MigDB

Relational to NoSQL Mapper

Project ID: 16-015

Project SRS Document

B.Sc. Special (Honors) Degree in Information Technology

Submitted on 01/04/2016

IT13093938 Kothalawala K.R.M.N

Team Members

Student ID	Name	Signature
IT13001476	Liyanaarachchi L.A.G.C	
IT13088156	Madhusanka K.P.L.K	
IT13093938	Kothalawala K.R.M.N	
IT13075422	Padmasiri H.R.K.L	

Supervisor

……………………………………

Dr. Anuradha Karunasena

Co-Supervisor

……………………………………

<Name of the Co-supervisor>

Declaration

I declare that this is my own work and this Software Requirements Specification does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

……………………………………….

Kothalawala K.R.M.N

IT13093938

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

The purpose of the Software Requirements Specification is to provide a detailed description on the purpose and functionality of the project 'MigDB' which focus on enabling users to easily transfer from a MySQL database to MongoDB along with table structures, relationships, data and queries. The document presents user and system requirements while describing functional and non-functional requirements of the system. Being intended for both stakeholders and the developers of the system, the document helps stakeholders to get a clear understanding on the end product and aids developers in understanding the constraints, what needs to be developed and how it needs to be developed by breaking problems down into parts. SRS can also be used to verify and validate the system after development.

1.2 Scope

The focus of the system 'MigDB' is on migration from MySQL tables along with proper relationship mapping, data as well as queries into MongoDB without requiring any prior knowledge on MongoDB commands or JSON. The intended system is primarily divided into three sections as Relational Database Handling module, Relationship Mapping module and Database Management module and these three modules mainly consist of the following sub components.

- Connection Manager
- SQL DB Analyzer
- **SQL DB Modification Evaluator**
- One-to-one Mapping
- **One-to-many Mapping**
- Many-to-many Mapping
- Semantic Network for Collection Mapping
- MongoDB Collection Modifier UI
- MongoDB Modification Evaluator
- Conversion of Referencing to Embedding
- Conversion of Embedding to Referencing
- MongoDB Data Manipulator
- Graphical Query Generator
- **SQL Query Converter**

The document addresses the sub components named SQL DB Modification Evaluator, One-to-many Mapping and SQL Query Converter under the system.

The software system ‘MigDB’ is capable of transferring a MySQL database into NoSQL document type database MongoDB along with table structures, relationships, existing data and queries with the use of incremental machine and human knowledge without requiring any MongoDB command or JSON skills from the user. The system is vastly beneficial to the users who focus on the migration from MySQL to MongoDB since existing migration tools are incapable of mapping relationships when migrating and the necessity for JSON or MongoDB command knowledge in the existing management tools.

The SQL DB Modification Evaluator component under the Relational Database Handling module of the system assists users in selecting and unselecting of tables and columns that are to be migrated. The objective of the component is to graphically represent MySQL table structures to the user, facilitate users to make changes to the existing table structures and finally to evaluate and record these changes. The advantage of the component over existing tools is its ability to evaluate user changes for any violations of constrains like deleting a column referenced as a foreign key in order to avoid further mistakes.

The One-to-many mapping component under the Mapping module focuses on properly mapping one-to-many relationships among the migrating MySQL database. The plus point of the component is the ability to map all one-to-many relationships within the MySQL database into MongoDB according to the optimal mapping option out of embedding and referencing selected with the help of the neural network and the database context.

The SQL Query Converter in Management module aids users in converting MySQL queries into MongoDB commands. The component assesses the user inputted MySQL query for syntax errors and would support table-level create, alter and drop statements, record-level insert, update, delete statements and select statement with table joins unlike the existing query converted which are only capable of converting select queries excluding table joins and therefore provides further assistance to the users by making the query conversion process effortless and efficient.

1.3 Definitions, Acronyms and Abbreviations

JSON	JavaScript Object Notation
DB	Database
UI	User Interface
SQL	Structured Query Language
MS	Microsoft
JDK	Java Development Kit
IDE	Integrated Development Environment

1.5 Overview

The expected outcome of 'MigDB' is a system which facilitates users to easily migrate MySQL tables, relationships, and data along with queries into MongoDB without requiring any prior knowledge on JSON or MongoDB commands from the user. The system is intended for users who desire to migrate from MySQL to MongoDB and in need of a tool which is time-saving, effective and easy to use. The system is mainly divided into three modules and drives towards following specific objectives.

- Connection Management module to establish connection with MySQL database and MongoDB database in local machine or remote server while having the capability to hold multiple database connections.
- SQL DB Analyzer module to facilitate upload of an SQL file and to analyze dump file and understand table structures, relationships along with data and then to map these information into intermediate JSON file.
- **SQL DB Modification Evaluator module to graphically represent the MySQL database structure, allow user to make changes, evaluate changes for any violations and then record the user changes to the database structure.**
- Neural network as a separate module to predict about database mapping technique with considering references, column count and data types of columns. This module has capability to trigger scheduled Neural Network supervised learning sessions using a CSV file. This module has capability to retrieve information from Neural Network and send to other modules through a restful interface.
- One-to-one mapping module to map one-to-one relationships in MySQL database into MongoDB with data migration.
- **One-to-many mapping module to map one-to-many relationships in MySQL database into MongoDB by using embedding or referencing.**
- Many-to-many mapping module to map many-to-many relationships in MySQL database into MongoDB by using embedding or referencing.
- MongoDB Collection Modifier UI module to illustrate collection structure generated by the system.
- MongoDB Modification Evaluator module to enable user to change collection structure by applying user experience on database mapping in user friendly way and to evaluate user changes on mapping.
- Conversion of Referencing to Embedding module to convert existing reference mapping into embedded mapping and update database.

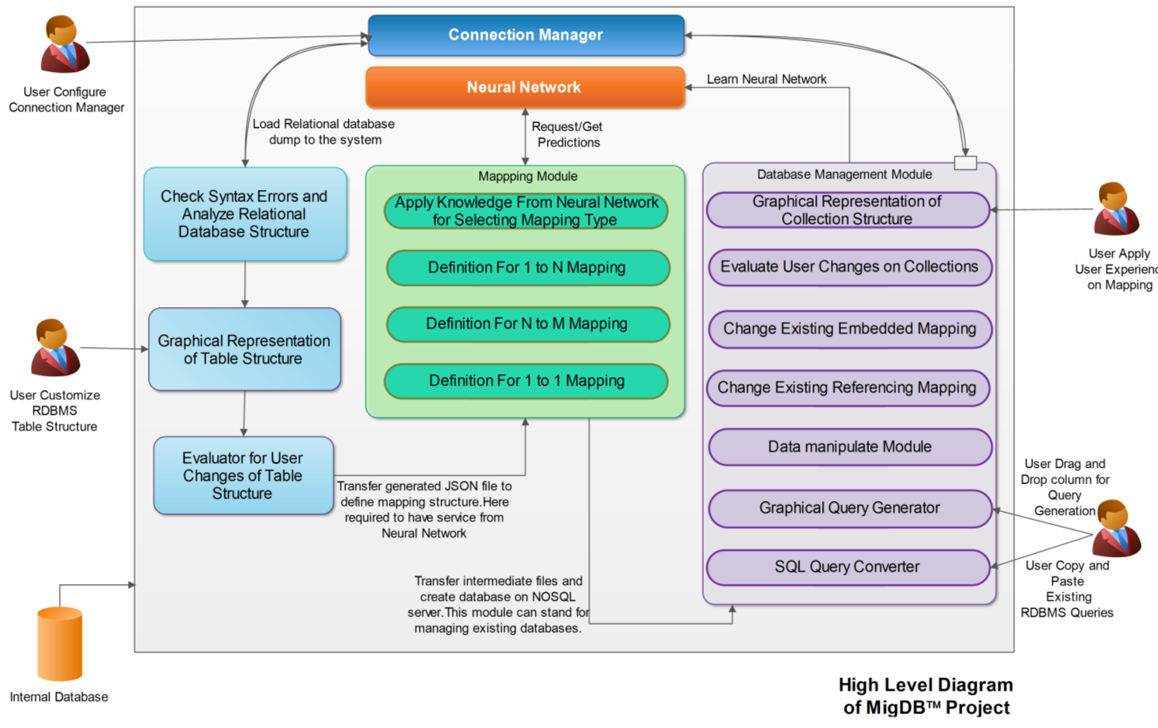
- Conversion of Embedding to Referencing module to convert existing embedded mapping into reference mapping.
- MongoDB Data Manipulator module to access NOSQL database on localhost and graphically represent data to the user and enable user to manipulate data and save changes.
- Graphical Query Generator module to generate MongoDB commands with the use of interactive graphical user interface.
- **SQL Query Converter module to convert given MySQL statements into MongoDB statements.**
- User friendly web portal to enable user for accessing above services with getting continuous user interaction. This web portal has capabilities to handle payments, file upload and download, user management and file version management.

The remaining of the SRS document elaborates overall descriptions, specific requirements and supporting information of the modules SQL DB Modification Evaluator, One-to-many Mapping and SQL Query Converter under the system.

The first section of the SRS document gave an introduction, describing the purpose of the document, scope indicating which aspects of the system is covered by the document and benefits and objectives of the system. The second section of the document provides overall descriptions of the modules SQL DB Modification Evaluator, One-to-many Mapping and SQL Query Converter under the system generally focusing on system, user, hardware, software, communication interfaces and constraints while also comparing and contrasting the system with existing products. It also describes product functions with the aid of diagrams, user characteristics and assumptions. The third section of the document is aimed at technical users and focuses on specific requirements explaining external interface requirements, performance requirements, design constraints and non-functional requirements in detail. Supporting information is presented in the forth section of the SRS document.

2 Overall Descriptions

'MigDB' is a standalone application which focus on its main goal of facilitating users to easily transfer from a MySQL database along with tables, relationships, data and queries into the NoSQL document type database MongoDB without necessitating prior knowledge on JSON or MongoDB commands. In order to accomplish the main objective, the intended system uses a step by step approach while interacting with the user periodically in the process.



Connection Manager
Neural Network
Learn Neural Network
User Configure Connection Manager
Load Relational database dump to the system
Request/Get Predictions
Check Syntax Errors and Analyze Relational Database Structure
Mapping Module
Apply Knowledge From Neural Network for Selecting Mapping Type
Definition For 1 to N Mapping
Definition For N to M Mapping
Definition For 1 to 1 Mapping
Database Management Module
Graphical Representation of Collection Structure
Evaluate User Changes on Collections
Change Existing Embedded Mapping
Change Existing Referencing Mapping
Data manipulate Module
Graphical Query Generator
SQL Query Converter
User Apply User Experience on Mapping
User Customize RDBMS Table Structure
Graphical Representation of Table Structure
Evaluator for User Changes of Table Structure
Transfer generated JSON file to define mapping structure.Here required to have service from Neural Network
Transfer intermediate files and create database on NOSQL server.This module can stand for managing existing databases.
User Drag and Drop column for Query Generation
User Copy and Paste Existing RDBMS Queries
Internal Database
High Level Diagram of MigDB™ Project

Figure 1 : System Diagram

Prior to starting the migration process, the system requires the MySQL database which needs to be migrated. The user has the capability of providing the MySQL database to be migrated either with the use of the connection manager by giving connection parameters and connecting local or remote MySQL database server or by uploading a MySQL database file. The SQL DB Analyzer module of the system then checks the uploaded MySQL database file for syntax errors and displays error messages to the user in case of errors. If there are no errors, the module creates an intermediate JSON file containing objects for table structures and data. The SQL DB Modification Evaluator module then uses the intermediate JSON file and represents the MySQL table structures graphically to the user and allows user to remove tables or columns that they find unnecessary to be migrated. The module validates these user changes with relevant constraints and if the changes violate any constraint it alerts user with an error message, otherwise the module updates the intermediate JSON file and the JSON file is then forwarded to the relationship mapping module where relationships are properly mapped and table and data objects are converted into document-like objects to be inserted into collections.

Initially, the one-to-one mapping module maps all table and data into document-like objects and stores them in a separate JSON file. If the tables in the database to be migrated contain one-to-one relationships, the module analyzes the table structures and embeds data in the most appropriate mechanism. Following the mapping of one-to-one relationships, the one-to-many relationship mapping module sends a request to the neural network in selecting the most optimal mapping option out of embedding and referencing to map one-to-many relationships within the MySQL database by passing the necessary parameters. Once the neural network response it compared with its own evaluations, the one-to-many mapping module decides which objects needs to be embedded or which objected needs to be referenced with the aid of its internal logics and then maps one-to-many relationships by using embedding or referencing appropriately. Proceeding the re-structuring of the second JSON file with one-to-many relationships, it is analyzed by the many-to-many mapping module. The many-to-many mapping module also uses neural network in selecting the most appropriate mapping option out of embedding and referencing and according to the neural network response, the module re-arranges the second JSON file which contains the document objects. Succeeding the completion of relationship mapping within the database to be migrated, the system creates collections in MongoDB with relevant documents.

The management module of 'MigDB' facilitates users in managing their existing MongoDB databases by representing MongoDB database structures graphically. The mapping conversion modules facilitates users to modify the way that relationship mapping is done by allowing users to convert embedded mapping into referencing and referenced mapping into embedding while the data manipulation module allows users to modify data within documents.

The Graphical Query Generator module of the system assist users who lacks familiarity of MongoDB commands or JSON to generate MongoDB commands graphically by only selecting items and typing values and tracking those actions in order to generate the commands. The SQL Query Converter module enables users to convert existing MySQL queries into MongoDB commands easily and instantly by identifying MySQL key words and mapping them into corresponding MongoDB commands.

2.1 Product Perspective

With NoSQL emerging as a much more flexible alternative to relational databases and MongoDB ranking high among non-relational databases, many researches have been conducted on the feasibility and issues of migrating a relational database into MongoDB and a few systems have been developed focusing on MongoDB and migration to MongoDB. These available products presents some sound features but still have some limitations in certain areas.

Pelica Migrator is one of the most known tools for the migration of an existing relational database into MongoDB. It is capable of migrating tables and data from any RDBMS (Relational Database Management System) including MySQL, Oracle, SQL Server 2008, PostgreSQL, DB2 and Sybase to MongoDB. Pelica Migrator also allows users to select/ unselect tables or columns which they wish to be migrated. The issue with the tools is that while it keenly migrates both tables and data into MongoDB, it does not consider table relationships when migrating. Moreover, the tool does not validate columns or tables user unselects when migrating which enables users to remove columns or tables which are referred by some other tables or columns. The NoSQL Manager for MongoDB tool which provides some sound features also facilitates users to migrate from MySQL or MS SQL Server database into MongoDB but it also does not take table relationships into consideration. Unlike in these tools, the intended system 'MigDB' is capable of migrating table relationships in the most appropriate option according to the application context and it also validates user changes to the MySQL database structure to ensure that user changes does not violate any constraints.

RoboMongo, UMongo, Admin mongo are management tools for MongoDB and while it these tools facilitate users in graphical database manipulation, the tools still requires some knowledge on JSON or MongoDB commands from the user whereas the system 'MigDB' does not require any prior knowledge on JSON or MongoDB from the user.

The existing MySQL to MongoDB query conversion tools Query Mongo and Klaus.dk are capable of converting MySQL select queries into MongoDB while Query Mongo is also capable of converting aggregate functions but the tools are still capable of converting only select queries and does not support table joins. The query converter of the system 'MigDB' facilitates users to convert table-level create, alter and drop statements, record-level insert, update, delete statements along with join statements into MongoDB commands.

2.1.1 System Interfaces

- One-to-many mapping module would need to access restful web service in order to get the neural network response on mapping and would also need connection to MongoDB server
- SQL Query Converter module would require access to internal SQLite database in order to get corresponding MongoDB keywords

2.1.2 User Interfaces

The application 'MigDB' uses a stage by stage methodology in achieving its main objective with continuous cooperation of the user in the process of migration and management. Following are some basic user interface sketches associated with the components SQL DB Modification Evaluator and SQL Query Converter.

- SQL DB Modification Evaluator

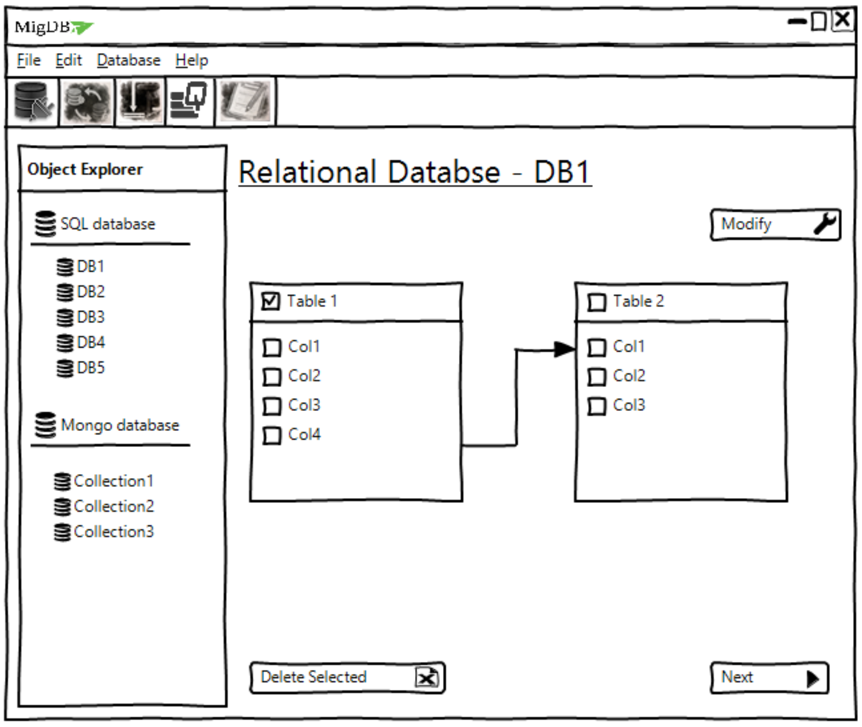


Figure 2 : User Interface Example 1

- One-to-many Mapping

  The one-to-many mapping module is not associated with a user interface as it is an intermediary module internal to the system.

- SQL Query Converter

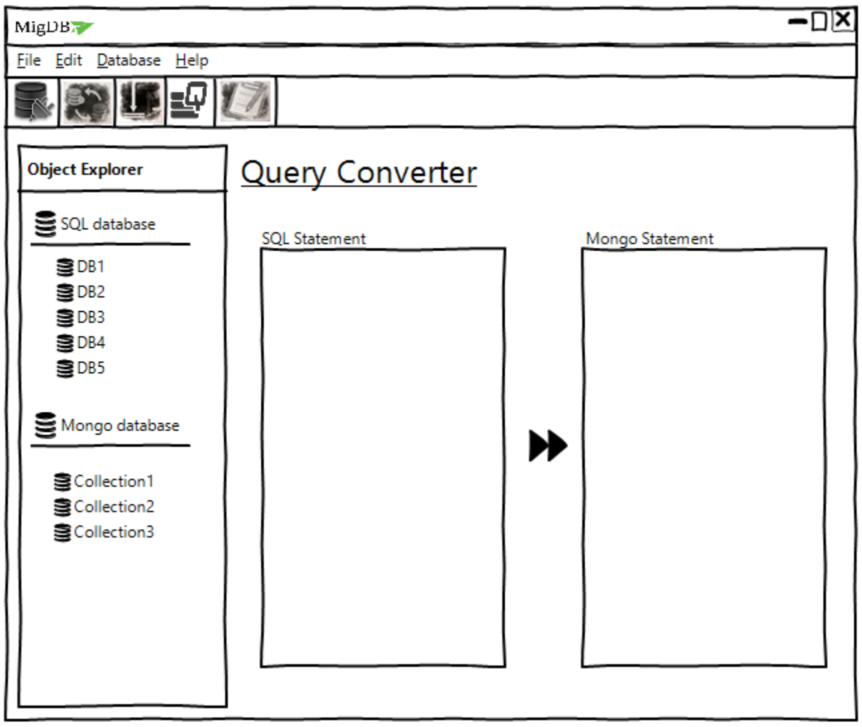


Figure 3 : User Interface Example 2

2.1.3 Hardware Interfaces

- Wifi router or dongle
- Service providing server
- Laptop/Desktop computers which are configured with OS such as Linux, Mac OS X or Windows

2.1.4 Software Interfaces

- Java JDK 8 is need to be configured in the machine
- MongoDB is need to installed on the machine an Mongo server need to be run to startup the machine
- Application is developed using Eclipse Mars.2 (4.5.2) and Scene Builder 8.0 is used for UI designing
- SQLite is used as an internal database to store MongoDB statements and other configuration files

2.1.5 Communication Interfaces

A modem or a router will be required to download the system as ‘MigDB’ is downloaded via the web. And internet connectivity is also used to access neural network services as it is hosted on the server.

- Dialup or Broadband connection with an internet provider
- Web hosting server

2.1.6 Memory Constraints

- Free disk space close to the size of the MySQL database to be migrated

2.1.7 Operations

- User downloads the ‘MigDB’ setup via the web application provided
- Install the application
- The first time user needs to provide username, hostname, port and password for MySQL connection on local or remote server and needs to provide hostname and port for Mongo server connection
- User can select a MySQL database from the lists of synchronized databases or can upload a MySQL database file for the migration to start
- In case of the file upload, it will be checked for syntax errors and will display it to the user if error are found
- System analyzes MySQL database and creates the intermediate JOSN file containing table structures and data
- System represents the MySQL database structure graphically
- User can remove unnecessary columns or tables that are to be migrated
- System evaluates user changes for violations of constraints and displays error messages if there are any violations
- System updates the intermediate JSON file according to the user changes
- The system initially analyzes table and data structures and creates a JSON file with document-like objects
- System maps one-to-one relationships via the use of embedding appropriately and updates the JSON file with document objects
- System maps one-to-many relationships appropriately with the use of neural network response and internal logics and updates the JSON file with document objects
- System maps many-to-many relationships appropriately with the use of neural network response and updates the JSON file with document objects
- System creates the database in MongoDB and inserts necessary collections and documents

- User can change existing mapping and system evaluates the mapping and updates database if there are no violations
- User can modify existing databases and system records the changes
- User can select items and type values in order to generate queries and system tracks those actions, get corresponding MongoDB commands from the SQLite database and generates the MongoDB commands
- User can convert existing MySQL queries into MongoDB by providing it as an input and system identifies the MySQL keywords and converts it to the corresponding MongoDB command.

2.1.8 Site Adaptation Requirements

- The system is available for download in the web. Therefore user needs to access the web via a web browser.
- User needs to provide Mongo server connection for the database to be migrated into MongoDB.
- User interfaces are available in English.

2.2 Product Functions

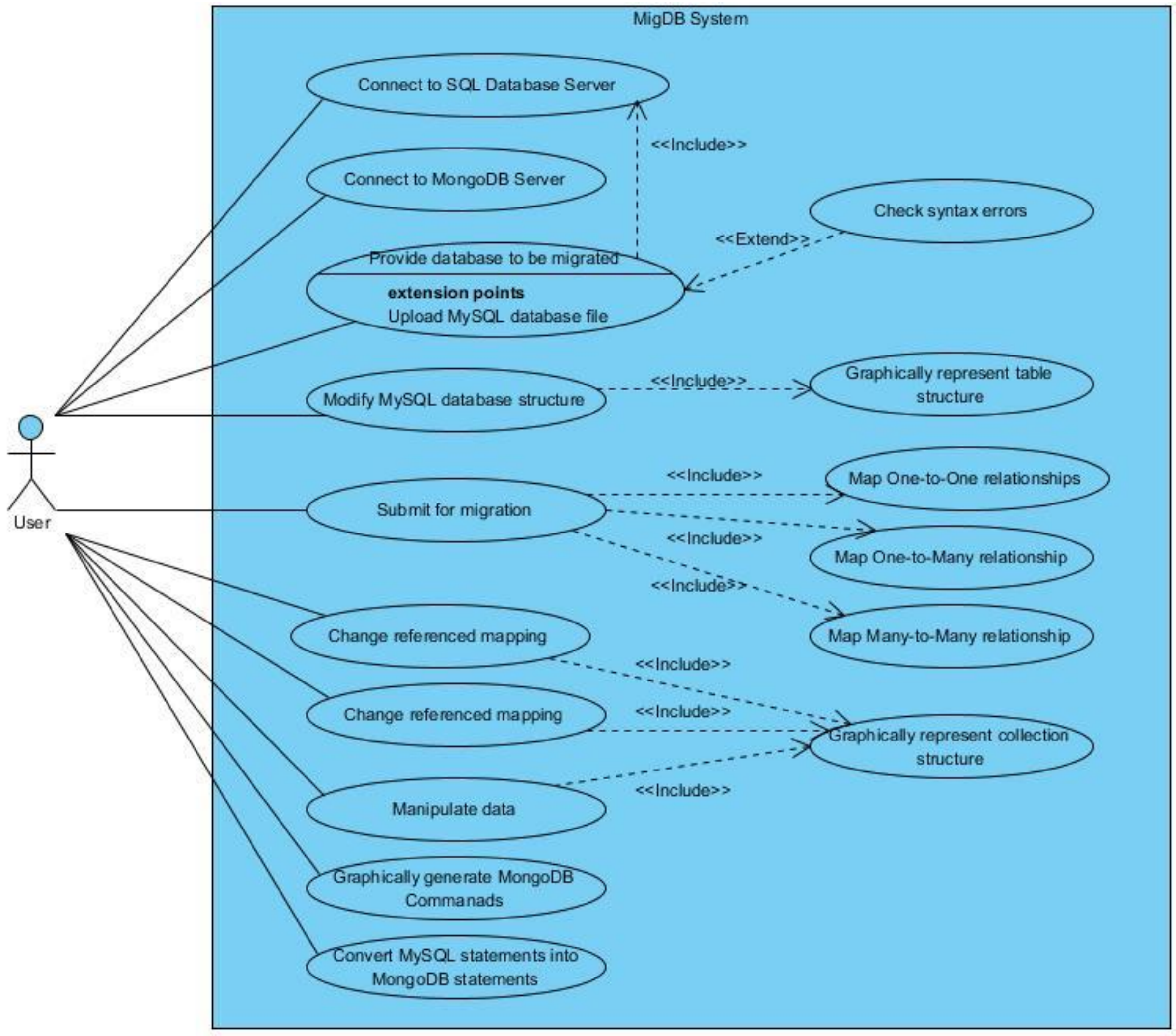

MigDB System
Connect to SQL Database Server
<<Include>>
Connect to MongoDB Server
Check syntax errors
<<Extend>>
Provide database to be migrated
extension points
Upload MySQL database file
Modify MySQL database structure
<<Include>>
Graphically represent table structure
User
Submit for migration
<<Include>>
Map One-to-One relationships
<<Include>>
Map One-to-Many relationship
<<Include>>
Map Many-to-Many relationship
Change referenced mapping
<<Include>>
Change referenced mapping
<<Include>>
Graphically represent collection structure
<<Include>>
Manipulate data
Graphically generate MongoDB Commanads
Convert MySQL statements into MongoDB statements

Figure 4 : Use Case Diagram

Use Case	Modify MySQL database structure
Description	Use case describes how the system allows users to select/ unselect tables and columns to be migrated
Pre-Conditions	User has provided a MySQL database to be migrated and intermediate JSON file which contains table structures and data has been created
Post-Conditions	User modifications are evaluated and recorded
Primary Actor	User
Main Success Scenario	1. Include : : (Graphically represent table structures) 2. User clicks on 'Modify' button 3. System displays checkboxes for user to tick 4. User ticks the checkbox with the column/table he/she wants to remove 5. User clicks on 'Delete Selected' button

	6. System validates user change 7. Use case ends with updating the JSON file
Extensions	6a. If user change violates any constraint, the system displays an error message

Table 1 : Use Case Scenario for Modify database structure

Use Case	Graphically represent table structures
Description	Use case describes how the system graphically represents MySQL database table structures
Pre-Conditions	User has provided a MySQL database to be migrated and intermediate JSON file which contains table structures and data has been created
Post-Conditions	Graphical display of table structures
Primary Actor	User
Main Success Scenario	1. Use case begins after the creation of intermediate JSON file 2. The system analyzes the JSON file table structures 3. System graphically displays the table structures to the user

Table 2 : Use Case Scenario for Graphically represent table structures

Use Case	Map one-to-many relationships
Description	Use case describes how the system maps one-to-many relationships within the MySQL database to be migrated into MongoDB
Pre-Conditions	System has mapped one-to-one relationships and the JSON file has been created with document objects There is proper internet connection to get neural network response
Post-Conditions	One-to-many relationships within the database has been properly mapped and the JSON file has been updated accordingly
Primary Actor	User
Main Success Scenario	1. The use case begins with the creation of the second JSON file with document objects and completion of one-to-one relationship mapping 2. The system analyzes both JSON files and identifies table objects with one-to-many relationships 3. The system sends a request to the neural network to get the most appropriate mapping option 4. The system analyzes the objects and decides which objects needs to embedded or referenced 5. System re-arranges the JSON file with document objects according to the mapped one-to-many relationships

Table 3 : Use Case Scenario for Map one-to-many relationships

Use Case	Convert MySQL statements into MongoDB statements
Description	Use case describes how the user can input a MySQL statement to the system and convert it into a MongoDB statement instantly
Pre-Conditions	System is up and running
Post-Conditions	The corresponding MongoDB statement is generated
Primary Actor	User
Main Success Scenario	1. Use case begins with the user entering the MySQL statement to be converted 2. The system analyzes the statement and calls the corresponding conversion method 3. Use case ends with the system outputting the converted MongoDB statement
Extensions	2a. If the user inputted MySQL statement contains any syntax errors, the system displays an error message to the user

Table 4 : Use Case Scenario for Convert MySQL statements into MongoDB statements

2.3 User Characteristics

'MigDB' is mainly intended for users who could range from developers to database administrators that desire to migrate from MySQL to MongoDB and in need of a tool which is time-saving, effective and easy to use in order to transfer from MySQL to MongoDB. The users could even be novices in MongoDB as the system does not require expertise in JSON or MongoDB commands.

2.4 Constraints

- The system would be available only in English language
- User should have a proper internet connection
- MongoDB should be installed in the user's computer
- Implementation would be done using Java 8

2.5 Assumptions and Dependencies

It is assumed that the user has the ability to understand simple words in English as all interfaces and messages are displayed in English language. Users should have knowledge on databases in general, should know the difference between relational database MySQL and NoSQL document type database MongoDB and should have proper understanding about the table relationships.

2.6 Apportioning of Requirements

The first section of the document gives a basic introduction to the intended system whereas the second section provides a general description of the system in non-technical reader perspective. The third section while being intended for technical readers describes the system in detail focusing on functional requirements, non-functional requirements and design constraints.

Since the system 'MigDB' is primarily divided into three modules such as Relational Database Handling module, Mapping module and Management module, the implementation also follows the same structure. Initially, since the main objective of the system is the migration of MySQL database into MongoDb, Relational Database Handling module and Mapping module are developed first. Since the development of Module completes the migration process, then the Management module is developed. While developing, the quality attributes required by the system are also taken into consideration.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

- SQL DB Modification Evaluator

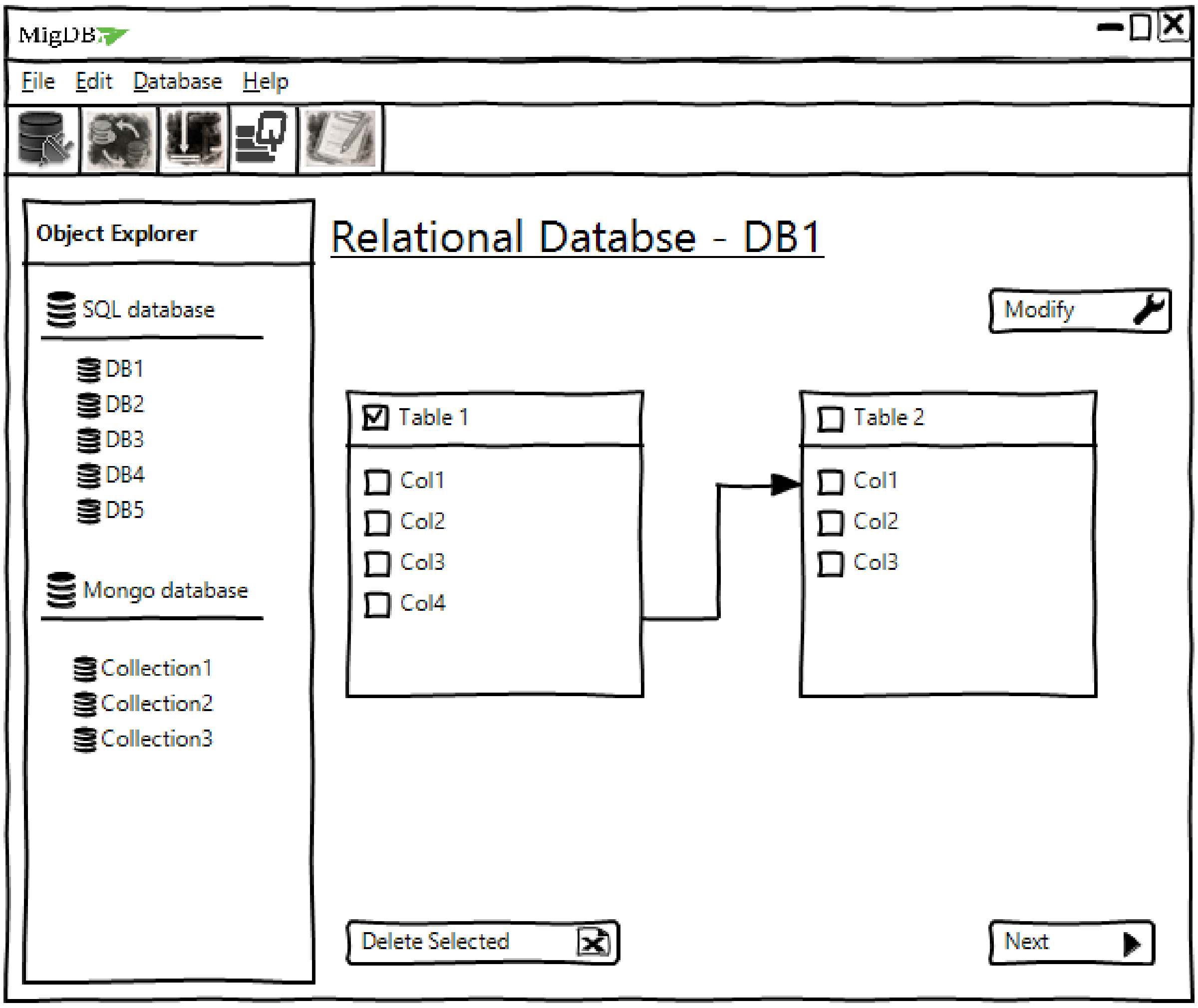


Figure 5 : User Interface Sketch 1 for SQL DB Modification Evaluator

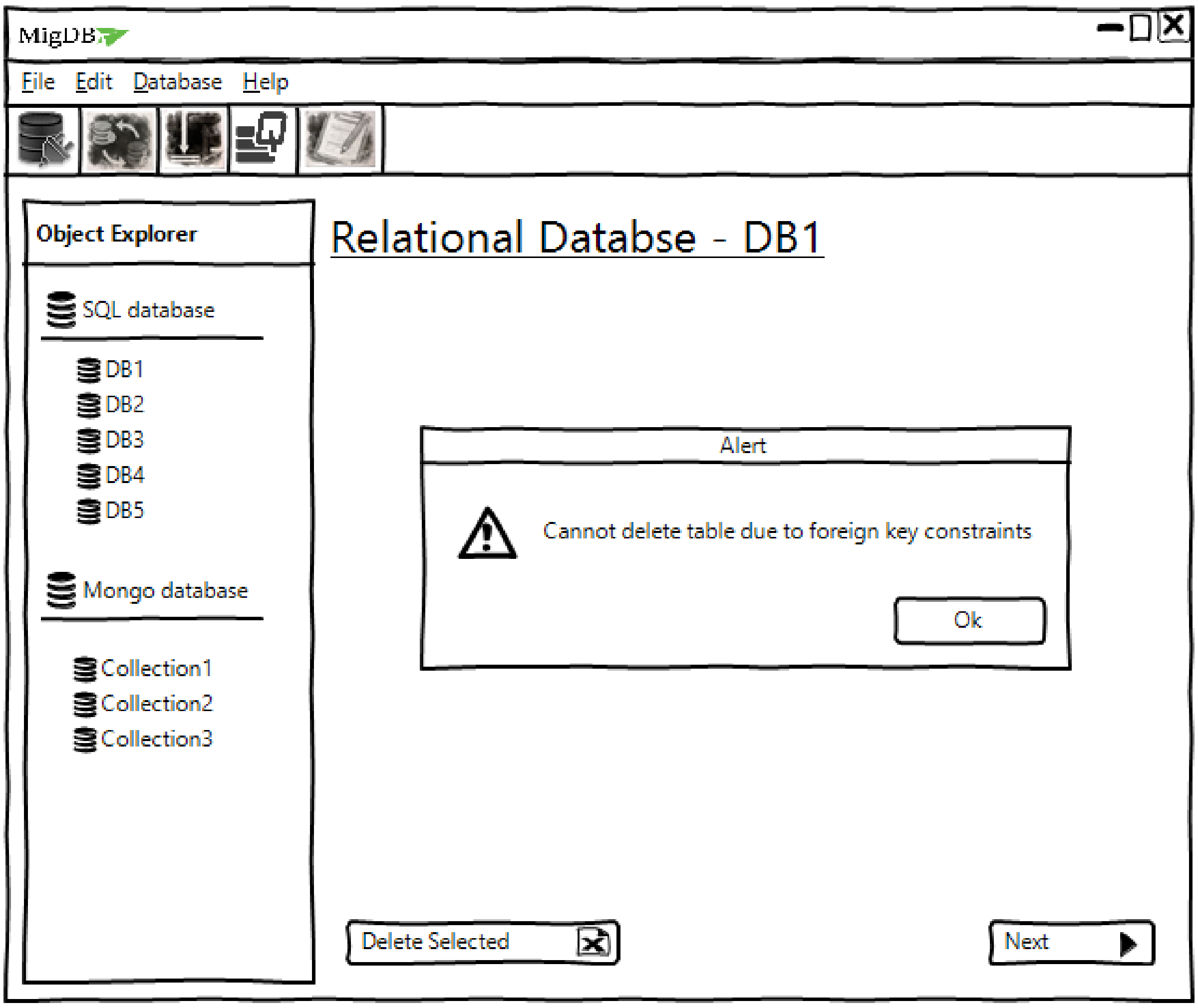


Figure 6 : User Interface Sketch 2 for SQL DB Modification Evaluator

The two figures above represent the user interface of the SQL DB Modification Evaluator module. Initially, the module analyzes the intermediate JSON file containing table objects which is outputted from the SQL DB Analyzer module and represents the table structures graphically. Once the user clicks on the 'Modify' button, the module enables users to select tables or columns to be removed as illustrated in figure 5. If the user removal of columns or tables results in any violations of constraints, the module outputs an error message to the user as shown in figure 6. The interface facilitates user to undo the changes as well. Following the validation of user changes, the module re-arranges the JSON file according to the user changes.

- One-to-many Mapping

  One-to-many mapping module does not contain a user interface as it is an internal processing module. It takes JSON file containing table and data structures and JSON file containing document objects outputted by the one-to-one mapping module as inputs and analyzes them. The module sends a request to the neural network passing the necessary parameters in order to get the most optimal mapping option out of embedding and referencing and then decides which objects needs to be embedded into which objects or which objects need to referenced by which objects with the aid of its internal logics. The module then re-structures the document objects accordingly mapping all one-to-many relationships appropriately.

- SQL Query Converter

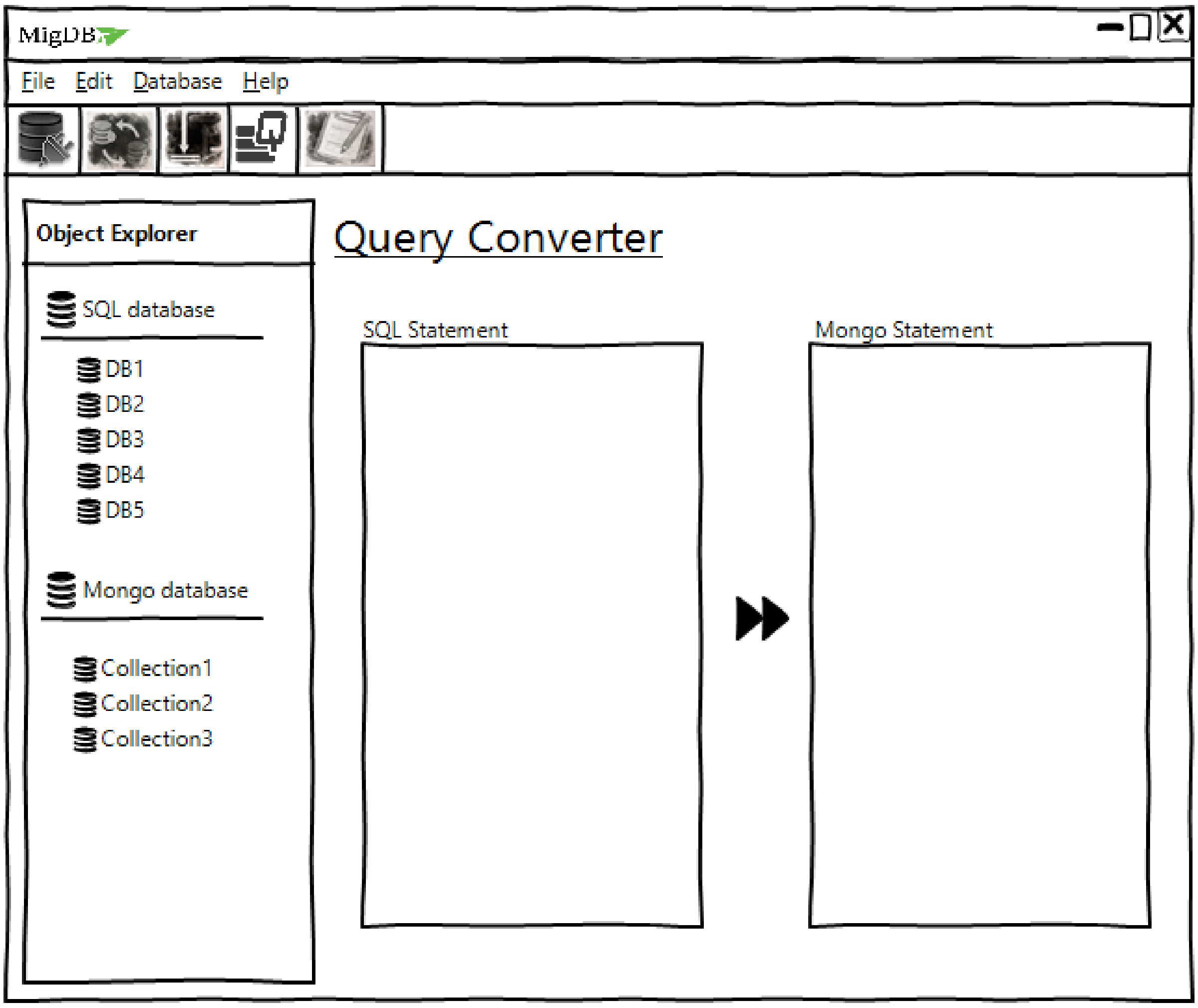


Figure 7 : User Interface Sketch for SQL Query Converter

The above figure demonstrates the user interface of the SQL Query Converter module. The module is responsible for converting a given MySQL statement into a MongoDB statement. The user can provide the existing MySQL statement in the provided space and click on the convert button in order to start the conversion. To start with, the module checks the user inputted MySQL statement for any syntax errors. In case of a syntax error, the system alerts the user of it. If there is no syntax errors found, the system analyzes the MySQL statement, identifies keywords and calls relevant methods within the module. The module then accesses the SQLite database containing MongoDB syntaxes, maps the MySQL keywords with the MongoDB keywords and outputs the converted MongoDB statement to the user.

3.1.2 Hardware Interfaces

- Wifi router or dongle
  A router or a dongle is needed since internet connectivity is required for downloading of the software and the mapping module to send request to the neural network web service
- Service providing server
  A server is required for hosting of the web application as well as for the neural network web service
- Laptop/Desktop computer
  A computer with a good processor and necessary disk space

3.1.3 Software Interfaces

The Eclipse Mars.2 (4.5.2) is used as the IDE. It's a full java IDE with advanced feature which enables to test, debug and interact with repositories. JavaFX is used as the software platform for creating desktop applications. Scene Builder as a tool that enables to quickly design rich JavaFX applications and it can be configured with Eclipse IDE.

- Latest Java JDK 8
- Eclipse 4.4 or greater with e(fx)clipse plugin
- Scene Builder 8.0

3.1.4 Communication Interfaces

A modem or a router will be required to download the system as 'MigDB' is downloaded via the web. And internet connectivity is also used to access neural network services as it is hosted on the server.

- Dialup or Broadband connection with an internet provider
- Web hosting server

3.2 Classes/ Objects

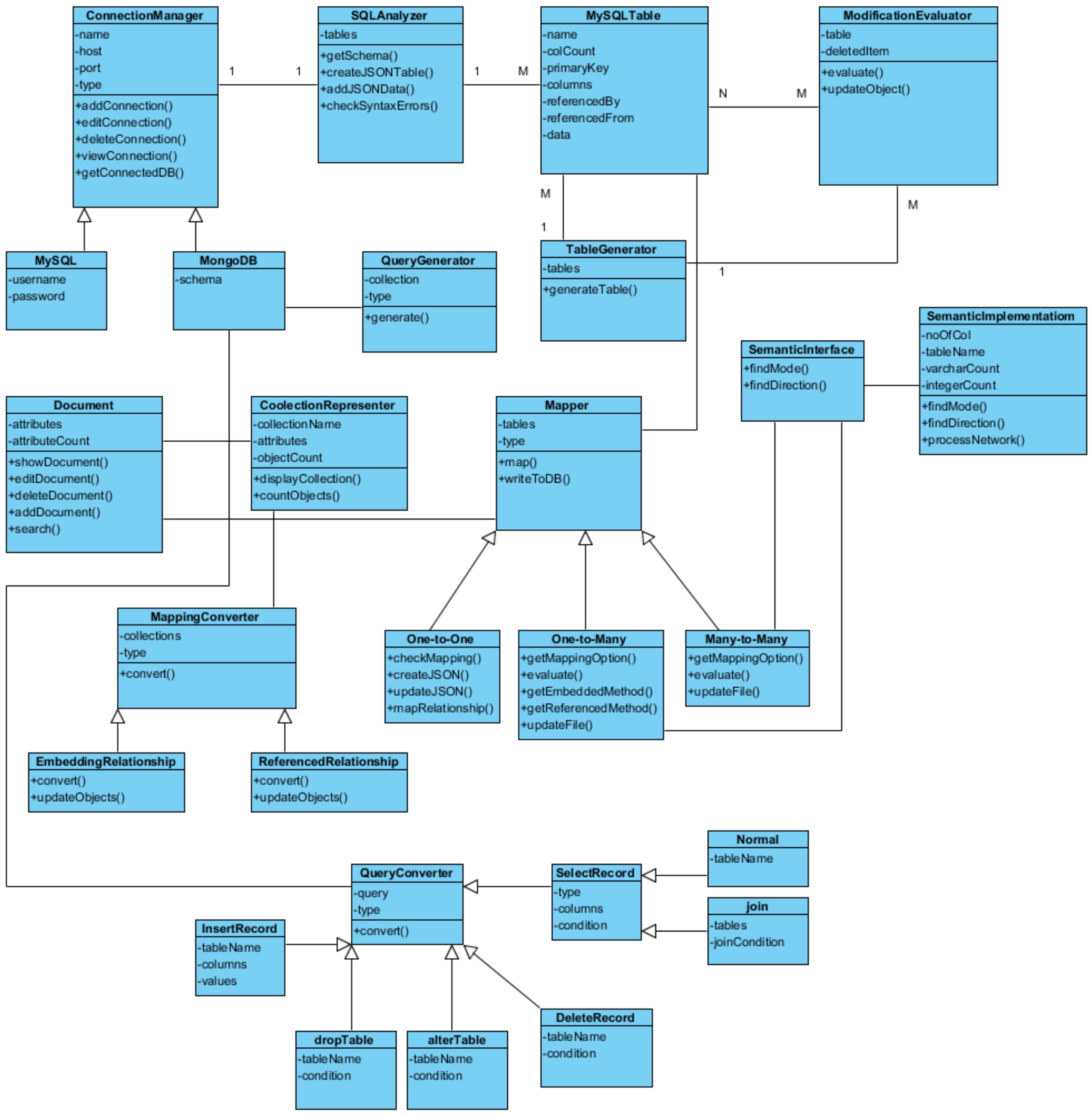


Figure 8 : Class Diagram

3.3 Performance Requirements

The client application is capable of handling all the operations that can be performed without freezing under required hardware and software requirements.

The cloud server can handle up to 1000 simultaneous users without freezing.

3.4 Design Constraints

Java should be used as the implementation language.

SQLite is used to store configurations, constraints and MongoDB syntaxes since it should be internal to the system.

The user interfaces should be simple and understandable to the user.

3.5 Software System Attributes

3.5.1 Reliability

Reliability focuses on how often the software system fails or in other words the capability of a system to provide expected functionality for a specific period of time [2]. The system 'MigDB' should support reliability by ensuring that the system does not fail more than twice a week. In order to safeguard reliability of the system, the end product should be fully tested in the working environment and appropriate software and hardware should be used to minimize failures.

3.5.2 Availability

Availability describes the ratio of time that a system is functional [2]. The system 'MigDB' aims at achieving high availability around 99% to ensure user satisfaction. Since third party commination interfaces and hosting services are used, it is difficult to maintain high availability but system should reduce the downtime which could occur due to poor developing or maintenance.

3.5.3 Security

"Security is the capability of a system to prevent malicious or accidental actions outside of the designed usage, and to prevent disclosure or loss of information" [2]. Security is to be achieved by the system with the use of secure payment gateways, encryption of user data and hiding of user database server information to the outsiders.

3.5.4 Maintainability

Maintainability represents the capability of the system to undergo modifications with ease [2]. 'MigDB' would support maintainability by the use of web services, following proper coding standards, separation of related modules in order to minimize dependencies and also by encouraging user for periodic software updates.

4 Supporting Information

4.1 Appendices

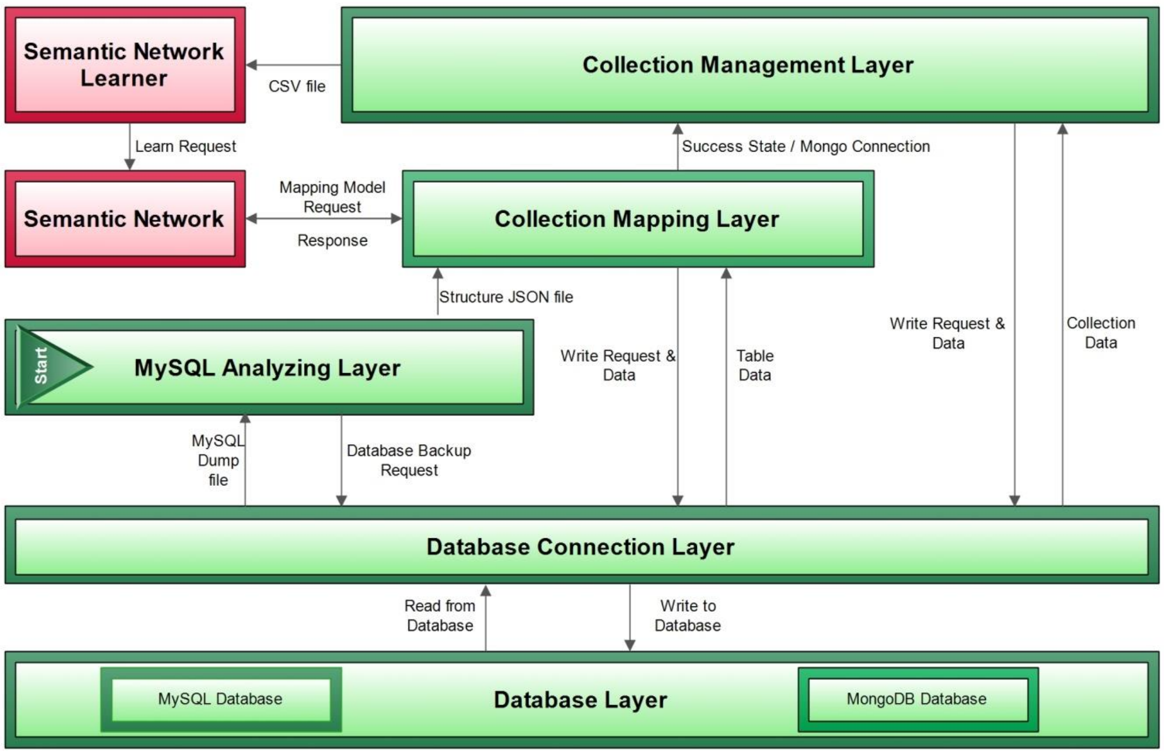


Figure 9 : System Architecture Diagram

References

[1] Kim Chan, “Writing a Software Requirements Specification Document”, *onedesk.com*, February 27, 2014. [Online]. Available : http://www.onedesk.com/writing-a-software-requirements-specification-document/ [Accessed: March 26, 2016].

[2]Microsoft Developer Network, “Microsoft Application Architecture Guide, 2nd Edition : Design Fundamentals : Chapter 16 – Quality Attributes”, *Microsoft Developer Network*, October 2009. [Online]. Available: https://msdn.microsoft.com/en-us/library/ee658094.aspx [Accessed: March 31, 2016].