STUDY PORTAL

A PROJECT REPORT

Submitted By

HARISH
(University Roll no. 2100290140066)
ROHIT SHARMA
(University Roll no. 2100290140113)
NIKUNJ TYAGI
(University Roll no. 2100290140097)

Submitted in partial fulfilment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
Dr. SHASHANK BHARDWAJ
Associate Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206

CERTIFICATE

Certified that **Harish 210029014030459,** have carried out the project work having "**Study Portal**" for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU**)** (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

Harish 2100290140066

Rohit Sharma 2100290140113

Nikunj Tyagi 2100290140097

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

Dr. SHASHANK BHARDWAJ
Associate Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Signature of Internal Examiner **Signature of External Examiner**

Dr. Arun Tripathi
Head, Department of Computer Applications

ABSTRACT

The purpose of Study Portal is to automate the existing manual system with the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy access and manipulation of the same. The required software and hardware are easily available and easy to work with.

Study Portal, as described above, can lead to an error-free, secure, reliable, and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on record keeping. Thus, it will help the organization in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant while being able to reach the information.

The aim is to automate its existing manual system with the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy access and manipulation of the same. Basically, the project describes how to manage for good performance and better services for the clients

ACKNOWLEDGEMENT

Success in life is never attained single handedly. My deepest gratitude goes to my thesis supervisor, **Dr. Shashank Bhardwaj** for his guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Arun Kumar Tripathi, Professor and Head, Department of Computer Applications**, for his insightful comments and administrative help at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Harish

Rohit Sharma

Nikunj Tyagi

TABLE OF CONTENT

1. Introduction of Project

1.1 Introduction

A website is a collection of web pages; thus, a portal is a collection of links to different websites. The study portal is a gateway to connect many web applications which is useful for doing student activities. The portal web pages provide access to many applications such as online courses and links to different websites.

Study Portal is a web application which is useful for learners to choose right information. It works like, initially the one who needs to get access only after his or her registeration to the application. The membership is only provided by the admin after checking the correctness of information provided by the user. This application can be implemented in a specificed organization or with a world-wide access.

Only the members can make use of services provided in study portal. Thus, a guest user can only view the home page of the web application. After the successful registeration the user can make use of the services only.

1.2 Objective of the project

The main objective behind construction of this portal is to provide a single place to students from where they can do all the study related activities in different websites. In other words, to provide a single place for all kind of works.

System Objective is of being useful in significant way by providing most basic and most essential functionalities and features to its users in efficient and effective manner.

1.3 Scope of the project

To make learning easy and interesting for students.

They can perform various activities in a single platform without switching to various sites.

It makes easy for student to create their profile where they can check out for their further activities.

This project touches all the related boundaries of study portal.

To search in other websites by being in a single platform, to make the learning easy which saves the time as well and which help student to concentrate in their stuff.

1.4 Overview of the project

This Project handle various platform related to study are-

- Home Page
- Register
- Login Page
- Profile Page
- Logout Page
- Notes
- Homework
- Youtube
- To Do List
- Books
- Dictionary
- Wikipedia
- Conversion

1.5 Technologies

- HTML 5
- CSS 3
- JAVASCRIPT
- BOOTSTRAP-5
- DJANGO
- PYTHON
- DATABASE, SQLITE 3
- VISUAL STUDIO CODE

2. Requirment Analysis

2.1 Functional Requirments

The user can navigate the website's main functions. If the user is interested in the search results and wishes to further their activities then they can login if already registered using their 'username' and 'password', otherwise they need to create an account. After login they will be redirected to the main screen of the home page of study portal where they will have access to various options depending upon their users rights.Administrators are registered already on the web application. It deals with queries and make the desired changes on the system.

2.2 Non-Functional Requirement

Login- i) Enter username and password
ii) Check username and password
iii) Verify login credentials

The system administrator monitors the user's activities.
Performance Requirements: The proposed system that we are going to develop will be used by the citizens of the Study Portal. Therefore, It is expected that the database would perform functionally all the requirements that are specified by the client.
Safety Requirements: Database is an important aspect of the any system. So it is required to take back up of the database.

2.3 Hardware Requirements

- Minimum Hardware Requirements -
  RAM - 128MB to 512MB
  PROCESSOR - Intel core i5
  HARD DISK - 80 GB

2.4 Software Requirements

- Minimum Software Requirements -
  Operating System - Windows 10 and Other Versions
  Front End - Html, CSS, Java Script, Bootstrap
  Server Side Script- Django
  Back-End Tool - My Sql
  IDE tool - Visual Studio Code

3. Requirement Spcecifiation Document

3.1 Purpose of the Document

This SRS Document contains the complete software requirements for the Study Portal and describes the design decisions, architectural design and the detailed design needed to implement the system.

It provides the visibility in the design and provides information needed for software support.

3.2 Scope of the Development Project

Study Portal is developing for students to make their learning easy.

This is to built upon the existing web-based project marking system in order to implement the project marking process and allocating supervisior/ideas to students.

This increase in efficiency of project marking, audit trails of marking process, give feedback to student.

Our project goals are to create a website to handle the on line recruitment procedure in the following manner:

Applicant: The applicant can register information for the further learning process.

Operator: Operator can issue the information to each users for the online. He is responsible for the online services.

Administrator: Control the whole website and give access rights to all user operators of the website.

3.3 Certain specific requirements

Inputs and Outputs:

Input Data: the software will be keyboard and mouse driven, all input data are from left mouse clicks on context buttons such as directional buttons or menu options. In case of filling information keyboard is required.

Output Data: Facts and graphics will be displayed to the monitor.

User interface:

The user interface for this product will be graphical and relatively simple in order to accommodate the target audience. All input to our program will be achieved through a mouse as well as keyboard while all output would be via a monitor.

Hardware interaction:

OS: Windows 10

HDD: Minimum 20GB

RAM: 256 MB

Color: 16 bit

Platform: Web

Software interaction:

Web Browsers: IE 6.0

4. Software Design

- ### Modules

  The project contains five modules:

  1. Welcome page of Study Portal
  2. Admin login
  3. Registeration page
  4. Home page of Study Portal
  5. Various Websites

➢ **Welcome page of Study Portal-** In this module student can check out for the welcome page, dashboard and various activities.

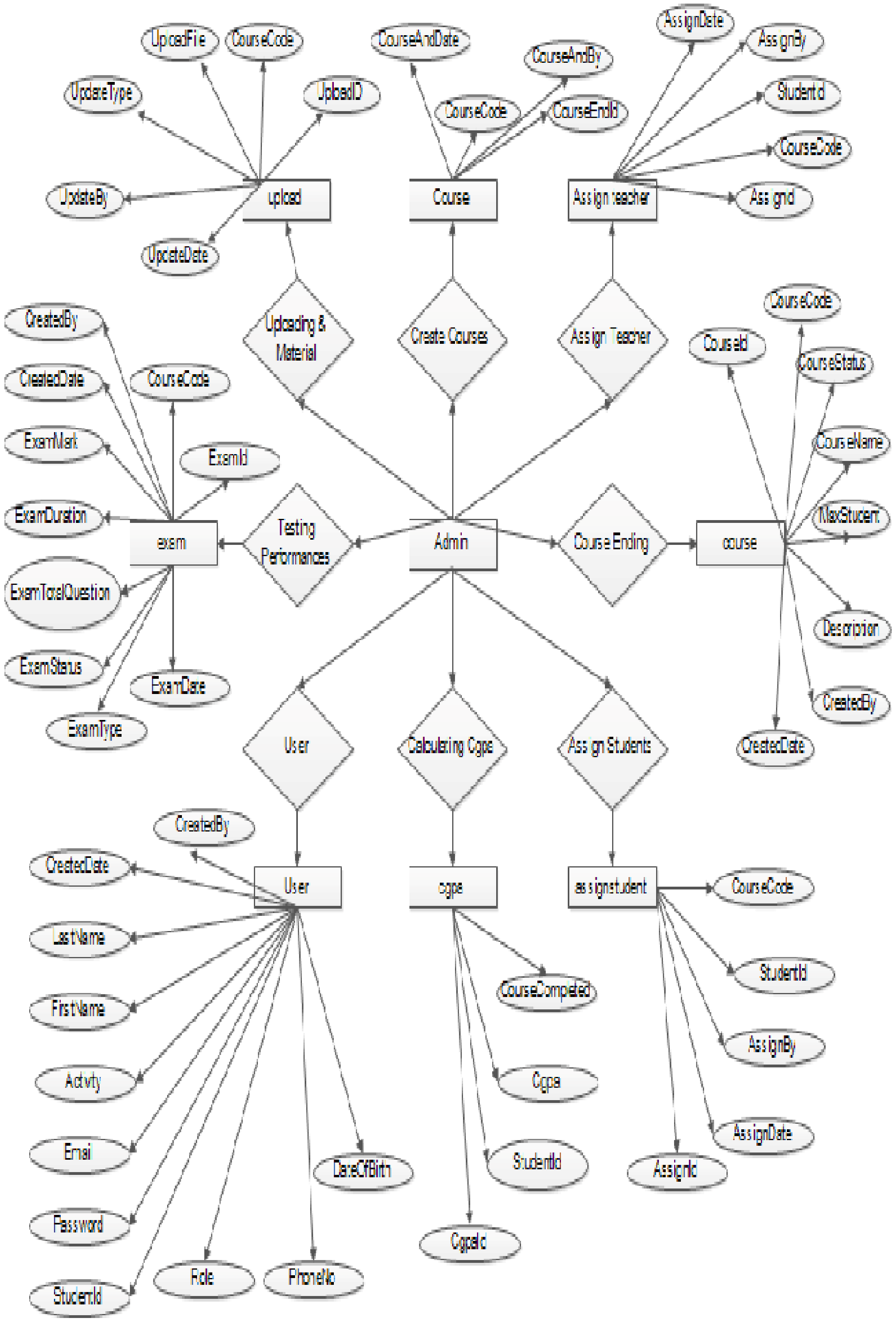
- **Admin Login-** Here student can login to work on various sites or to create their stuff.

- **Registeration page-** In this the new comer can register with their name to get start with their work and making learning easy.

- **Home page of Study Portal-** In this module, after login student can use the various sites.

- **Various Websites-** In this module, the site is with various websites were student can prepare using other sites also.

• ER Diagram:



UploadFile
CourseCode
UpdateType
UploadID
UpdateBy
upload
UpdateDate
CourseAndDate
CourseAndBy
CourseCode
CourseEndId
Course
AssignDate
AssignBy
StudentId
CourseCode
AssignId
Assign teacher
Uploading & Material
Create Courses
Assign Teacher
CreatedBy
CreatedDate
CourseCode
ExamMark
ExamId
ExamDuration
exam
Testing Performances
Admin
Course Ending
course
CourseCode
CourseId
CourseStatus
CourseName
MaxStudent
Description
CreatedBy
CreatedDate
ExamTotalQuestion
ExamStatus
ExamDate
ExamType
User
Calculating Cgpa
Assign Students
CreatedBy
CreatedDate
User
cgpa
assignstudent
CourseCode
LastName
FirstName
CourseCompleted
StudentId
Activity
Cgpa
AssignBy
Email
DateOfBirth
StudentId
AssignDate
AssignId
Password
CgpaId
Role
PhoneNo
StudentId

5. Tools and Languages

5.1 HTML5

HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and last major HTML version that is a World Wide Web Consortium (W3C) recommendation. The current specification is known as the HTML Living Standard. It is maintained by the Web Hypertext Application Technology Working Group (WHATWG), a consortium of the major browser vendors (Apple, Google, Mozilla, and Microsoft).

HTML5 was first released in a public-facing form on 22 January 2008,[2] with a major update and "W3C Recommendation" status in October 2014. Its goals were to improve the language with support for the latest multimedia and other new features; to keep the language both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc., without XHTML's rigidity; and to remain backward-compatible with older software. HTML5 is intended to subsume not only HTML 4 but also XHTML 1 and DOM Level 2 HTML

HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves, and rationalizes the markup available for documents and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a candidate for cross-platform mobile applications because it includes features designed with low-powered devices in mind.

HTML5 is designed, as much as possible, to be backward compatible with existing web browsers. Its new features have been built on existing features and allow you to provide fallback content for older browsers.

5.2 CSS3

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. CSS3 is a latest standard of css earlier versions(CSS2). The main difference between css2 and css3 is follows −

- Media Queries
- Namespaces
- Selectors Level 3
- Color

CSS3 is the latest version of the CSS specification. CSS3 adds several new styling features and improvements to enhance the web presentation capabilities.

CSS3 is collaboration of CSS2 specifications and new specifications, we can called this collaboration is **module**. Some of the modules are shown below −

- Selectors
- Box Model
- Backgrounds
- Image Values and Replaced Content
- Text Effects
- 2D Transformations
- 3D Transformations
- Animations
- Multiple Column Layout
- User Interface

5.3 Bootstrap

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

Bootstrap is an HTML, CSS & JS Library that focuses on simplifying the development of informative web pages (as opposed to web apps). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browsers. In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight.

Bootstrap also comes with several JavaScript components in the form of jQuery plugins. They provide additional user interface elements such as dialog boxes, tooltips, and carousels. Each Bootstrap component consists of an HTML structure, CSS declarations, and in some cases accompanying JavaScript code. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields.

Example of a webpage using Bootstrap framework rendered in Firefox

The most prominent components of Bootstrap are its layout components, as they affect an entire web page. The basic layout component is called "Container", as every other element in the page is placed in it. Developers can choose between a fixed-width container and a fluid-width container.

➢ **Applications of Bootstrap-**

- Scaffolding − Bootstrap provides a basic structure with Grid System, link styles, and background. This is is covered in detail in the section Bootstrap Basic Structure
- CSS − Bootstrap comes with the feature of global CSS settings, fundamental HTML elements styled and enhanced with extensible classes, and an advanced grid system. This is covered in detail in the section Bootstrap with CSS.
- Components − Bootstrap contains over a dozen reusable components built to provide iconography, dropdowns, navigation, alerts, pop-overs, and much more. This is covered in detail in the section Layout Components.
- JavaScript Plugins − Bootstrap contains over a dozen custom jQuery plugins. You can easily include them all, or one by one. This is covered in details in the section Bootstrap Plugins.
- Customize − You can customize Bootstrap's components, LESS variables, and jQuery plugins to get your very own version.

➢ Why to Learn Bootstrap?

- Mobile first approach − Bootstrap 3, framework consists of Mobile first styles throughout the entire library instead them of in separate files.
- Browser Support − It is supported by all popular browsers.
- Easy to get started − With just the knowledge of HTML and CSS anyone can get started with Bootstrap. Also the Bootstrap official site has a good documentation.
- Responsive design − Bootstrap's responsive CSS adjusts to Desktops, Tablets and Mobiles. More about the responsive design is in the chapter Bootstrap Responsive Design.
- Provides a clean and uniform solution for building an interface for developers.
- It also provides web based customization.
- And best of all it is an open source.

5.4 DJANGO

Django is a web development framework that assists in building and maintaining quality web applications. Django helps eliminate repetitive tasks making the development process an easy and time saving experience.

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Django makes it easier to build better web apps quickly and with less code.Django is a registered trademark of the Django Software Foundation, and is licensed under BSD License.

Django comes with the following design philosophies −

- Loosely Coupled − Django aims to make each element of its stack independent of the others.
- Less Coding − Less code so in turn a quick development.
- Don't Repeat Yourself (DRY) − Everything should be developed only in exactly one place instead of repeating it again and again.
- Fast Development − Django's philosophy is to do all it can to facilitate hyper-fast development.
- Clean Design − Django strictly maintains a clean design throughout its own code and makes it easy to follow best web-development practices.

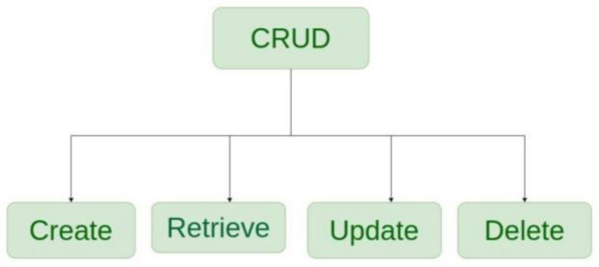
➢ Advantages of Django

Here are few advantages of using Django which can be listed out here –

- Object-Relational Mapping (ORM) Support − Django provides a bridge between the data model and the database engine, and supports a large set of database systems including MySQL, Oracle, Postgres, etc. Django also supports NoSQL database through Django-nonrel fork. For now, the only NoSQL databases supported are MongoDB and google app engine.
- Multilingual Support − Django supports multilingual websites through its built-in internationalization system. So you can develop your website, which would support multiple languages.
- Framework Support − Django has built-in support for Ajax, RSS, Caching and various other frameworks.
- Administration GUI − Django provides a nice ready-to-use user interface for administrative activities.
- Development Environment − Django comes with a lightweight web server to facilitate end-to-end application development and testing.

Django CRUD-

Django is a Python-based web framework which allows you to quickly create web application without all of the installation or dependency problems that you normally will find with other frameworks. Django is based on MVT (Model View Template) architecture and revolves around CRUD (Create, Retrieve, Update, Delete) operations. CRUD can be best explained as an approach to building a Django web application. In general CRUD means performing Create, Retrieve, Update and Delete operations on a table in a database. Let's discuss what actually CRUD means,



- **Create** – create or add new entries in a table in the database.
- **Retrieve** – read, retrieve, search, or view existing entries as a list(List View) or retrieve a particular entry in detail (Detail View).
- **Update** – update or edit existing entries in a table in the database.
- **Delete** – delete, deactivate, or remove existing entries in a table in the database.

5.5 Database SQLite

SQLite is a self-contained, file-based SQL database. SQLite comes bundled with Python and can be used in any of your Python applications without having to install any additional software

SQLite is a C library that provides a lightweight disk-based database that doesn't require a separate server process and allows accessing the database using a nonstandard variant of the SQL query language. Some applications can use SQLite for internal data storage. It's also possible to prototype an application using SQLite and then port the code to a larger database such as PostgreSQL

SQLite is an open-source and simple database engine that allows you to create a relational database and interact with it. In general, it is very lightweight and can be used within almost all programming languages including Python.

Why SQLite?

- Simple: SQLite does not require any "setup" process and there is no need to start, stop, or configure any server to work on.
- Concurrency: it gives the ability to execute multiple queries or access multiple database files simultaneously in a single connection.
- Reliability: it can face any maliciously designed database files and **SQL** strings.
- Control: the content can be accessed and updated using powerful **SQL** queries.

- Scalability: SQLite is scalable, as long as you don't need it for multi-user in high availability cases, and I don't recommend using it in production with big data.

Data Types Available in SQLite for Python-

SQLite for Python offers fewer data types than other SQL implementations. This can be a bit restricting. However, as you'll see, SQLite makes a lot of other things easier. Let's take a quick look at the data types that are available:

- **NULL** — Includes a NULL value
- **INTEGER** — Includes an integer
- **REAL** — Includes a floating-point (decimal) value
- **TEXT**. — Includes text
- **BLOB**. — Includes a binary large object that is stored exactly as input

From this list, you may notice a number of missing data types such as dates. Unfortunately, when using SQLite, you're restricted to these data types.

- ➢ SQLite is a lightweight database that can provide a relational database management system with zero-configuration because there is no need to configure or set up anything to use it.

- ➢ SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. It is a database, which is zero-configured, which means like other databases you do not need to configure it in your system.
- ➢ SQLite engine is not a standalone process like other databases, you can link it statically or dynamically as per your requirement with your application. SQLite accesses its storage files directly.

6. Coding and Implementation

6.1 Introduction of Coding

At the most basic level, programming is a broader discipline whereas coding is a narrower one. Coding involves writing many lines of code in order to create a software program Programming involves not only coding but also other tasks. such as analytical and implementing algorithms, understanding data structures, solving problems, and more. Programmers are typically technically-minded and have strong analytical skills. To put it simply, all programmers are coders but not all coders are programmers. Some experienced programmers use the word "coder" as jargon that refers to a beginner (junior) software developer.

Approaches Used

Top-down and bottom-up approach are strategies of information processing and knowledge ordering. A top-down approach is essentially breaking down a system to gain insight into its subsystems. In a top-down approach an overview of the system is first formulated, specifying but not detailing any first-level subsystems. Each subsystem is then refined in yet greater detail, sometimes in many additional subsystem levels, until the entire specification is reduced to base elements.

A bottom-up approach is essentially piecing together systems to give rise to grander systems, thus making the original systems sub-systems of the emergent system. In a bottom-up approach the individual base elements of the system are first specified in great detail. These elements are then linked together to form larger subsystems, which then in turn are linked,sometimes in many levels, until a complete top-level system is formed.

6.2 Modules and Explanation

Modules:

Overall description of elegant kichha:

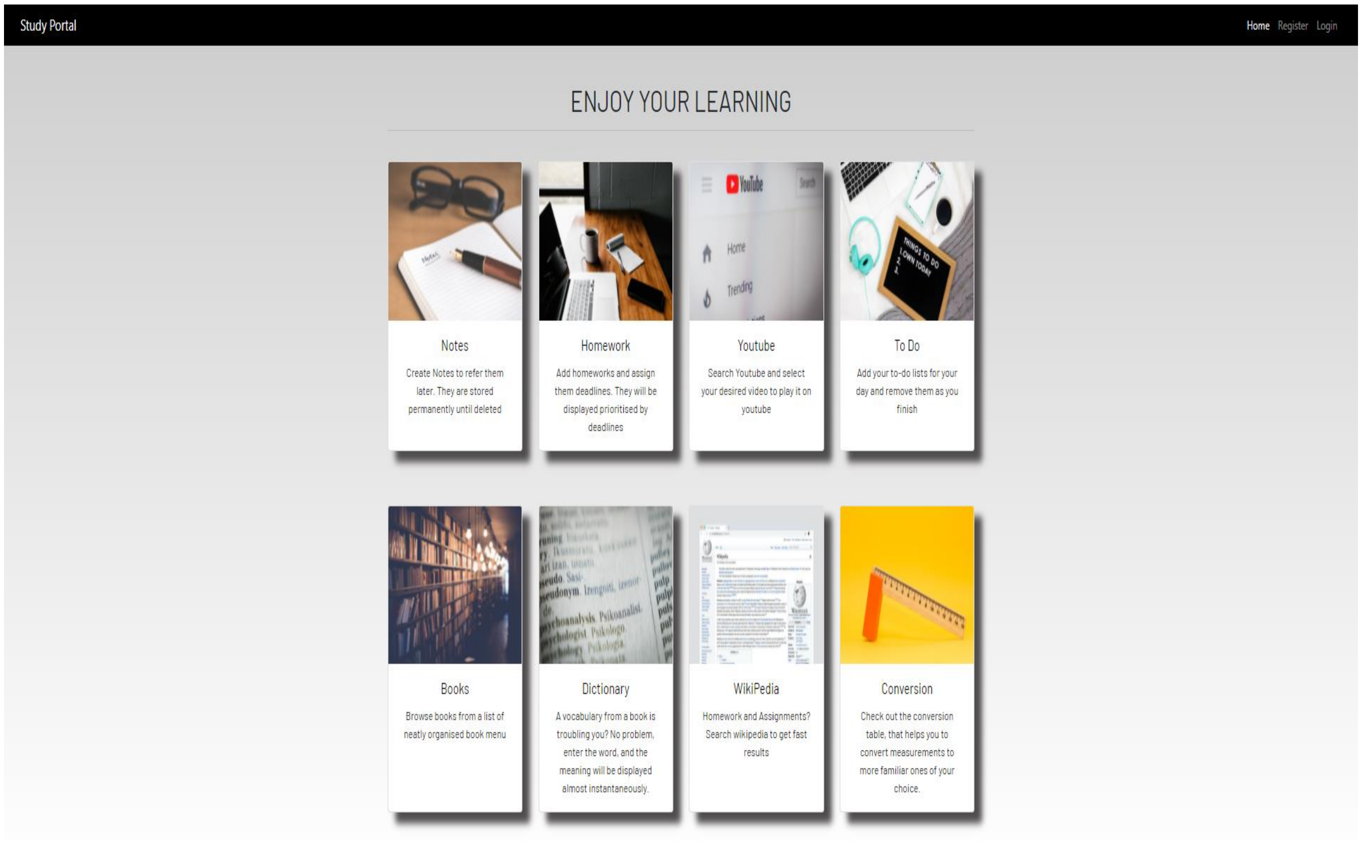
- Home Page

- Register Page
- Login Page
- Profile Page
- Logout Page
- Notes
- Homework
- Youtube
- To Do list
- Books
- Dictionary
- Wikipedia
- Conversion

Modules Description:

6.3 Home Page

- In this, the first page or welcome page of the study portal, here they will see about the sites and dashboards, but to work on them the student must be registered and then can login with their desired username and password.



ENJOY YOUR LEARNING

Notes
Create Notes to refer them later. They are stored permanently until deleted

Homework
Add homeworks and assign them deadlines. They will be displayed prioritised by deadlines

Youtube
Search Youtube and select your desired video to play it on youtube

To Do
Add your to-do lists for your day and remove them as you finish

Books
Browse books from a list of neatly organised book menu

Dictionary
A vocabulary from a book is troubling you? No problem, enter the word, and the meaning will be displayed almost instantaneously.

WikiPedia
Homework and Assignments? Search wikipedia to get fast results

Conversion
Check out the conversion table, that helps you to convert measurements to more familiar ones of your choice.

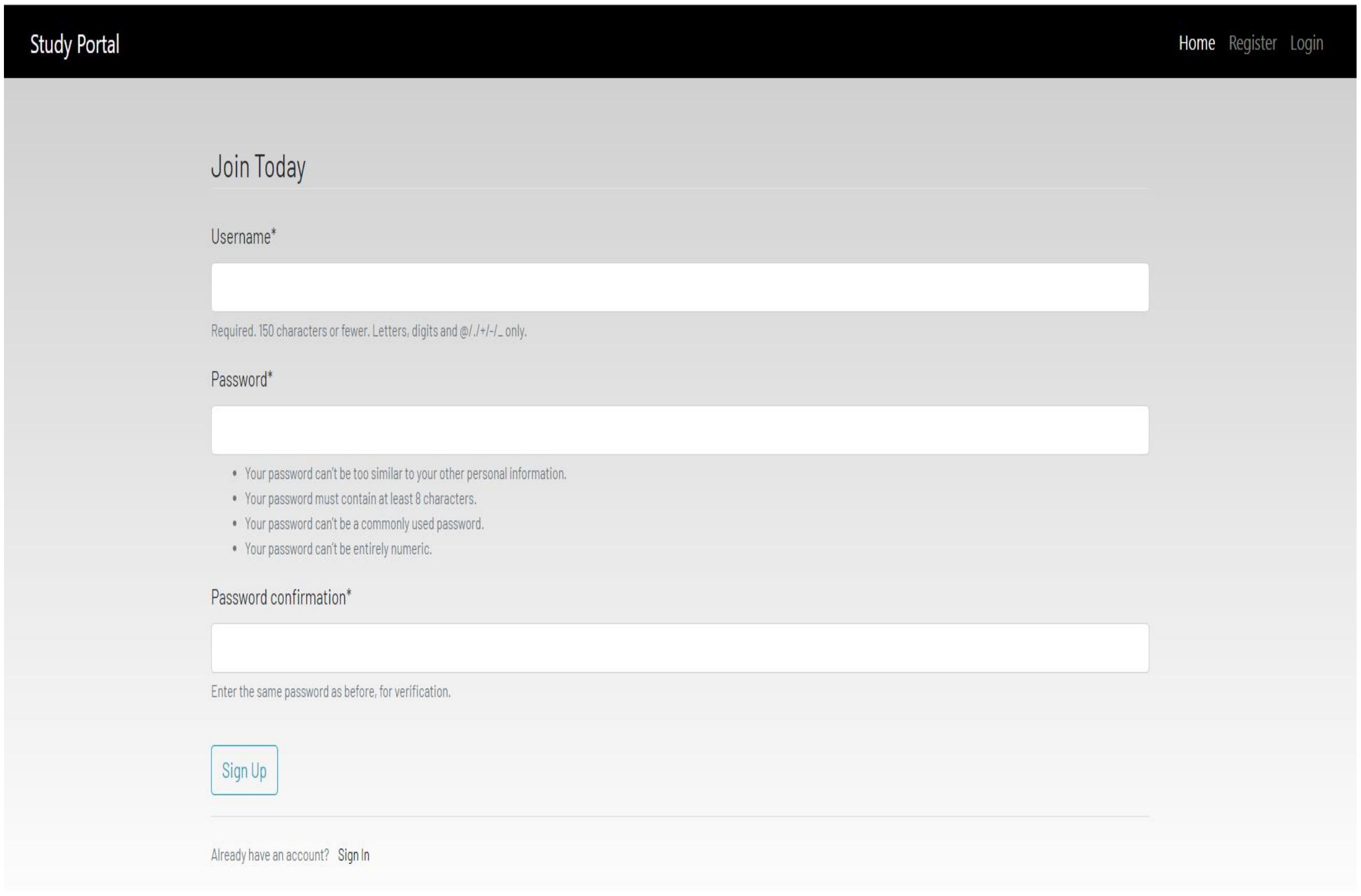
➢ Code of Home page-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% block content %}

<section class="container text-center">
  <h1>ENJOY  YOUR  LEARNING</h1>

  <hr> <br>
  <div class="row">
    <div class="col-md-3">
      <a href="{% url 'notes' %}">
        <div class="card">
          <img class="card-img-top" src="{% static 'images/notes.jpg' %}" alt="Notes image">
          <div class="card-body">
            <h5 class="card-title ">
              Notes
            </h5>
            Create Notes to refer them later. They are stored permanently until deleted
          </div>
        </div>
      </a>
    </div>
    <div class="col-md-3">
      <a href="{% url 'homework' %}">
        <div class="card">
          <img class="card-img-top" src="{% static 'images/homework.jpg' %}" alt="Notes image">
          <div class="card-body">
            <h5 class="card-title ">
              Homework
            </h5>
            Add homeworks and assign them deadlines. They will be displayed prioritised by deadlines
          </div>
        </div>
      </a>
    </div>
    <div class="col-md-3">
      <a href="{% url 'youtube' %}">
        <div class="card">
          <img class="card-img-top" src="{% static 'images/youtube.jpg' %}" alt="Notes image">
          <div class="card-body">
            <h5 class="card-title ">
              Youtube
            </h5>
            Search Youtube and select your desired video to play it on youtube
          </div>
        </div>
      </a>
    </div>
    <div class="col-md-3">
      <a href="{% url 'todo' %}">
        <div class="card">
          <img class="card-img-top" src="{% static 'images/todo.jpg' %}" alt="Notes image">
          <div class="card-body">
            <h5 class="card-title ">
              To Do
            </h5>
            Add your to-do lists for your day and remove them as you finish
          </div>
        </div>
      </a>
    </div>
  </div>
  <br><br><br>
  <div class="row">
    <div class="col-md-3 ">
      <a href="{% url 'books' %}">
        <div class="card mt-20">
          <img class="card-img-top" src="{% static 'images/books.jpg' %}" alt="Notes image">
          <div class="card-body">
            <h5 class="card-title ">








  Books
            </h5>
            Browse books from a list of neatly organised book menu
          </div>
        </div>
      </a>
    </div>



    <div class="col-md-3">
      <a href="{% url 'dictionary' %}">
        <div class="card">
          <img class="card-img-top" src="{% static 'images/dictionary.jpg' %}" alt="Notes image">
          <div class="card-body">
            <h5 class="card-title ">
              Dictionary
            </h5>
            A vocabulary from a book is troubling you? No problem, enter the word, and the meaning will be displayed
            almost instantaneously.
          </div>
        </div>
      </a>
    </div>
    <div class="col-md-3">
      <a href="{% url 'wiki' %}">
        <div class="card">
          <img class="card-img-top" src="{% static 'images/wiki.jpg' %}" alt="Notes image">
          <div class="card-body">
            <h5 class="card-title ">
              WikiPedia
            </h5>
            Homework and Assignments? Search wikipedia to get fast results
          </div>
        </div>
      </a>
    </div>
    <div class="col-md-3">
      <a href="{% url 'conversion' %}">
        <div class="card">
          <img class="card-img-top" src="{% static 'images/conversion.jpg' %}" alt="Notes image">
          <div class="card-body">
```

```
            <h5 class="card-title ">
              Conversion
            </h5>
            Check out the conversion table, that helps you to convert measurements to more familiar ones of your choi
</div>
        </div>
      </a>
    </div>
  </div>
</section>
{% endblock content %}
```

6.4 Register Page

- In this page the students can register themselves by their name and making a strong password, which will help them to make their profile and login to the site through which they can create their stuff and make their learning easy.



Join Today

Username*

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password*

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation*

Enter the same password as before, for verification.

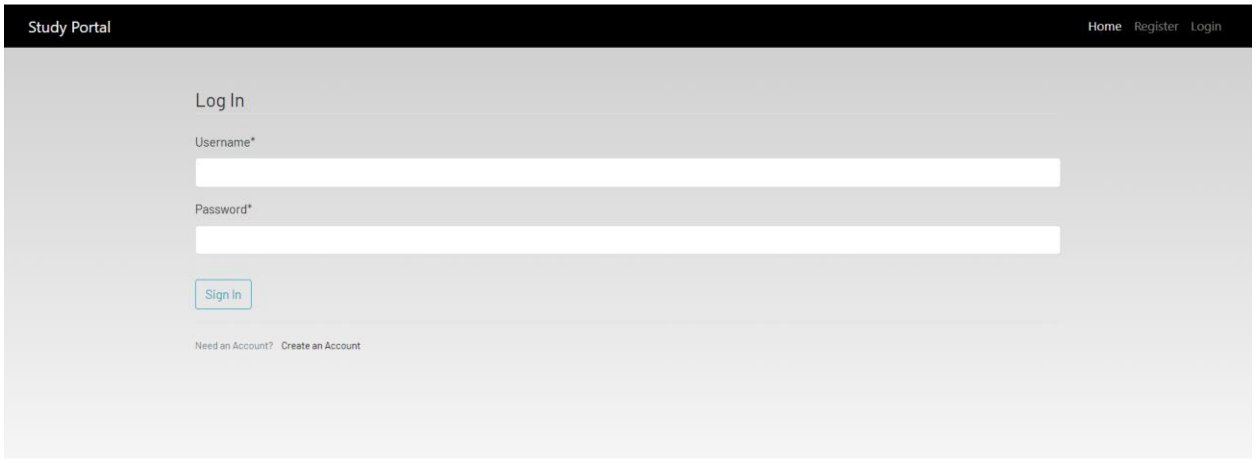
Already have an account? Sign In

➢ Code of the Register Page-

```
{% extends 'dashboard/base.html' %}

{% load static %}
{% load crispy_forms_tags %}
{% block content %}
<div class="container">
  <form method="POST">
    {% csrf_token %}
    <fieldset class="form-group">
      <legend class="border-bottom mb-4">Join Today</legend>
      {{form|crispy}}
    </fieldset>
    <div class="form-group">
      <button href="" class="btn btn-outline-info" type="submit">
        Sign Up
      </button>
    </div>
  </form>
  <div class="border-top pt-3">
    <small class="text-muted">Already have an account?
      <a class="ml-2" href="{% url 'login' %}">Sign In</a>
    </small>
  </div>
</div>
{% endblock content %}
```

6.5 Login Page

- In this page student can login their particular profile to connect with their stuff, and all their stuff by searching and learning.



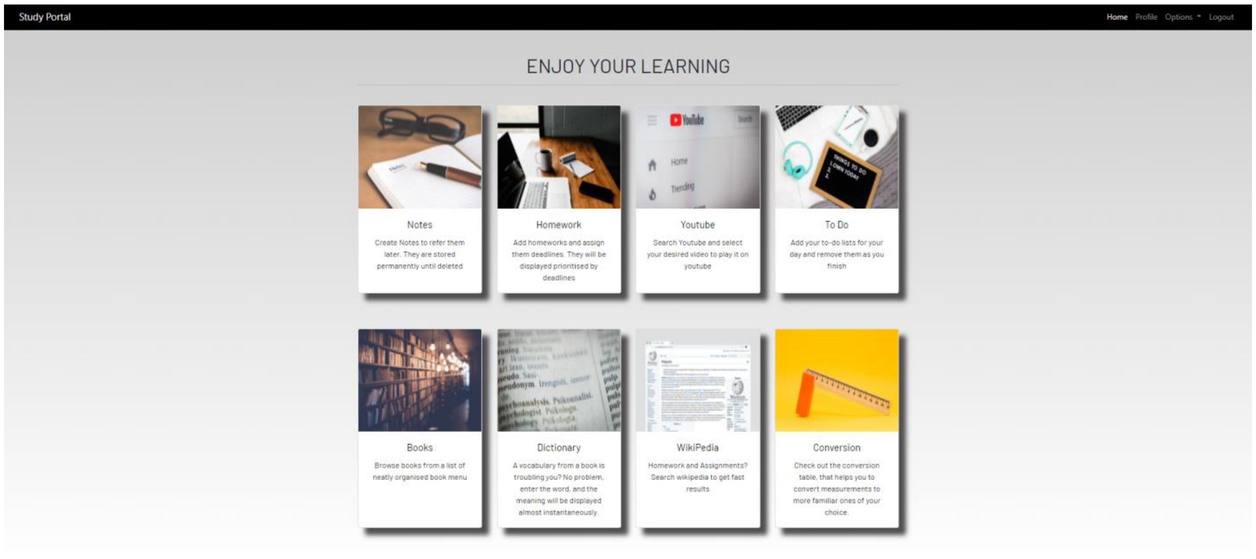
➢ Code of the Login Page-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% load crispy_forms_tags %}
{% block content %}
<div class="container">
  <form method="POST">
    {% csrf_token %}
    <fieldset class="form-group">
      <legend class="border-bottom mb-4">Log In</legend>
       {{form|crispy}}
    </fieldset>
    <div class="form-group">
      <button href="" class="btn btn-outline-info" type="submit">
        Sign In
      </button>
    </div>
  </form>
```

```
<div class="border-top pt-3">
    <small class="text-muted">Need an Account?
      <a class="ml-2" href="{% url 'register' %}">Create an Account</a>
    </small>
  </div>
</div>
{% endblock content %}
```

6.6 Profile Page

- After login to the site, student will again get the home page with their profile and their stuff. They can continue with their work in their profile by knowing about their current list and activities.



ENJOY YOUR LEARNING

Notes
Create Notes to refer them later. They are stored permanently until deleted

Homework
Add homeworks and assign them deadlines. They will be displayed prioritised by deadlines

Youtube
Search Youtube and select your desired video to play it on youtube

To Do
Add your to-do lists for your day and remove them as you finish

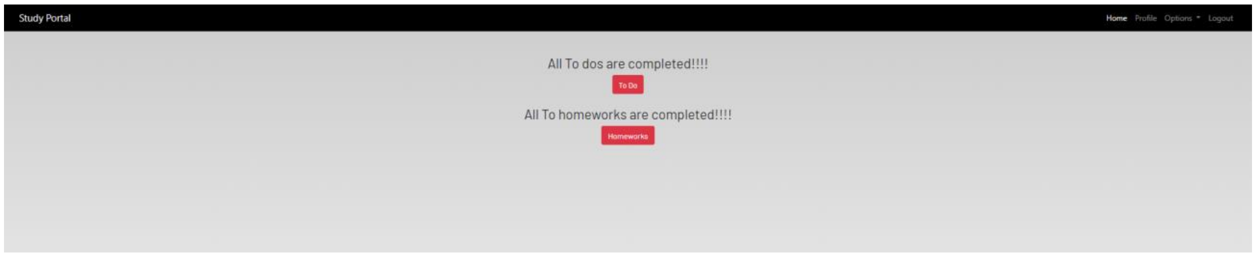
Books
Browse books from a list of neatly organised book menu

Dictionary
A vocabulary from a book is troubling you? No problem, enter the word, and the meaning will be displayed almost instantaneously.

WikiPedia
Homework and Assignments? Search wikipedia to get fast results

Conversion
Check out the conversion table, that helps you to convert measurements to more familiar ones of your choice.

- All pendind work will be shown here or else comleted if there is no work pending.



All To dos are completed!!!!

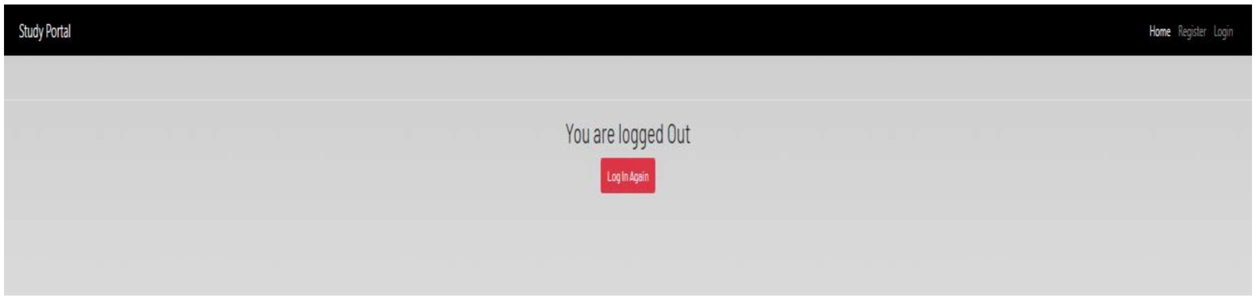
All To homeworks are completed!!!!

➢ Code of the Profile Page-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% load crispy_forms_tags%}
{% block content %}
<section class="text-center">
    <div class="container">
      {% if not todos_done %}
        <h2>Due ToDos</h2>
        <table class="table table-striped table-bordered table-hover table-dark">
            <thead>
                <tr>
                    <th scope="col">#</th>
                    <th scope="col">Title</th>
                    <th scope="col">Status</th>
                    <th scope="col">Delete</th>
                </tr>
            </thead>
            <tbody>
                {% for todo in todos %}
                <tr>
                    <th scope="row">{{forloop.counter}}</th>
                    <td>{{todo.title}}</td>
                    <td>
                        <a href="{% url 'update-todo' todo.id %}">
                            <div class="form-check">
                                {% if todo.is_finished == True %}
                                <input class="form-check-input" type="checkbox" value="" checked>
                                {% else %}
                                <input class="form-check-input" type="checkbox" value="">
                                {% endif %}
                                <label class="form-check-label text-light" for="flexCheckDefault">
                                    Mark as Completed
                                </label>
                            </div>
                        </a>
                    </td>

                    <td><a href="{% url 'delete-todo' todo.id %}"><i class="fa fa-trash fa-2x"></i></a></td>
                </tr>
                {% endfor %}
            </tbody>
        </table>







{% else %}
        <h3>All To dos are completed!!!!</h3>
    {% endif %}
        <a href="{% url 'todo'%}" class="btn btn-danger">To Do</a>
        <br><br>

        {% if not homework_done %}
        <h2>Due Homeworks</h2>
        <table class="table table-striped table-bordered table-hover table-dark">
            <thead>
                <tr>
                    <th scope="col">#</th>
                    <th scope="col">Subject</th>
                    <th scope="col">Title</th>
                    <th scope="col">Description</th>
                    <th scope="col">Due</th>
                    <th scope="col">Status</th>
                    <th scope="col">Delete</th>
                </tr>
            </thead>
            <tbody>
                {% for homework in homeworks %}
                <tr>
                    <th scope="row">{{forloop.counter}}</th>
                    <td>.{{homework.subject}}</td>
                    <td>{{homework.title}}</td>
                    <td>{{homework.description}}</td>
                    <td>{{homework.due}}</td>  <td>
                        <a href="{% url 'update-homework' homework.id %}">
                            <div class="form-check">
                                {% if homework.is_finished == True %}
                                <input class="form-check-input" type="checkbox" value="" checked>
                                {% else %}
                                <input class="form-check-input" type="checkbox" value="">
                                {% endif %}
                                <label class="form-check-label text-light" for="flexCheckDefault">
                                    Mark as Completed
                                </label>
                        </div>
                    </a>
                </td>
                <td><a href="{% url 'delete-homework' homework.id %}"><i class="fa fa-trash fa-
2x"></i></a></td>
            </tr>




   {% endfor %}
            </tbody>
        </table>
        {% else %}
        <h3>All To homeworks are completed!!!!</h3>
        {% endif %}
        <a href="{% url 'homework' %}" class="btn btn-danger">Homeworks</a>
    </div>
</section>
{% endblock content %}
```

6.7 Logout Page

- After completing their stuff they can save all their documents and logout by which none other can check out for all their activities or documents they worked in. For visiting again they have to login again to their respective id.

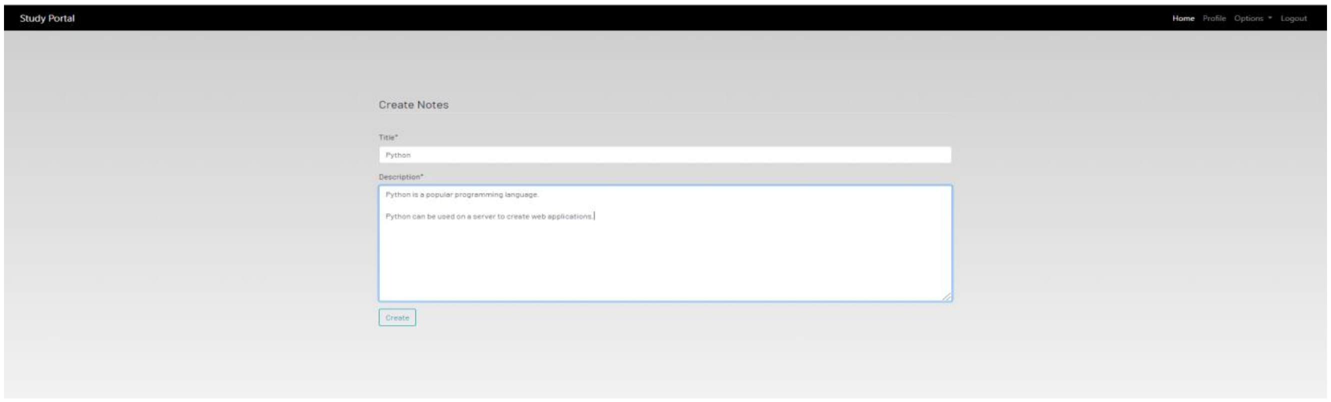


➢ **Code of the Logout Page-**

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% block content %}
<div class="border-top pt-3 text-center">
  <h2>You are logged Out</h2>

  <small class="text-muted"
    ><a class="btn btn-danger" href="{% url 'login' %}">Log In Again</a>
  </small>
</div>
{% endblock content %}
```

6.8 NOTES

- In this page students can create their notes topicwise or subjectwise. Here they can write all the stuff regarding topic or subject in description.



Create Notes

Title*

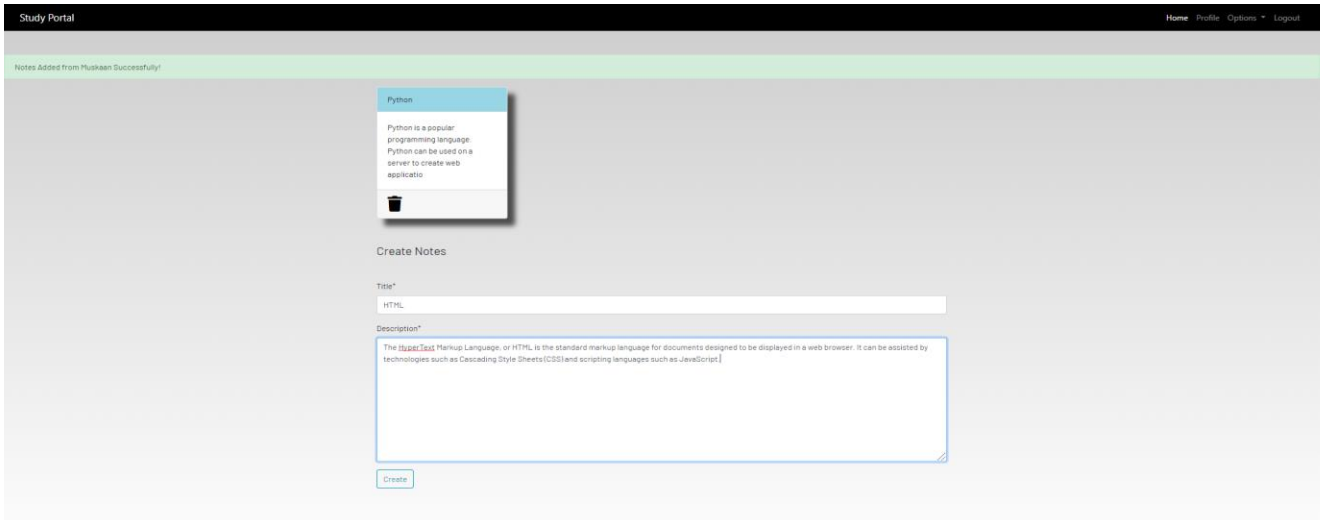
Python

Description*

Python is a popular programming language.

Python can be used on a server to create web applications.

- After creating they will get their notes and other data procedure for their next working process.



Notes Added from Muskaan Successfully!

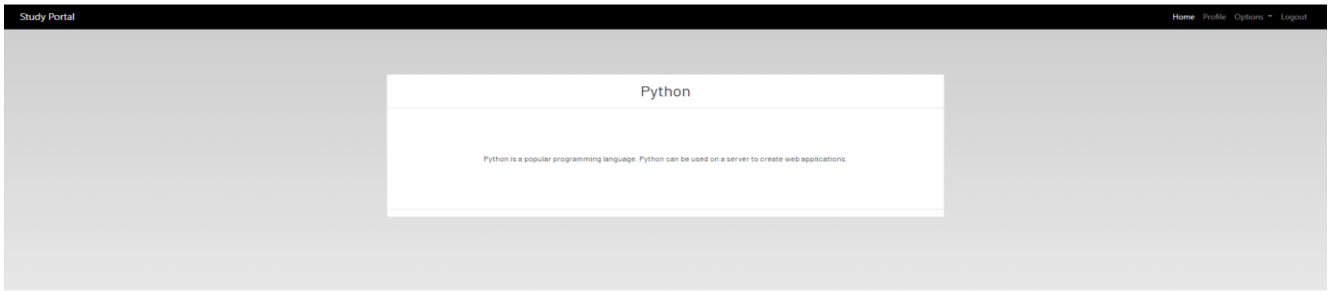
Python

Python is a popular programming language. Python can be used on a server to create web applicatio

HTML

The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

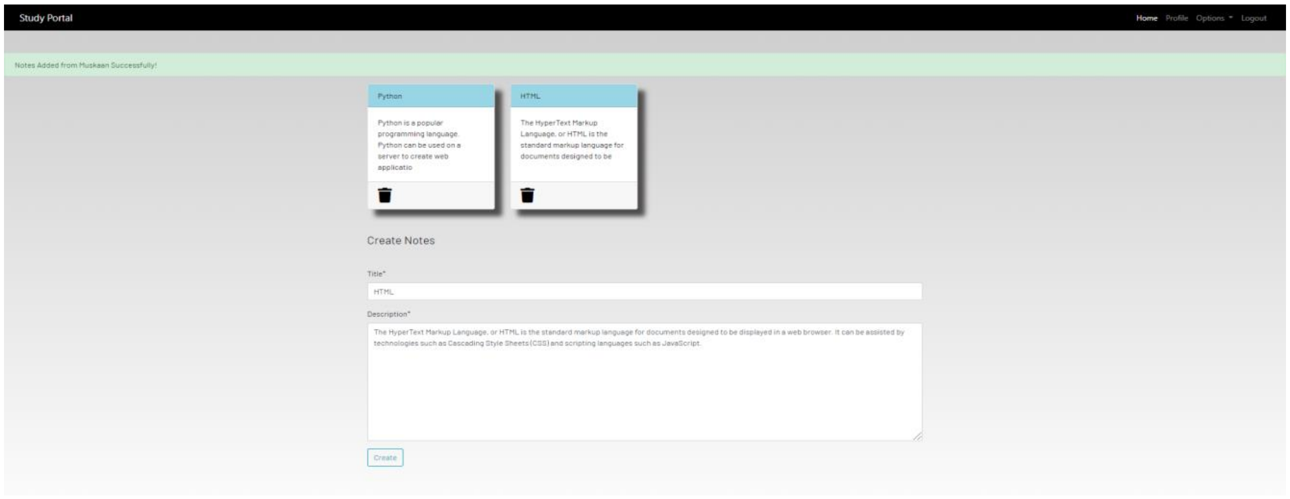
- Their notes will open in another page where they can read their notes after completing it as shown below:



Python

Python is a popular programming language. Python can be used on a server to create web applications.

- After creating other notes they will get the page in such a way and direction for creating further notes.



Notes Added from Muskaan Successfully!

Python

Python is a popular programming language. Python can be used on a server to create web applicatio

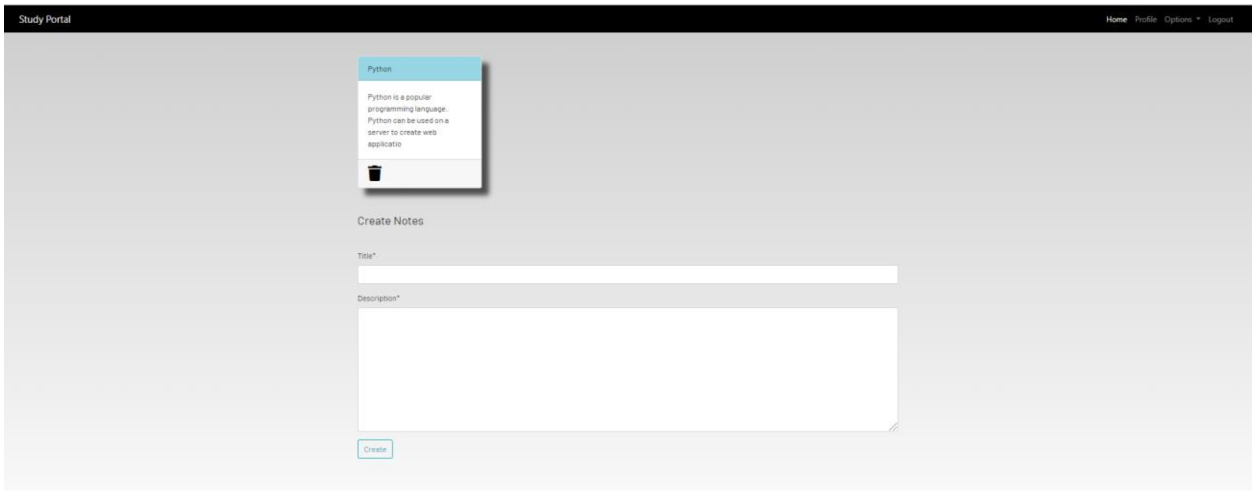
HTML

The HyperText Markup Language, or HTML is the standard markup language for documents designed to be

HTML

The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

- Here they have option for deleting their notes, after deleting the page will be shown:



Python is a popular programming language. Python can be used on a server to create web applicatio

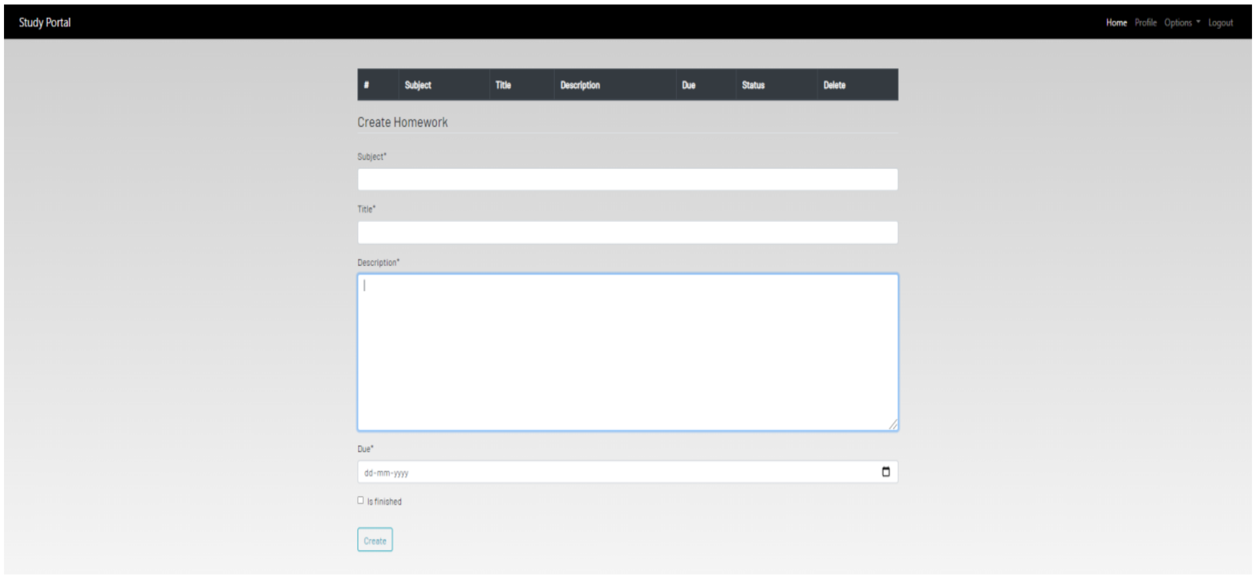
➢ Code of the Notes-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% load crispy_forms_tags %}
{% block content %}

<div class="container">
    <div class="row">
        {% for note in notes %}
        <div class="col-md-3">
            <a href="{% url 'notes-detail' note.id %}">
                <div class="card">
                    <div class="card-header">{{note.title}}</div>
                    <div class="card-body">{{note.description|slice:"0:100"}}</div>
                    <div class="card-footer mt-auto "><a href="{% url 'delete-note' note.id %}">
                        <i class="fa fa-trash fa-2x"></i></a>






</div>
                </div>
            </a>
        </div>
        {% endfor %}
        <br><br>
 </div>
</div>
<br><br>
<div class="container">
    <form method="POST">
```

```
      {% csrf_token %}
        <fieldset class="form-group">
            <legend class="border-bottom mb-4">Create Notes</legend>
        </fieldset>
        {{form|crispy}}
        <div class="form-group">
            <button href="" class="btn btn-outline-info" type="submit">
                Create
            </button>
        </div>
    </form>
</div>
{% endblock content %}
@login_required
def notes(request):
    if request.method == "POST":
        form = NotesForm(request.POST)
        if form.is_valid():
            notes = Notes(user=request.user,title=request.POST['title'],description=request.POST['description'
])
            notes.save()
        messages.success(request,f"Notes Added from {request.user.username} Successfully!")
    else:
        form = NotesForm()
    notes = Notes.objects.filter(user=request.user)
    context= {'notes':notes,'form':form}
    return render(request,'dashboard/notes.html',context)
@login_required
def delete_note(request,pk=None):
    Notes.objects.get(id=pk).delete()
    return redirect("notes")

class NotesDetailView(generic.DetailView):
    model = Notes
```

6.9 HOMEWORK

- In this page, students can list out all their homeworks. They can create their list using subject, title, description, the due date, status of their homework whether they have completed or not.



#	Subject	Title	Description	Due	Status	Delete

Create Homework

Subject*

Title*

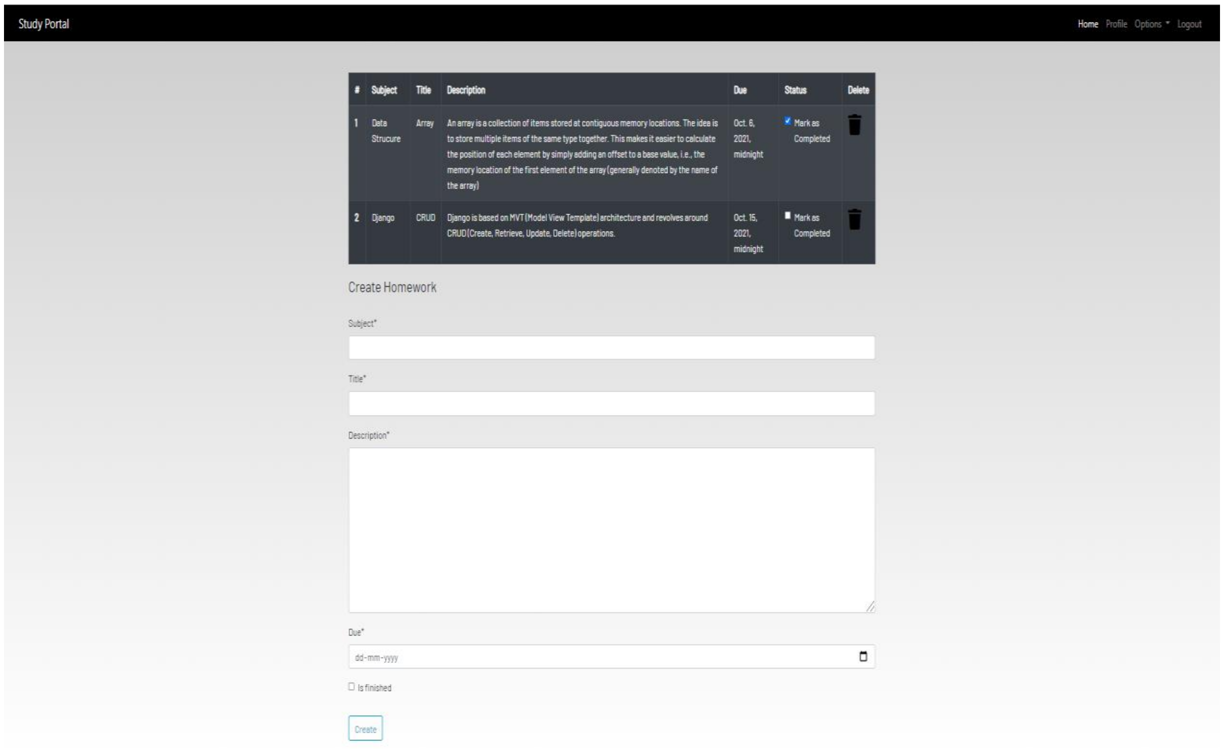
Description*

Due*

dd-mm-yyyy

☐ Is finished

- They can update their status as after completing their task, and their homework schedule will list out in such a manner shown below:



#	Subject	Title	Description	Due	Status	Delete
1	Data Strucure	Array	An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together. This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (generally denoted by the name of the array)	Oct. 6, 2021, midnight	✓ Mark as Completed	
2	Django	CRUD	Django is based on MVT (Model View Template) architecture and revolves around CRUD (Create, Retrieve, Update, Delete) operations.	Oct. 15, 2021, midnight	☐ Mark as Completed	

Create Homework

Subject*

Title*

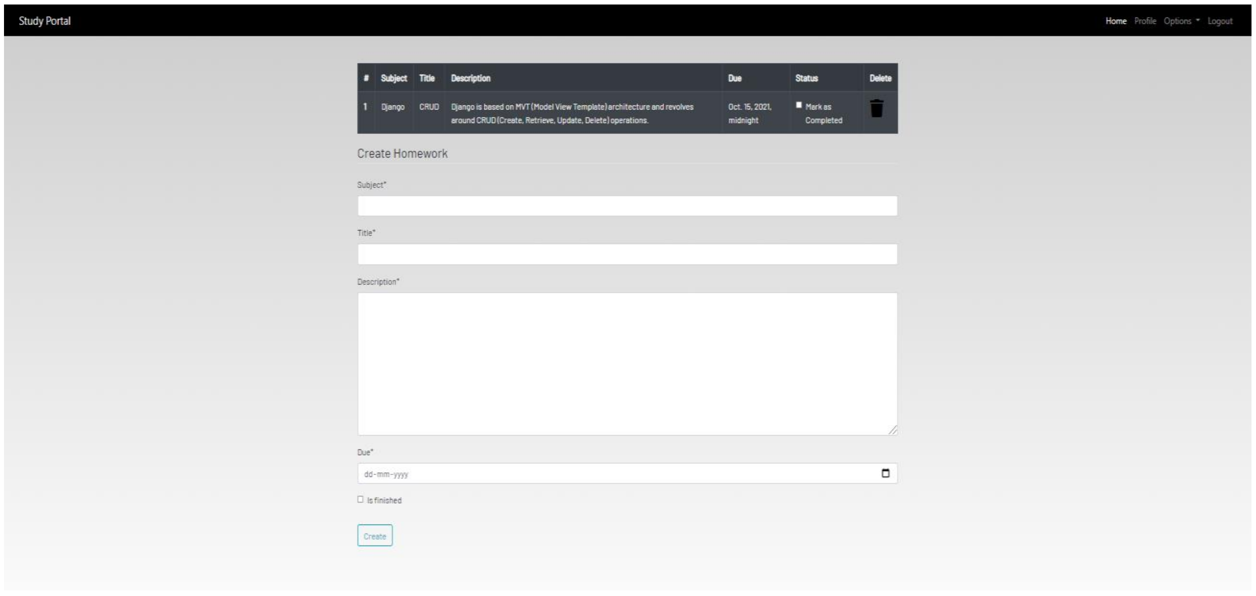
Description*

Due*

dd-mm-yyyy

☐ Is finished

- After all of creating their stuff, they can delete their homeworks.



#	Subject	Title	Description	Due	Status	Delete
1	Django	CRUD	Django is based on MVT (Model View Template) architecture and revolves around CRUD (Create, Retrieve, Update, Delete) operations.	Oct. 15, 2021, midnight	Mark as Completed	

Create Homework

Subject*

Title*

Description*

Due*

dd-mm-yyyy

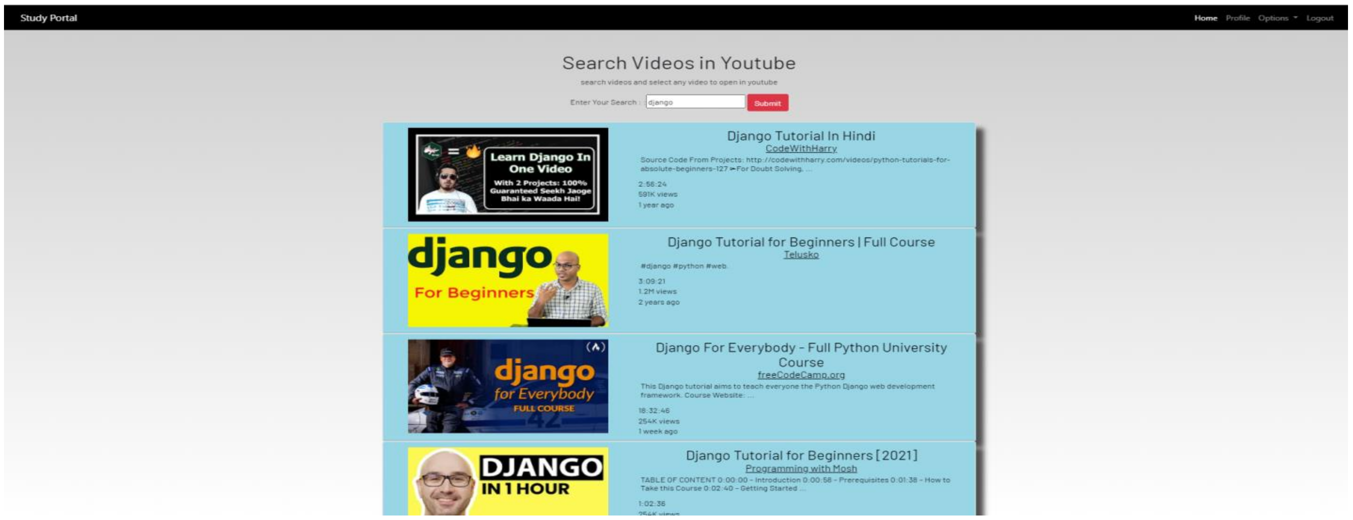
Is finished

➢ Code of the Homework-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% load crispy_forms_tags %}
{% block content %}

<div class="container">
{% if not homework_done %}
    <table class="table table-striped table-bordered table-hover table-dark">
        <thead>
            <tr>
                <th scope="col">#</th>
                <th scope="col">Subject</th>
                <th scope="col">Title</th>
                <th scope="col">Description</th>
                <th scope="col">Due</th>
                <th scope="col">Status</th>
                <th scope="col">Delete</th>
            </tr>
        </thead>
        <tbody>
            {% for homework in homeworks %}
            <tr>
                <th scope="row">{{forloop.counter}}</th>
                <td>{{homework.subject}}</td>
                <td>{{homework.title}}</td>
                <td>{{homework.description}}</td>
                <td>{{homework.due}}</td>
```

```
 <td>
                   <a href="{% url 'update-homework' homework.id %}">
                       <div class="form-check">
                           {% if homework.is_finished == True %}
                           <input class="form-check-input" type="checkbox" value="" checked>
                           {% else %}
                           <input class="form-check-input" type="checkbox" value="" >
                             {% endif %}
                           <label class="form-check-label text-light" for="flexCheckDefault">
                               Mark as Completed
                           </label>
                       </div>
                   </a>
               </td>
               <td><a href="{% url 'delete-homework' homework.id %}"><i class="fa fa-trash fa-2x"></i></a></td>
           </tr>
           {% endfor %}
       </tbody>
    </table>
    {% else %}
    <h4>All homeworks are completed!! Create to have more!!</h4>
    {% endif %}
</div>
<div class=" container">
    <form method="POST">
    {% csrf_token %}
        <fieldset class="form-group">
            <legend class="border-bottom mb-4">Create Homework</legend>
            {{form|crispy}}
        </fieldset>
        <div class="form-group">
            <button href="" class="btn btn-outline-info" type="submit">
                Create
            </button>
        </div>
    </form>
</div>
{% endblock content %}
@login_required
def homework(request):
    if request.method == "POST":
        form = HomeworkForm(request.POST)
        if form.is_valid():
            try:
                finished = request.POST['is_finished']
                if finished == 'on':
```

```
                finished = True
            else:
                finished = False
        except:
            finished = False




 homeworks = Homework(
            user = request.user,
            subject = request.POST['subject'],
            title = request.POST['title'],
            description = request.POST['description'],
            due = request.POST['due'],
            is_finished = finished
        )
        homeworks.save()
        messages.success(request,f'Homework Added from {request.user.username}!!')
    else:
            form = HomeworkForm()
    homework = Homework.objects.filter(user=request.user)
    if len(homework) == 0:
        homework_done = True
    else:
        homework_done =False

    context = {'homeworks':homework,
             'homeworks_done':homework_done,
             'form':form,
            }
    return render(request,'dashboard/homework.html',context)

@login_required
def update_homework(request,pk=None):
    homework = Homework.objects.get(id=pk)
    if homework.is_finished == True:
        homework.is_finished = False
    else:
        homework.is_finished = True
    homework.save()
    return redirect('homework')

@login_required
def delete_homework(request,pk=None):
    Homework.objects.get(id=pk).delete()
    return redirect("homework")
```



Search Videos in Youtube

search videos and select any video to open in youtube

Enter Your Search : django Submit

Django Tutorial In Hindi
CodeWithHarry
Source Code From Projects: http://codewithharry.com/videos/python-tutorials-for-absolute-beginners-127 ➢For Doubt Solving, ...
2:56:24
591K views
1 year ago

Django Tutorial for Beginners | Full Course
Telusko
#django #python #web.
3:09:21
1.2M views
2 years ago

Django For Everybody - Full Python University Course
freeCodeCamp.org
This Django tutorial aims to teach everyone the Python Django web development framework. Course Website: ...
18:32:46
254K views
1 week ago

Django Tutorial for Beginners [2021]
Programming with Mosh
TABLE OF CONTENT 0:00:00 - Introduction 0:00:58 - Prerequisites 0:01:38 - How to Take this Course 0:02:40 - Getting Started ...
1:02:36

6.10 YOUTUBE

- In this, students can easily search for any particular topics, they will top 10 videos of respective search topics. This will help them out to make their topic clear and learning easy and interesting.

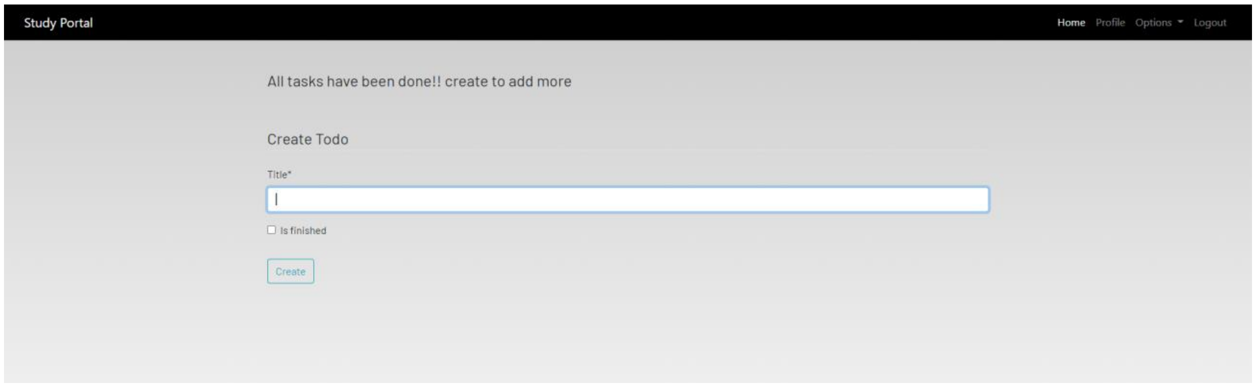
➢ Code of the Youtube-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% block content %}
<section class='text-center container'>
   <h1>Search Videos in Youtube</h1>
   <p>search videos and select any video to open in youtube</p>
   <form action="" method="post">
      {% csrf_token %}
      {{form}}
      <input class="btn btn-danger" type="submit" value="Submit">
   </form><br>
```

```
    {% for result in results %}
    <a href="{{result.link}}" target="_blank">
        <div class="card">
            <div class="card-header">
                <div class="row">
                    <div class="col-md-5">
                        <img class="img-fluid" src="{{result.thumbnail}}" alt="">
                    </div>
                    <div class="col-md-7">
                        <h3 class="p-0 m-0">{{result.title}}</h3>
                      <b>
                            <u>
                                <h5 class="p-0 m-0">{{result.channel}}</h5>
                            </u>




                  </b>
                        <h6 class="p-0 m-1">{{result.description}}</h6>
                        <b>
                            <h6 class="ml-0 mt-3">{{result.duration}}</h6>
                            <h6 class="ml-0 mt-1">{{result.views}}</h6>
                            <h6 class="ml-0 mt-1">{{result.published}}</h6>
                        </b>
                    </div>
                </div>
            </div>
        </div>
    </a>
    {% endfor %}
    <br></section>
{% endblock content %}
def youtube(request):
    if request.method == "POST":
        form = DashboardForm(request.POST)
        text = request.POST['text']
        video = VideosSearch(text,limit=10)
        result_list = []
        for i in video.result()['result']:
            result_dict = {
                 'input':text,
                 'title':i['title'],
              'duration':i['duration'],
                 'thumbnail':i['thumbnails'][0]['url'],
                 'channel':i['channel']['name'],
                 'link':i['link'],
                 'views':i['viewCount']['short'],
                 'published':i['publishedTime']
            }
            desc = ''
            if i['descriptionSnippet']:
                for j in i['descriptionSnippet']:
                    desc += j['text']
            result_dict['description'] = desc
            result_list.append(result_dict)
            context={
                'form':form,
                'results':result_list
            }
        return render(request,'dashboard/youtube.html',context)
    else:
        form = DashboardForm()
    context = {'form':form}
    return render(request,"dashboard/youtube.html",context)
```

6.11 TO DO LIST

- Here, in this page student can list out all their activities or tasks that are needed to be completed.



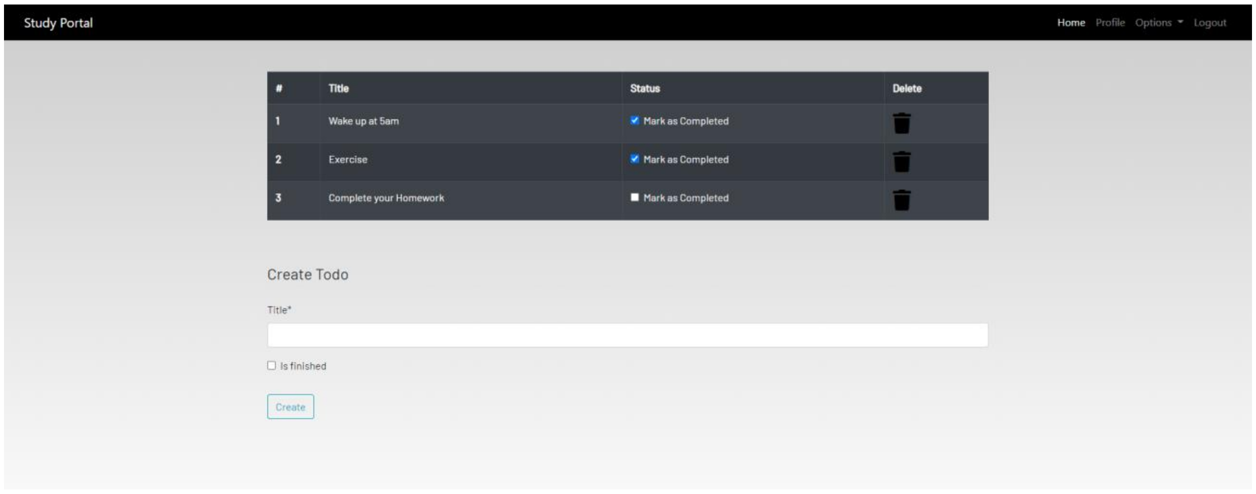
All tasks have been done!! create to add more

Create Todo

Title*

Is finished

- After creating their todo list it will list out in such a way, and after the completion of the tasks it would get updated.


Study Portal

Home Profile Options Logout

#	Title	Status	Delete
1	Wake up at 5am	✓ Mark as Completed	🗑
2	Exercise	✓ Mark as Completed	🗑
3	Complete your Homework	☐ Mark as Completed	🗑

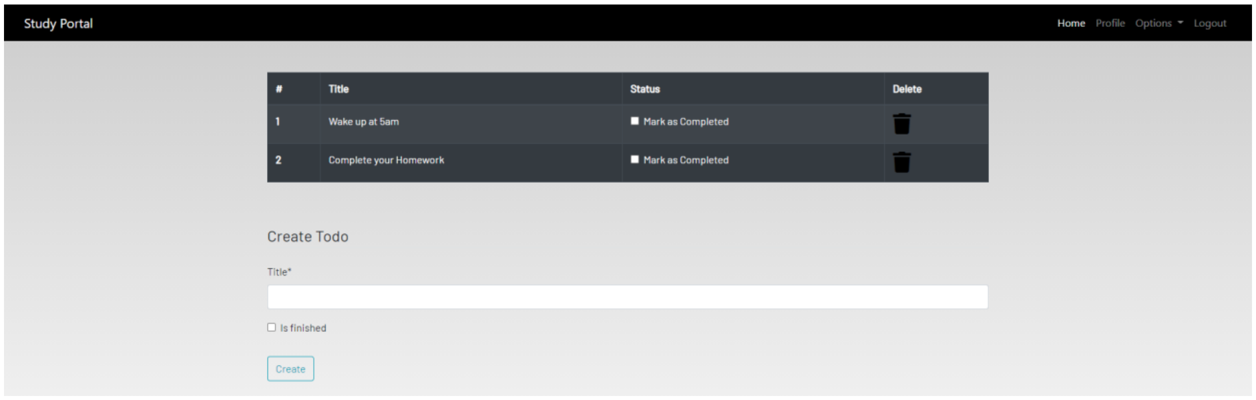
Create Todo

Title*

☐ Is finished

Create

- Here student get the option to delete the particular activity or task.


Study Portal

Home Profile Options Logout

#	Title	Status	Delete
1	Wake up at 5am	☐ Mark as Completed	🗑
2	Complete your Homework	☐ Mark as Completed	🗑

Create Todo

Title*

☐ Is finished

Create

➢ Code of the Todo list-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% load crispy_forms_tags %}
{% block content %}

<div class="container">
    {% if not todos_done %}
   <table class="table table-striped table-bordered table-hover table-dark">
       <thead>
           <tr>
               <th scope="col">#</th>
               <th scope="col">Title</th>
```

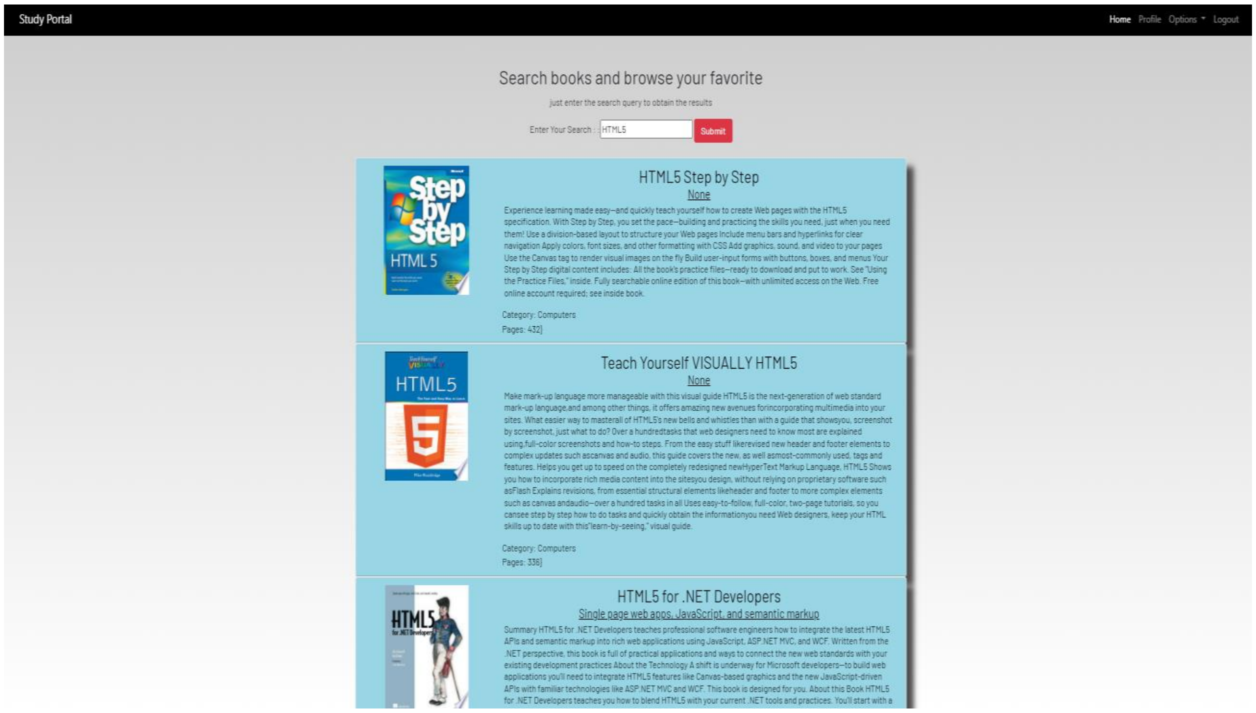
```
            <th scope="col">Status</th>
            <th scope="col">Delete</th>
          </tr>
        </thead>
        <tbody>
          {% for todo in todos %}
          <tr>
            <th scope="row">{{forloop.counter}}</th>
            <td>{{todo.title}}</td>
                        <td>
              <a href="{% url 'update-todo' todo.id %}">
                <div class="form-check">
                  {% if todo.is_finished == True %}
                  <input class="form-check-input" type="checkbox" value="" checked>
                  {% else %}
                  <input class="form-check-input" type="checkbox" value="" >
                  {% endif %}
                  <label class="form-check-label text-light" for="flexCheckDefault">
                    Mark as Completed
                  </label>
                </div>
              </a>
            </td>
            <td><a href="{% url 'delete-todo' todo.id %}"><i class="fa fa-trash fa-2x"></i></a></td>
          </tr>
          {% endfor %}
        </tbody>
    </table>
    {% else %}
    <h4>All tasks have been done!! create to add more</h4>
    {% endif %}
</div><br><br>
<div class="container">
    <form method="POST">
```

```
{% csrf_token %}
        <fieldset class="form-group">
          <legend class="border-bottom mb-4">Create Todo</legend>
          {{form|crispy}}
        </fieldset>
        <div class="form-group">
          <button href="" class="btn btn-outline-info" type="submit">
            Create
          </button>
        </div>
```

```
    </form>
</div>

{% endblock content %}
```

6.12 BOOKS

- Here, students can easily go to page and find any of the book they are willing to read or learn. This page direct them to the google page easily. It make easy for students instead of switching various sites, they can do all the things in a single site.



Search books and browse your favorite

just enter the search query to obtain the results

Enter Your Search : HTML5 Submit

HTML5 Step by Step

None

Experience learning made easy—and quickly teach yourself how to create Web pages with the HTML5 specification. With Step by Step, you set the pace—building and practicing the skills you need, just when you need them! Use a division-based layout to structure your Web pages Include menu bars and hyperlinks for clear navigation Apply colors, font sizes, and other formatting with CSS Add graphics, sound, and video to your pages Use the Canvas tag to render visual images on the fly Build user-input forms with buttons, boxes, and menus Your Step by Step digital content includes: All the book's practice files—ready to download and put to work. See "Using the Practice Files," inside. Fully searchable online edition of this book—with unlimited access on the Web. Free online account required; see inside book.

Category: Computers

Pages: 432]

Teach Yourself VISUALLY HTML5

None

Make mark-up language more manageable with this visual guide HTML5 is the next-generation of web standard mark-up language,and among other things, it offers amazing new avenues forincorporating multimedia into your sites. What easier way to masterall of HTML5's new bells and whistles than with a guide that showsyou, screenshot by screenshot, just what to do? Over a hundredtasks that web designers need to know most are explained using,full-color screenshots and how-to steps. From the easy stuff likerevised new header and footer elements to complex updates such ascanvas and audio, this guide covers the new, as well asmost-commonly used, tags and features. Helps you get up to speed on the completely redesigned newHyperText Markup Language, HTML5 Shows you how to incorporate rich media content into the sitesyou design, without relying on proprietary software such asFlash Explains revisions, from essential structural elements likeheader and footer to more complex elements such as canvas andaudio—over a hundred tasks in all Uses easy-to-follow, full-color, two-page tutorials, so you cansee step by step how to do tasks and quickly obtain the informationyou need Web designers, keep your HTML skills up to date with this"learn-by-seeing," visual guide.

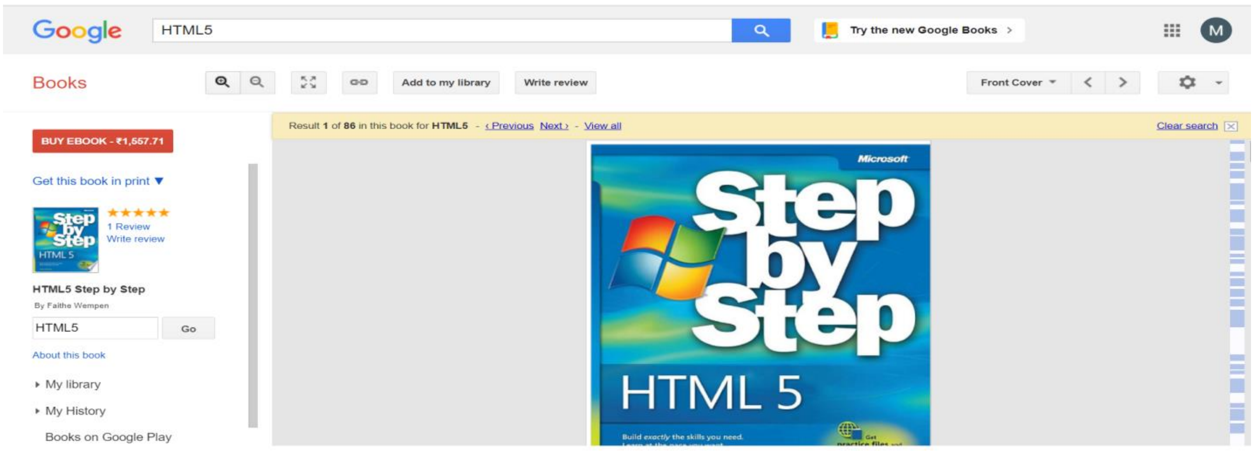
Category: Computers

Pages: 336]

HTML5 for .NET Developers

Single page web apps, JavaScript, and semantic markup

Summary HTML5 for .NET Developers teaches professional software engineers how to integrate the latest HTML5 APIs and semantic markup into rich web applications using JavaScript, ASP.NET MVC, and WCF. Written from the .NET perspective, this book is full of practical applications and ways to connect the new web standards with your existing development practices About the Technology A shift is underway for Microsoft developers—to build web applications you'll need to integrate HTML5 features like Canvas-based graphics and the new JavaScript-driven APIs with familiar technologies like ASP.NET MVC and WCF. This book is designed for you. About this Book HTML5 for .NET Developers teaches you how to blend HTML5 with your current .NET tools and practices. You'll start with a

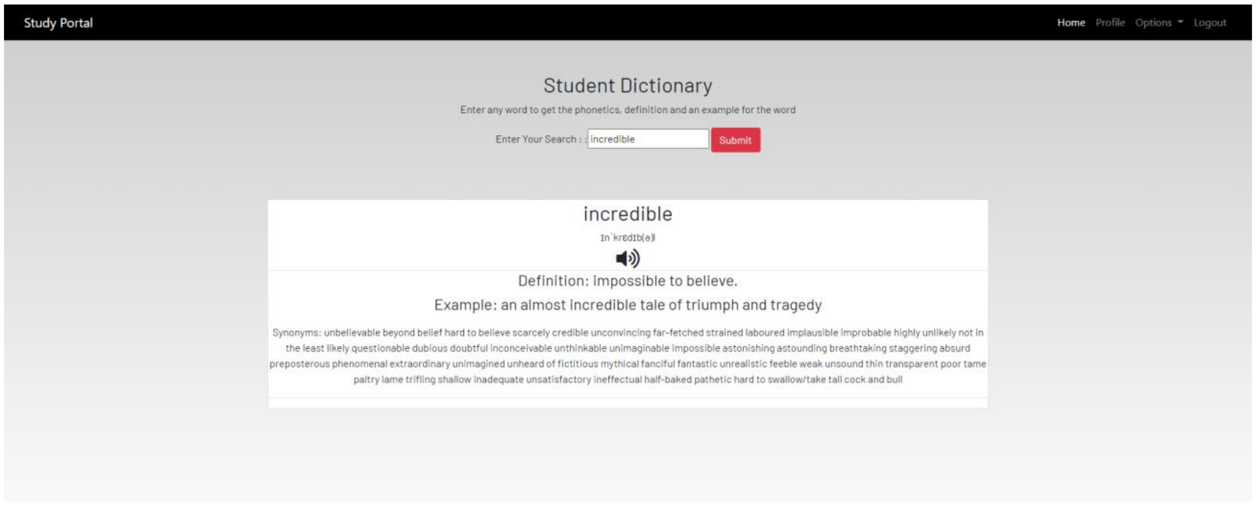


- **Code of the Books-**

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% block content %}
<section class='text-center container'>
    <h2>Search books and browse your favorite</h2>
    <p>just enter the search query to obtain the results</p><b></b>
    <form action="" method="post">
        {% csrf_token %}
            {{form}}
        <input class="btn btn-danger" type="submit" value="Submit">
    </form><br>
 {% for result in results %}
    <a href="{{result.preview}}" target="_blank">
        <div class="card">
            <div class="card-header">
                <div class="row">
                    <div class="col-md-3">
                        <img class="img-fluid" src="{{result.thumbnail}}" alt="">
                    </div>
                    <div class="col-md-9">
                        <h3 class="p-0 m-0">{{result.title}}</h3>
                        <b>
                            <u>
                                <h5 class="p-0 m-0">{{result.subtitle}}</h5>
                            </u>
                        </b>
                        {% if result.description %}
                        <h6 class="p-0 m-1">{{result.description}}</h6>
                        {% endif %}
                        <b>
                       {% if result.categories %}
                            <h6 class="ml-0 mt-3">Category:









                                      {% for category in result.categories %}
                                       {{category}}
                                  {% endfor %}
                            </h6>
                        {% endif %}
                         {% if result.count %}
                            <h6 class="ml-0 mt-1">Pages: {{result.count}}</h6>
                         {% endif %}
                        {% if result.rating %}
                            <h6 class="ml-0 mt-1">Rating: {{result.rating}}</h6>
                        {% endif %}
                    </b>
                </div>
            </div>
        </div>
    </div>
</a>
{% endfor %}
<br>
</section>
{% endblock content %}
def books(request):
    if request.method == "POST":
        form = DashboardForm(request.POST)
        text = request.POST['text']
url = "https://www.googleapis.com/books/v1/volumes?q="+text
        r = requests.get(url)
        answer = r.json()
        result_list = []
        for i in range(10):
            result_dict = {
                'title':answer['items'][i]['volumeInfo']['title'],
                'subtitle':answer['items'][i]['volumeInfo'].get('subtitle'),
                'description':answer['items'][i]['volumeInfo'].get('description'),
                'count':answer['items'][i]['volumeInfo'].get('pageCount'),
                'categories':answer['items'][i]['volumeInfo'].get('categories'),
                'rating':answer['items'][i]['volumeInfo'].get('pageRating'),
                'thumbnail':answer['items'][i]['volumeInfo'].get('imageLinks').get('thumbnail'),
                'preview':answer['items'][i]['volumeInfo'].get('previewLink')
            }
            result_list.append(result_dict)
            context={
                'form':form,
                'results':result_list
            }
        return render(request,'dashboard/books.html',context)
    else:
        form = DashboardForm()
    context = {'form':form}
    return render(request,"dashboard/books.html",context)
```

6.13 DICTIONARY

- In this page, students can learn about the new words or meaning of words as well as they will learn how pronounce the word they are searching for.



Student Dictionary

Enter any word to get the phonetics, definition and an example for the word

Enter Your Search : : incredible

incredible

ɪnˈkrɛdɪb(ə)l

Definition: impossible to believe.

Example: an almost incredible tale of triumph and tragedy

Synonyms: unbelievable beyond belief hard to believe scarcely credible unconvincing far-fetched strained laboured implausible improbable highly unlikely not in the least likely questionable dubious doubtful inconceivable unthinkable unimaginable impossible astonishing astounding breathtaking staggering absurd preposterous phenomenal extraordinary unimagined unheard of fictitious mythical fanciful fantastic unrealistic feeble weak unsound thin transparent poor tame paltry lame trifling shallow inadequate unsatisfactory ineffectual half-baked pathetic hard to swallow/take tall cock and bull

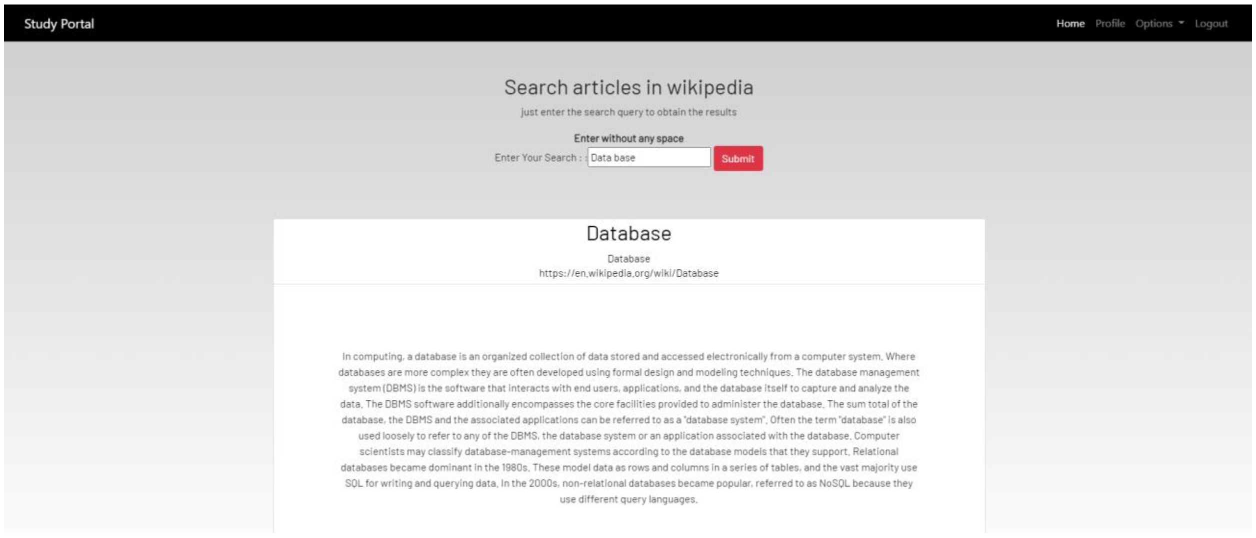
➢ Code of Dictionary-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% block content %}
<section class='text-center container'>
    <h2>Student Dictionary</h2>
    <p>Enter any word to get the phonetics, definition and an example for the word</p>
    <form action="" method="post">
      {% csrf_token %}
      {{form}}
       <input class="btn btn-danger" type="submit" value="Submit">
    </form><br>
    <div class="content-section p-0 mt-5">
        {% if input %}
        <div class="custom-header">
            <h2>{{input}}</h2>
            <h6>{{phonetics}}</h6>
            <audio id="player" src="{{audio}}"></audio>
            <div>
                <a onclick="document.getElementById('player').play()"<i class='fa fa-volume-up fa-2x'></i></a>
            </div>
        </div>




  <hr class="p-0 m-0">
```

```
        <p class="float-left">
            <h4>Definition: {{definition}}</h4>
            <h4>Example: {{example}}</h4>
        </p>
        Synonyms:
         {% for synonym in synonyms %}
            {{synonym}}
         {% endfor %}
        <hr>

    </div>
     {% else %}
    <h2>Sorry, API request limit exceeded</h2>
    {% endif %}
</section>
{% endblock content %}
def dictionary(request):
     if request.method == "POST":
        form = DashboardForm(request.POST)
        text = request.POST['text']
        url =  "https://api.dictionaryapi.dev/api/v2/entries/en_US/"+text
        r = requests.get(url)
        answer = r.json()

 try:
            phonetics = answer[0]['phonetics'][0]['text']
            audio = answer[0]['phonetics'][0]['audio']
            definition = answer[0]['meanings'][0]['definitions'][0]['definition']
            example = answer[0]['meanings'][0]['definitions'][0]['example']
            synonyms = answer[0]['meanings'][0]['definitions'][0]['synonyms']
            context ={
                'form':form,
                'input':text,
                'phonetics':phonetics,
                'audio':audio,
                'definition':definition,
                'example':example,
                'synonyms':synonyms
            }
        except:
            context = {
                 'form':form,
                 'input':''
            }
        return render(request,"dashboard/dictionary.html",context)
     else:
           form = DashboardForm()
           context = {'form':form}
           return render(request,"dashboard/dictionary.html",context)
```

6.15 WIKIPEDIA

- In this page, student can search about any of the particular topic in the Wikipedia site directly, instead of switching to the various sites. They will get a paragraph as an intro or about that topic through which student will get to know basic about the topic.



Database

Database
https://en.wikipedia.org/wiki/Database

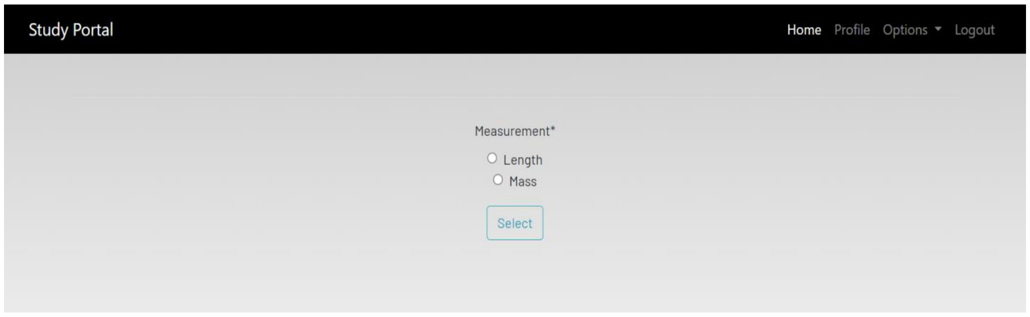
In computing, a database is an organized collection of data stored and accessed electronically from a computer system. Where databases are more complex they are often developed using formal design and modeling techniques. The database management system (DBMS) is the software that interacts with end users, applications, and the database itself to capture and analyze the data. The DBMS software additionally encompasses the core facilities provided to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a "database system". Often the term "database" is also used loosely to refer to any of the DBMS, the database system or an application associated with the database. Computer scientists may classify database-management systems according to the database models that they support. Relational databases became dominant in the 1980s. These model data as rows and columns in a series of tables, and the vast majority use SQL for writing and querying data. In the 2000s, non-relational databases became popular, referred to as NoSQL because they use different query languages.

➢ Code of the Wikipedia-

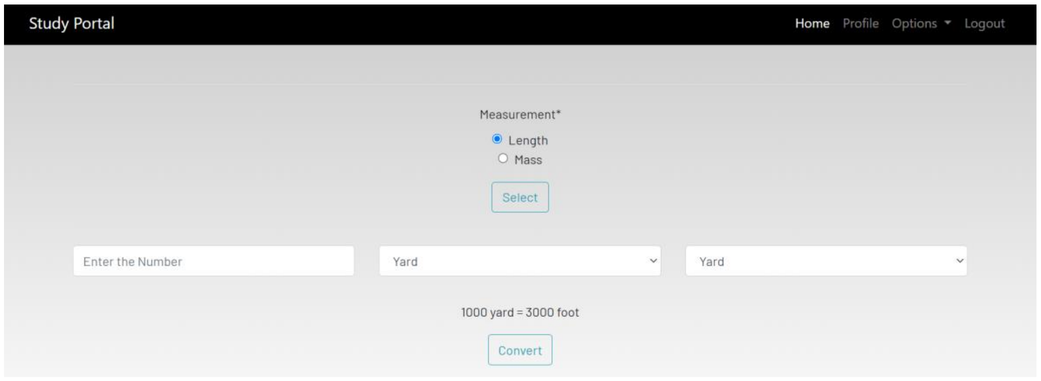
```
{% extends 'dashboard/base.html' %}
{% load static %}
{% block content %}

<section class='text-center container'>
    <h2>Search articles in wikipedia</h2>
    <p>just enter the search query to obtain the results</p><b>Enter without any space</b>
    <form action="" method="post">
      {% csrf_token %}
      {{form}}
        <input class="btn btn-danger" type="submit" value="Submit">
    </form><br>

    <div class="container">
        <div class="content-section p-0 mt-5">
            <a href="" class="p-0 m-0" target="_blank">
                <div class="custom-header">
                    <h2>{{title}}</h2>{{title}}
                    <h6>{{link}}</h6>
                </div>
            </a>




     <hr class="p-0 m-0">
            <p class="description m-5 p-5">
                {{details}}
            </p>
            <hr>
        </div>
    </div>
</section>
{% endblock content %}
```

6.15 CONVERSION

- This page will help out students to make the conversions of Length and Mass.

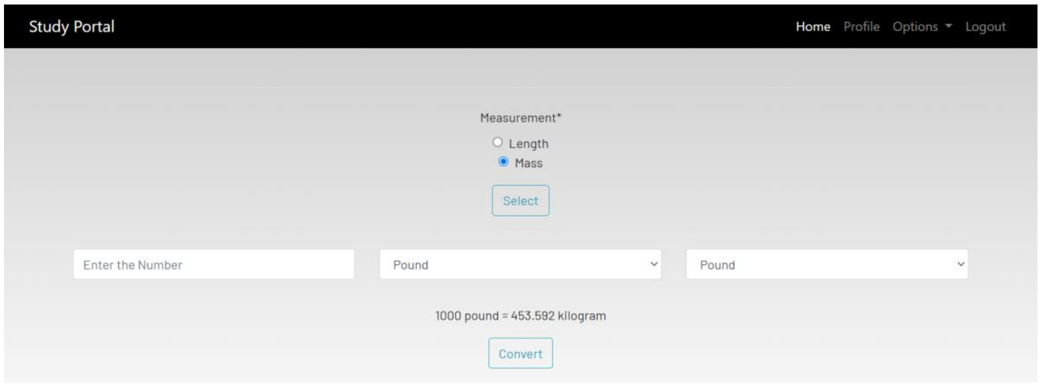


- After selecting one of them they can enter the number and select the form in which they want the conversion.



Measurement*: (●) Length ( ) Mass

1000 yard = 3000 foot



1000 pound = 453.592 kilogram

➢ Code of the Conversion-

```
{% extends 'dashboard/base.html' %}
{% load static %}
{% load crispy_forms_tags %}
{% block content %}
<div class="container text-center">
    <form method="POST">
        {% csrf_token %}
        <div class="form-group">
         <legend class="border-bottom mb-4"></legend>
            {{form|crispy}}
            <button href="" class="btn btn-outline-info" type="submit">
                Select
            </button>
        </div>
        {% if input %}
        <fieldset class="form-group">
            <legend class="border-bottom mb-4"></legend>
            <div class="row">
                <div class="col-md-4">
                    {{m_form.input|as_crispy_field}}
                </div>
                <div class="col-md-4">
                    {{m_form.measure1|as_crispy_field}}
                </div>
                <div class="col-md-4">
             {{m_form.measure2|as_crispy_field}}
                </div>
            </div>
        </fieldset>
        <fieldset class="form-group">
            {{answer}}
        </fieldset>
        <div class="form-group">
            <button href="" class="btn btn-outline-info" type="submit">
                Convert
            </button>
        </div>
        {% endif %}
    </form>
</div>

{% endblock content %}
```

7. Testing

Testing plays a critical role in quality assurance for software Due to the limitation of the verification method for the previous phases, design and requirement fault also appear in the code. Testing is used to detect these errors, in edition to the error introduced during coding phase. Testing is a dynamic method for verification and validation, where the system is to be tested is executed and behavior of the system is observed. Due to this testing the failure of the system can be observed, from which the presence of fault can be deduced. However, separate activities have to be performed to identify the faults.

There are two method of testing: **functional** and **structural**. In functional testing, the internal logic of the system under testing is not considered and the test cases are decided from the specification or the requirements. It is often called "Black Box Testing". Equivalence class partitioning, boundary analysis, and cause effect graphing are examples of methods for selecting test cases for functional testing. In structural testing, the test cases are decided entirely on the internal logic of the program or module being tested.

As the goal of testing is to detect any errors in the programs different flavor of testing are often used. Unit testing are used to test a module or a small collection of modules and the focus is on detecting coding errors in modules. During integration testing modules are combined into sub-system, which are then tested.

The goal here is to test the system design. In system testing and acceptance testing, the entire System is tested. The goal here is to test the requirement themselves. Structural testing can be used for unit testing while at higher level mostly functional testing is used.

System testing is a critical phase in systems implementation Testing of a system hardware device testing and debugging of computer programs and testing information processing procedures. Testing can be done with test data, which attempts to simple all possible conditions that may arise during processing. The plane for testing are prepared and ben implemented.

Principles of Testing:

(i) All the test should meet party the customer requirements.
(ii) To make our software testing should be performed by third party
(iii) Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk Assessment of the application.
(iv) All the test to be conducted should be planned before implementing it.
(v) It follows pareto rule(80/20 rule) which states that 80% of errors comes from 20% of components.
(vi) Start testing with small parts and extend it to large parts.

Types of Testing:

- **Unit Testing-**

  It focuses on smallest unit of software design. In this we test an individual unit or group of inters related units. It is often done by programmer by using sample input and observing its corresponding outputs.
  Example:
  a) In a program we are checking if loop, method or function is working fine
  b) Misunderstood or incorrect, arithmetic precedence.
  c)Incorrect initialization.

- **Integration Testing-**

  The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components are combined to produce output.

- **Alpha Testing-**

  This is a type of validation testing. It is a type of acceptance testing which is done before the product is released to customers. It is typically done by QA people. Example: When software testing is performed internally within the organization.

- **Beta Testing-**

  The beta test is conducted at one or more customer sites by the end-user of the software. This version is released for the limited number of users for testing in real time environment.
  Example: When software testing is performed for the limited number of people.

- **System Testing**-

  In this software is tested such that it works fine for different operating system .It is covered under the black box testing technique. In this we just focus on required input and output on internal In this we have security testing, recovery testing stress testing and performance testing without focusing on internal working. In this we have security testing, recovery testing , stress testing and performance testing. Example: This include functional as well as non functional Testing.

- **Performance Testing-**

  It is designed to test the run-time performance of software within the context of an integrated system. It is used to test speed and effectiveness of program. Example: Checking number of processor cycle.

7.1 Test Criteria

- **Stability (ST) –** Focusing on the application being stable on the device.

- **Application Launch (AL) -** Once an application is loaded it must start (launch) and stop correctly in relation to the device and other applications on the device.

- **User Interface (UI) -** The intent is to not specify exactly how to design a user interface but rather to give general guidelines. It is expected that publishers and network operators will further define the look and feel of an application's user interface to make it more in conformance with their overall look and feel.

- **Localization (LO) -** Applications that are to be deployed to localities other than their point of origin must account for changes in language, alphabets, date and money formats, etc.

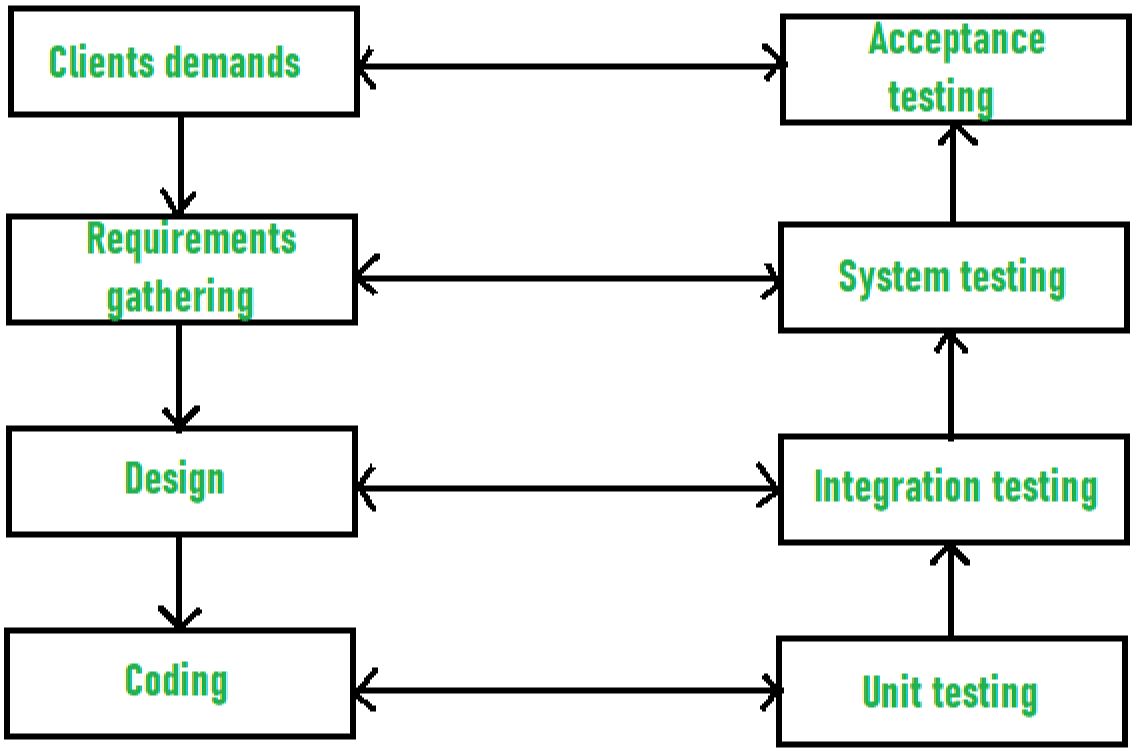
- **Functionality (FN) -** Documented features are implemented in the application and work as expected. Sources for the information are user manuals, formatted application specification documents and online documentation.

- **Connectivity (CO) -** If an application has communication capabilities then it must demonstrate its ability to communicate over a network correctly. It must be capable of dealing with both network problems and server-side problems.

- **Personal Information Management (PI) -** The application accessing user information needs to be able to do it in an appropriate manner and not to destroy the information.

7.2 Levels of Testing

- In order to uncover the errors present in different phases, we have the concept of levels of testing. The basic levels of testing are:



LEVELS OF TESTING

Unit testing

- Unit testing focuses verification effort on the smallest unit of software i.e. the module Using the detailed design and the process specifications, testing is done to uncover errors within the boundary of the module. All modules must be successful in the unit test before the start of the integration testing begins.

- In this project each service can be thought of a module. There are so many modules like Login, HR Department, Interviewer Section, etc. Each module has been tested by giving different sets of inputs. When developing the module as well as finishing the development, the module works without any error. The inputs are validated when accepting them from the user.

Integration Testing

- After unit testing, we have to perform integration testing. The goal here is to see if modules can be integrated properly, the emphasis being on testing interfaces between modules. This testing activity can be considered as testing the design and hence the emphasis on testing module interactions.

- In this project the main system is formed by integrating all the modules. When integrating all the modules I have checked whether the integration effects working of any of the services by giving different combinations of inputs with which the two services run perfectly before Integration.

System Testing

- Here the entire software system is tested. The reference document for this process is the requirements document, and the goal is to see if software meets its requirements.

- Here entire 'HRRP has been tested against requirements of project and it is checked whether all requirements of project have been satisfied or not.

Acceptance Testing

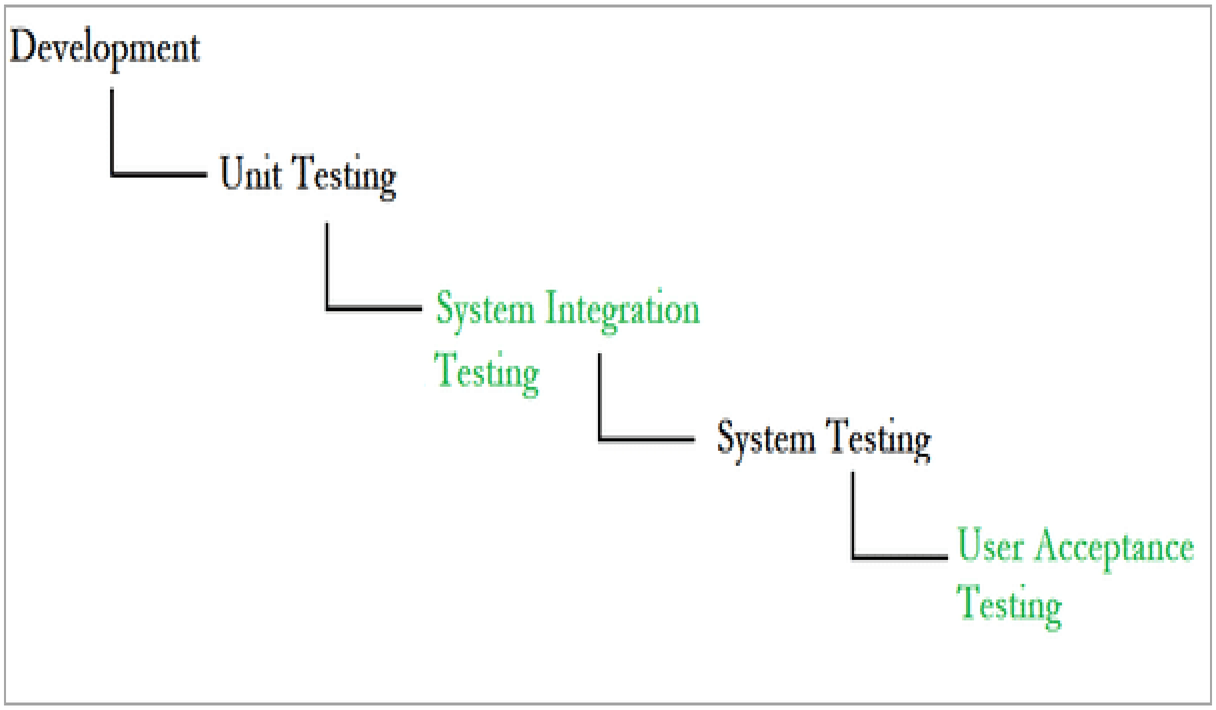
- Acceptance Testing is performed with realistic data of the client to demonstrate that the software is working satisfactorily. Testing here is focused on external behavior of the system: the internal logic of program is not emphasized.

- Test cases should be selected so that the largest number of attributes of an equivalence class is exercised at once. The testing phase is an important part of software development. It is the process of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied.

7.3 Test Information Flow-

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vortex of the spiral and, concentrates on each unit, component of the software as implemented in source code. Testing progresses moving outward along the spiral to integration testing, where the focus is on designed the construction of the software architecture. Taking another turn outward on spiral, we encounter validation testing, where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally, we arrive at system testing, where the software and other system elements are tested as a whole. To test computer software, we spiral out along stream lines that broaden the scope of testing with each turn.

Considering the process from a procedural point of view, testing within the context of software engineering is actually a series of four steps that are implemented sequentially. The steps are shown in Figure. Initially, tests focus on each component individually, ensuring that it functions properly as unit. Hence, the name unit testing Unit testing makes heavy use of white-box testing techniques, exercising specific paths in module's control structure to ensure complete coverage and maximum error detection.



7.4 Website Security

1. Physical security

Damage due to natural causes like earth tremor, flooding, water logging, fire hazards, atmospheric or environmental conditions etc.
For overcoming these difficulties, the replica of the data is automatically stored at various networks and for environmental conditions Air conditioning environment is created.

2. Data security

There are basically two problems associated with data security:

1.Data not being available to the authorized person at the time of need.
2. Data becoming available to the unauthorized person.

To overcome these difficulties the following access facilities has been provided:

i) Identification:
Unique Ids for the different users have been provided.

ii) Authentication:
System checks the password under the particular user identification. The computer permits the various resources to the authorized person.

iii) Authorization:
The access control mechanism to prevent unauthorized logging to the system.

8. Maintenance

The maintenance starts after the final software product is delivered to the client. The maintenance phase identifies and implements the change associated with the correction of errors that may arise after the customer has started using the developed software. This also maintains the change associated with changes in the software environment and customer requirements. Once the system is a live one, Maintenance phase is important. Service after sale is a must and users/ clients must be helped after the system is implemented. If he/she faces any problem in using the system, one or two trained persons from developer's side can be deputed at the client's site, so as to avoid any problem and if any problem occurs immediate solution may be provided.

Even though the definition of equivalence partitioning states that testing one value from a class is equivalent to testing any other value from that class, we need to look at the boundaries of equivalent classes more closely. This is so since boundaries are more error prone.

To design two valid cases at both the ends test cases using boundary value analysis, for a range of values:

• Two valid cases at both the ends.
• Two invalid cases just beyond the range limits.

Cause Effect Analysis

• The main drawback of the previous two techniques is that they do not explore the combination of input conditions.

• Cause effect analysis is an approach for studying the specifications carefully and identifying the combinations of input conditions (causes) and their effect in the form of a table and designing test cases.

• It is suitable for applications in which combinations of input conditions are few and readily visible.

Cause Effect Graphing

- This is a rigorous approach, recommended for complex systems only. In such systems′ the number of inputs and number of equivalent classes for each input could be many and hence the number of input combinations usually is astronomical. Hence we need a systematic approach to select a subset of these input conditions.

Software maintenance is widely accepted part of SDLC now a days. It stands for all the modifications and updations done after the delivery of software product. There are number of reasons, why modifications are required, some of them are briefly mentioned below:

- **Market Conditions -** Policies, which changes over the time, such as taxation and newly introduced constraints like, how to maintain bookkeeping, may trigger need for modification.

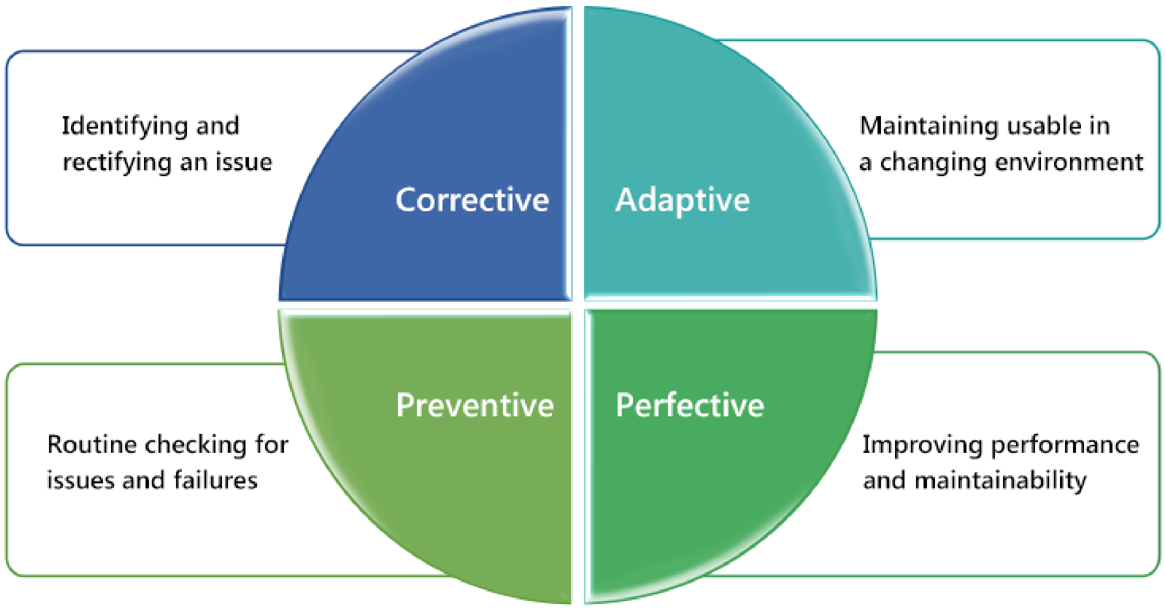
- **Client Requirements** - Over the time, customer may ask for new features or functions in the software.

- **Host Modifications** - If any of the hardware and/or platform (such as operating system) of the target host changes, software changes are needed to keep adaptability.

- **Organization Changes** - If there is any business level change at client end, such as reduction of organization strength, acquiring another company, organization venturing into new business, need to modify in the original software may arise.

8.1 Types of Maintenance

Software Maintenance is the process of modifying a software product after it has been delivered to the customer. The main purpose of software maintenance is to modify and update software applications after delivery to correct faults and to improve performance.

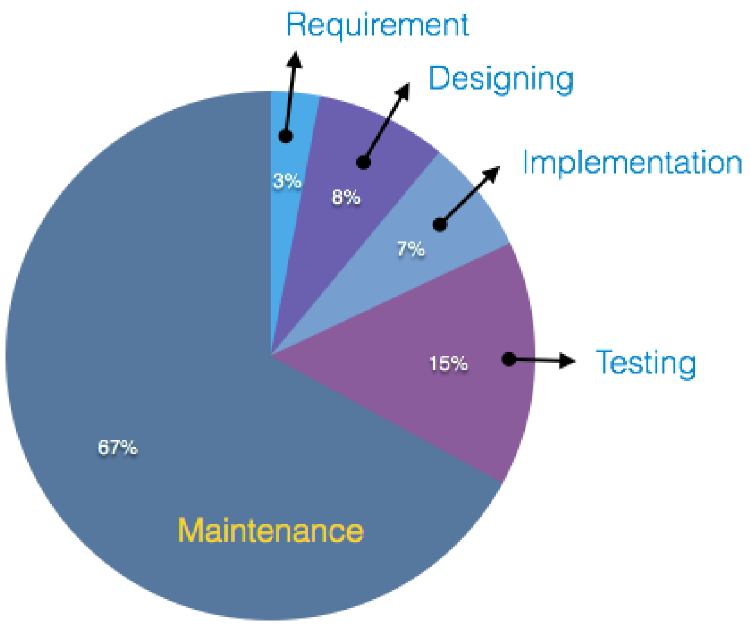

Identifying and rectifying an issue
Corrective
Adaptive
Maintaining usable in a changing environment
Routine checking for issues and failures
Preventive
Perfective
Improving performance and maintainability

- Corrective Software Maintenance- Corrective software maintenance is what one would typically associate with the maintenance of any kind. Correct software maintenance addresses the errors and faults within software applications that could impact various parts of your software, including the design, logic, and code.

- **Adaptive Software Maintenance-** Adaptive software maintenance becomes important when the environment of your software changes. This can be brought on by changes to the operating system, hardware, software dependencies, Cloud storage, or even changes within the operating system.
- **Perfective Software Maintenance-** Perfective software maintenance focuses on the evolution of requirements and features that existing in your system. As users interact with your applications, they may notice things that you did not or suggest new features that they would like as part of the software, which could become future projects or enhancements.
- **Preventive Software Maintenance-** Preventative Software Maintenance helps to make changes and adaptations to your software so that it can work for a longer period of time. The focus of the type of maintenance is to prevent the deterioration of your software as it continues to adapt and change. These services can include optimizing code and updating documentation as needed.

8.2 Cost of Maintenance

Reports suggest that the cost of maintenance is high. A study on estimating software maintenance found that the cost of maintenance is as high as 67% of the cost of entire software process cycle.



On an average, the cost of software maintenance is more than 50% of all SDLC phases. There are various factors, which trigger maintenance cost go high, such as:

REAL-WORLD FACTORS AFFECTING MAINTENANCE COST-

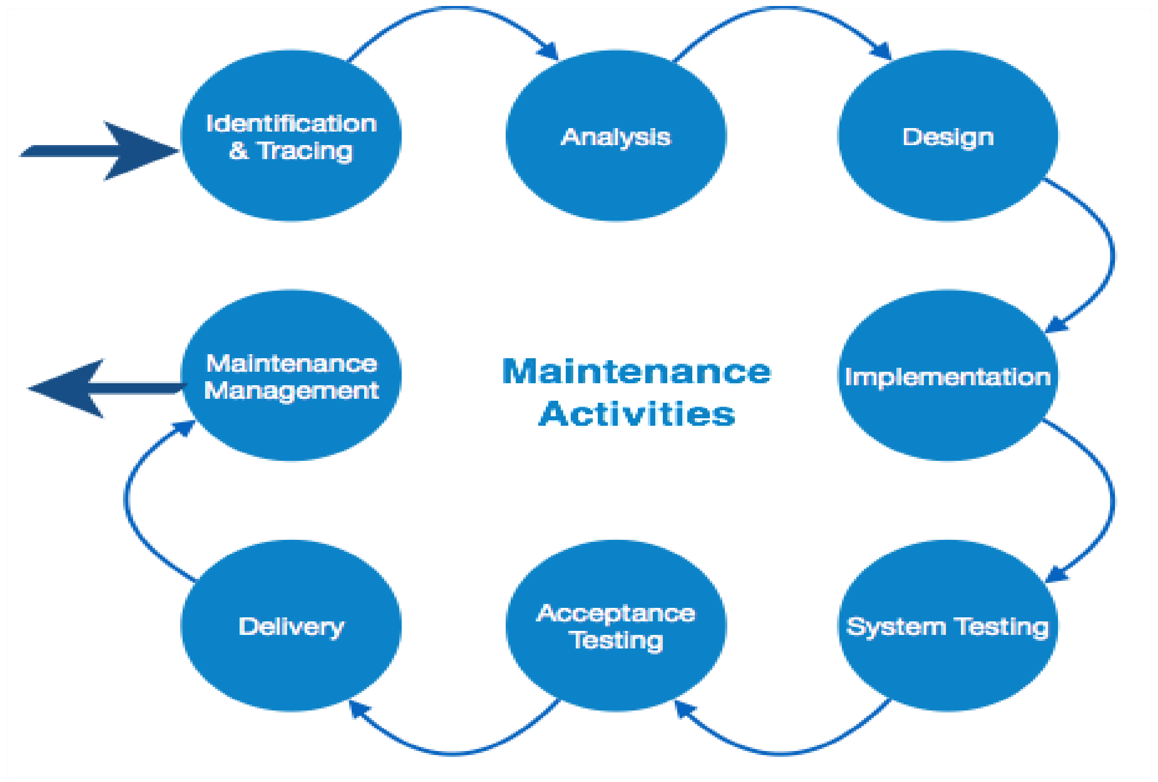
• The standard age of any software is considered up to 10 to 15 years.
• Older softwares, which were meant to work on slow machines with less memory and storage capacity cannot keep themselves challenging against newly coming enhanced softwares on modern hardware.
• As technology advances, it becomes costly to maintain old software.
• Most maintenance engineers are newbie and use trial and error method to rectify problem.
• Often, changes made can easily hurt the original structure of the software, making it hard for any subsequent changes.
• Changes are often left undocumented which may cause more conflicts in future.

SOFTWARE-END FACTORS AFFECTING MAINTENANCE COST

• Structure of Software Program
• Programming Language
• Dependence on external environment
• Staff reliability and availability

8.3 Maintenance Activities

IEEE provides a framework for sequential maintenance process activities. It can be used in iterative manner and can be extended so that customized items and processes can be included.



These activities go hand-in-hand with each of the following phase:

• **Identification & Tracing** - It involves activities pertaining to identification of requirement of modification or maintenance. It is generated by user or system may itself report via logs or error messages.Here, the maintenance type is classified also.

• **Analysis** - The modification is analyzed for its impact on the system including safety and security implications. If probable impact is severe, alternative solution is looked for. A set of required modifications is then materialized into requirement specifications. The cost of modification/maintenance is analyzed and estimation is concluded.

• **Design** - New modules, which need to be replaced or modified, are designed against requirement specifications set in the previous stage. Test cases are created for validation and verification.

• **Implementation** - The new modules are coded with the help of structured design created in the design step.Every programmer is expected to do unit testing in parallel.

• **System Testing** - Integration testing is done among newly created modules. Integration testing is also carried out between new modules and the system. 55 Finally the system is tested as a whole, following regressive testing procedures.

• **Acceptance Testing** - After testing the system internally, it is tested for acceptance with the help of users. If at this state, user complaints some issues they are addressed or noted to address in next iteration.

• **Delivery** - After acceptance test, the system is deployed all over the organization either by small update package or fresh installation of the system. The final testing takes place at client end after the software is delivered. Training facility is provided if required, in addition to the hard copy of user manual.

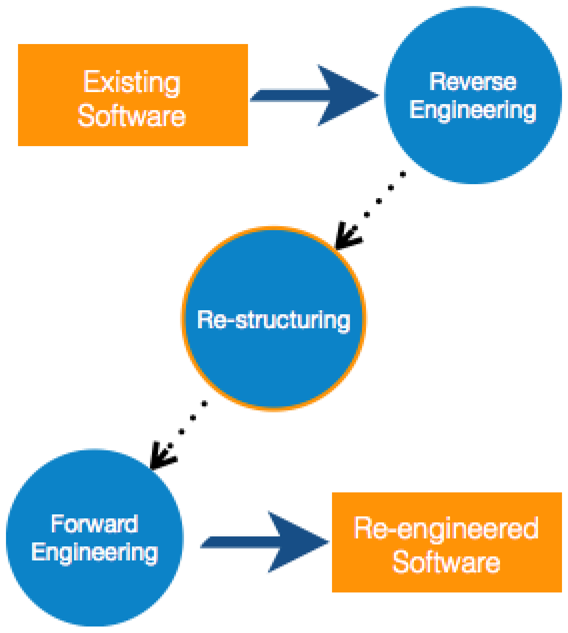
• **Maintenance management** - Configuration management is an essential part of system maintenance. It is aided with version control tools to control versions, semi-version or patch management.

8.4 Software Re-engineering

When we need to update the software to keep it to the current market, without impacting its functionality, it is called software re-engineering. It is a thorough process where the design of software is changed and programs are re-written.

Legacy software cannot keep tuning with the latest technology available in the market. As the hardware become obsolete, updating of software becomes a headache. Even if software grows old with time, its functionality does not.
For example, initially Unix was developed in assembly language. When language C came into existence, Unix was re-engineered in C, because working in assembly language was difficult.

Other than this, sometimes programmers notice that few parts of software need more maintenance than others and they also need re-engineering.



8.5 RE-ENGINEERING PROCESS

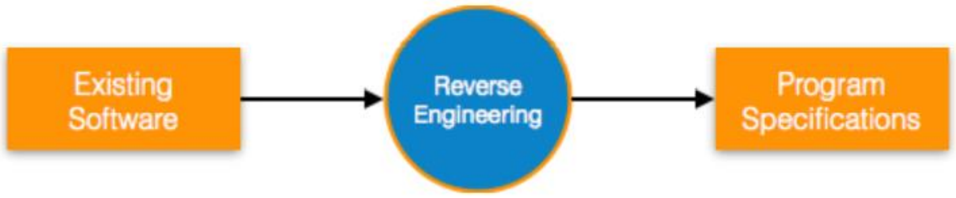
• **Decide** what to re-engineer. Is it whole software or a part of it?
• **Perform** Reverse Engineering, in order to obtain specifications of existing software.
• **Restructure Program** if required. For example, changing function-oriented programs into object-oriented programs.
• **Re-structure data** as required.
• **Apply Forward engineering** concepts in order to get re-engineered software.

There are few important terms used in Software re-engineering

REVERSE ENGINEERING

It is a process to achieve system specification by thoroughly analyzing, understanding the existing system. This process can be seen as reverse SDLC model, i.e. we try to get higher abstraction level by analyzing lower abstraction levels.

An existing system is previously implemented design, about which we know nothing. Designers then do reverse engineering by looking at the code and try to get 57 the design. With design in hand, they try to conclude the specifications. Thus , going in reverse from code to system specification.

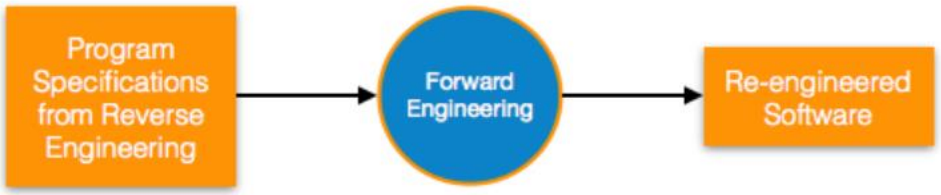


PROGRAM RESTRUCTURING

It is a process to re-structure and re-construct the existing software. It is all about re-arranging the source code, either in same programming language or from one programming language to a different one. Restructuring can have either source code-restructuring and data-restructuring or both.
Re-structuring does not impact the functionality of the software but enhance reliability and maintainability. Program components, which cause errors very frequently can be changed, or updated with re-structuring. The dependability of software on obsolete hardware platform can be removed via re-structuring.

FORWARD ENGINEERING

Forward engineering is a process of obtaining desired software from the specifications in hand which were brought down by means of reverse engineering. It assumes that there was some software engineering already done in the past.
Forward engineering is same as software engineering process with only one difference – it is carried out always after reverse engineering.



Component reusability

A component is a part of software program code, which executes an independent task in the system. It can be a small module or sub-system itself.

EXAMPLE:

- The login procedures used on the web can be considered as components, printing system in software can be seen as a component of the software.
- Components have high cohesion of functionality and lower rate of coupling, i.e. they work independently and can perform tasks without depending on other modules.
- In OOP, the objects are designed are very specific to their concern and have fewer chances to be used in some other software.
- In modular programming, the modules are coded to perform specific tasks which can be used across number of other software programs.
- There is a whole new vertical, which is based on re-use of software component, and is known as Component Based Software Engineering (CBSE).

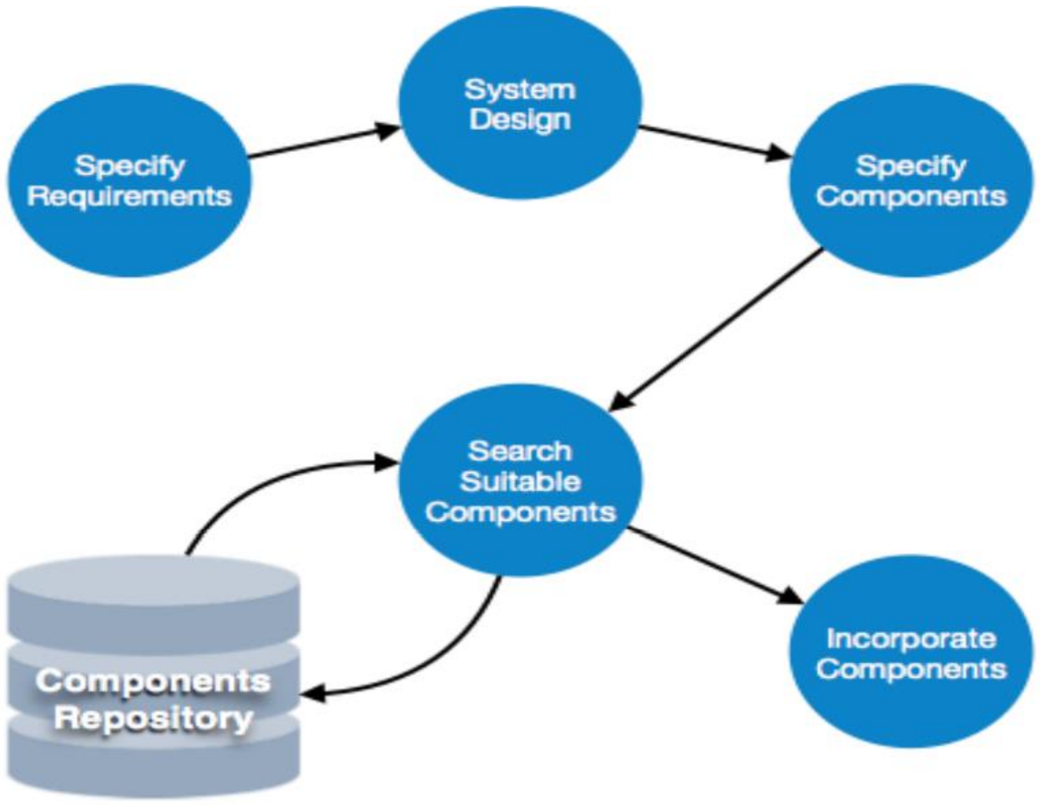
Re-use can be done at various levels

- **Application level** - Where an entire application is used as sub-system of new software.
- **Component level** - Where sub-system of an application is used.
- **Modules level** - Where functional modules are re-used.

Software components provide interfaces, which can be used to establish communication among different components.

REUSE PROCESS

Two kinds of method can be adopted: either by keeping requirements same and adjusting components or by keeping components same and modifying requirements.



• **Requirement Specification** - The functional and non-functional requirements are specified , which a software product must comply to, with the help of existing system, user input or both.
• **Design** - This is also a standard SDLC process step, where requirements are defined in terms of software parlance. Basic architecture of system as a whole and its sub-systems are created.
• **Specify Components** - By studying the software design, the designers segregate the entire system into smaller components or sub-systems. One complete software design turns into a collection of a huge set of components working together.
• **Search Suitable Components** - The software component repository is referred by designers to search for the matching component, on the basis of functionality and intended software requirements.
• **Incorporate Components** - All matched components are packed together to shape them as complete software.

9. Conclusion

This project was successfully completed within the time span allotted. The project Study Portal has been developed in Django. All the modules are tested separately and put together to form the main system. Finally the system is tested with real data and everything worked successfully. Thus the system has fulfilled the entire objective identified.

The system had been developed in an attractive dialogs fashion. So user with minimum knowledge about computers can also operate the system easily. It will make easy interactions between users and store. The speed and accuracy are maintained in proper way.

Overall the Online System procedure the online features and the customer to save the time Effort ,and price

10. Bibliography

WEBSITES

- **https://www.google.com**
- **https://www.w3schools.com**

- **https://www.tutorialspoint.com**
- **https://www.codewithharry.com**
- **https://getbootstrap.com**