**CHAPTER 1**

**INTRODUCTION**

**1.1 Introduction to DBMS with Architecture Diagram**

A database is a collection of related data. However, the common use of the term database is usually more restricted. A database has the following implicit properties:

* A database represents some aspect of the real world, sometimes called the mini world or the universe of discourse (UOD). Changes to the mini world are reflected in the database.
* A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.
* A database is designed, built, and populated with data for a specific purpose.

It has an intended group of users and some preconceived applications in which these users are interested. In other words, a database has some source from which data is derived, some degree of interaction with events in the real world, and an audience that is actively interested in its contents.

A **database management system (DBMS)** is a computerized system that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications. Defining a database involves specifying the data types, structures, and constraints of the data to be stored in the database. The database definition or descriptive information is also stored by the DBMS in the form of a database catalog or dictionary; it is called meta-data. Constructing the database is the process of storing the data on some storage medium that is controlled by the DBMS. Manipulating a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the mini world, and generating reports from the data. Sharing a database allows multiple users and programs to access the database simultaneously

The goal of the three-schema architecture, illustrated in Figure 1.1, is to separate the user applications from the physical database.

In the below given Figure 1.1 DBMS architecture, schemas can be defined at the following three levels:

1. The **internal level** has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.
2. The **conceptual level** has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. This implementation conceptual schema is often based on a conceptual schema design in a high-level data model.
3. The **external or view level** includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group. As in the previous level, each external schema is typically implemented using a representational data model, possibly based on an external schema design in a high-level conceptual data model.

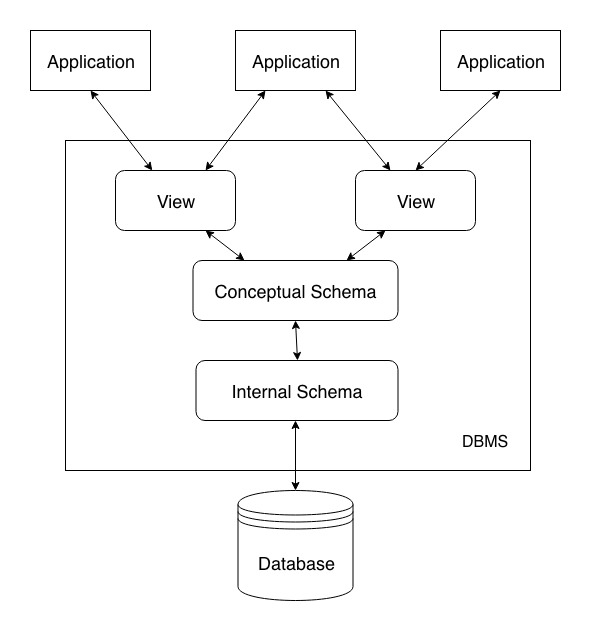


Figure 1.1 Three Schema Architecture (DBMS Architecture)

**1.2 Overview of the project**

* The goal of this project is to create a result sheet that can be downloaded and sent through sms.
* This database management system can be effectively used to get the results on time with better security management.
  + 1. **Problem Statement**

To maintain and manipulate the data to be stored in Marks Report Card database system.

* + 1. **Objectives of the Project**
* To be able to generate report cards from the marks database.
* To be able to export results in JSON format.
* To be able to send the result via sms.
* To be able to manage marks of the students.

**CHAPTER 2**

**SYSTEM DESIGN AND METHODOLOGY**

System design is a process of planing a new system or to replace the existing system.

Simply, system design is a blueprint for a building, it specifies all the features that are to be in the finished product

**2.1 System Architecture**

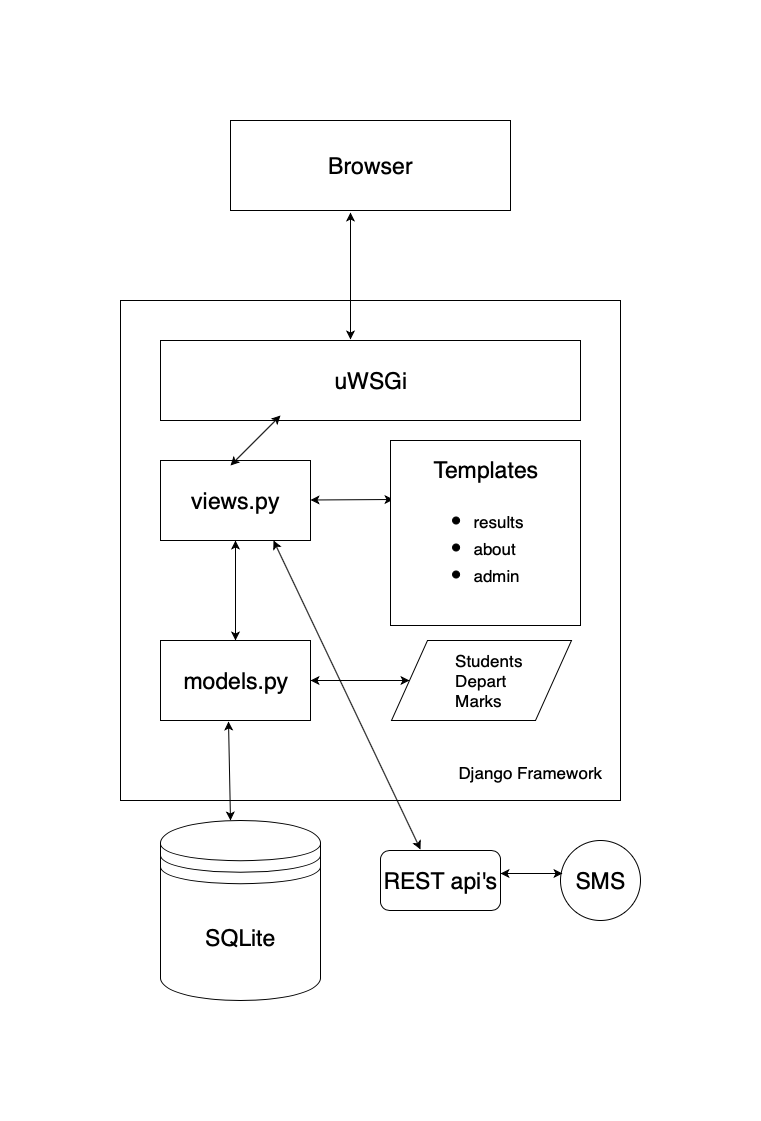


Figure 2.1 System Architechture Diagram for Marks Card Generation Tool

The above Figure 2.1 represents the System Architecture of the project. It mainly contains the framework called Django and a centralized Database from which data can be accessed by only the admin and the students.

Administrative access is granted to the teacher wherein he/she can login using the correct username and password.

Once the login is successful the admin page is seen wherein the admin can add, check, update or even delete the students details.

**2.2 ER Diagram**

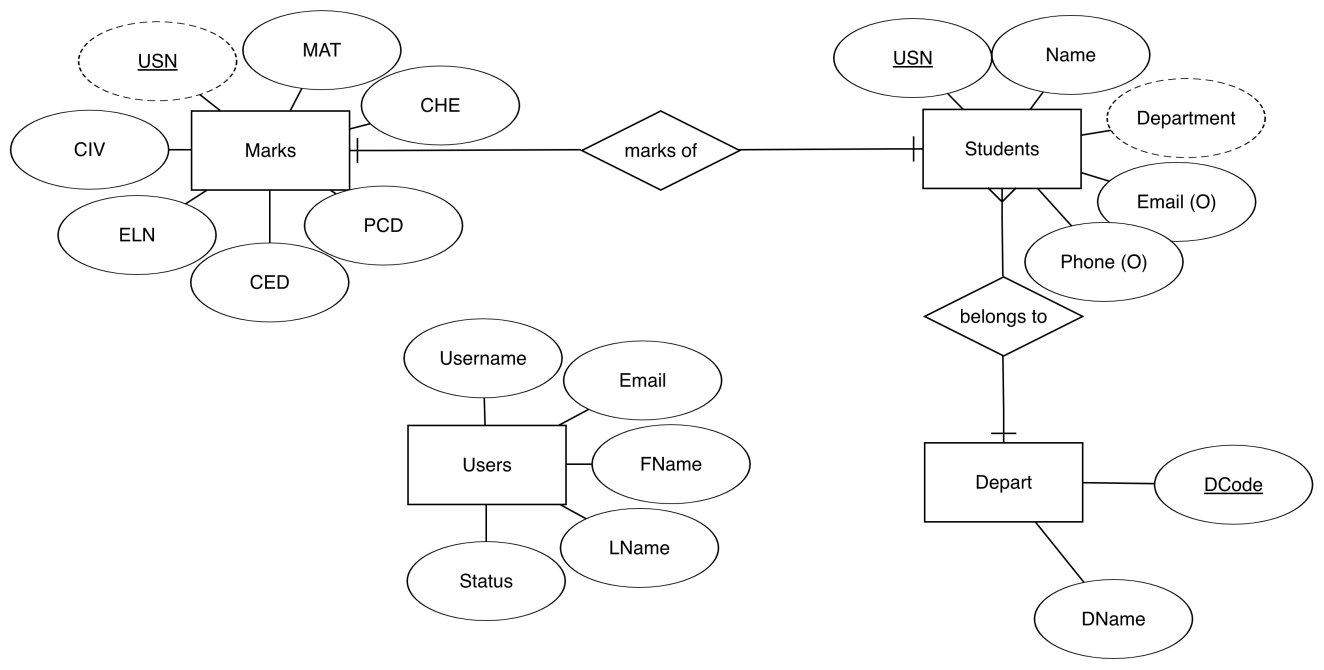
****

Figure 2.2 ER Diagram

The Figure 2.2 shows the Entity-Relationship Diagram (ERD) which is a data modelling technique that illustrates an information system’s entities and relationships between those entities. An ERD is a conceptual and a representational model of data used to represent the entity framework infrastructure.

The ER diagram shown in Figure 2.2 consists of 4 entities namely

1. Marks
2. Students
3. Depart
4. Users

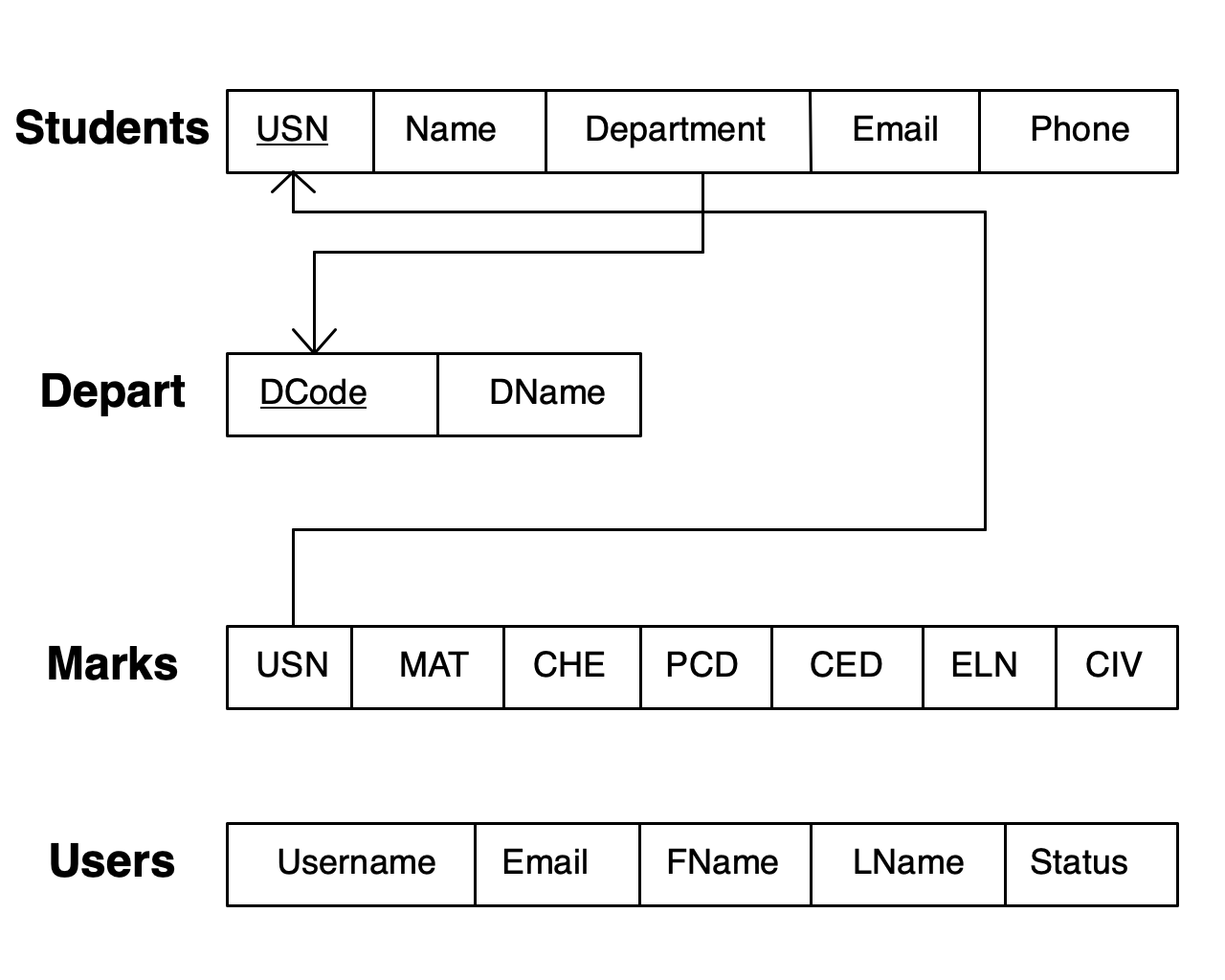
The Marks contains the following attributes: **USN, MAT, CHE, CHE, PCD, CED, ELN**.

The Students contains the following attributes: **USN, NAME, DEPARTMENT, EMAIL, PHONE**.

The Depart table contains following attributes: **DCODE, DNAME**.

The Users table contains following attributes: **USERNAME, EMAIL, FNAME, LNAME, STATUS.**

**2.3 Schema Diagram**

Figure 2.3 Schema Diagram

The logical mapping of ER diagram to a relational model is as shown in the Figure 2.3

**2.4 Algorithms**

**Stored procedure**

Step 1 : BEGIN

Step 2 : DECLARE PERCENTAGE INT;

Step 3 : ADD all (Marks) and store in PERCENTAGE;

Step 4 : DIVIDE PERCENTAGE by 6 and store in it;

Step 5 : DISPLAY PERCENTAGE;

Step 6 : Based upon the value of PERCENTAGE PASS (Green) or FAIL (Red) is decided.

Step 7 : END

**Trigger**

Step 1 : BEGIN;

Step 2 : INSERT / UPDATE into Students TABLE;

Step 3 : DISPLAY the actions performed with Date/Time and USER;

Step 4 : END

**CHAPTER 3**

**SYSTEM IMPLEMENTATION**

To implement this system we have used Sqlite, Django for the backend and HTML, CSS for the frontend (GUI).

**Features of HTML are**:

1) It is a very easy and simple language. It can be easily understood and modified.

2) It is very easy to make effective presentation with HTML because it has a lot of formatting tags.

3) It is a markup language so it provides a flexible way to design web pages along with the text.

4) It facilitates programmers to add link on the web pages (by html anchor tag) so it enhances the interest of browsing of the user.

5) It is platform-independent because it can be displayed on any platform like Windows, Linux and Macintosh etc.

6) It facilitates the programmer to add Graphics, Videos, and Sound to the web pages which makes it more attractive and interactive

**Features of CSS are:**

* CSS stands for Cascading Style Sheet.
* CSS is used to design HTML tags.
* CSS is a widely used language on the web.
* HTML, CSS and JavaScript are used for web designing. It helps the web designers to apply style on HTML tags.

**Features of SQLite are:**

* Transactions are atomic, consistent, isolated, and durable (ACID) even after system crashes and power failures.
* Zero-configuration - no setup or administration needed.
* Full-featured SQL implementation with advanced capabilities like partial indexes, indexes on expressions, JSON, and common table expressions. (Omitted features)
* A complete database is stored in a single cross-platform disk file. Great for use as an application file format.
* Supports terabyte-sized databases and gigabyte-sized strings and blobs.
* Small code footprint: less than 600KiB fully configured or much less with optional features omitted.
* Simple, easy to use API.
* Fast: In some cases, SQLite is faster than direct filesystem I/O
* Written in ANSI-C. TCL bindings included. Bindings for dozens of other languages available separately.
* Well-commented source code with 100% branch test coverage.
* Available as a single ANSI-C source-code file that is easy to compile and hence is easy to add into a larger project.
* Self-contained: no external dependencies.
* Cross-platform: Android, \*BSD, iOS, Linux, Mac, Solaris, VxWorks, and Windows (Win32, WinCE, WinRT) are supported out of the box. Easy to port to other systems.
* Sources are in the public domain. Use for any purpose.
* Comes with a standalone [command-line interface](https://www.sqlite.org/cli.html) (CLI) client that can be used to administer SQLite databases.

**Features of Django are:**

* Rapid Development:  The project implementation phase is a very time taken but Django creates it rapidly.
* Secure: Django takes security seriously and helps developers to avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery etc. Its user authentication system provides a secure way to manage user accounts and passwords.
* Scalable: Django is scalable in nature and has ability to quickly and flexibly switch from small to large scale application project.
* Fully loaded: Django includes various helping task modules and libraries which can be used to handle common Web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds etc.
* Versatile: Django is versatile in nature which allows it to build applications for different-different domains. Now a days, Companies are using Django to build various types of applications like: content management systems, social networks sites or scientific computing platforms etc.
* Open Source: Django is an open source web application framework. It is publicly available without cost. It can be downloaded with source code from the public repository. Open source reduces the total cost of the application development.
* Vast and Supported Community: Django is a one of the most popular web frameworks. It has widely supportive community and channels to share and connect.

**3.1 Module Description**

The modules included in this project are:

**1. USN Search**

**Input :** USN of the student

**Output:** A successful search would lead to the respective result page of that student with the input USN.

**Description:** We have used HTML and CSS for front-end. The homepage contains a textbox indicating to enter the USN of the user and a submit button. If the entered USN is correct then it takes to the result page where it contains the result of the user

**2. Result Page**

This page gives the description about the student and provides the result with the percentage and give options like printing, exporting it to JSON and sending result via SMS.

**3. Admin Login**

**Input:** Username and password

**Output:** If succesfully logged in, then it leads to the admin page where any changes can be made. If wrong credentials are entered then an error message pops us indicating that the entered username or password is incorrect.

**Description:** This page contains two textboxes one for username and the other for password and contains the log in button if logged in properly then it leads to admin page or else shows an error.

**4. Admin Page**

This page would allow the admin to Insert, Delete and Update the student’s information and provides features like log out and change password for the admin.

**5. Student Details**

**Input:** USN, Name, Department, Email, Phone.

**Outpu**t**:** Student details are inserted successfully.

**Description:** This page contains 5 textboxes for filling the details. Saving options like save and add another, save and continue editing and save are featured.

**6. Marks Details**

**Input:** Marks for each subject.

**Output:** Marks are inserted successfully.

**Description:** It contains a total of 7 textboxes. It will ask the user for the USN and the 6 subject’s marks.

**7. Users Details**

**Input:** Username, Email, First Name, Last Name, Status.

**Output:** New user created successfully.

**Description:** It contains the above mentioned fields. It asks the user to provide the details for the new user. It creates a user and gives the option to choose user permissions.

**CHAPTER 4**

**RESULTS AND SCREENSHOTS**

**4.1 Home Page**

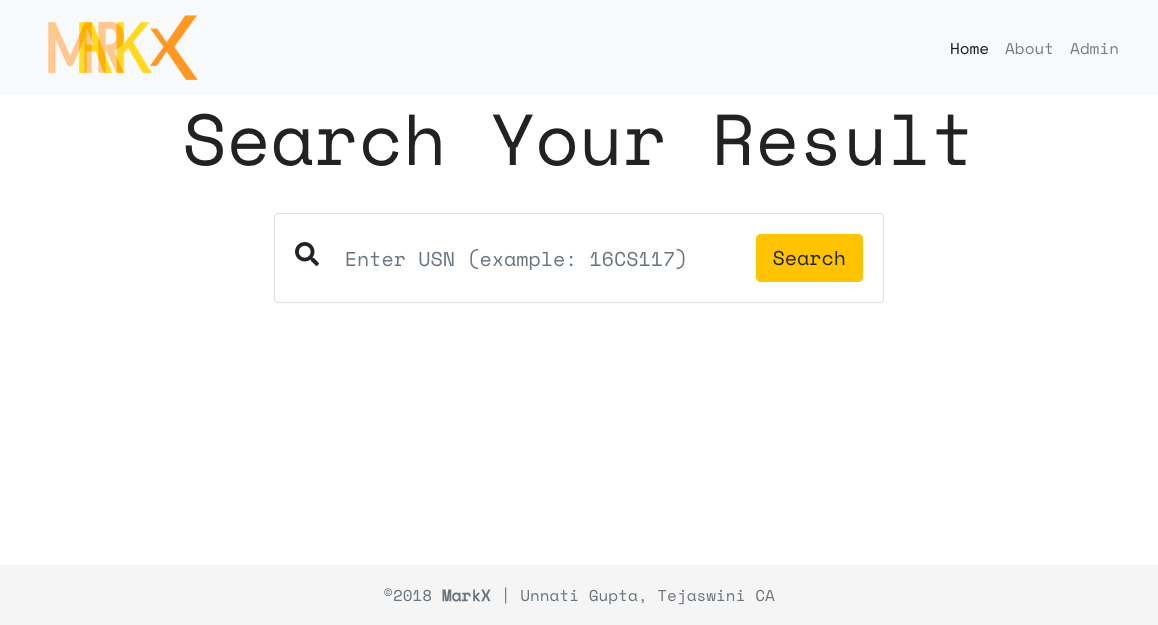
****

Figure 4.1 Home Page

The above Figure 4.1 shows the Home Page to the user where USN can be entered.

**4.2 Admin Login Page**

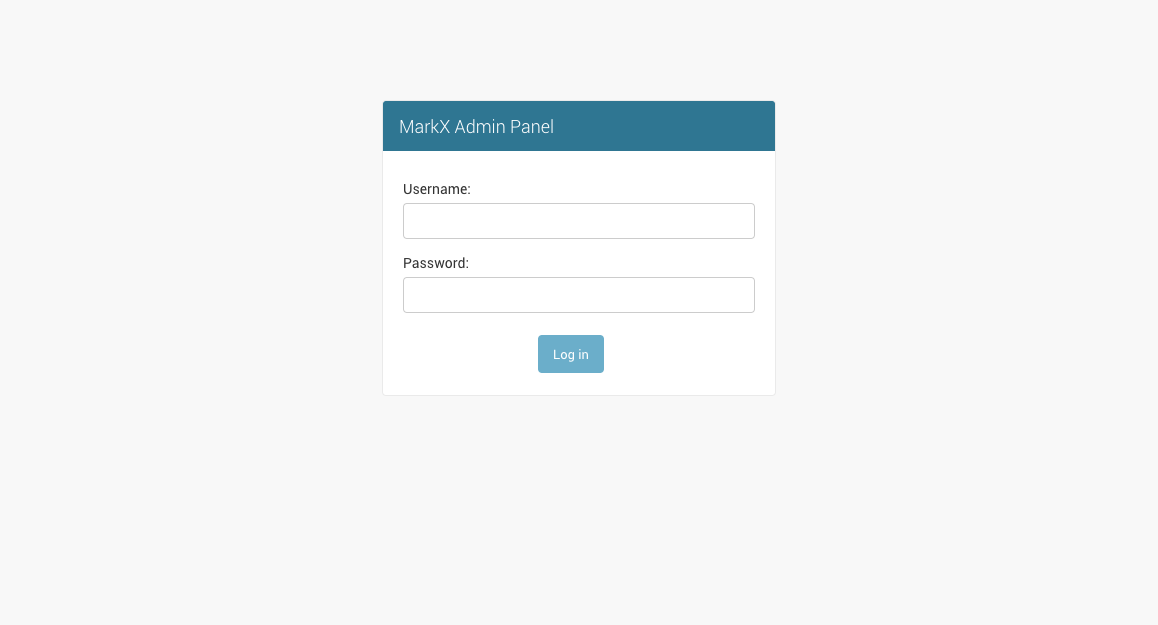


Figure 4.2 Admin Login Page

The above Figure 4.2 shows the Admin Login Page where it asks for admin credentials for logging in.

**4.3 Admin Panel Page**

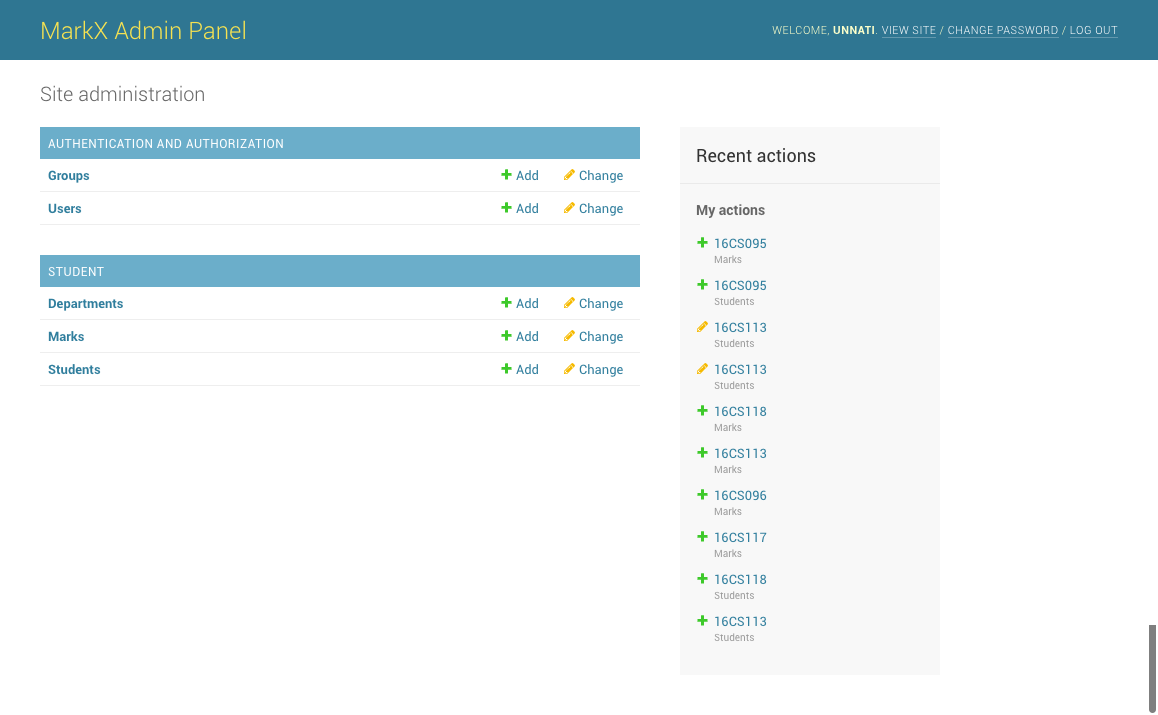
****

Figure 4.3 Admin Panel Page

The above Figure 4.3 shows the Admin Panel Page where it displays all the tables in the system.

**4.4 Students Table Page**

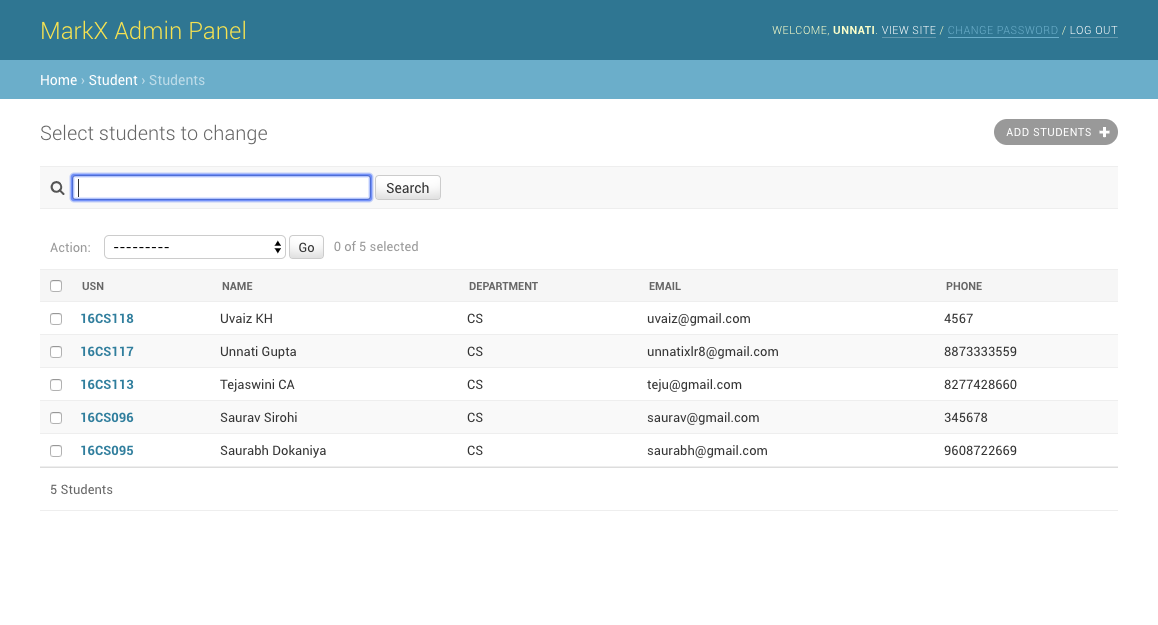
****

Figure 4.4 Students Table Page

The above Figure 4.4 shows the Students Table Page where it displays the contents of the Students table.

**4.5 Marks Table Page**

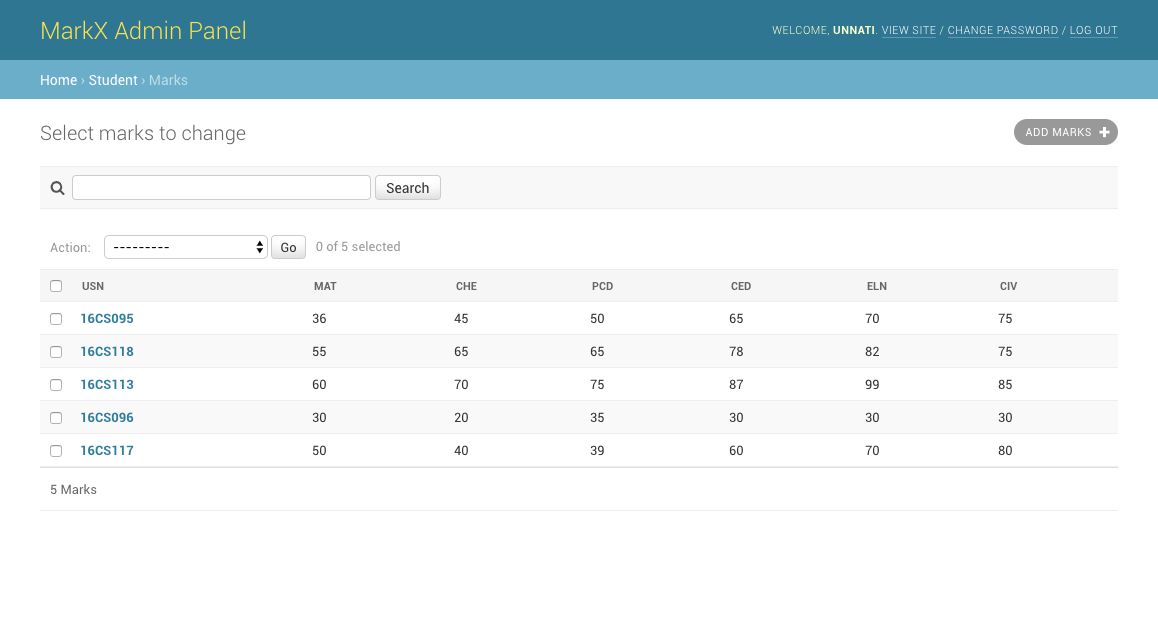
****

Figure 4.5 Marks Table Page

The above Figure 4.5 shows the Marks Table Page where it displays the contents of the Marks table.

**4.6 Departments Table Page**

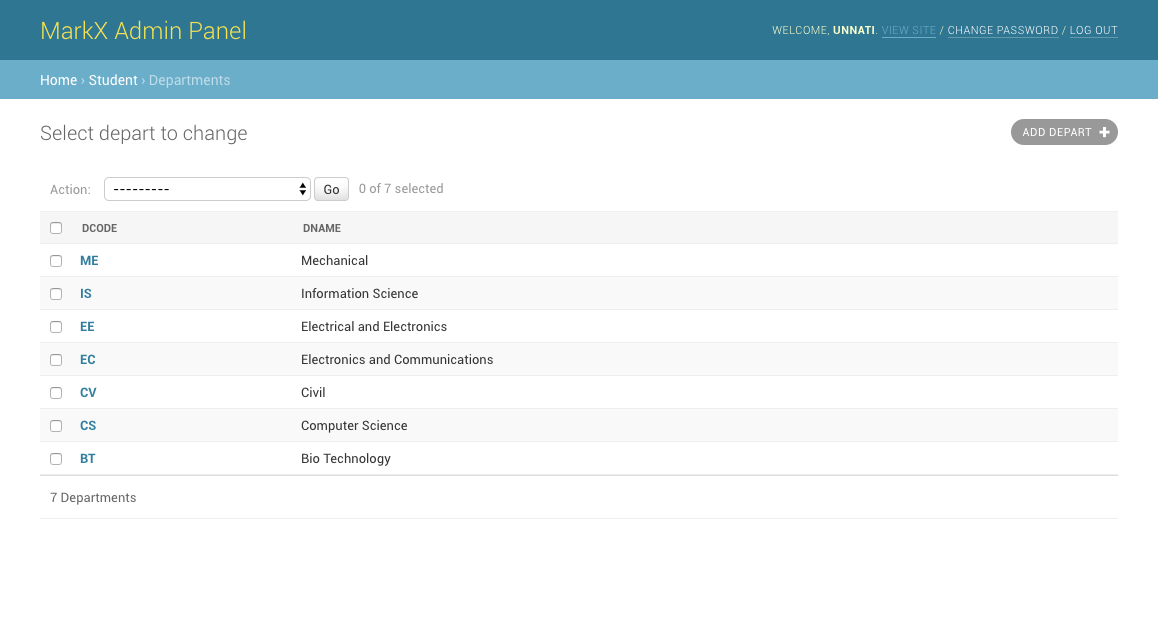
****

Figure 4.6 Departments Table Page

The above Figure 4.6 shows the Departments Table Page where it displays the contents of the Department table.

**4.7 Users Table Page**

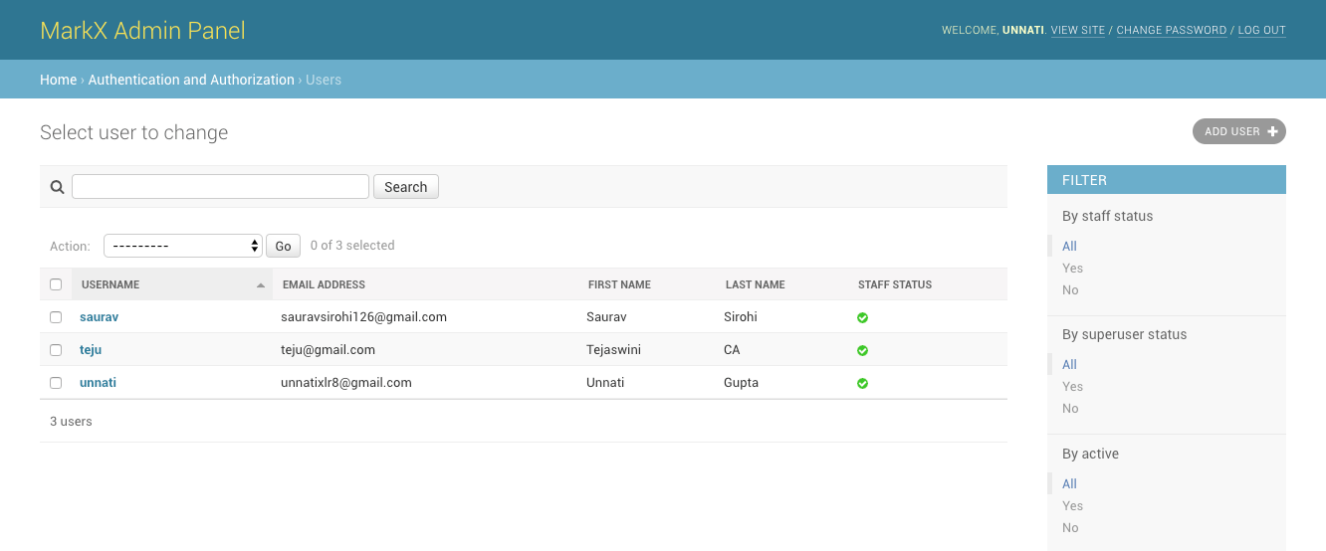
****

Figure 4.7 Users Table Page

The above Figure 4.7 shows the Users Table Page where it displays the contents of the Users table.

**4.8 Add Student Page**

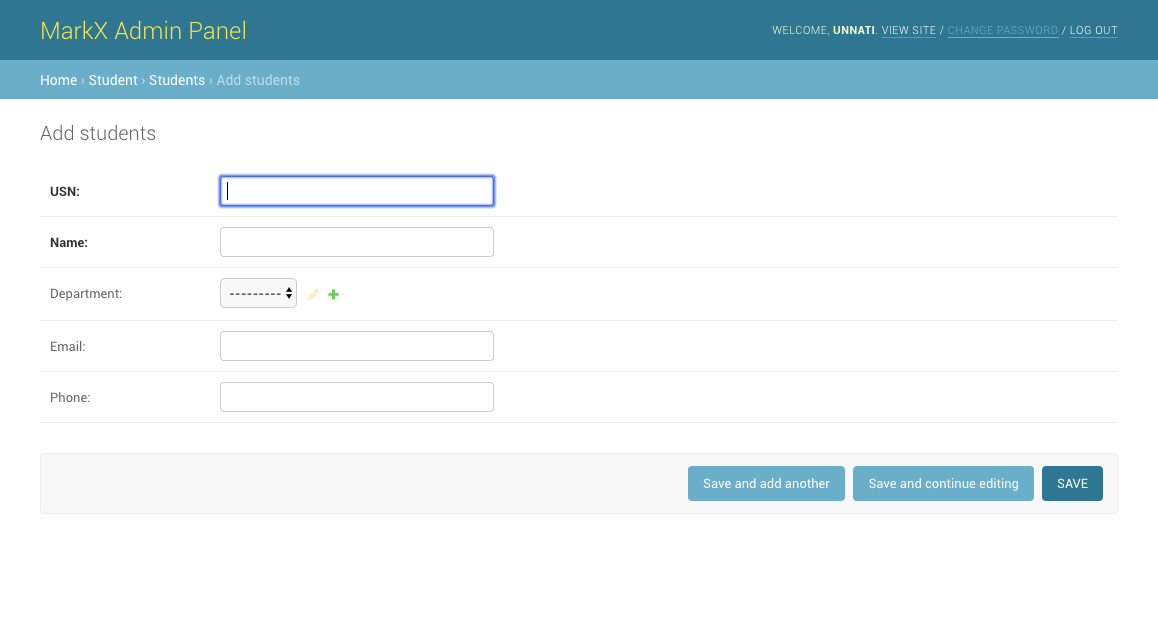
****

Figure 4.8 Add Students Page

The above Figure 4.8 shows the Add Student Page where it provides a form to add new student. It asks for USN, Name, Department, Email and Phone.

**4.9 Update Student Page**

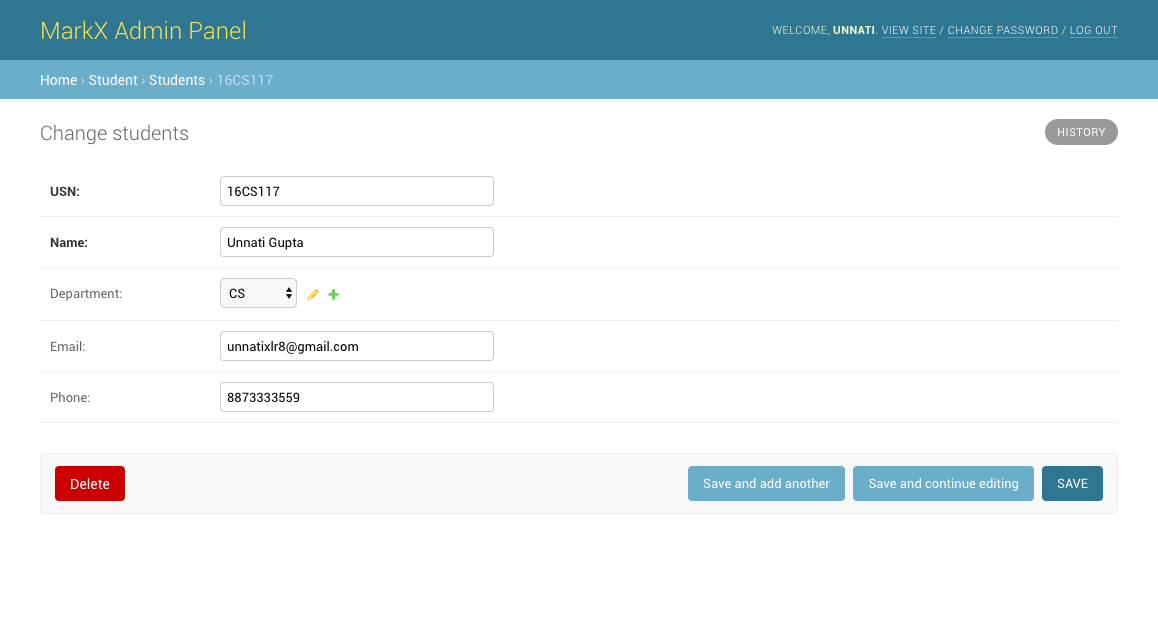


Figure 4.9 Update Student Page

The above Figure 4.9 shows the Update Student Page where it allows to update the Student details. It provides an option to delete the entry.

**4.10 Add Marks Page**

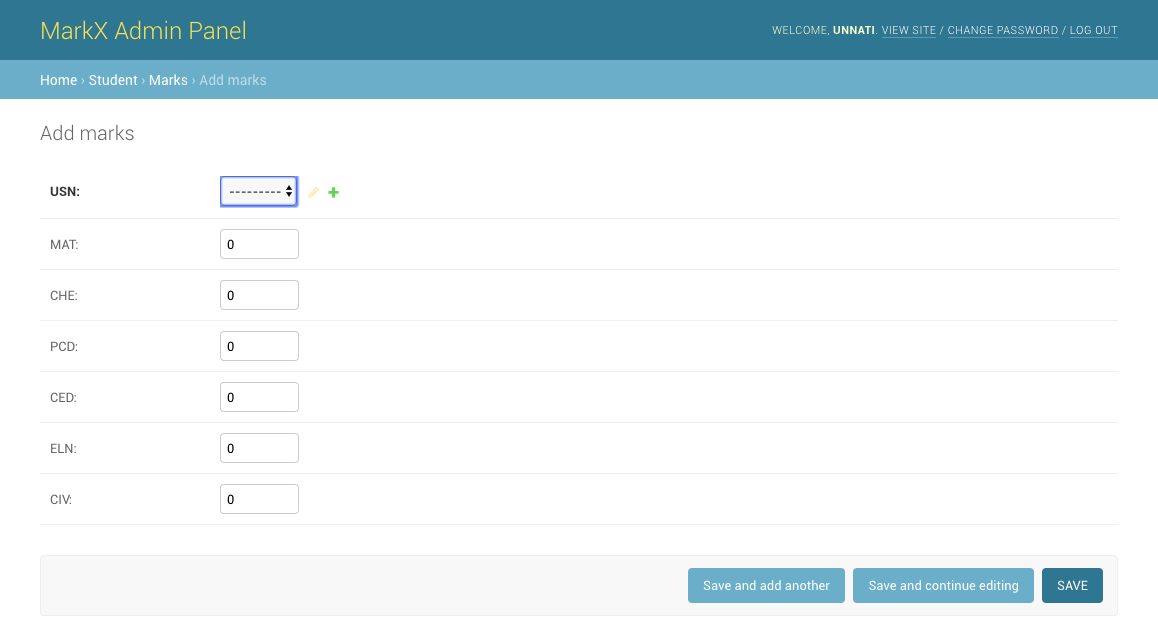
****

Figure 4.10 Add Marks Page

The above Figure 4.10 shows the Add Marks Page where it provides a form to add marks details for the 6 subjects for a particular USN.

**4.11 Update Marks Page**

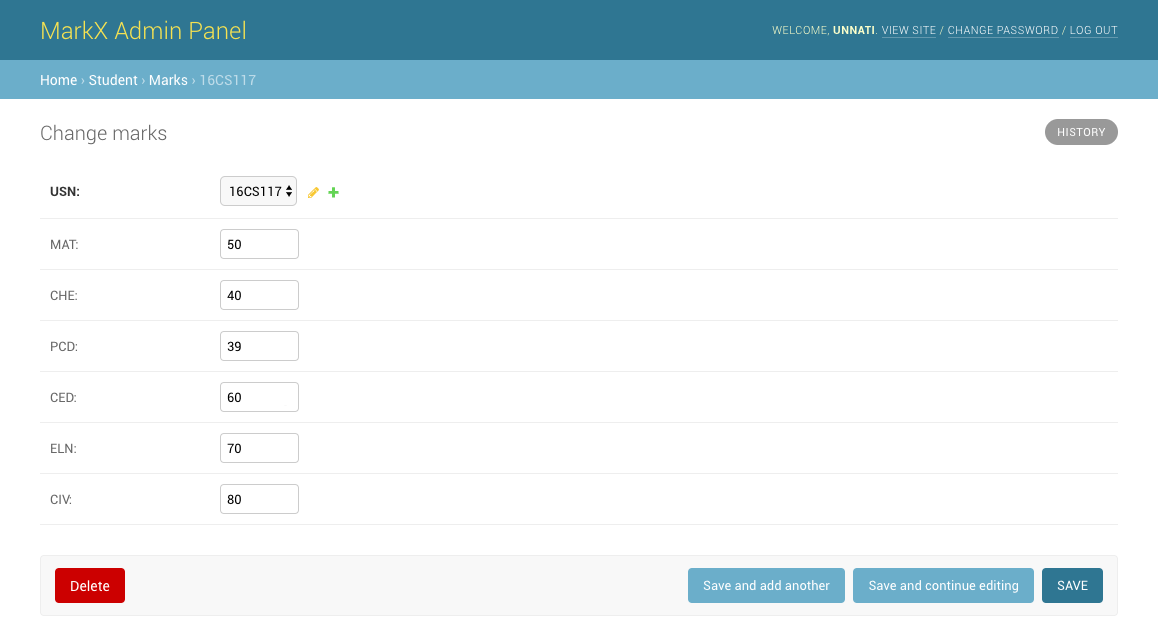
****

Figure 4.11 Update Marks Page

The above Figure 4.11 shows the Update Marks Page where it allows to update the marks details of a particular USN. It provides an option to delete the entry.

**4.12 Add Department Page**

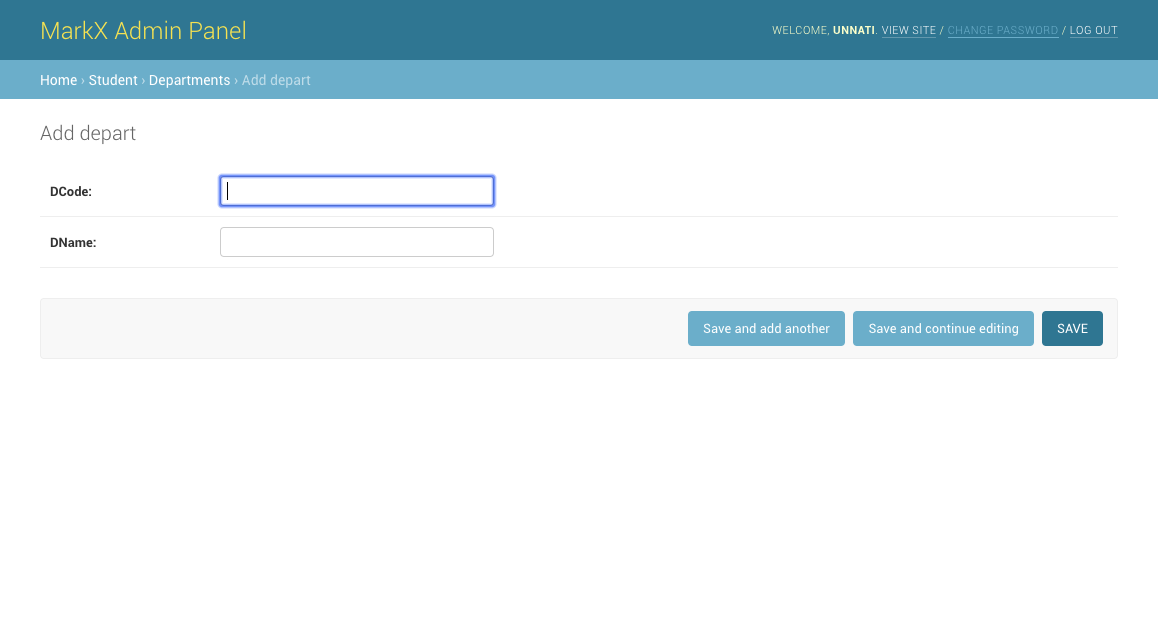
****

Figure 4.12 Add Department Page

The above Figure 4.12 shows the Add Department Page where it provides a form to add new Department. It asks for Department Code and Department Name.

**4.13 Update Department Page**

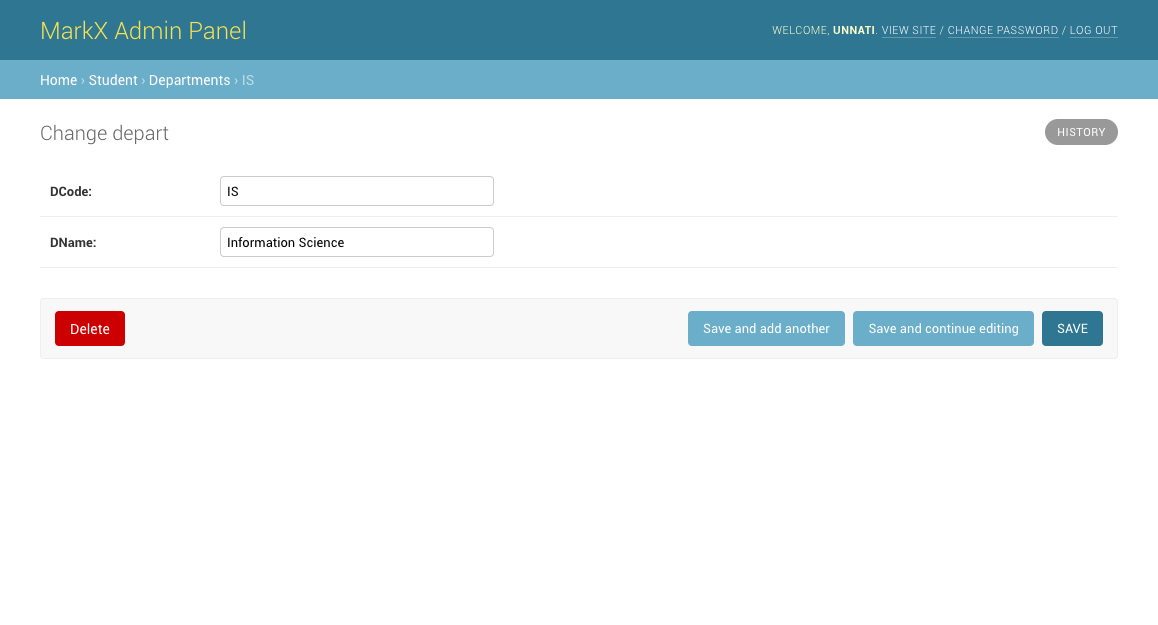
****

Figure 4.13 Update Department Page

The above Figure 4.13 shows the Update Department Page where it allows to update the details of a particular Department. It provides an option to delete the entry.

**4.14 Add User Page**

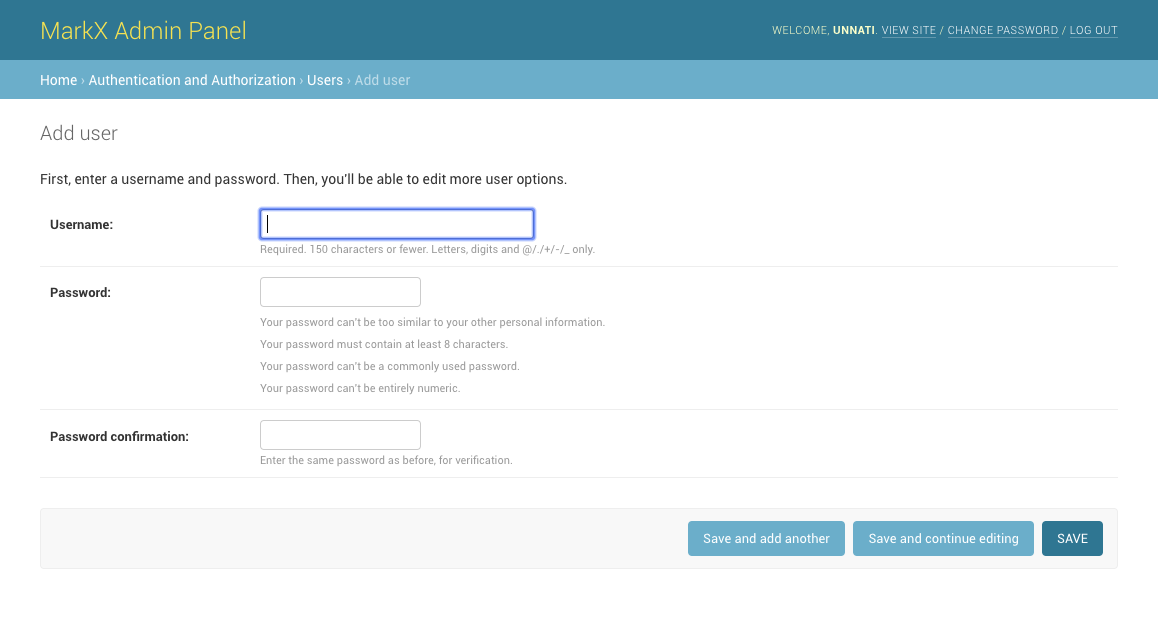
****

Figure 4.14 Add User Page

The above figure 4.14 shows the Add User Page where it provides a form to add new user. It asks for a new username and password.

**4.15 Change Password Page**

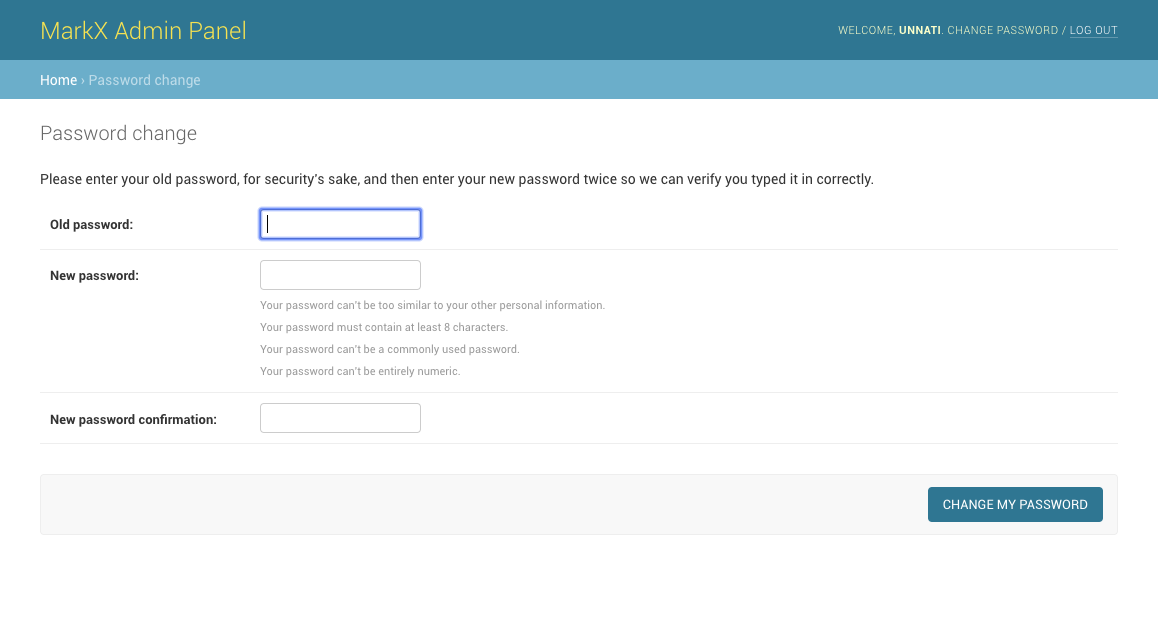
****

Figure 4.15 Change Password Page

The above figure 4.15 shows the Change Password Page where it provides a form to change the password of the current logged in User.

**4.16 Search Result Page**

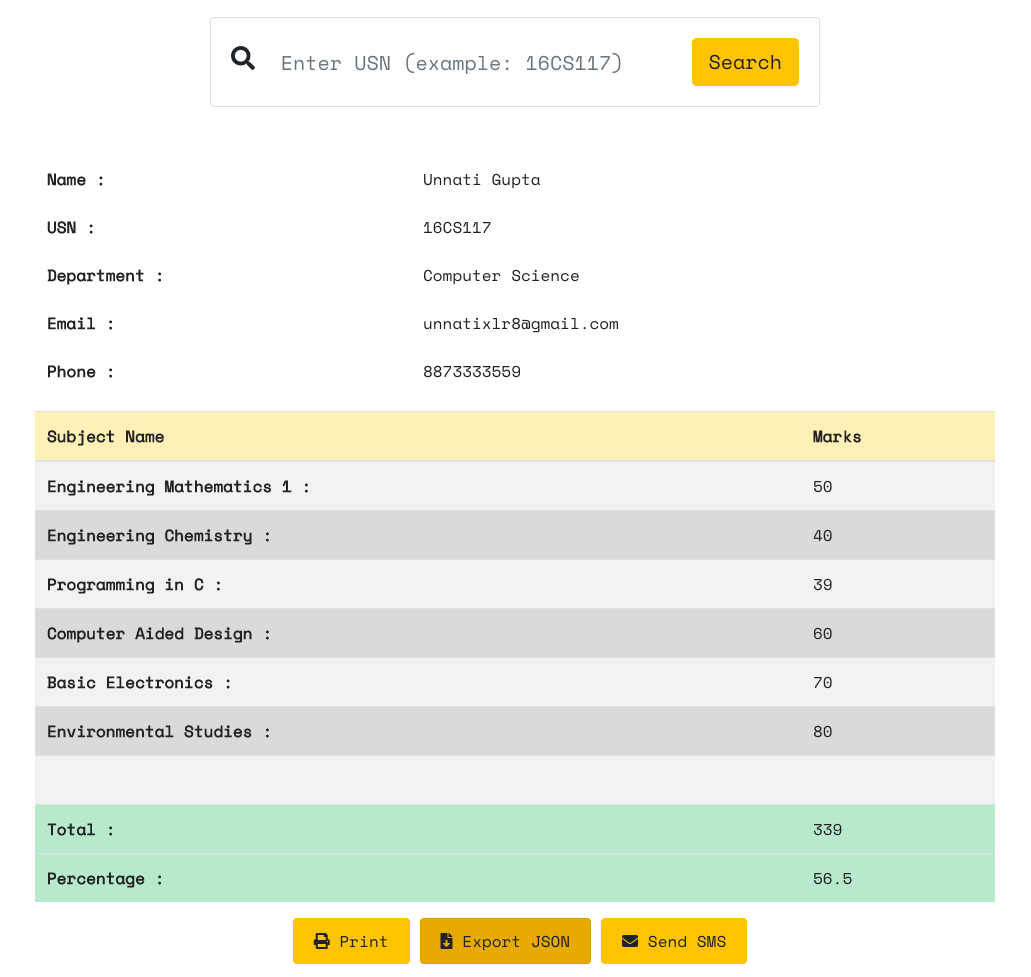
****

Figure 4.16 Search Result Page

The above Figure 4.16 shows Search Result Page where it shows the result of the student by providing USN. It features the result to be Printed, Exported and be sent via SMS.

**4.17 Log Page**

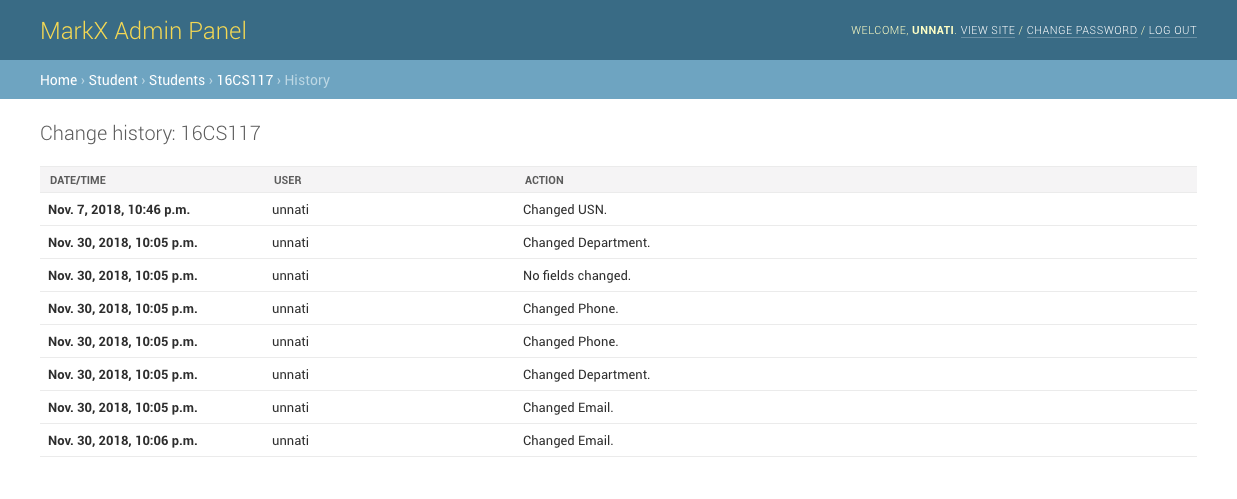


Figure 4.17 Log Page

The above Figure 4.17 shows Log Page where it shows the logs of the operations done for the particular student.

**CHAPTER 5**

**Conclusion and Future works**

The main objective of this project is to generate result sheet and to manage the student’s records safely. Several user-friendly codes have been adopted. The project shall prove to be a powerful package which satisfies all the requirements of the user.

The following are the planned future works:

1. The project shall host the platform on online servers to make it accessible worldwide.
2. The project shall integrate multiple load balancers to distribute loads on system.
3. The project shall include a master-slave database structure to reduce overload on databases on regular basis on different servers.

**Bibliography**

[1] https://docs.djangoproject.com/en.2.1

[2] https://www.sqlite.org/docs.html

[3] https://djangobook.com/

[4] https://tutorial.djangogirls.org/en/

[5] https://www.w3schools.com/

[6] https://getbootstrap.com/docs/4.0/getting-started/introduction/

[7] Database System Concepts, Fifth Edition, Avi Silberschatz, 2005