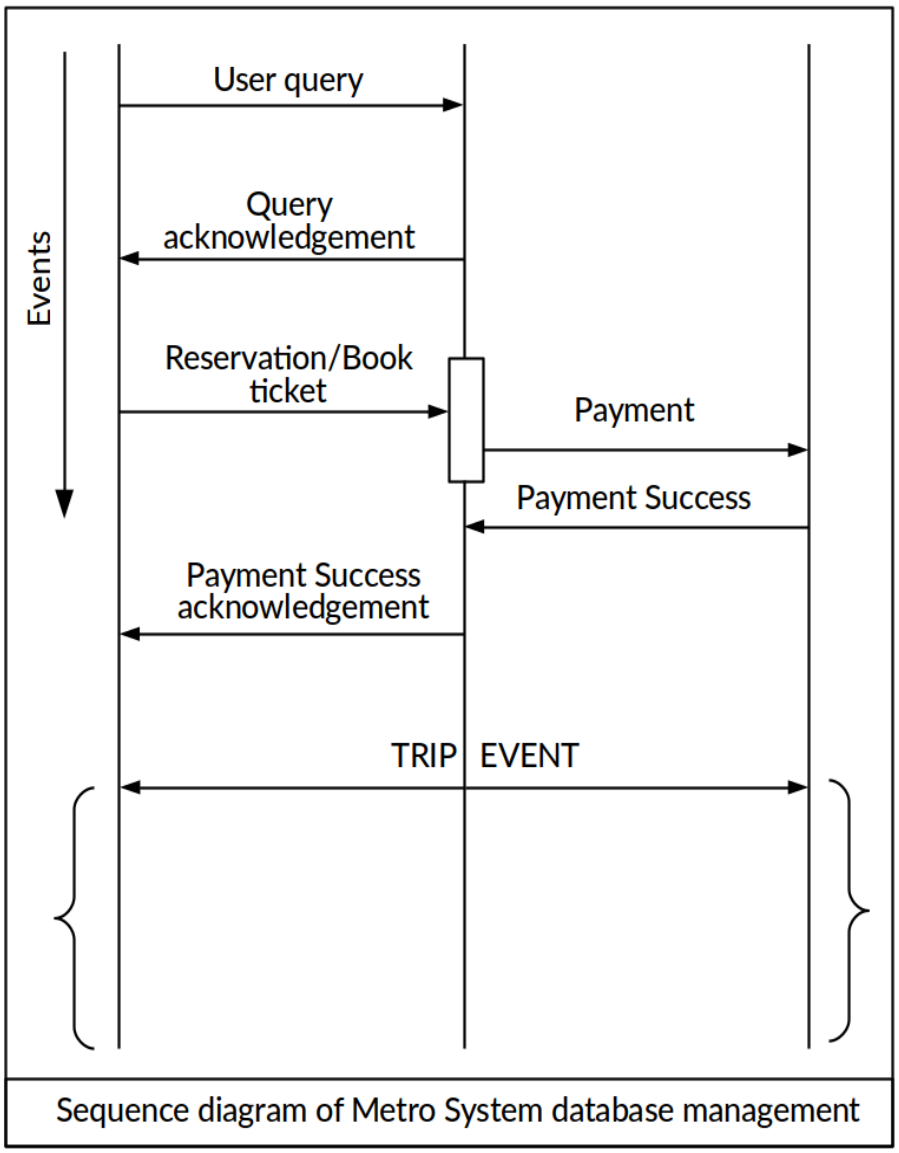

Figure 3: Sequence Diagram.

such as injection techniques, Reliability issues, data inconsistency risks, etc. Hence it requires to construct a proper Risk Mitigation, Monitoring, and Management plan (RMMM Plan) for specific usage of databases practically.

We keep our project small and simple which led to some major improvements in costs, efforts, and time usage in complete life cycle of project development. We've tried to include all major functionalities which must be there in a metro/railway system database but still it is far behind than a practical working software. It requires large number of attributes their manipulation and maintenance. We've completed major objectives of our project and the project is now in its maintenance phase which will continue at GitHub where it will be proper version controlled and is properly maintained by subject experts, reviewers, students, etc.

We had not built RMMM plan for every risk instead we take look at each and every feasible risk in development procedure, and simple eliminate it by whatsoever reasons.

9.1 Future Scope

In near future the usage of databases will probably be decreased due to huge sources of data and new technologies such as Big data analytic, which will further improve the drawbacks of database management system.

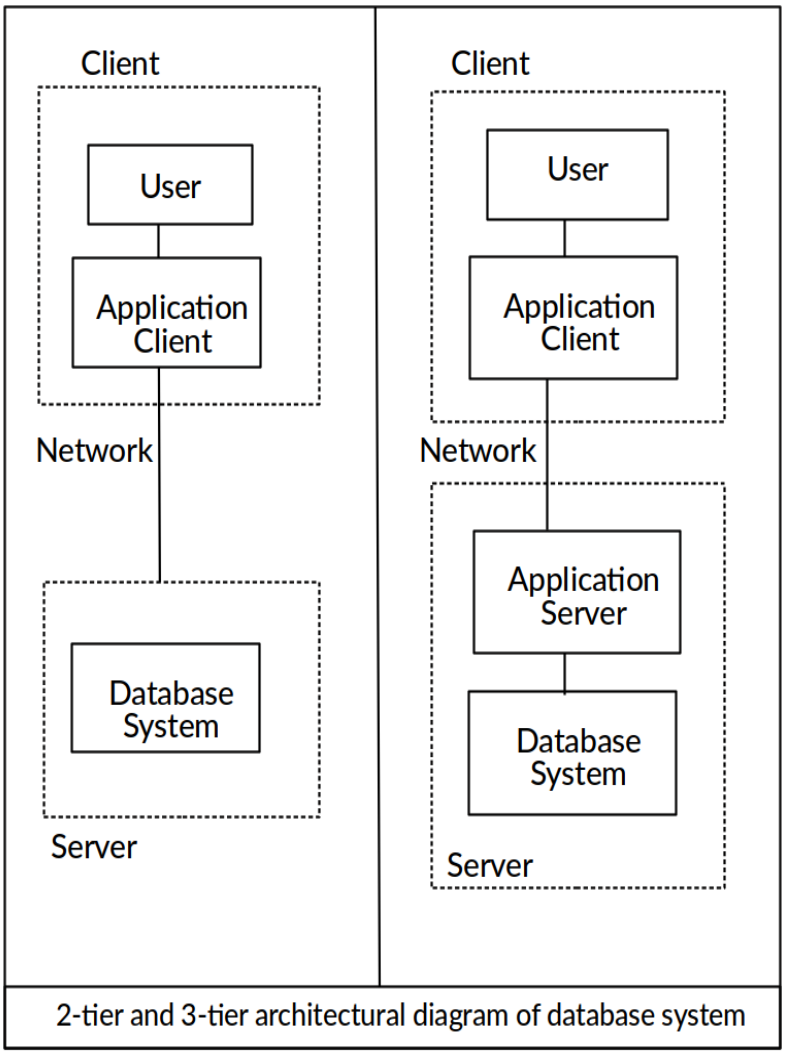

Figure 4: Database Architectural Diagram (2-tier and 3-tier).

The project output can be used as raw input to a bigger project which may led to practical implementation of various concepts and debugging. This can be enhanced much on following points-

1. **User Interface:**
   (a) It can be ported to a simple application using android studio/iOS builder which can then be published on Play Store, or iOS Store.
   (b) It can be reconstructed by Graphics programming user Java Swing, Java awt, or Python Tkinter for web applications.
2. **Implementation:**
   (a) It can be implemented using Oracle database 12C, DB2, PostgreSQL, MongoDB. LibreOffice is currently using HSQLEmbedded as embedded database, instead of which other databases can be used as per requirements.
   (b) The relational schema for Metro System can also be implemented on Object Oriented database which has its own pros and cons.
3. **Methodology:** We've used simple yet hybrid methodology for customization of events in development of out project.
   (a) Agile methodology can be used completely which will take the performance of project development at high level.

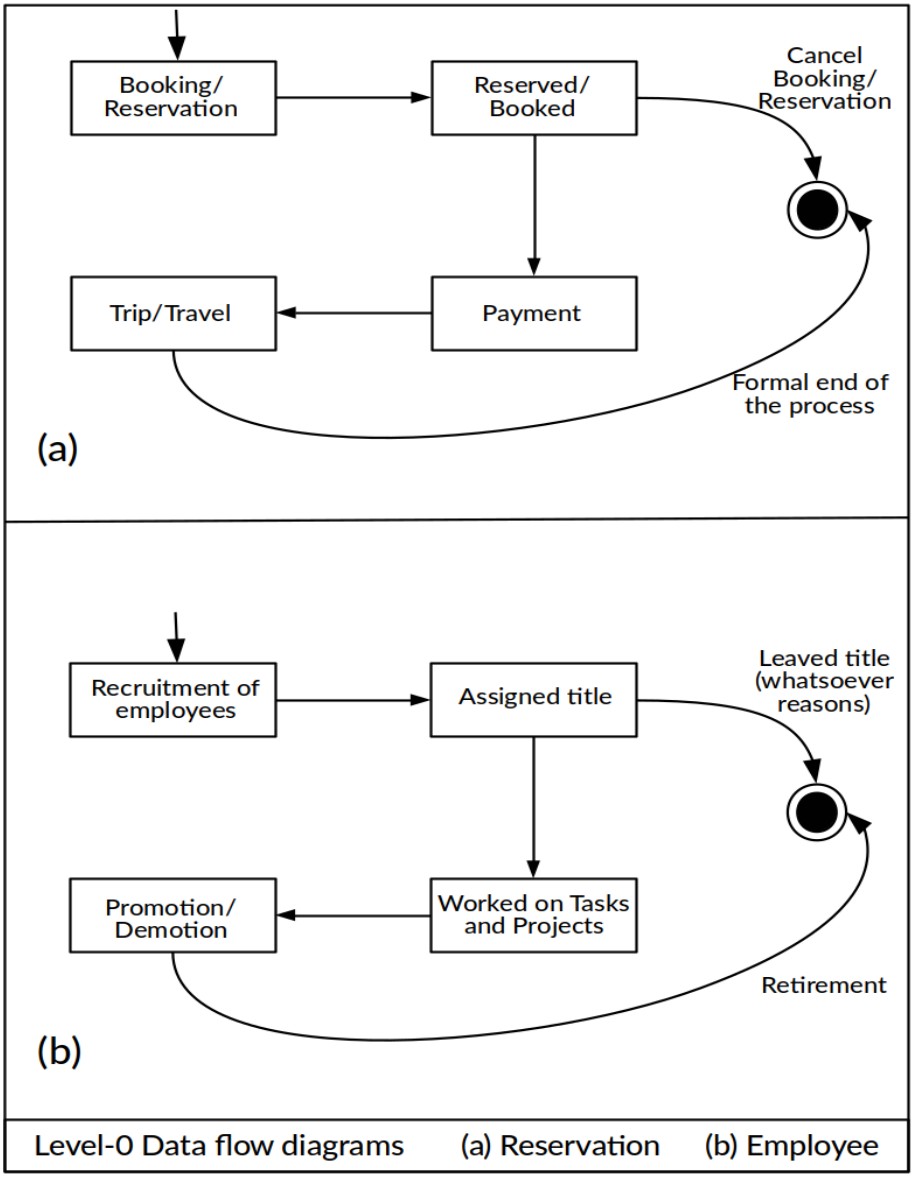

Figure 5: Data Flow Diagrams (Level-0).

(b) Extreme Programming practices and activities can also be used in Project development life cycle which has its own further improvements.

4. **Standards:** We've simply used unorganized guidelines and develop project using random software standards.

   (a) Instead of random software standards we can use organized ISO (International standards organization) standards and guidelines such as ISO 9001-2008.

   (b) Capability Maturity Model (CMM) Guidelines can be used in organized manner in at least level-3.

   (c) Six Sigma certification aspects can also be incorporated in project development.

References

[1] *Database System concepts.* Abraham Silberschatz, Henry F. Korth, S. Sudarshan. 6^{th} Internatonal edition. McGraw-Hill, New York. ISBN 978-0-07-352332-3. MHID 0-07-352332-1.

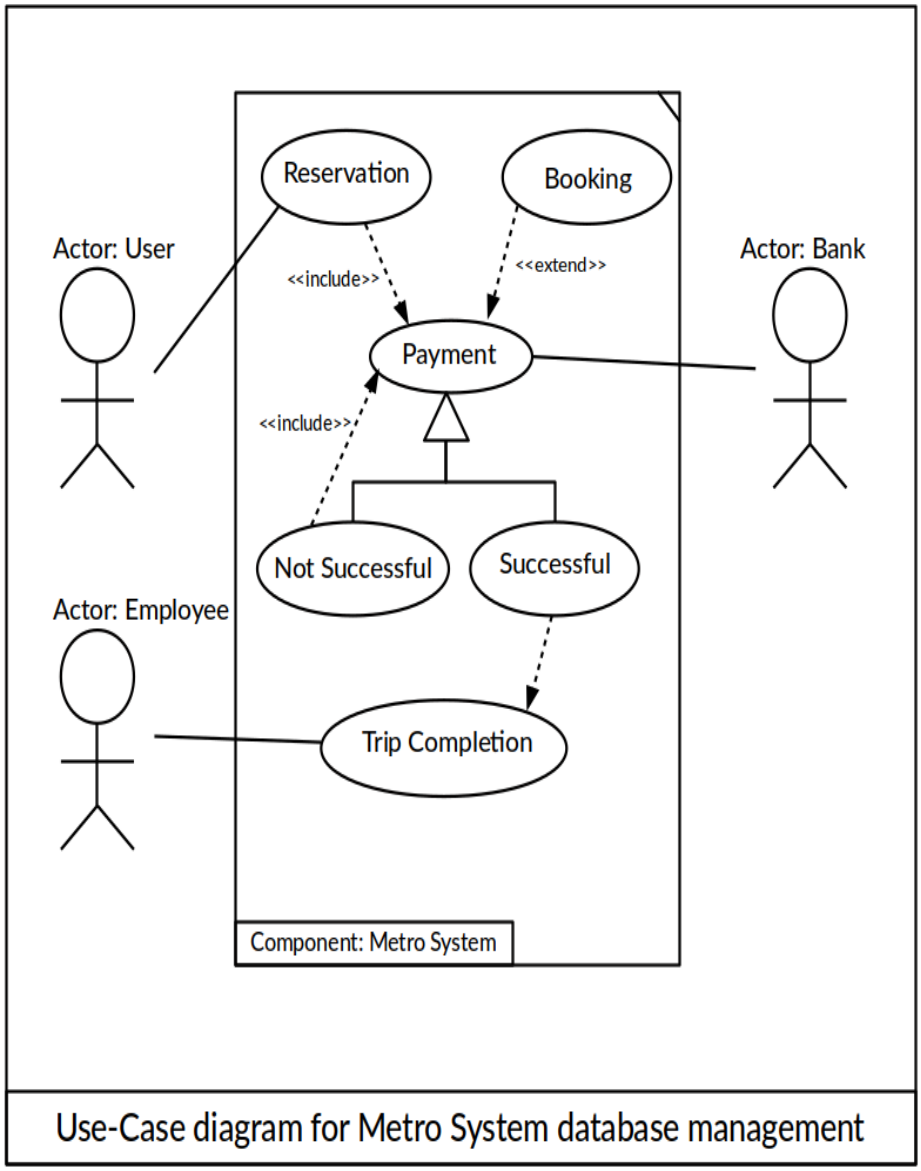

Figure 6: Use Case diagram.

[2] *Database Systems The Complete Book.* Hector Garcia-Molina, Jefrey D. Ullman, Jennifer Widom. 2nd edition. Prentice Hall, New Jersey. ISBN 0-13-606701-8, 978-0-13-606701-6.

[3] *Database Management Systems.* Raghu Ramakrishnan, Johannes Gehrke. 2^{nd} edition. McGraw Hill higher education. ISBN 0-07-246535-2.

[4] *Database Management System tutorial.* Database Management Systems tutorial. Tutorials Point. www.tutorialspoint.com.

[5] *SQL tutorial.* Structured Query Language Tutorials Point. www.tutorialspoint.com.

[6] *Database Management Systems* Scribe class notes on DBMS (CSE-224) at MANIT, Bhopal. Instructor: Ms. Shweta Jain.

[7] *Colorlib* Login form 13. www.colorlib.com Retrieved March 2017.

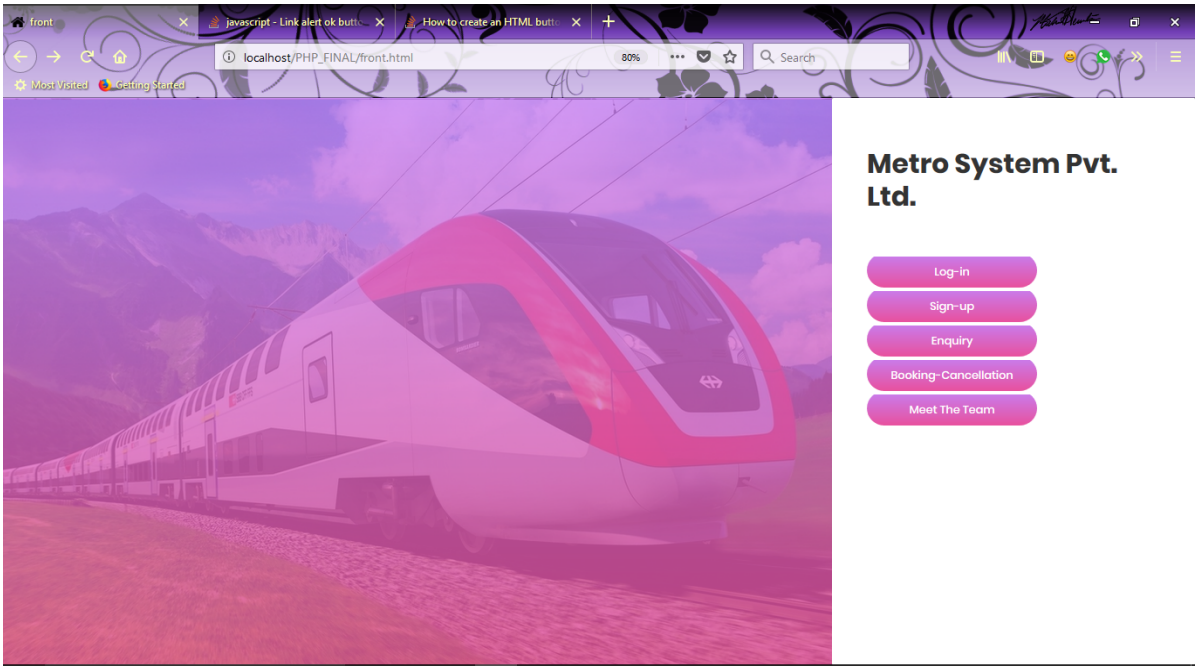

Figure 7: Home page (Metro System)

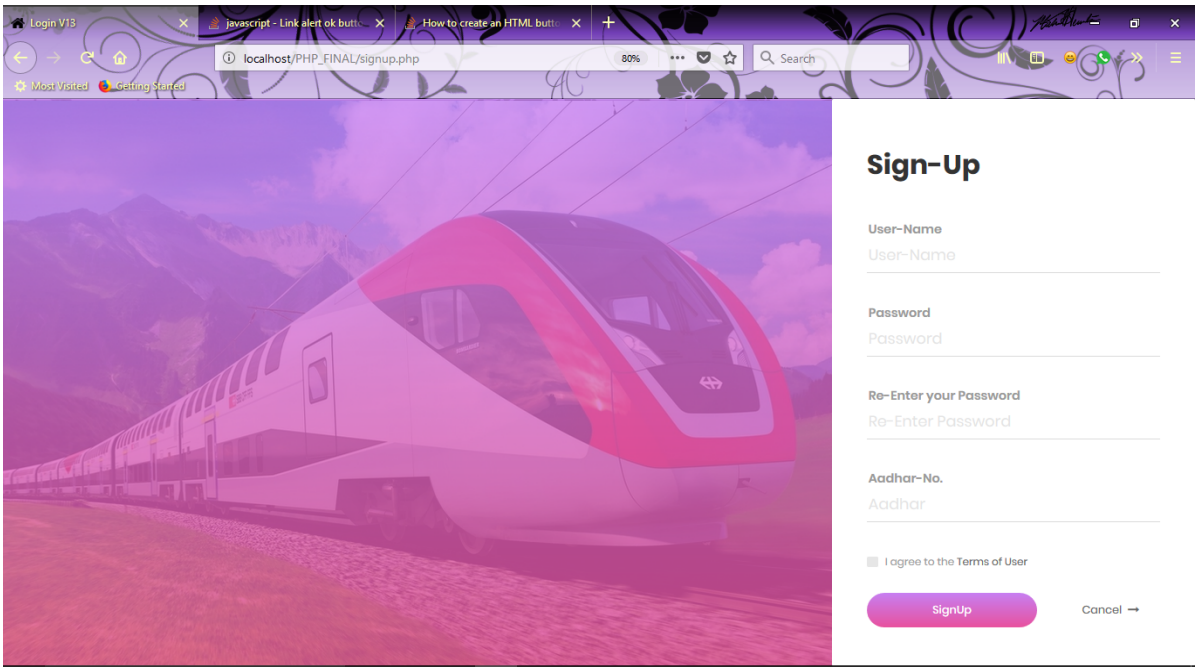

Figure 8: Sign-Up page (Metro System)

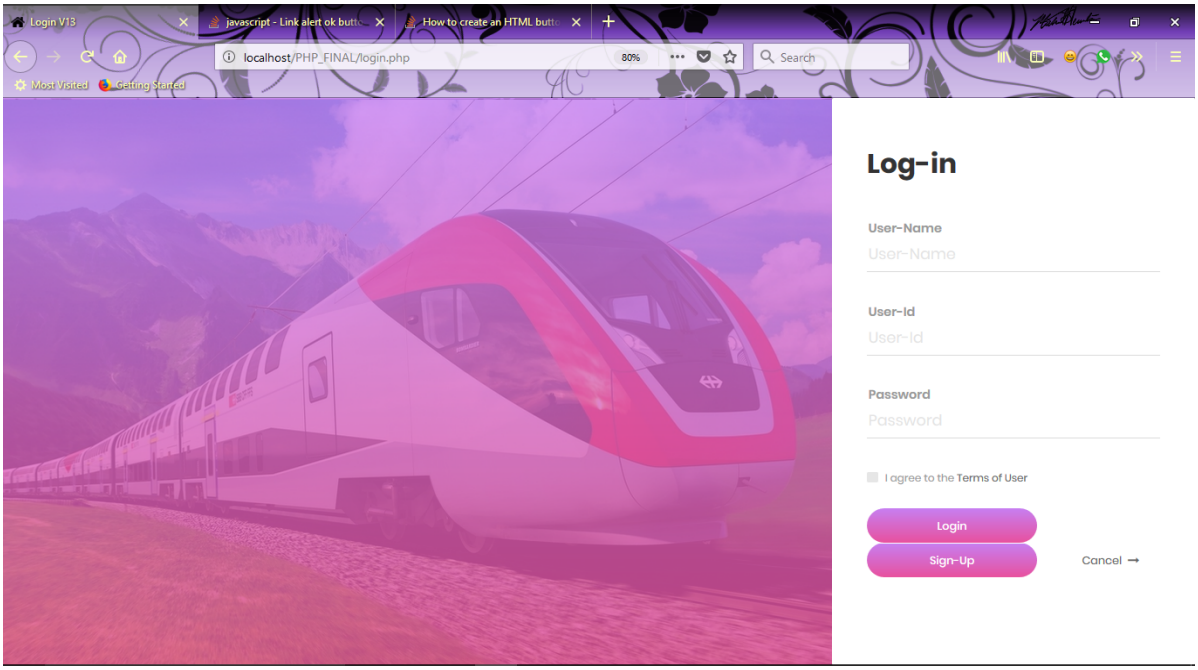

Figure 9: Log-in page (Metro System)

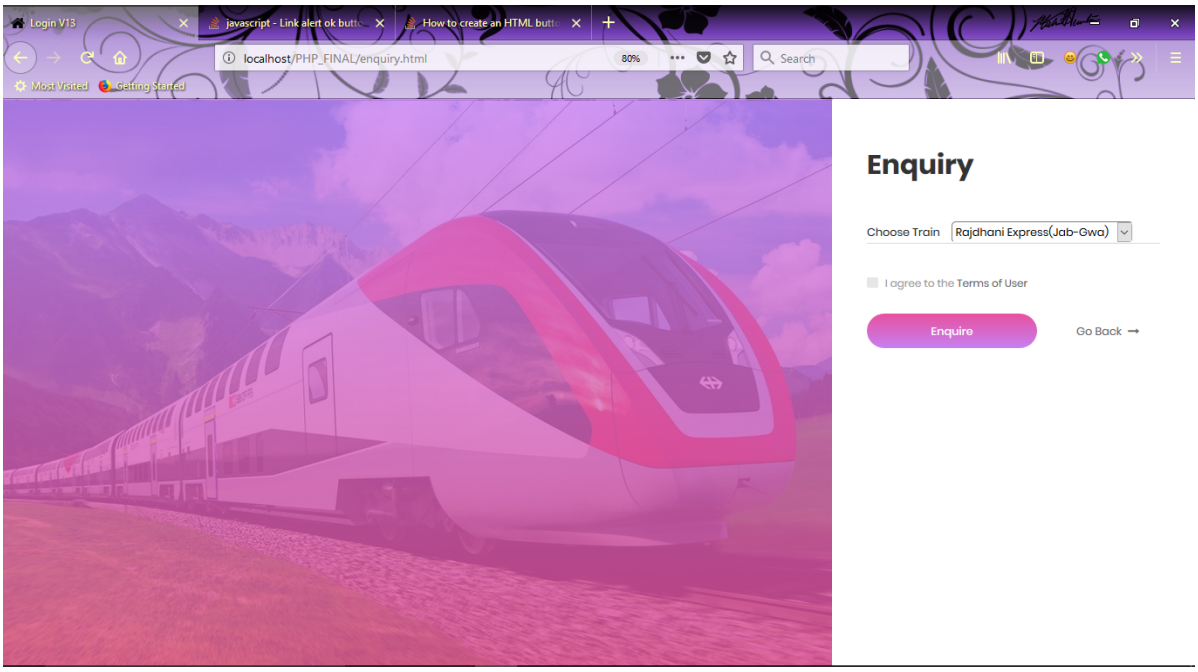

Figure 10: Enquiry page (Metro System)

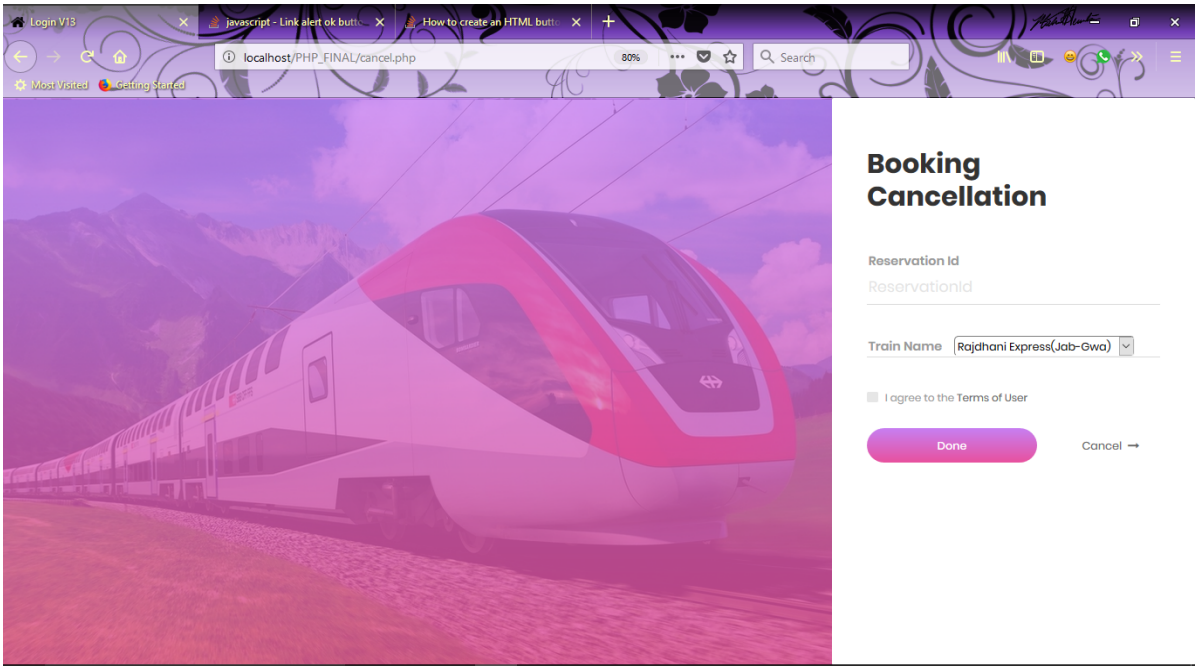

Figure 11: Booking Cancellation page (Metro System)

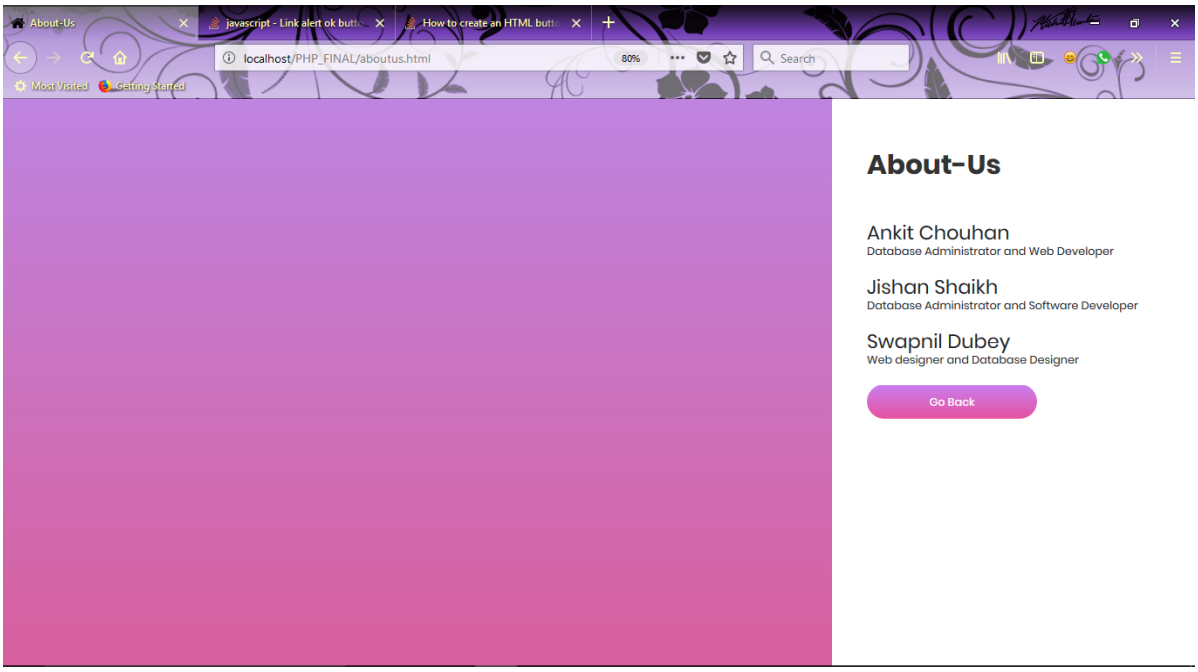

Figure 12: About-Us page (Metro System)

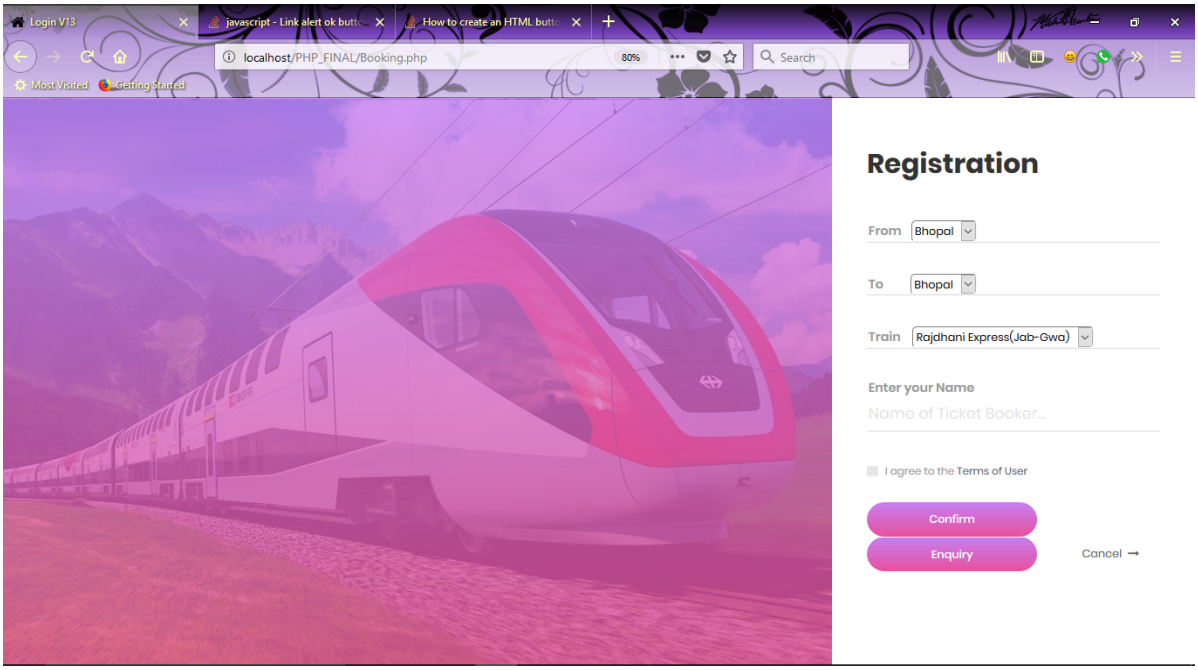

Figure 13: Registration page (Metro System)

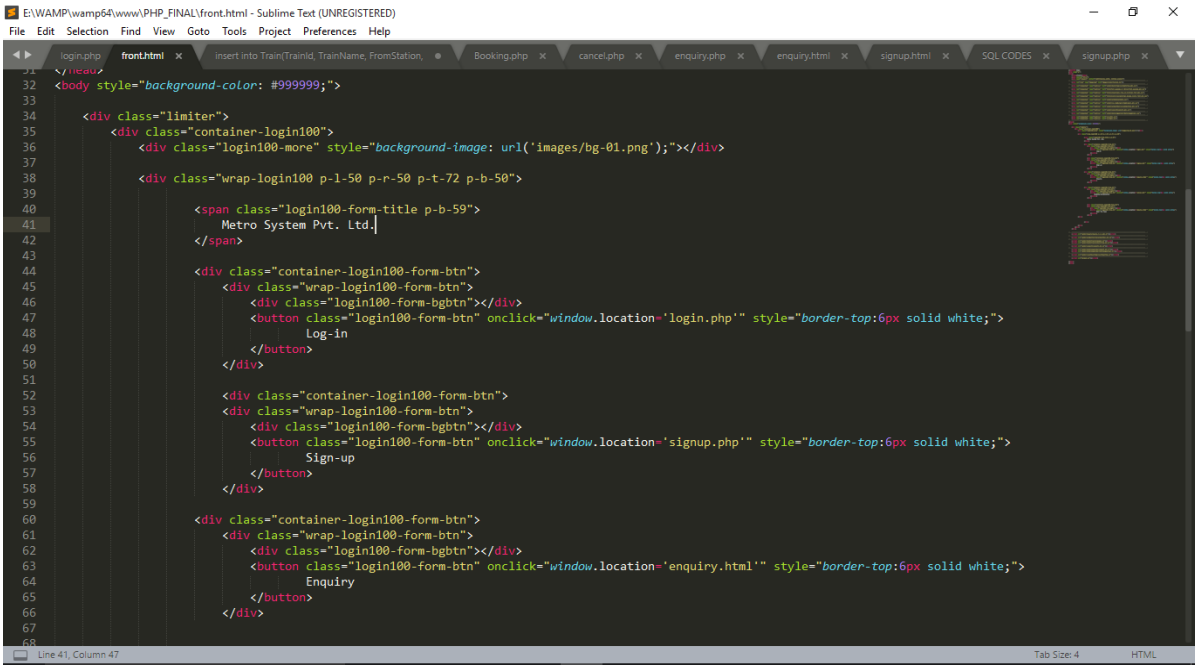

Figure 14: Back-end codes snapshots (Front.html)

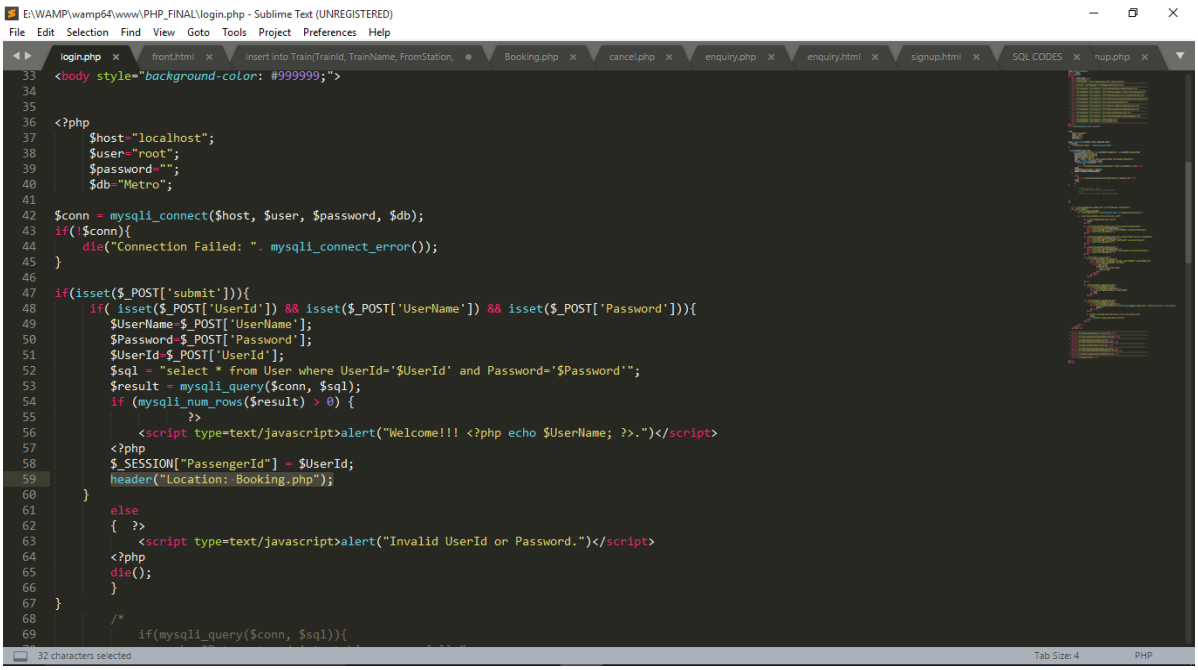

Figure 15: Back-end codes snapshots (login.php)

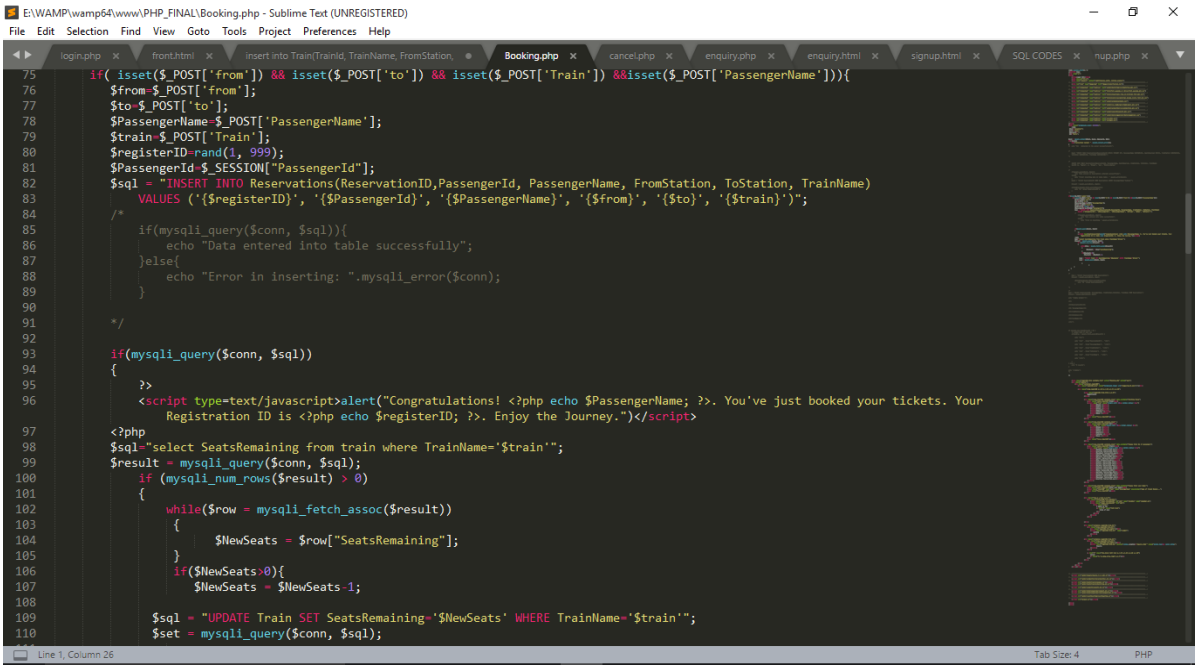

Figure 16: Back-end codes snapshots (Booking.php)

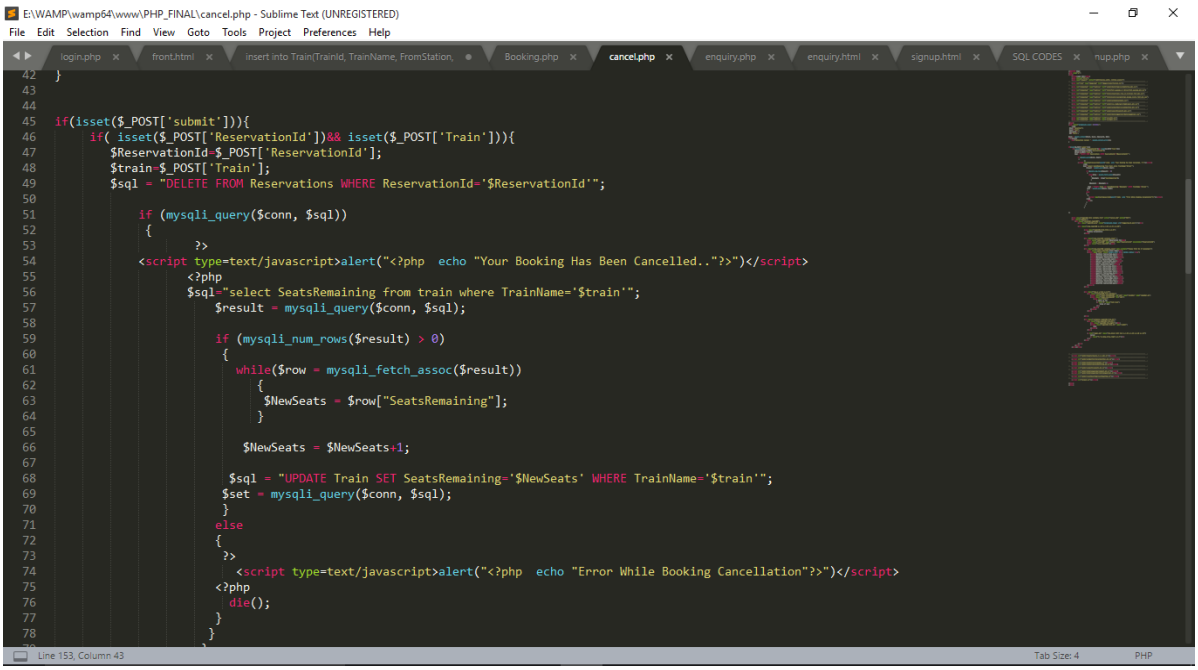

Figure 17: Back-end codes snapshots (cancel.php)

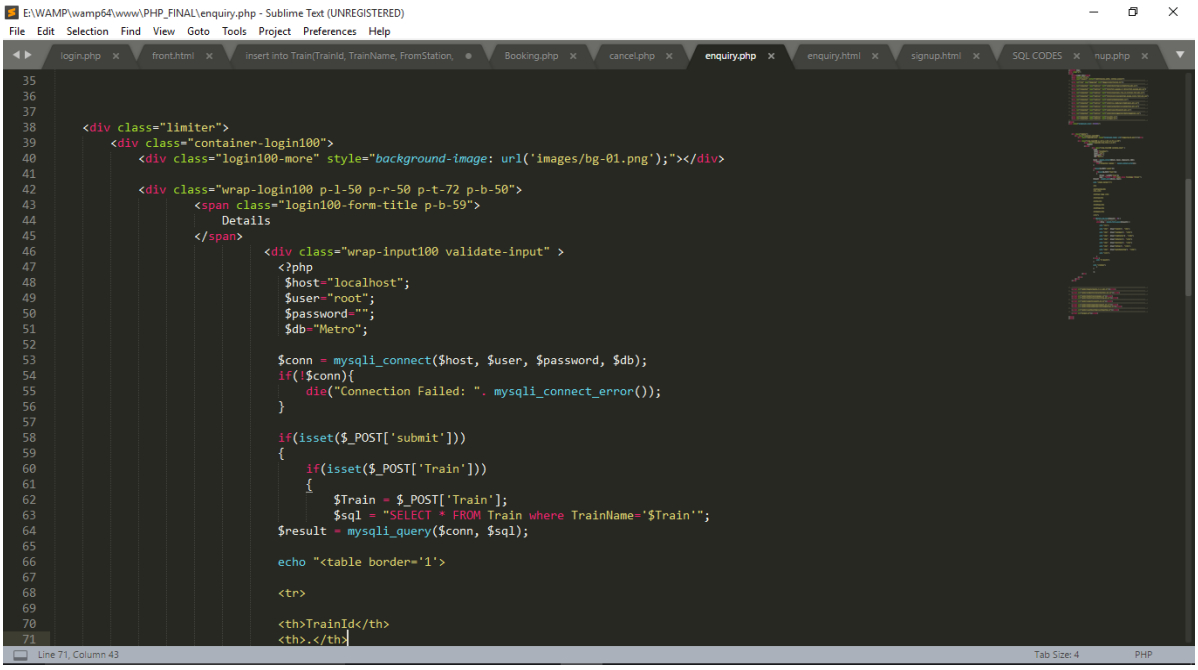

Figure 18: Back-end codes snapshots (enquiry.php)

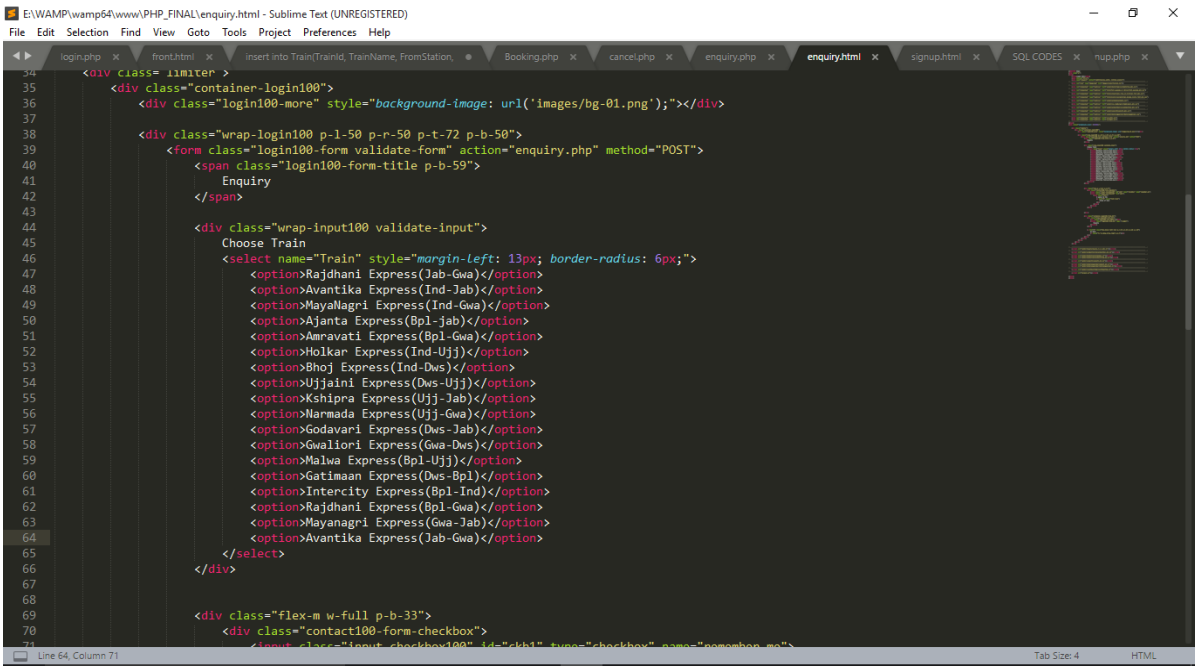

Figure 19: Back-end codes snapshots (enquiry.html)

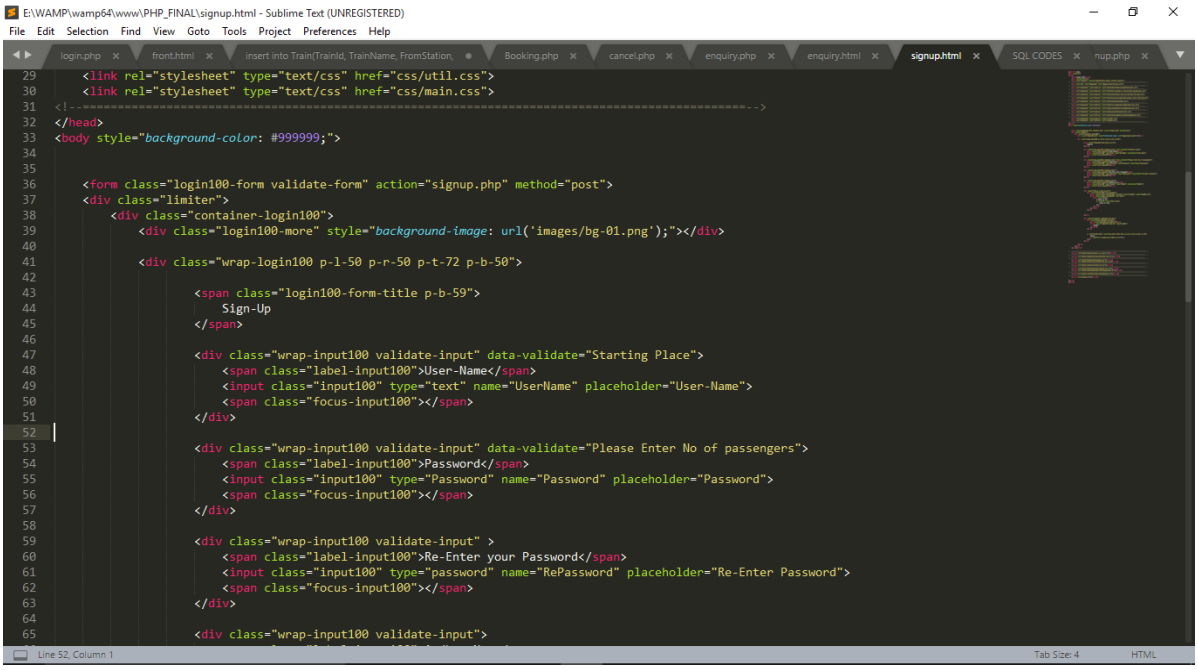

Figure 20: Back-end codes snapshots (signup.html)

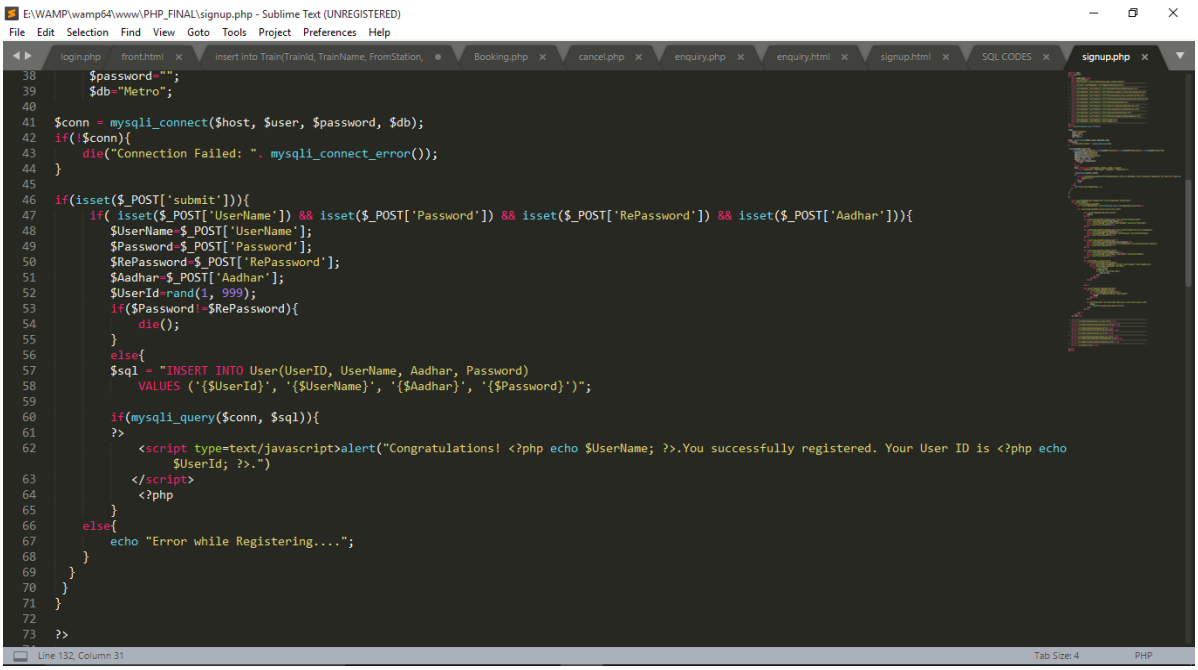

Figure 21: Back-end codes snapshots (signup.php)

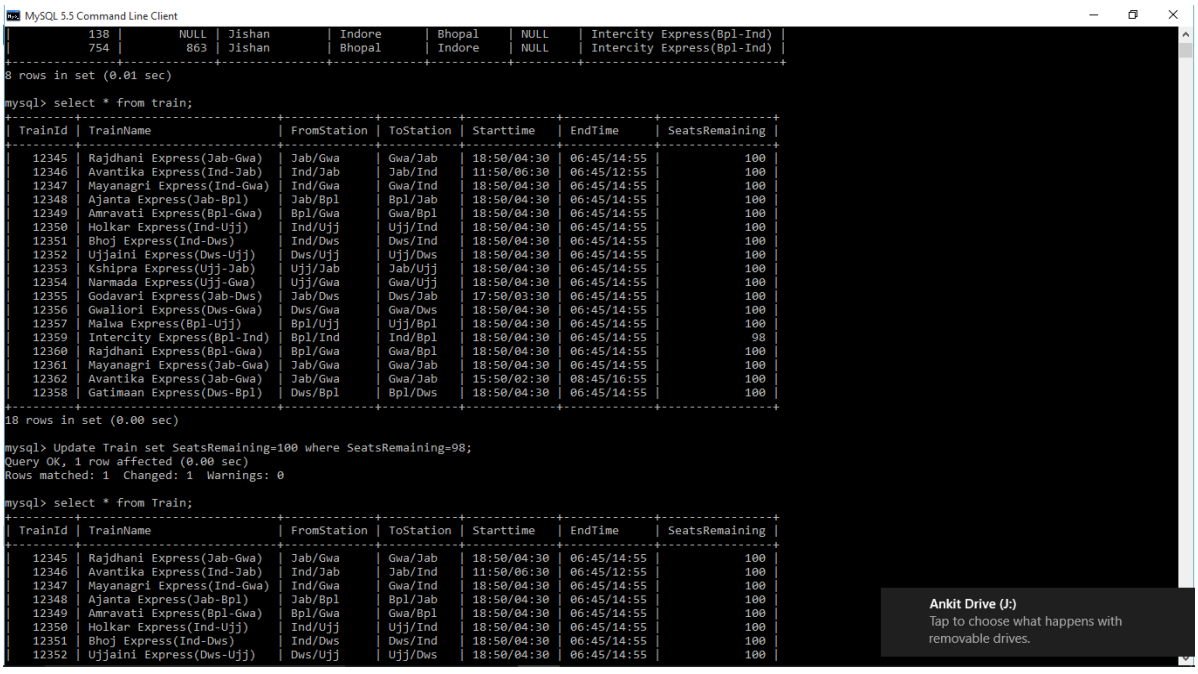

Figure 22: MySQL Command Line Client (Snapshot-1)

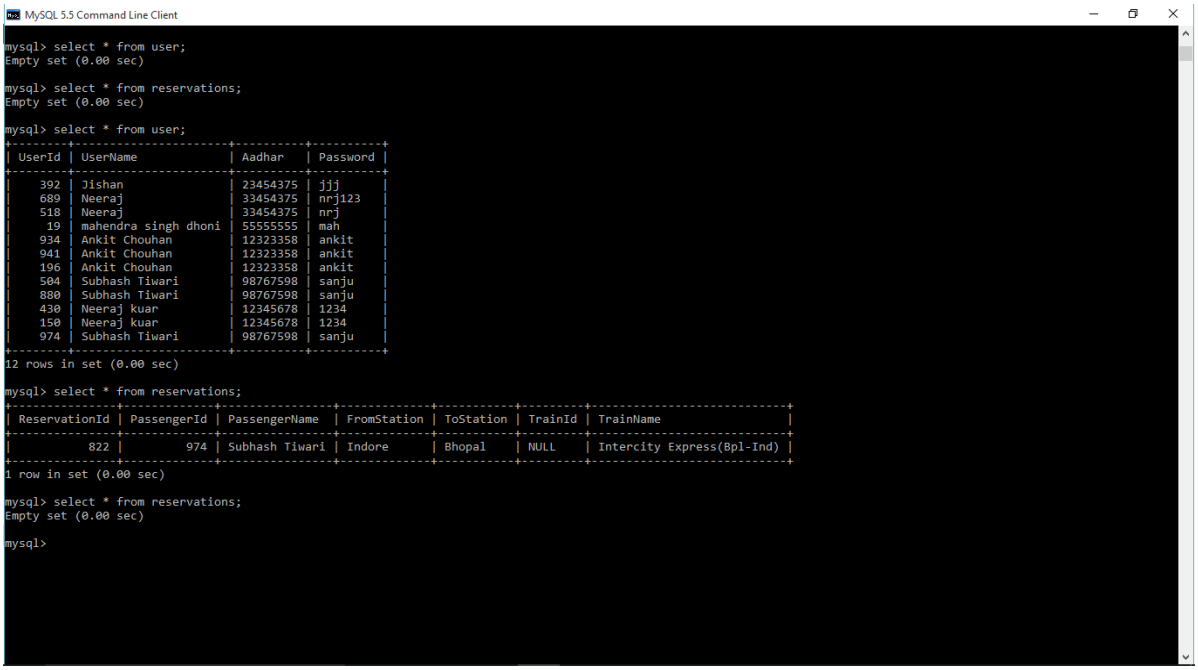

Figure 23: MySQL Command Line Client (Snapshot-2)

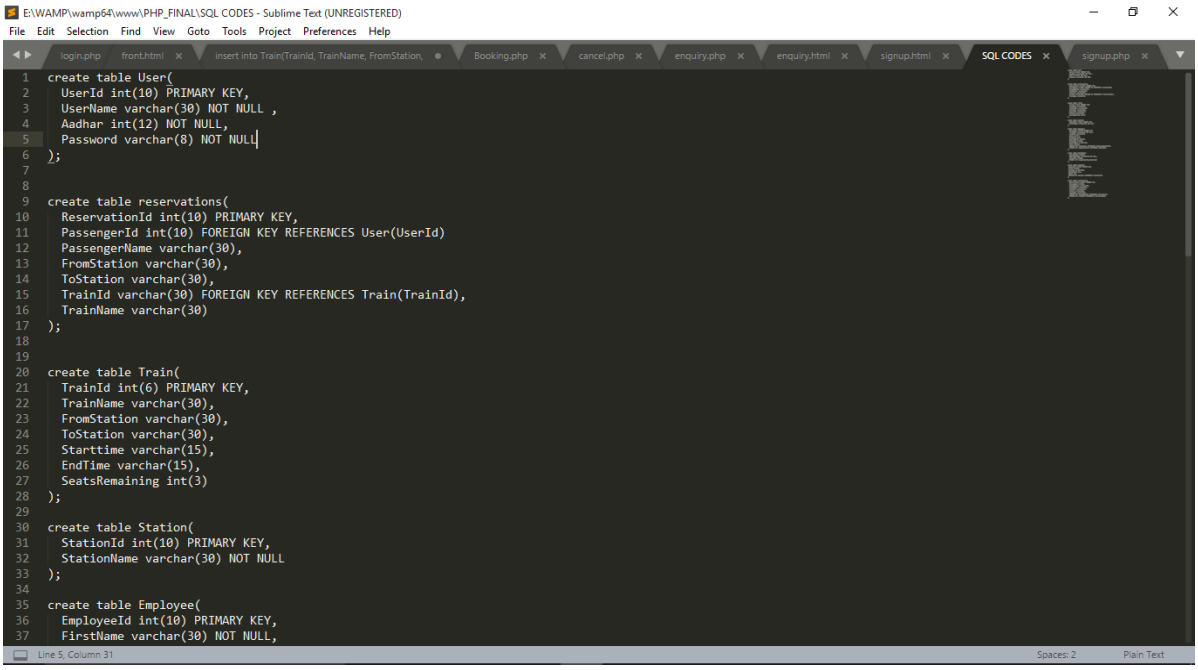

Figure 24: MySQL Command Line Client (Snapshot-3)

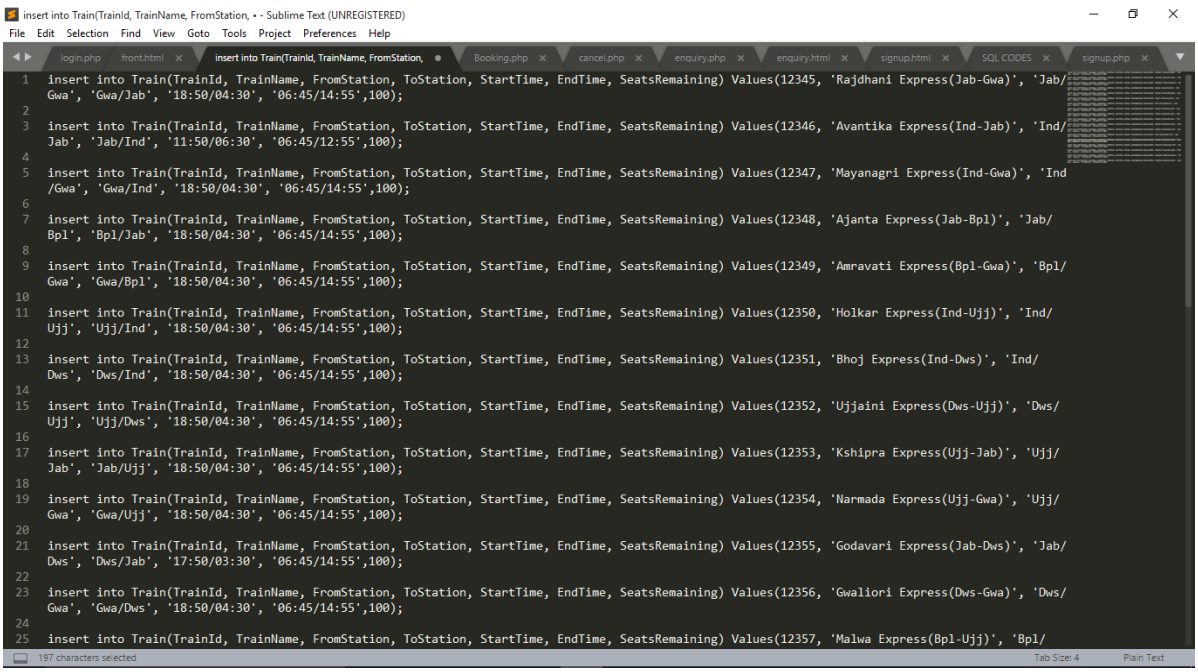

Figure 25: MySQL Command Line Client (Snapshot-4)