Design Document

for

Admission Help System

Version 1.0 approved

Prepared by

Team 12

20th Feb. 2017

Table of Contents

1. Introduction

1.1 Purpose

This software design specification is made with the purpose of outlining the software architecture and design of the Student Registration System in detail. The document will provide developers an insight in meeting client's needs efficiently and effectively. Moreover the document facilitates communication and understanding of the system by providing several views of the system design.

1.2 Scope

The Software design document would demonstrate how the design will accomplish the functional and non- functional requirements captured in the Software Requirement specification (SRS). The document will provide a framework to the programmers through describing the high level components and architecture, sub-systems, interfaces, database design and algorithm design. This is achieved through the use of architectural patterns, design patterns, sequence diagrams, class diagrams, relational models and user interfaces.

1.3 Overview

The next chapter of the document has described architectural design of the Student Registration System. The high level components and their interactions, suitable architectural patterns, physical arrangement of components and design decisions applied to the whole system. The 3 rd and final chapter of the System Design Specification is on Component and detailed design. Includes design patterns, sequence diagrams, database design in detail and user interface design with screen shots of the interfaces.

1.4 Reference Material

- IEEE

1.5 Definitions and Acronyms

Algorithm Design: Specific method to create a mathematical process in solving problems.
Architectural Design: Establishing the overall structure of software system.
Database: A collection of stored related data.
Sequence Diagram: An interaction diagram that shows how processes interact with one another and in what order.
SRS: Software Requirements Specification

2. System Architecture

2.1 Architectural Design

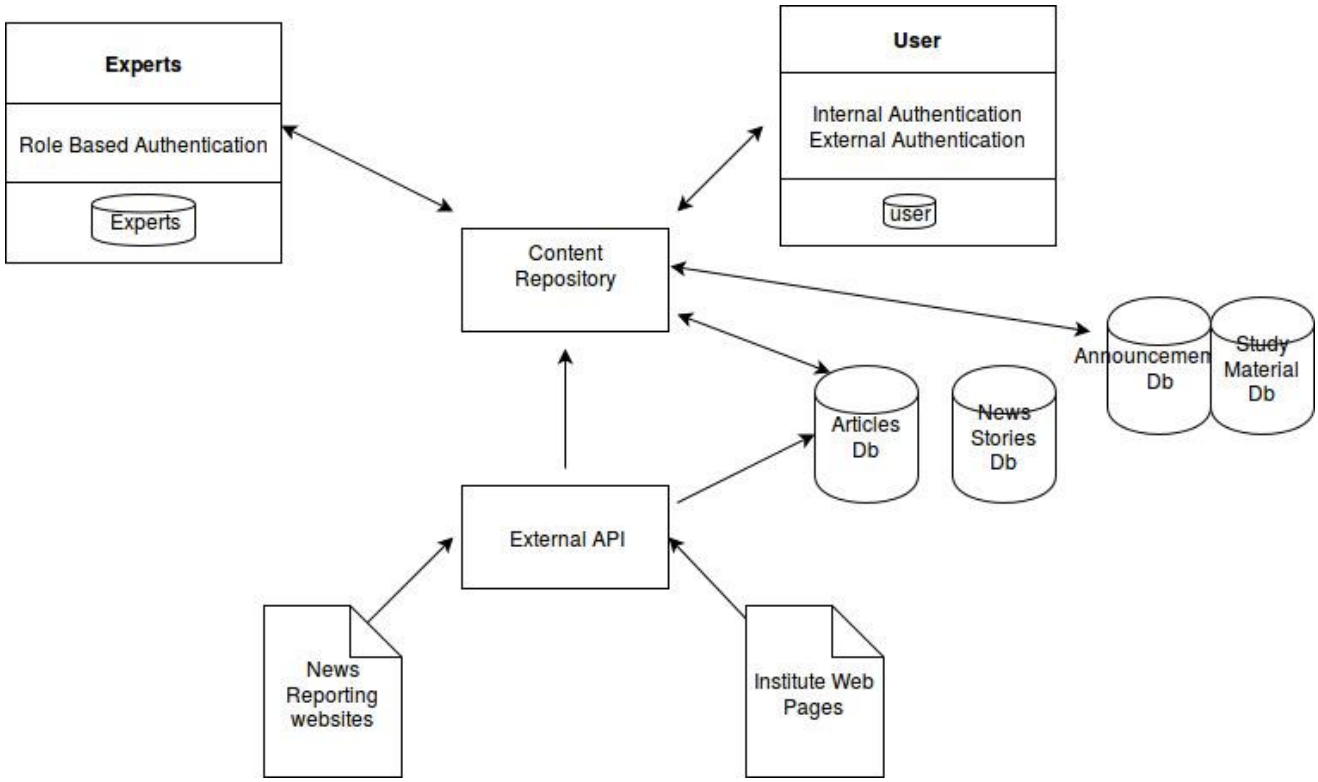

Figure 2.1.1. Architectural Design of the Admission Help system

2.2 Decomposition Description

The architectural diagram has the subsystems: User, Experts, External API, content repository and the databases. Each subsystem can be decomposed into smaller modules again depending on their functionality and structure. Each of these subsystems is described using the top-level data flow diagram and structural diagrams.

2.2.1. User

This subsystem contains basic user information, credentials and basic functionalities to interact with the environment. The user interacts with the system by asking queries, searching the information, updating his/her information etc.

Structural Diagram

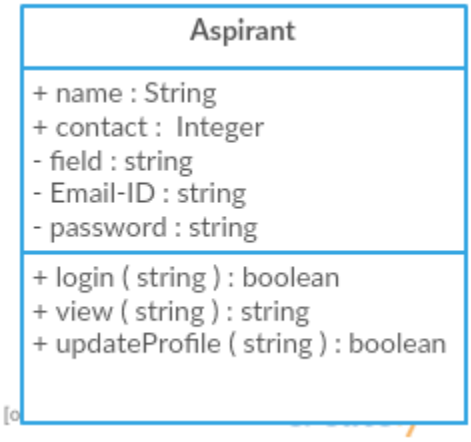

Figure 2.2.1 Structural Diagram of Aspirant

Data flow diagram

This is the data flow diagram of one of the functionality of the user i.e. searching for articles or new stories. The aspirant searches using Key words and the repository in turn gets the information from the storage and responds to the aspirant.

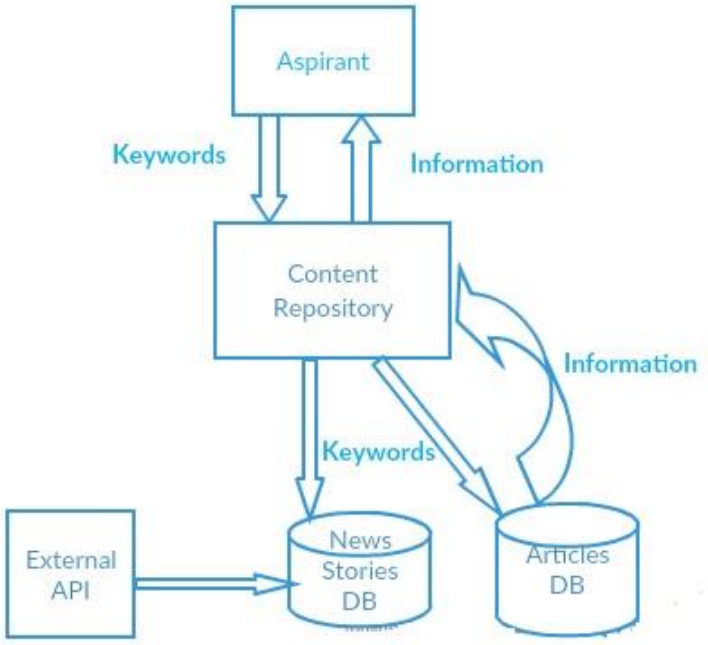

Figure 2.2.2 Data Flow Diagram of Searching Information

2.2.2. Experts

This subsystem contains the information about the expert, credentials and the functionality required to interact with the system. The expert interacts with the system by viewing and updating the study material, tips required.

Structural Diagram

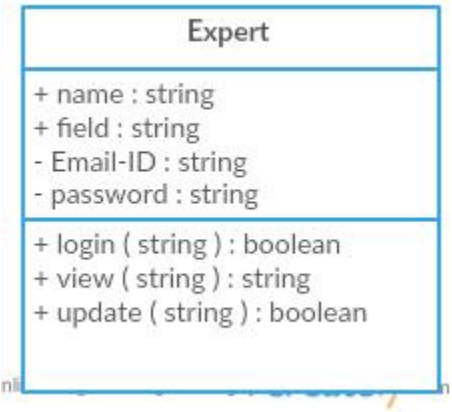

Figure 2.2.3 Structural diagram of Expert

Data Flow Diagram

The data flow diagram of the expert functionality: update study materials. The expert can view the study material and make necessary changes to maintain the content up-to-date.

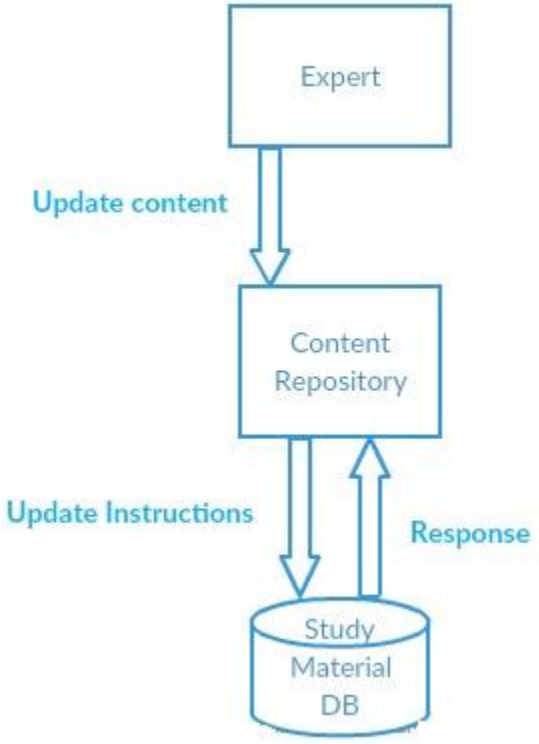

Figure 2.2.4 Data Flow Diagram of Updating study Material

2.2.3. External API

The External API subsystem is the entry point for the information gathered from other systems into our system. The information from News Reporting Sites and Institute Web pages is collected, ordered and stored in the respective Data Bases up-to-date.

Structural Diagram

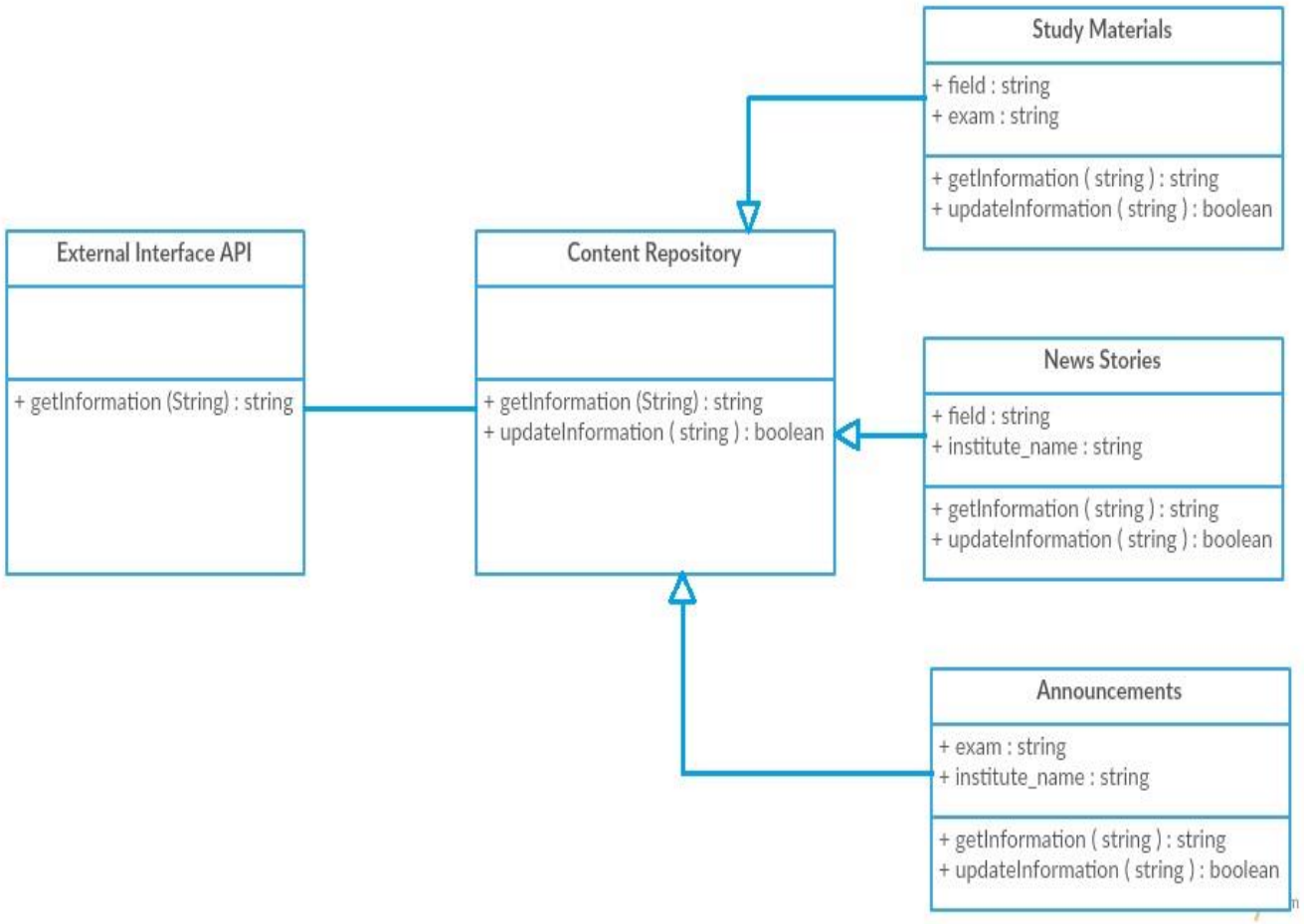

Figure 2.2.5 Structural Diagram of External API Interface

Data Flow Diagram

The information is collected from News Reporting Sites and Institute Web Pages through the External Interface API from time to time and added to the Repository which in turns stores in suitable Data bases.

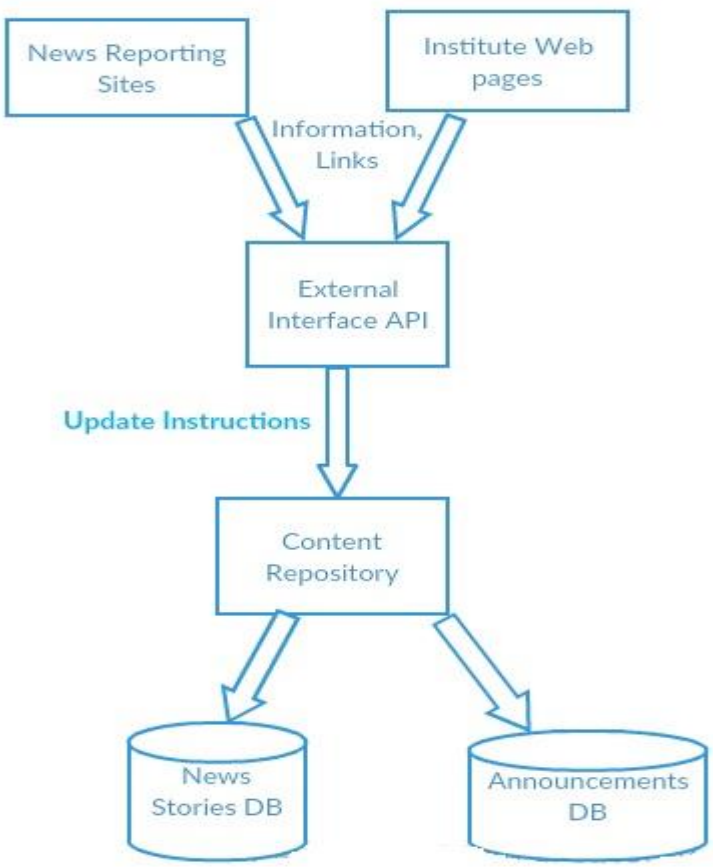

Figure 2.2.6 Data Flow Diagram of gathering Information from external interfaces

2.2.4. Content Repository

The Content Repository subsystem contains all the information that is maintained in the system. It can further be decomposed into various data bases as Study Material DB, News Stories DB, Announcements DB and Articles DB.

Structural Diagram

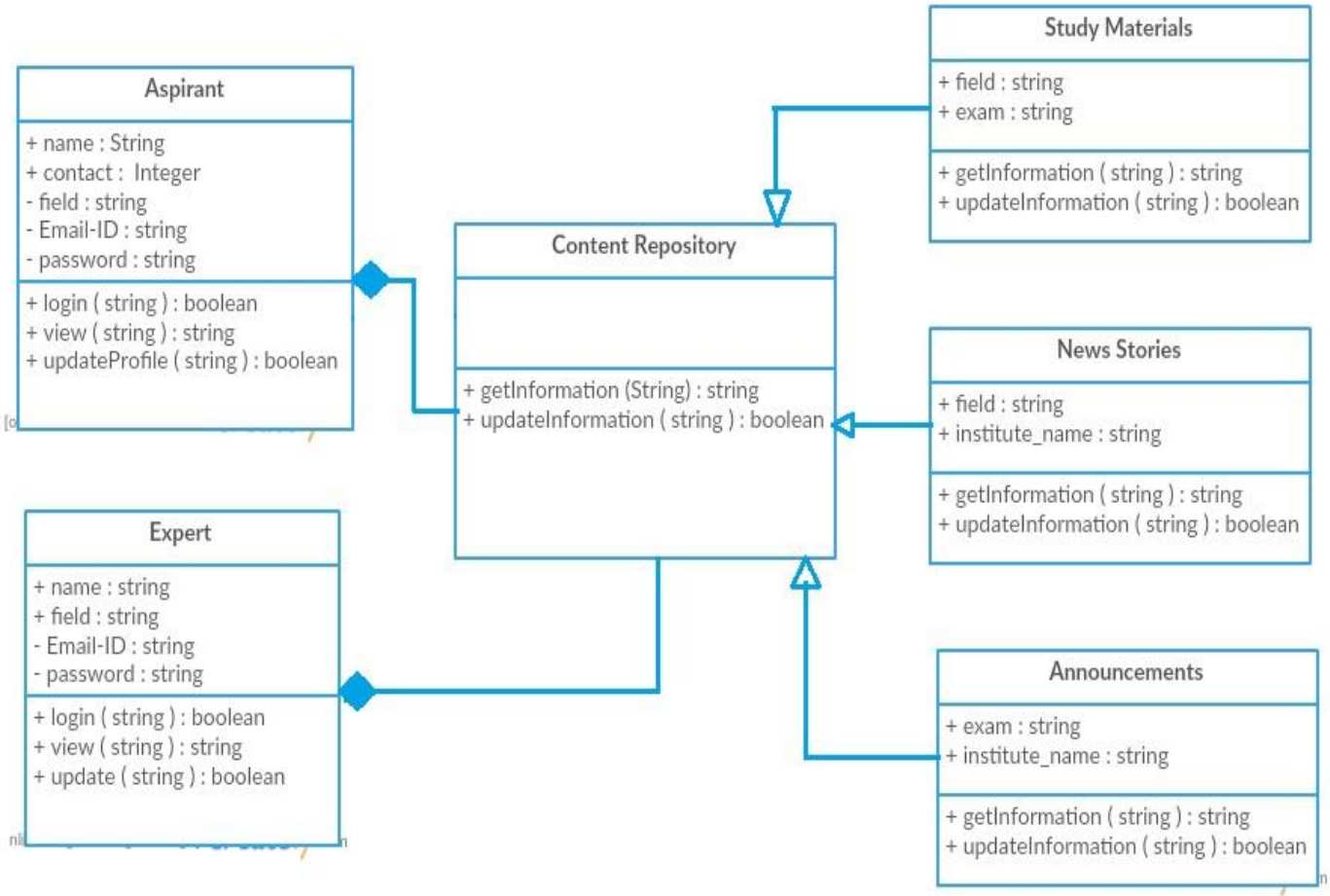

Figure 2.2.7 Structural diagram of Content Repository sub-system

Data Flow Diagram

The content Repository gets the data from Institute Web pages and News Reporting Sites through the External Interface API and stores in the Data bases News stories and Announcements. The data obtained from the Expert is stores in the Articles and Study Material Data bases.

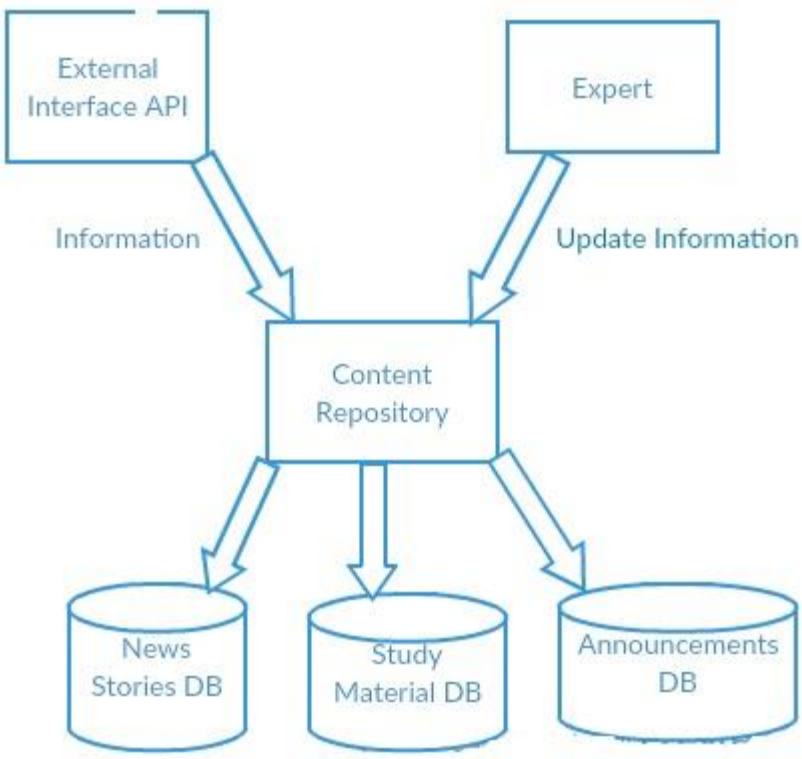

Figure 2.2.8 Data Flow Diagram of searching and updating Information in the storage

2.3 Design Rationale

Object oriented software development methods
Reasons:

- Improved software maintainability.
- Faster development
- Lower cost development
- Improved software development productivity
- Higher quality software

MVC Architecture
Reasons:

- It should interact with other machines or users effectively.
- For more efficient interaction system should have flexible interfaces.
- MVC can be taken as for a popular and easy to handle web application development style that has the feature of separating the presentation, Business & intermediate logics.
- Ease to coding and provide well defined interfaces within each logic.

3. Database Design:

3.1 Introduction:

The major requirements for database in Online Admission portal is for storing the user information details, courses information, examination details, study material, course content, metadata etc. Online Admission portal is using two databases in parallel to manage content on the data one is SQLite which is the relational database and MongoDB which is a NoSQL non-relational database.

3.2 SQLite:

This is a relational database system where various tables are formed in order to store data in a conventional way which includes storing of tuples having various attributes. For local development environment SQLite has been used in order to keep the dependencies simple but for production environments mysql comes into role. As online admission portal is majorly built on the Web2py framework.

In the online admission portal, the role of SQLite is to store the user profile, user information and course details, courseware details and fields.

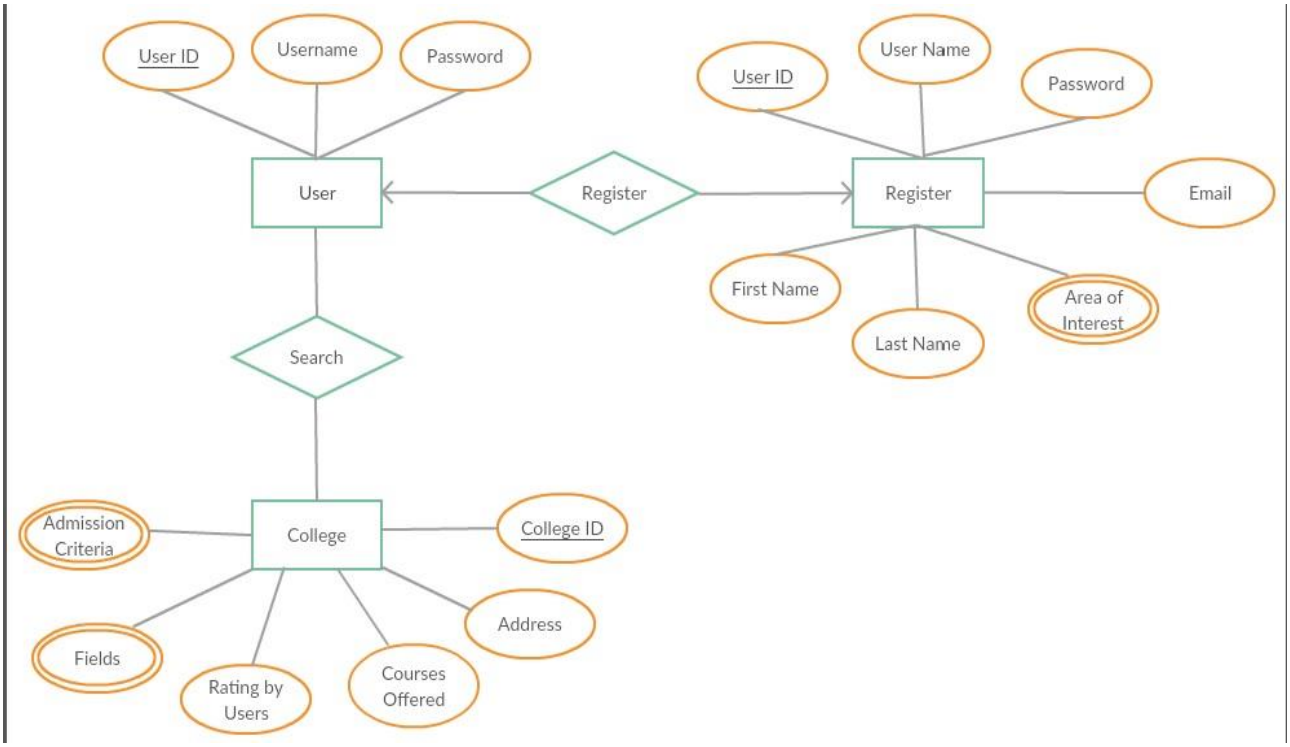

Figure 3.2.1: E-R diagram of data for online admission portal.

The basic information of users are included in user and register. The example of schema(of user and register respectively) are:

CREATE TABLE "user" ("username" varchar(30) NOT NULL PRIMARY KEY, "username" varchar(75) NOT NULL UNIQUE, "password" varchar(128) NOT NULL);
CREATE TABLE "register" ("userID" integer NOT NULL PRIMARY KEY, "username" varchar(30) NOT NULL UNIQUE, "password" varchar(128) NOT NULL, "email" varchar(75) NOT NULL UNIQUE , "first_name" varchar(30) NOT NULL, "last_name" varchar(30) NOT NULL, “area of interest” varchar(30));
These are the various categories in which the tables can be organized and the relation between these relations have been included in the ER diagram of the database.

3.3 MongoDB

Since Mongodb is a non-relational NoSql document based database thus facilitates to organize and store the course content and study materials. It has no predefined schema rather uses a dynamic schema in form of collections (which contains documents). Collection are different form table as, they don't have fixed and predefined typed set of attributes which are there in relational databases. The Documents are different from tuples as, they don't need to give information for all the attributes. Each document contains _id information which uniquely identifies a document serves as a primary key.
Key-value pairs are created for the data stored in Mongodb databases so it can be optimized whenever any kind of operation is performed in these databases. Thus Mongodb is much scalable and faster.

4. Component and Detailed Design

4.1 Design Patterns and Techniques used

Singleton Pattern

This is a creational design pattern and is one of the simplest patterns in the field of software engineering. It involves only one class which is responsible to instantiate itself, so that it creates no more than one instance. The singleton pattern is useful when access to limited resource needs to be controlled.

Applications:

In the system this pattern will be used for database manager.

Abstract factory pattern

Abstract factory pattern is creational design pattern that provides a way to encapsulate a group of individual factories that have a common theme without specifying their concrete classes. Abstract factory pattern offers the interface for creating a family of related objects without explicitly specifying their classes.

Applications:

The design pattern will be applied in creating different user accounts which are the different factories.

Observer Pattern

The observer pattern is a behavioural pattern which defines a one-to-many dependency between objects where a state change in one object results with all its dependents being notified and updated automatically. This pattern may be used in all situations where more than one display format for state information is required and where it is not necessary for the object that maintains the state information to know about the specific display formats used.

Applications:

Observer pattern will be used in the system for the operations of the system users.

Adapter Pattern

The adapter pattern is a structural pattern that translates one interface for a class to a compatible interface. This will convert the interface of a class into another interface that the user expects. Adapter gives the opportunity for the classes with incompatible interfaces to work together.

Applications:

This pattern will be used in the system when displaying information from the database.

4.2 Class Diagram

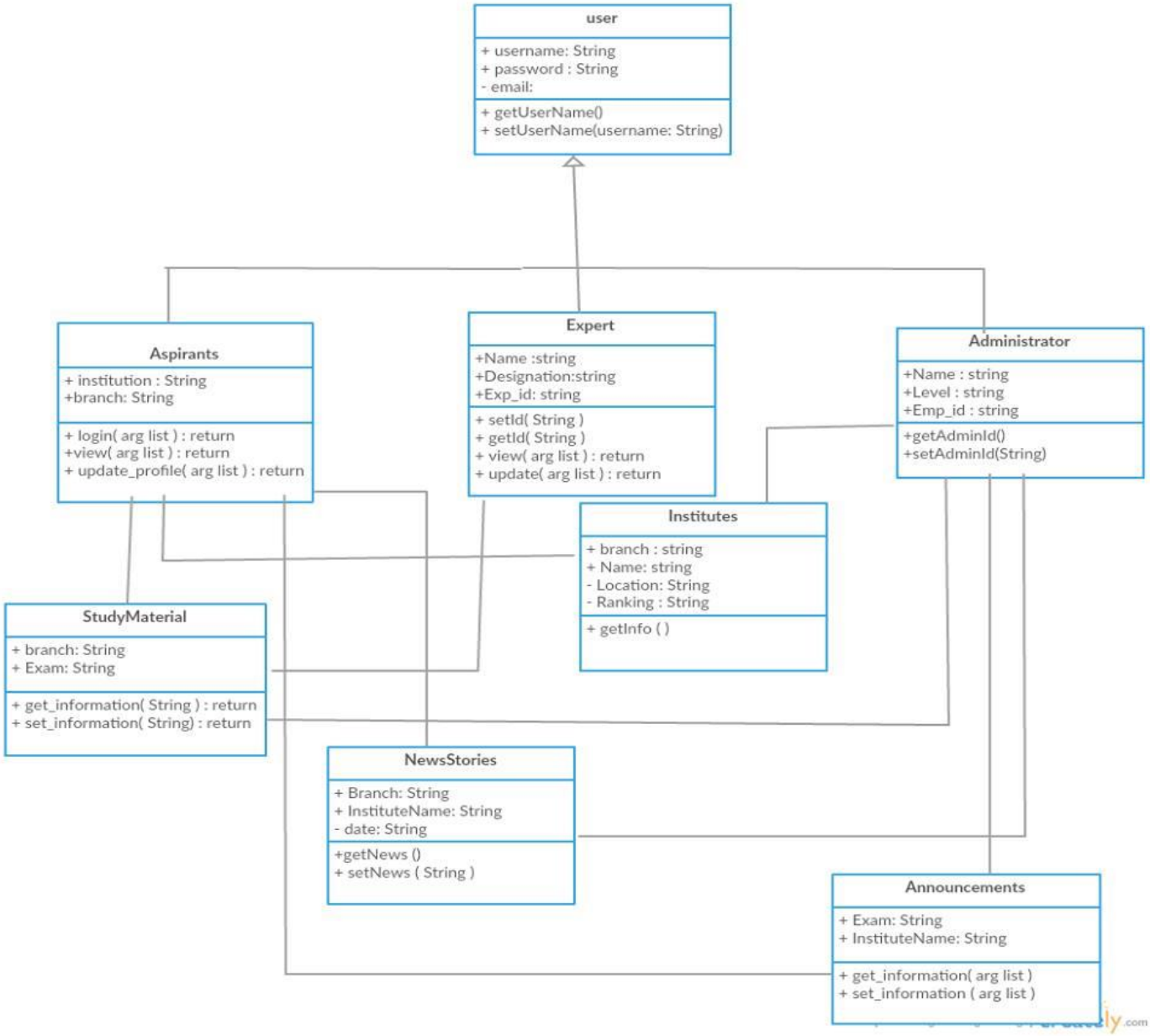

Figure 4.2.1 Class Diagram of Admission Help system

4.3 Sequence diagrams

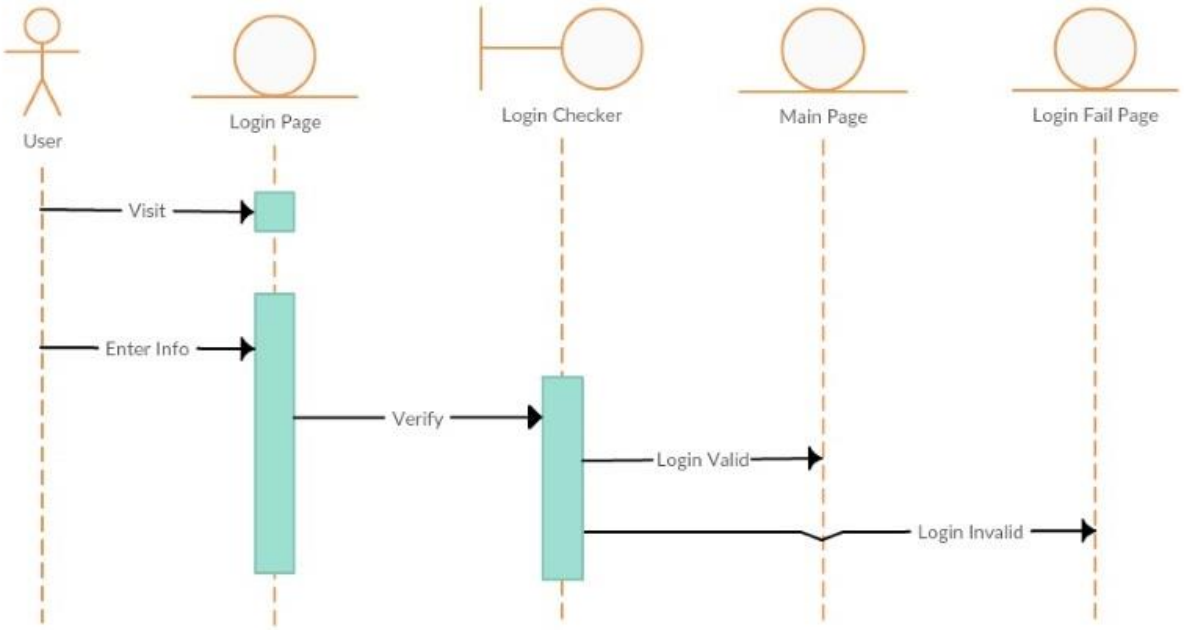

Figure 4.3.1: Sequence diagram of updating information for users profile and other information

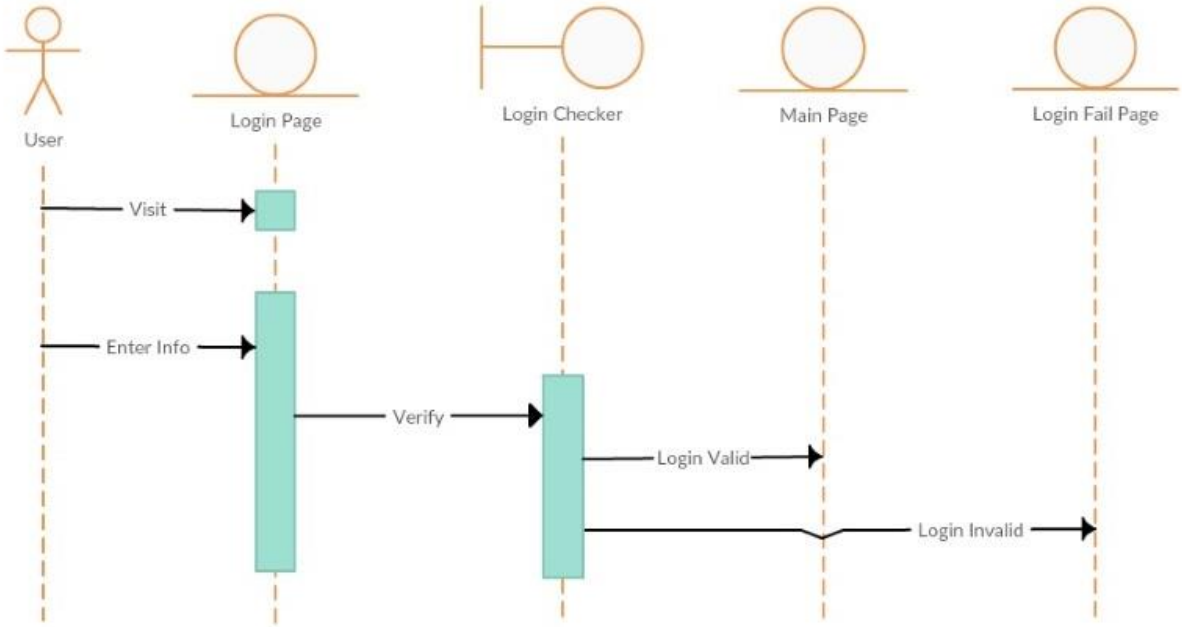

Figure 4.3.2: Sequence diagram for user/admin logins

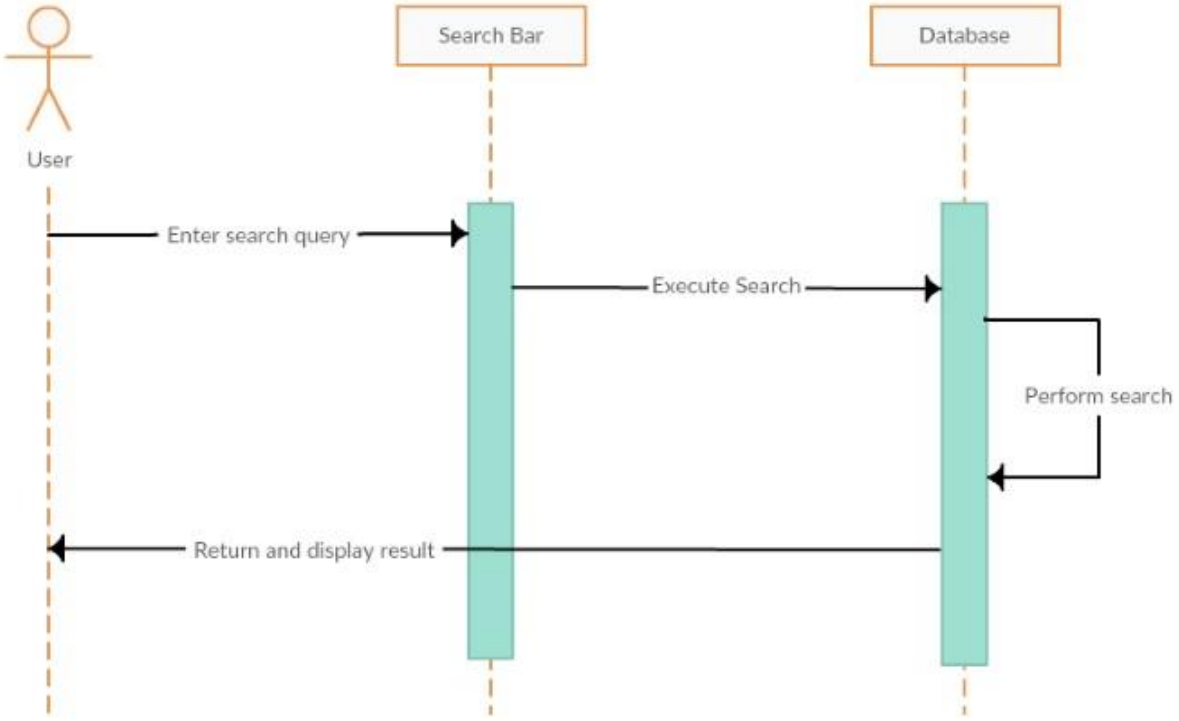

Figure 4.3.3: Sequence diagram for querying for information

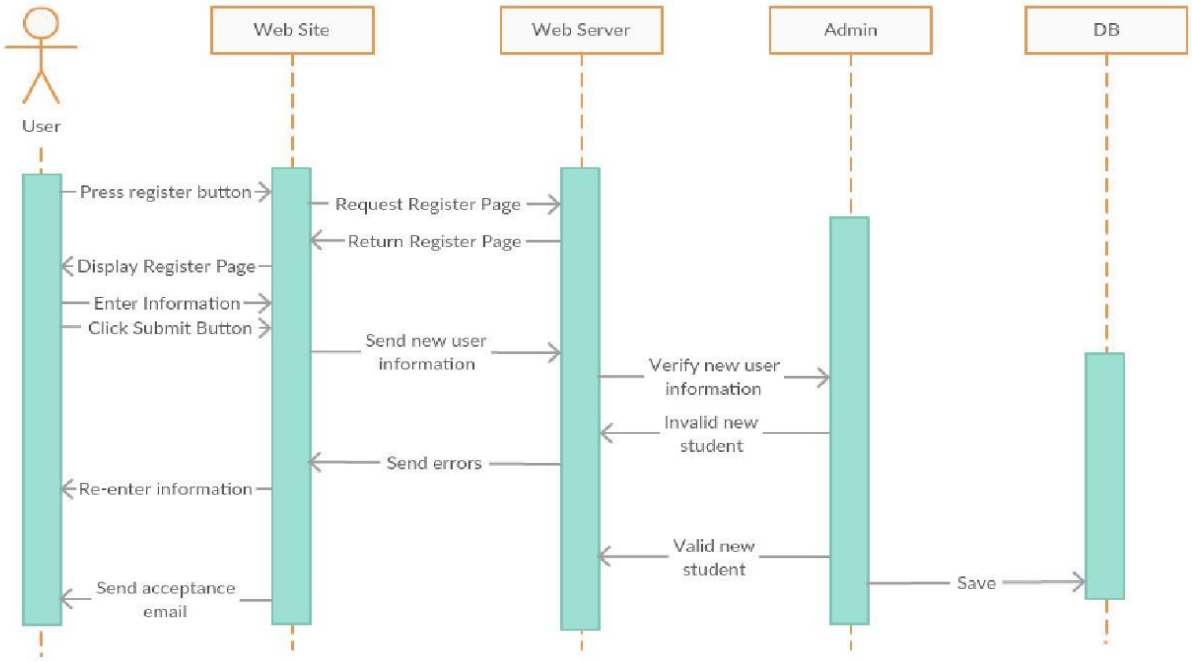

Figure 4.2.4 Registration for the Aspirant

4.4 User Interface Design

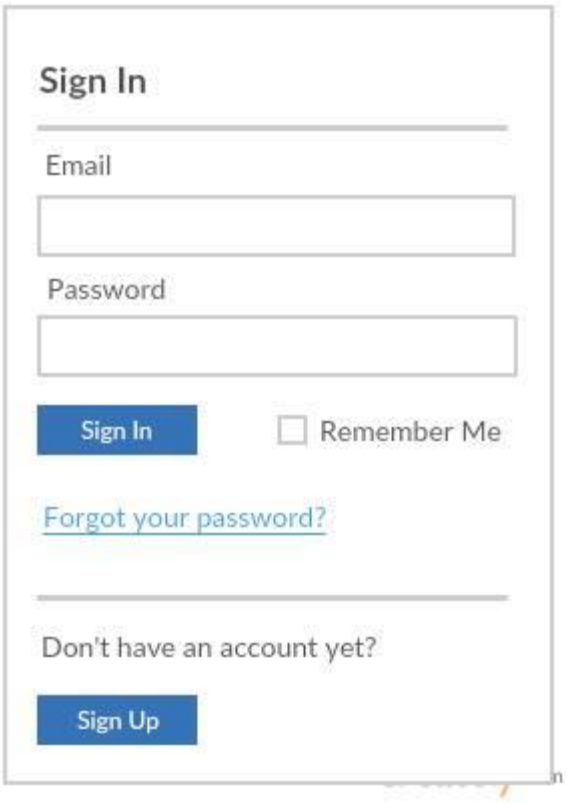

Figure 4.4.1: Sign in page for User

Figure 4.4.2: Sign up page for User

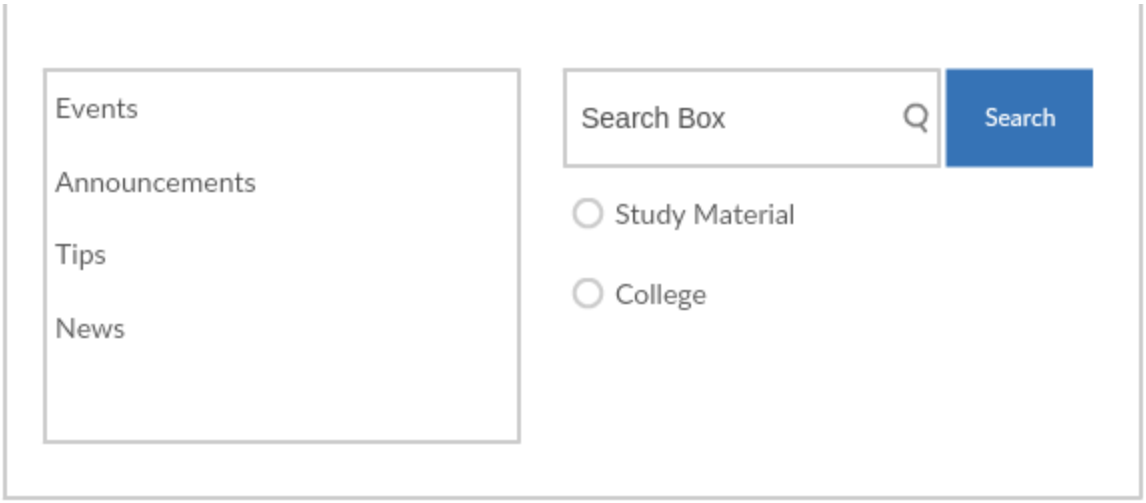

Figure 4.4.3: Search Page and major links for getting information