1.INTRODUCTION

1.1 PROBLEM DEFINITION

During the pandemic time, Students faced a lot of problems in terms of their education. Most of the schools /colleges had online platform of learning. But in that, There was really less / No interaction. And students were not clearly understanding the concepts.As we move further with online based system, the learning has become quite easy, but it is not effective.

According to report by O. Simpson— "78% of students fail to finish their courses.The predominant reason of students dropping an online course is lack of student engagement".The requisite activities and materials for the next assessment".Since more than required notes are available to study.

1.2 SCOPE OF THE APP

The Objective of This "EDUCATOR" App Is to Provide an interface so that people can share their thoughts.

- Easy and highly interactive user interface.
- Enhanced Learning
- Create And Recommend Modules
- Create And Analize Learning Pattern
- Implement Recommendation Through Machine Learning Algorithm

MODULES

- **User Validation :** To Validate Users
- **Course Recommendation :**Recommend New Courses
- **Payment Gateway:** Payment of Course To Access It
- **Module / Syllabus Recommendation :** Decide Learning Roadmap.
- **Content Delivery :** Learning Place.
- **User's Statistics:** Analyze User's Performance
- **Wishlist:** Bookmark A Course For Later Use.

2.SYSTEM STUDY

2.1 EXISTING SYSTEM

In modern online learning app, all students are who enroll in a particular course are provided with same content. But learning capability of every student is not same. Therefore, grasping the knowledge may become too much of some student and too less to others. In other online app. The content provided was vague and general to all student. It was not specific to a particular student group.

2.2 FEASIBILITY STUDY

In other online app. The content provided was vague and general to all student. It was not specific to a particular student group. But our app “Educator” Gives a customized learning experience which Increases productivity in learning by providing a learning roadmap to it.

The pace of student’s learning is quite different, some might learn fast, some takes a lot of time.If the notes are quite easy, a fast learner might feel bored and lose interest. If we keep too complicated, slow learners might feel this is beyond us and never try to learn.So by making average type of learning is one solution, but we will not get an optimized way of doing it. So by customizing our learning pattern, we can achieve our objective.

2.3 PROPOSED SYSTEM

This is an online learning platform in which students can learn any course at their own pace. It is different from other online learning platform. That it not only customize and recommend courses, but also it customizes modules / syllabus to each and every student with the help of Deep/Machine learning algorithms.So, in order to provide required learning, This app “Educator” Cluster students based on their performance and provide required modules in order to fulfill the course. This creates less burden on students in learning period and fully understand the taught concepts.

ADVANTAGES OF PROPOSED SYSTEM

- ✔ Easy To Use
- ✔ Highly Interactive UI
- ✔ Real Time Learning
- ✔ Customized Learning To Each Particular Group
- ✔ More Learning Capabilities

OBJECTIVES OF PROPOSED SYSTEM

The study of existing system is the exact nature of problem manual system. Also, if has been decided that problem in the existing system can be solved only through computerization.

MACHINE LEARNING MODELS USED

Liner Discriminat Analizer:

Linear discriminant analysis is an extremely popular dimensionality reduction technique. Dimensionality reduction techniques have become critical in machine learning since many high-dimensional datasets exist these days. Linear Discriminant Analysis was developed as early as 1936 by Ronald A. Fisher. The original Linear discriminant applied to only a 2-class problem. It was only in 1948 that C.R. Rao generalized it to apply to multi-class problems. It is used as a dimensionality reduction technique. Also known as a commonly used in the pre-processing step in machine learning and pattern classification projects.

Cosine Based Recommendation System

These algorithms recommend items similar to the ones a user liked in the past. We'll review different similarity functions and you'll then be able to choose the more suitable one for your system. The main input is the Item-Content Matrix (ICM) which describes all the attributes for each item. We'll see how we can improve the quality of content-based techniques, by normalising and tuning the importance of each attribute in the ICM: you'll be able to use some specific tuning strategies in order to obtain the best quality recommendations from your system. So, at the end of this module, you'll know how to build a content-based recommender system, how to clean and normalize your input data.

Dataset Used:

The Data from a group of college students of 'Kristu Jayanti College' who're pursuing BCA [Bachelor's in Computer Application]. They took a test on a particular subject, "Operating System Concepts & UNIX/LINUX". This was a pre-course test to perceive the simple idea of the subject from the college students. The test was split into 10 questions. Each question has its very own weights based on its importance. The questions are divided into 3 levels easy, average, hard.

3. System Design

It Is Also One of The Most Crucial Step In Software Development. In the design phase, the architecture is established. This phase starts with the requirement document delivered by the requirement phase und maps the requirements into an architecture.The design may include the usage of existing components Analysing the trade-offs of necessary complexity allows for many things to remain simple which, in turn, will eventually lead to a higher quality product. The architecture team also converts the typical scenarios into a test plan.System design is the process of defining the architecture components, modules interfaces and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of system analysis, systems architecture and systems engineering,

Logical design

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted viu modelling, using an over-abstract (and sometimes graphical) model of the actual system, In the context of systems, designs are included. Logical design includes entity-relationship diagrams (ER diagrams).

Physical design

The physical design relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified/authenticated, how it is processed, and how it is displayed. In physical design, the following requirements about the system are decided.

• Input requirement
• Output requirements
• Storage requirements
• Processing requirements
• System control and backup or recovery:

Put another way, the physical portion of system design can generally be broken down i**nto three** sub-tasks:

User Interface Design is concerned with how users add information to the system and with how the system presents information hack to them.

Data Design is concerned with how the data is represented and stored within the system.

Process Design is concerned with how data moves: through the system, and with how and where it is validated, secured and/or transformed as it flows into through and out of the system.

3.1 ER DIAGRAM

An Entity Relationship Diagram (ERD) shows the Relationships of Entity sets stored in a Database. An Entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases.At first glance an entity relationship diagram looks very much like a flowchart. It is the specialized symbols, and the meanings of those symbols, that make it unique. Because this ER tutorial focuses on beginners below are some tips that will help you build effective ER diagrams:

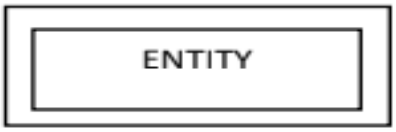
- Identify all the relevant entities in a given system and determine the relationships among These entities.
- An entity should appear only once in a particular diagram.
- Provide a precise and appropriate name for each entity, attribute, and relationship in the diagram. Terms that are simple and familiar always beats vague, technical-sounding words.
- In naming entities, remember to use singular nouns. However, adjectives may be used to distinguish entities belonging to the same class. Meanwhile attribute names must be meaningful, unique, system independent, and easily understandable.
- Remove vague, redundant or unnecessary relationships between entities.
- Never connect a relationship to another relationship.
- Make effective use of colours. You can use colours to classify similar entities or to highlight key areas in your diagrams.

You can draw entity relationship diagrams manually, especially when you are just informally showing simple systems to your peers. However, for more complex systems and for external audiences, you need diagramming software such as flow.io to craft visually engaging and precise ER diagrams. The ER Diagram Software offered by flow.io as an online service is pretty easy to use and is a lot more affordable than purchasing licensed software.

It is also perfectly suited for development teams because of its strong support for collaboration. There are five main components of an ERD:

- **Entities,** which are represented by rectangles. An entity is an object or concept about which you want to store information

- **A weak entity** is an entity that must defined by a foreign key relationship with another entity as it



cannot be uniquely identified by its own attributes alone.

- **Relationship** , which are represented by diamond shapes, show how two entities share

information in the database. In some cases, entities can be self-linked.

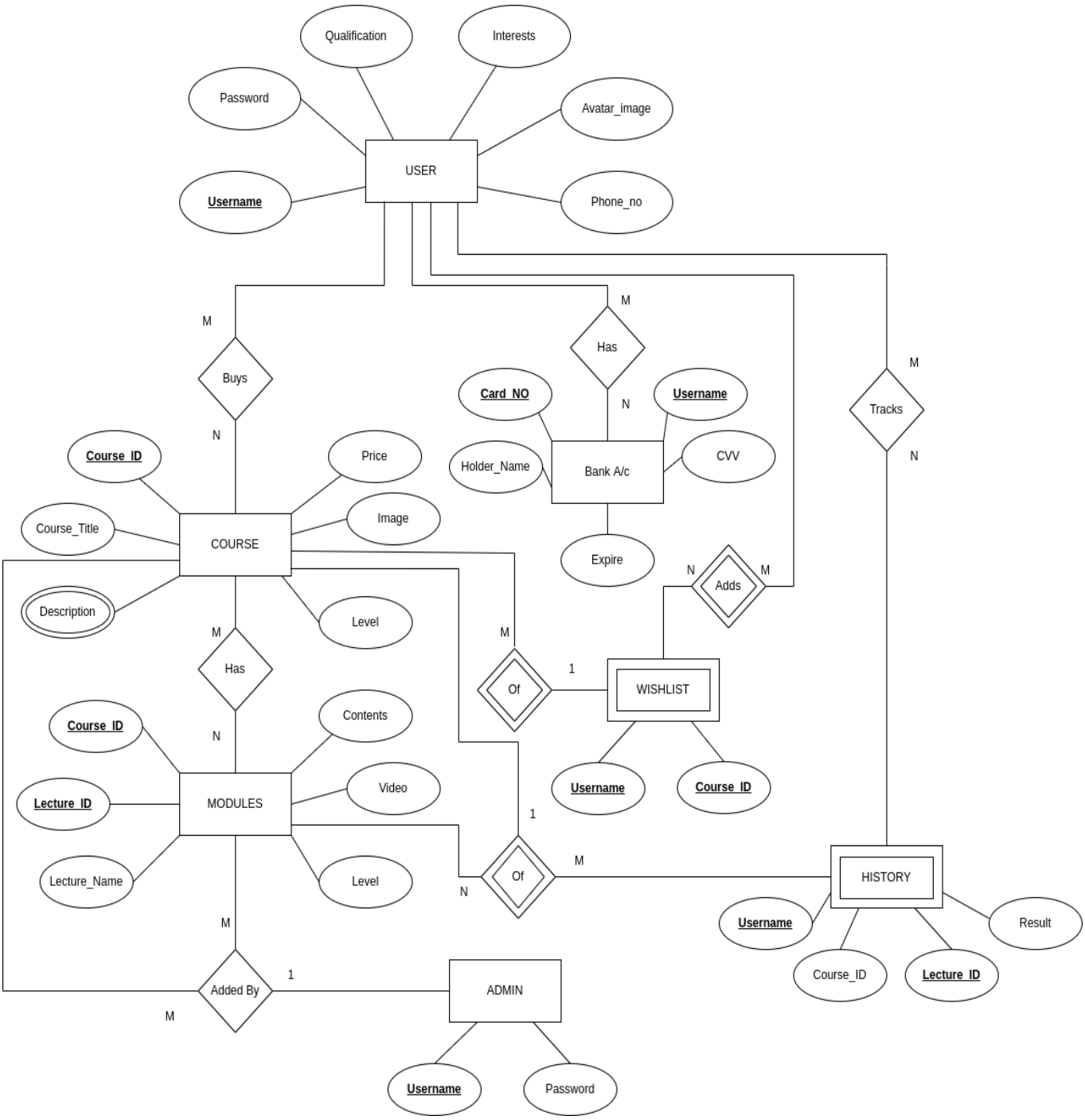
- **Attributes**, which are represented by ovals. A key attribute is the unique, distinguishing characteristic of the entity.

- A **multivalve attribute** can have more than one value. A derived attribute is based on another attribute.

- **Connecting lines**, solid lines that connect attributes to show the relationships of entities in the ER Diagram.

Cardinality specifies how many instances of an entity relate to one instance of another entity

ER Diagram


Qualification
Interests
Password
Avatar_image
USER
Username
Phone_no
M
Buys
N
M
Has
N
Card_NO
Username
Holder_Name
Bank A/c
CVV
Expire
M
Tracks
N
Course_ID
Price
Course_Title
Image
COURSE
Description
Level
N
Adds
M
M
Has
N
M
Of
1
WISHLIST
Username
Course_ID
Course_ID
Contents
Lecture_ID
MODULES
Video
Lecture_Name
Level
1
M
Of
N
HISTORY
Username
Result
Course_ID
Lecture_ID
M
Added By
M
1
ADMIN
Username
Password

3.2 Data flow diagram (level 0 and level 1)

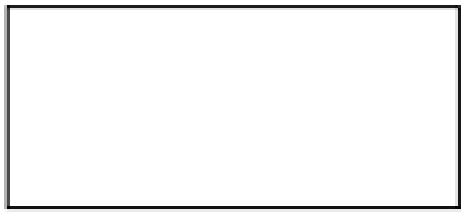
The Data Flow Diagrams (DFDs) are used for structure analysis and design. DFDs show the flow of data from external entities into the system. DFDs also show how the data moves and is transformed from one process to another, as well as its logical storage.

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart).

Theory:

Data flow diagram example:

Entity



This Represents an **Entity.** A source of data or a destination for data.

Process/Function

Process: Represented by circles in the diagram. Processes are responsible for manipulating the data. They take data as input and output an altered version of the data.

Database

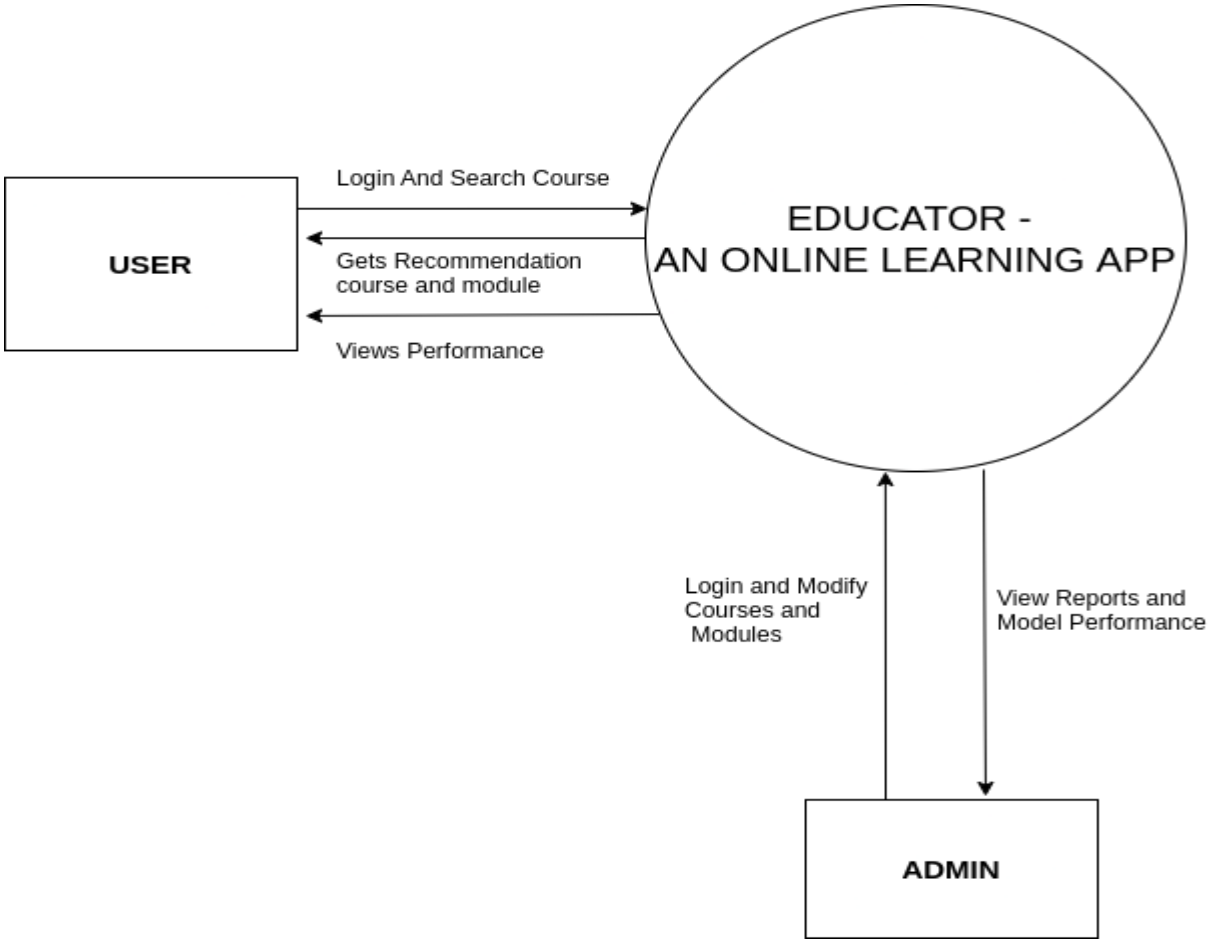
Database: Represented by a segmented rectangle with an open end on the right. Data Stores are both electronic and physical locations of data. Examples include databases, directories, files, and even filing cabinets and stacks of paper.

Data Flow

Data Flow Direction: Represented by a unidirectional arrow. Data Flows show how data is moved through the System. Data Flows are labelled with a description of the data that is being passed through it.

Data flow diagrams are also known as **bubble charts**. DFD is a designing tool used in the top-down approach to Systems Design. This context-level DFD is next "exploded", to produce a Level 1 DFD that shows some of the detail of the system being modelled. The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present.In order for the system to do its job, and shows the flow of data between the various parts of the system. In the course of developing a set of levelled data flow diagrams the analyst/designer is forced to address how the system may be decomposed into component sub-systems, and to identify the transaction data in the data model.

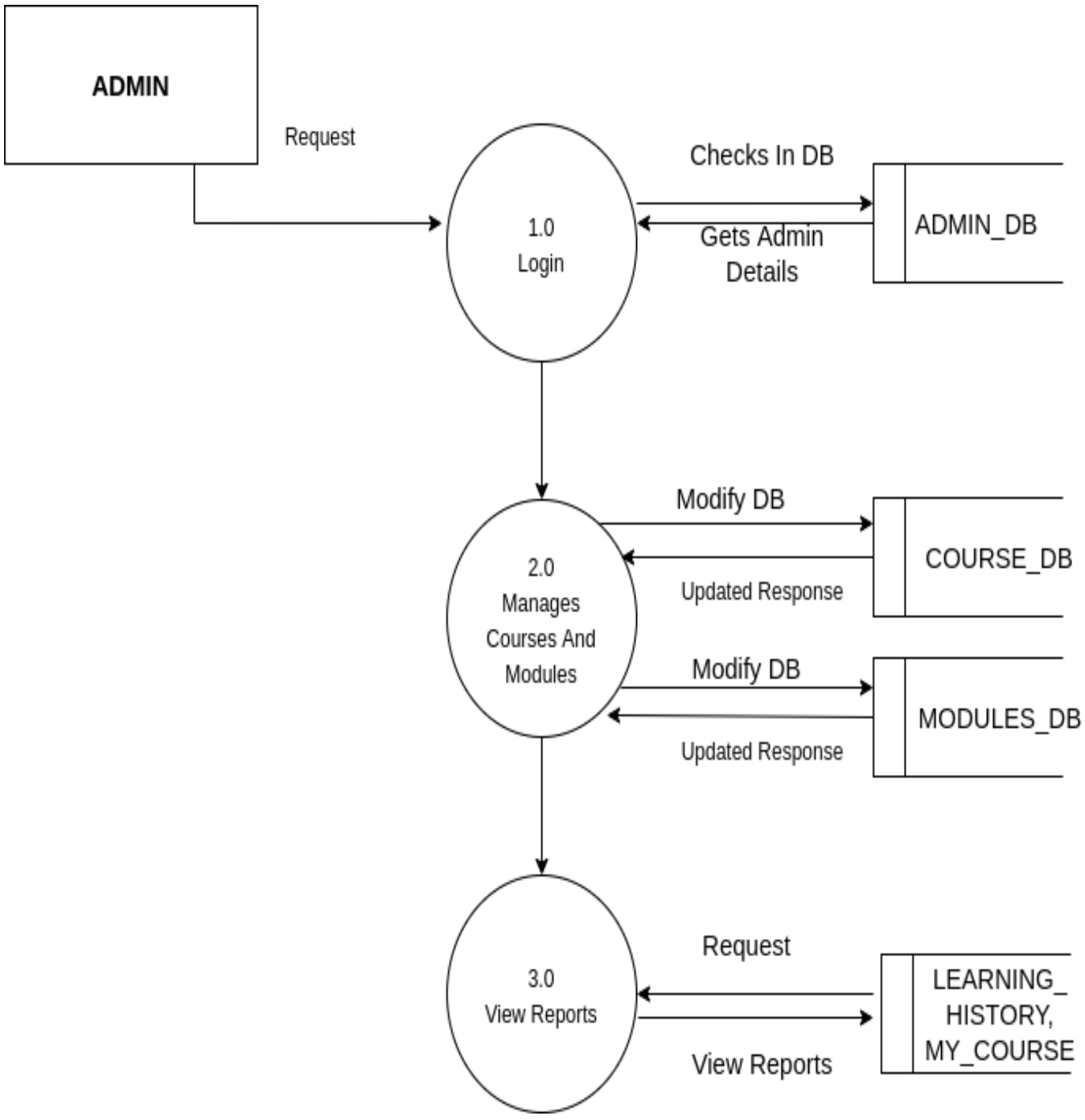
Level-0 DFD.



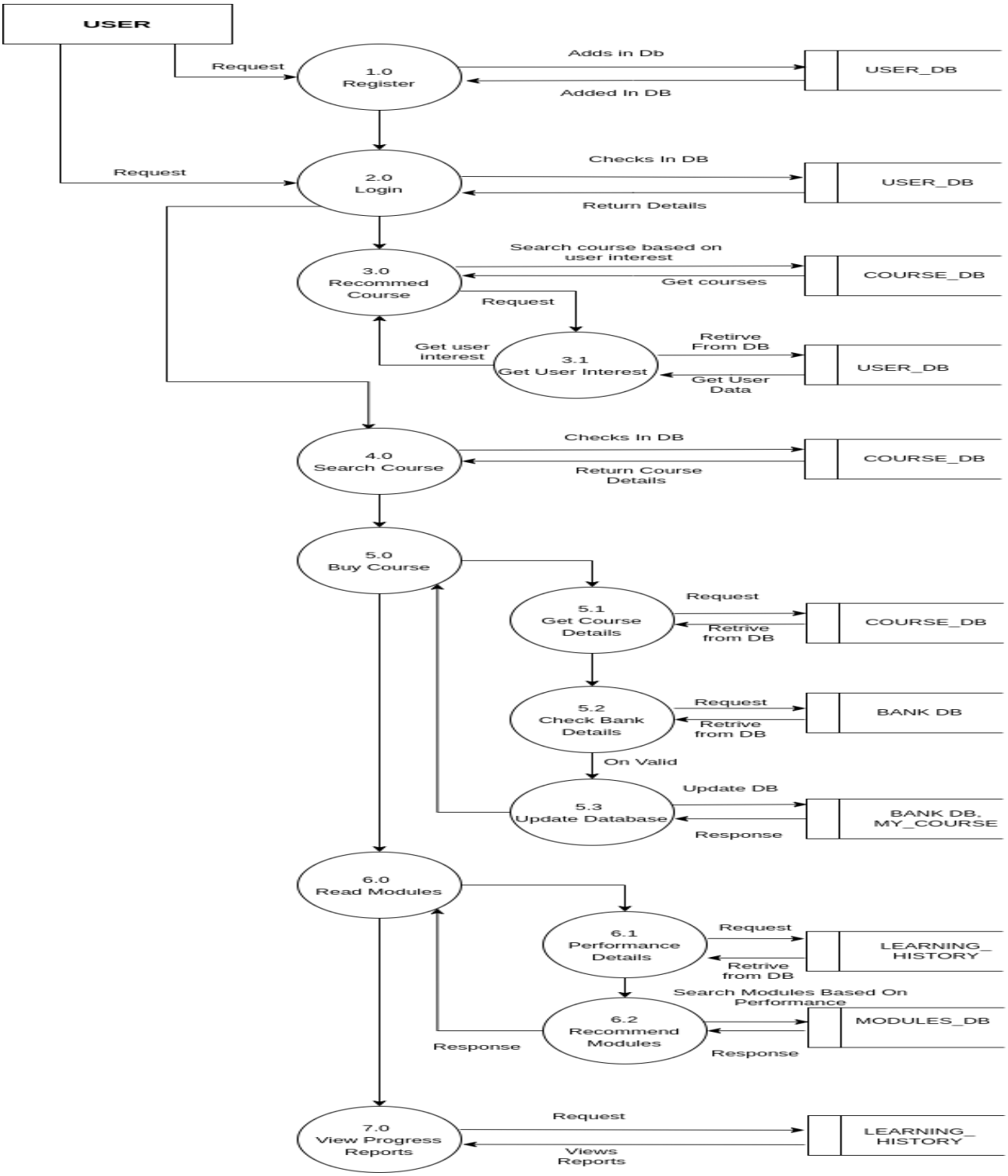
Level 1 DFD.

Level 1 DFD ‘s aim to give an overview of the full system. They look at the system in more detail. Major processes are broken down into sub-processes. Level 1 DFD ‘s also identifies data stores that are used by the major processes. When constructing a Level 1 DFD, we must start by examining the Context Level DFD. We must break up the single process into its sub-processes. We must then pick out the data stores from the text we are given and include them in our DFD. Like the Context Level DFD‘s, all entities, data stores and processes must be labelled. We must also state any assummade from the text.

Level 1 DFD (FOR ADMIN)



DFD LEVEL -1 (FOR USER)

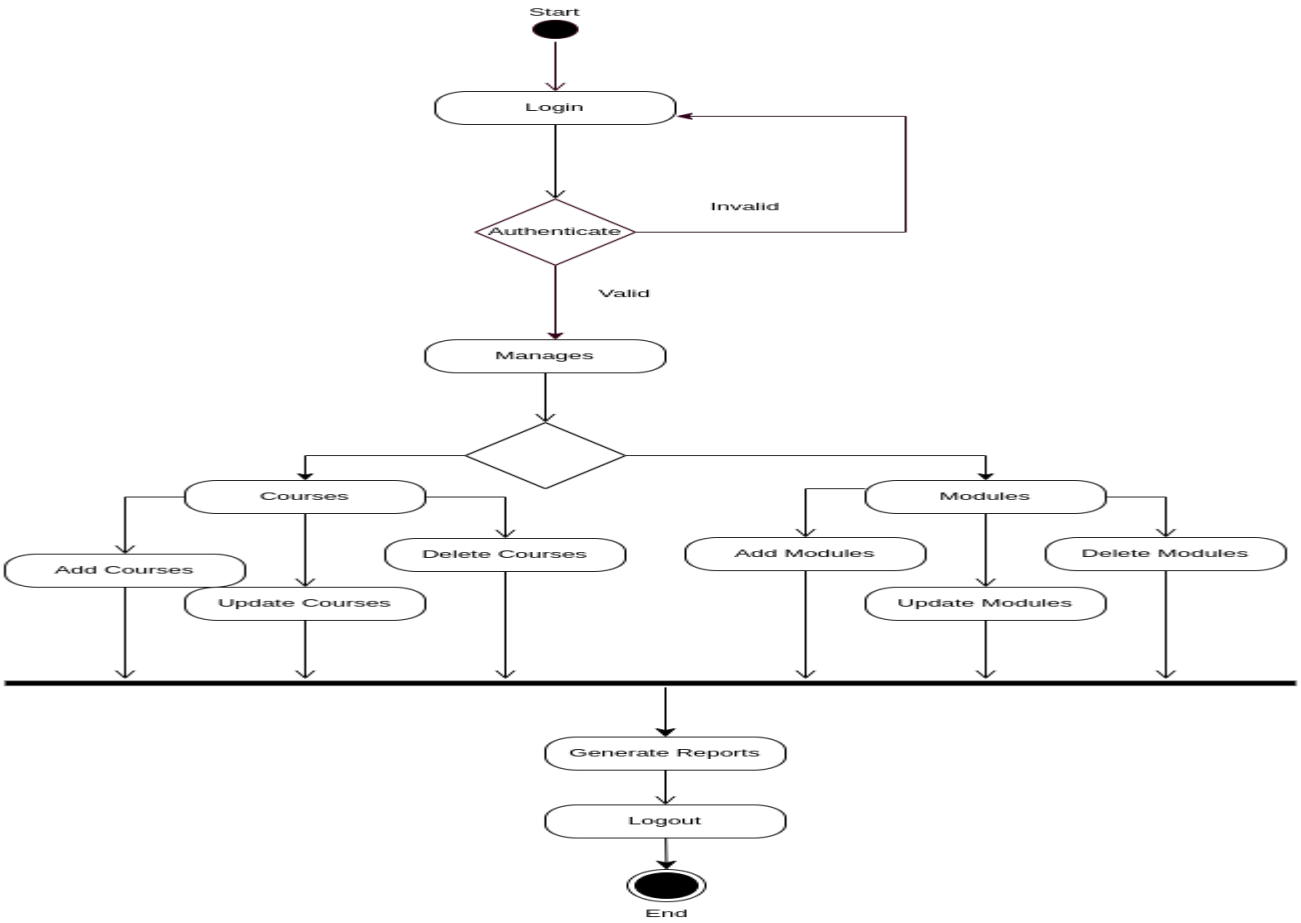


3.3 Activity Diagram

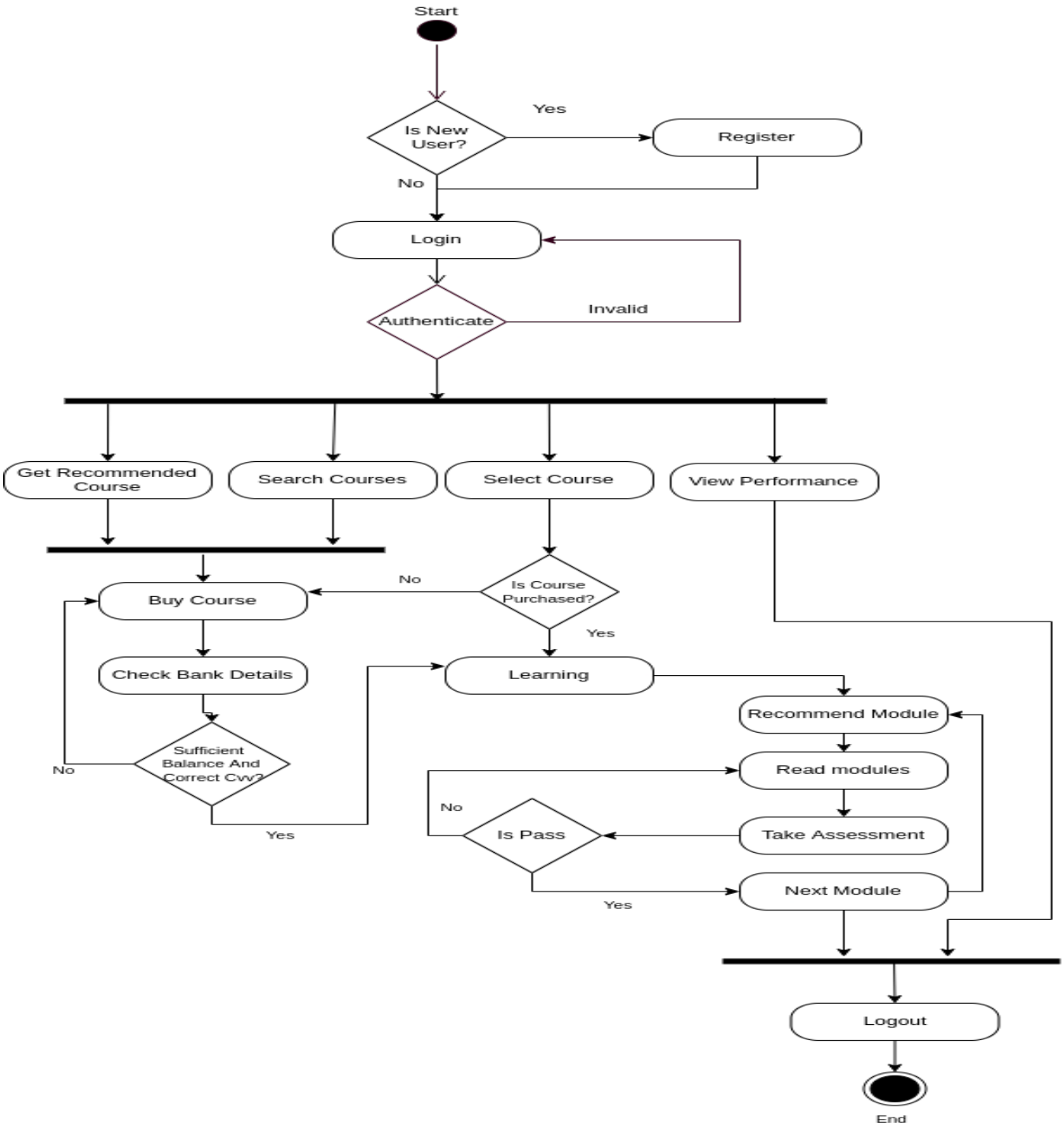
This Module Provides All The Activity Provided in The App. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed. We can depict both sequential processing and concurrent processing of activities using an activity diagram.

They are used in business and process modelling where their primary use is to depict the dynamic aspects of a system. Flowcharts were typically invented earlier than activity diagrams. Non-programmers use Flow charts to model workflows. For example: A manufacturer uses a flow chart to explain and illustrate how a particular product is manufactured. We can call a flowchart a primitive version of an activity diagram. Business processes where decision making is involved is expressed using a flow chart.

Admin Activity Diagram



User Activity Diagram



3.4 GANTT CHART

A Gantt chart is a type of bar chart, devised by Henry Gantt in the 1910s, that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements comprise the work breakdown structure of the project. Modern Gantt charts also show the dependency (i.e., precedence network) relationships between activities. Gantt charts can be used to show current schedule status using percent-complete shadings and a vertical "TODAY" line as shown here.

GANTT CHART BENEFITS:

Clarity: One of the biggest benefits of a Gantt chart is the tool's tool's ability to boil down multiple tasks and timelines into a single document. Stakeholders throughout an organization can easily understand where teams are in a process while grasping the ways in which independent elements come together toward project completion.

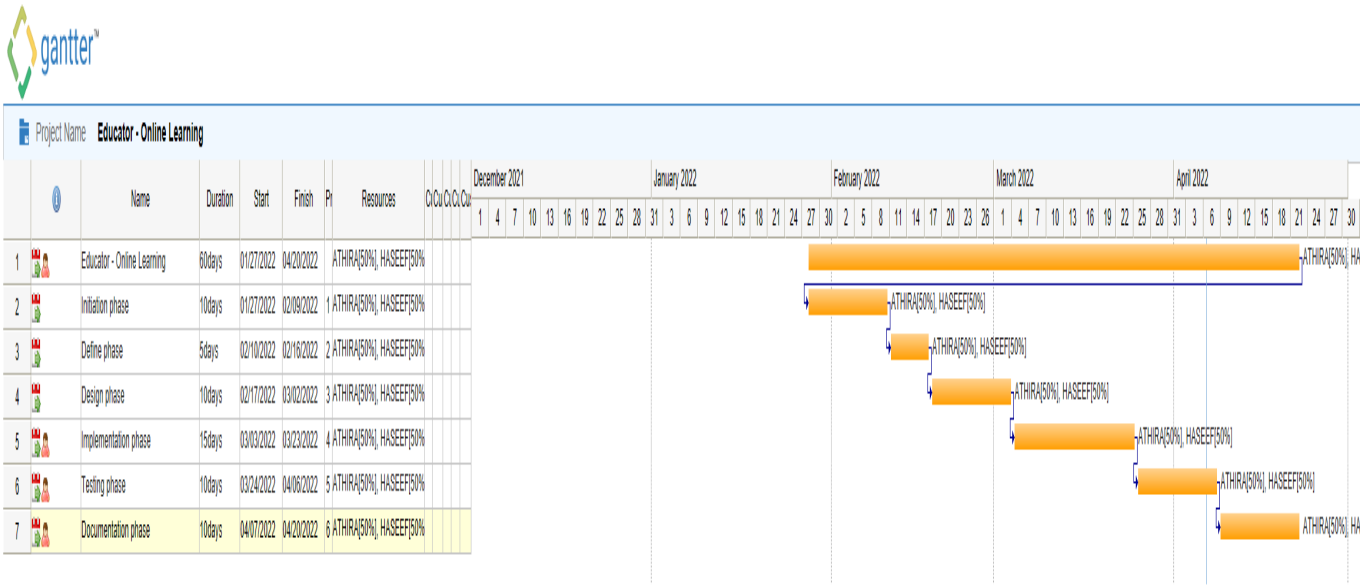
Communication: Teams can use Gantt charts to replace meetings and enhance other status updates. Simply clarifying chart positions offers an easy, visual method to help team members understand task progress.

Time Management: Most managers regard scheduling as one of the major benefits of Gantt charts in a creative environment. Helping teams understand the overall impact of project delays can foster stronger collaboration while encouraging better task organization.

Efficiency: Another one of the benefits of Gantt charts is the ability for team‘s members to leverage each other‘s deadlines for maximum efficiency. For instance, while one team member waits on the outcome of three other tasks before starting a crucial piece of the assignment, he or she can perform other project tasks.

Accountability: When project teams face major organizational change, documenting effort and outcomes becomes crucial to career success. Using Gantt charts during critical projects allows both project managers and participants to track team progress, highlighting both big wins and major failures.

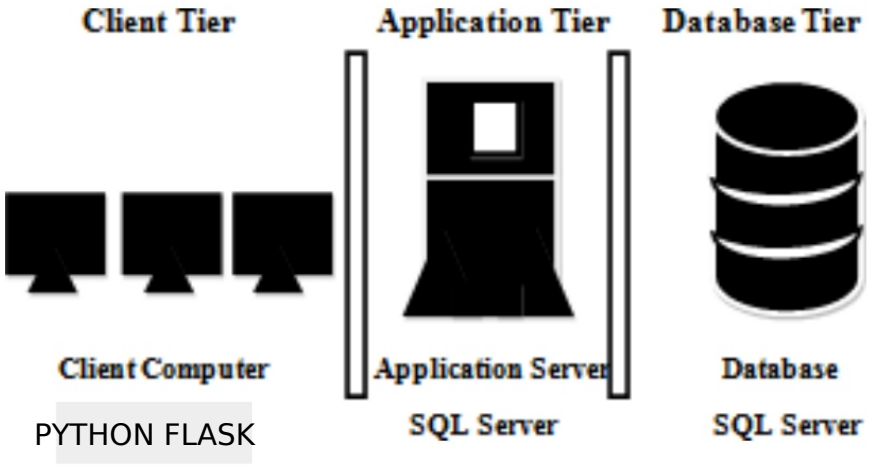
Gantt Chart


gantter
Project Name Educator - Online Learning
December 2021
January 2022
February 2022
March 2022
April 2022
1 Educator - Online Learning 60days 01/27/2022 04/20/2022 ATHIRA[50%], HASEEF[50%]
2 Initiation phase 10days 01/27/2022 02/09/2022 1 ATHIRA[50%], HASEEF[50%]
3 Define phase 5days 02/10/2022 02/16/2022 2 ATHIRA[50%], HASEEF[50%]
4 Design phase 10days 02/17/2022 03/02/2022 3 ATHIRA[50%], HASEEF[50%]
5 Implementation phase 15days 03/03/2022 03/23/2022 4 ATHIRA[50%], HASEEF[50%]
6 Testing phase 10days 03/24/2022 04/06/2022 5 ATHIRA[50%], HASEEF[50%]
7 Documentation phase 10days 04/07/2022 04/20/2022 6 ATHIRA[50%], HASEEF[50%]
PNG Generated On: 4/5/2022, 8:40:09 PM

3.5 Architecture Diagram

An "Architecture" can be defined as an abstract description of entities in a system and the relationships between them. It involves a series of decision-making processes. The architecture is a structure and a vision. A "system architecture" is the embodiment of concepts and the distribution of the correspondences between the functions of things or information and formal elements. It defines the relationships among elements as well as between elements and the surrounding environment.

Building a sound architecture is a complex task and a great topic for us to discuss here. After you build an architecture, relevant parties must understand it and follow its dictates. An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap.



3.6 Input/Output Design

Login and Register Page

```
<!DOCTYPE html>
<html lang="en">
<head>
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css" integrity="sha384-zCbKRCUGaJDkqS1kPbPd7TveP5iyJE0EjAuZQTgFLD2ylzuqKfdKlfG/eSrtxUkn" crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.min.js" integrity="sha384-VHvPCCyXqtD5DqJeNxl2dtTyhF78xXNXdkwX1CZeRusQfRKp+tA7hAShOK/B/fQ2" crossorigin="anonymous"></script>
                <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js" integrity="sha384-fQybjgWLrvvRgtW6bFlB7jaZrFsaBXjsOMm/tB9LTS58ONXgqbR9W8oWht/amnpF" crossorigin="anonymous"></script>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet" href="{{ url_for('static', filename= 'css/index.css') }}">
   <title>Educator |Login </title>
</head>
<body>
   <h1 class="Logo"> EDUCATOR</h1>
   <span class="slogan">An Online Learning App</span>
   <div class="login-wrap">
      <div class="login-html">
         <img src="/static/images/E-LOGO2.png" class="img-logo"/>
               <input id="tab-1" type="radio" name="tab" class="sign-in" checked><label for="tab-1" class="tab">Sign
            In</label>
         <input id="tab-2" type="radio" name="tab" class="sign-up"><label for="tab-2" class="tab">Sign Up</label>
         <div class="login-form">
            <form action="/login" method="POST">
```

```
            <div class="sign-in-htm">
              <div class="group">
                <label for="user" class="label">Username</label>
                <input id="user" type="email" class="input" name="username" required="">
              </div>
              <div class="group">
                <label for="pass" class="label">Password</label>
                                <input id="pass" type="password" class="input" data-type="password" name="password" required="">
              </div>
              <div class="group">
                <input id="check" type="checkbox" class="check" checked>
                <label for="check"><span class="icon"></span> Keep me Signed in</label>
              </div>
              <div>
                {% with messages = get_flashed_messages(with_categories=true) %}
                {% if messages %}
                  {% for category, message in messages %}
                  <div  class="alert alert-warning" role="alert">{{ message }}</div>
                  {% endfor %}
                {% endif %}
                {% endwith %}
              </div>
              <div class="group">
                <input type="submit" class="button" value="Sign In">
              </div>
              <div class="hr"></div>
              <div class="foot-lnk">
                <a href="{{url_for('admin_login')}}">Admin?</a>
              </div>
            </div>
          </form>
          <div class="sign-up-htm">
```

```
            <div class="group">
                <label for="pass" class="label">Repeat Password</label>
                    <input id="pass" type="password" class="input" data-type="password" name="conf_pass" required="">
                </div>
                <div class="group">
                    <input type="submit" class="button" value="Sign Up">
                </div>
                <div class="hr"></div>
                <div class="foot-lnk">
                    <label for="tab-1">Already Member?</a>
                </div>
            </form>
          </div>
        </div>
      </div>
   </div>
</body>
</html>
```

Home Page:

```
{% extends 'base.html'%}
{% block title%}
HomePage
{% endblock %}
{% block home%}
active
{%endblock%}
{% block cssblock%}
<link rel="stylesheet" href="{{ url_for('static', filename= 'css/home.css') }}">
<!-- <link href="https://unpkg.com/aos@2.3.1/dist/aos.css" rel="stylesheet"> -->
{% endblock %}
{% block Bodyblock%}
<div class="Home-Content">
  <div class="Top-Content">
    <h1 class="Username-style">
      Welcome {{session['username'].split('@').0 }}
    </h1>
    <div class="Content-top">
```

```
    A Best Place For Online And Customized Studies.
    <p>
     Personalized learning is a teaching model based on that premise.
     Each student gets a “learning plan” based on how they learn, what they know,
     and what their skills and interests are. It’s the opposite of the “one size fits all”
     approach used in most schools.
     Students work with their teachers to set both short-term and long-term goals.
     This process helps students take ownership of their learning.
    </p>
   </div>
   <img src="/static/images/sitting.png" class="sitting" />
   <img src="/static/images/miscellaneous.png" class="misc" />
  </div>
  <div class="second-top">
   <img src="/static/images/SVG1.svg" class="svg_image_top" />
   <h1 class="learning">
    Let's Start With Your Learning ....
   </h1>
   <div class="Contents">
    <div class=" row row-col-sm-6 card-container">
      {%for i in courselist%}
      <div class="card" data-aos="fade-up">
       <img class="card-img-top" src="{{url_for('static', filename= 'images/courseimage/'+i.url) }}"
        alt="Image Not Found....">
       <div class="card-body">
        <h5 class="card-title">{{i.course_title}}</h5>
        <p class="card-text">{{i.description}}</p>
        {% if i.registered %}
        <a href="{{url_for('payment_course',course_id = i.id)}}"> <button class="btn btn-outline-
success"> Buy @
          RS {{i.price}} </button></a>
        <button class="btn btn-outline-danger"> Add to Wishlist </button>
        {% endif %}
       </div>
      </div>
      {%endfor%}
    </div>
   </div>
  </div>
</div>
{% endblock %}
\
```

Modules Page

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

{% block title%}
```

```
Learning
{% endblock %}
{% block learning%}
active
{%endblock%}
{% block cssblock%}
<link rel="stylesheet" href="{{ url_for('static', filename= 'css/module.css') }}">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
{% endblock %}
{% block Bodyblock %}
<div class="top-content">
 {%if data%}
 <section>
  <div class="heading ">
   <h1 class="row">
    {{subjectname}}
   </h1>
   <div class="cert" >
    {% for i in data%}
     {% if i[2] == 'Finaltest' %}
     {% if i[7] != 0 %}
      <button class="btn btn-lg btn-primary" onclick="show('{{i[0]}}')">View Certificate</button>
     {% endif %}
     {% endif %}
    {% endfor %}
   </div>
  </div>
  <div class="row ">

   <div class="col-lg description">
    description
    <div class="row">
```

```
<div class="notes-details col-lg">
                    <video width="100%" height="100%" controls id="myVideo" src="/static/course_content/educator-front.mp4" type="video/mp4" >
  Your browser does not support the video tag.
 </video>
</div>
</div>
{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}
  {% for category, message in messages %}
  <div  class="alert">{{ message }} <br>
    <button type="button"  class="close-btn" data-dismiss="alert" >
     Okay
    </button>
  </div>
  {% endfor %}
{% endif %}
{% endwith %}
</div>
<div class=" course-content col-lg">
 <h1>
  Modules
 </h1>
 <hr>
 <div class="drop-down ">
  {% set count = namespace(value=0) %}
  {% for i in data%}
   {% if i[2] == 'test' or i[2] == 'Finaltest' %}
                {% if i[7] == 1 %}                <a class="dropdown-item" href="{{url_for('test',testname=i[3],c_id=i[0],l_id=i[1])}}">{{i[3]}} <i class="fa fa-check" style="font-size:24px;  color: rgb(80, 245, 80);"></i></a>
                {%else%}                <a class="dropdown-item" href="{{url_for('test',testname=i[3],c_id=i[0],l_id=i[1])}}">{{i[3]}} <i class="fa fa-edit" style="font-size:24px"></i></a>
```

```
            {% endif %}
          {%else%}
            {% if i[7] == 0 %}
              {% set count.value = count.value + 1 %}
            {%else%}
              <a class="dropdown-item"  onclick="playvideo('{{i[5]}}','{{i[0]}}', '{{i[1]}}' )">{{i[3]}}    <i class="fa fa-check" style="font-size:24px; color: rgb(80, 245, 80);"></i> </a>
            {% endif %}
          {% endif %}
        {% endfor %}
      </div>
      {% endif %}
    </div>
  </section>
  {%else%}
  <section>
    An Error Occured Try Again Later...
  </section>
  {% endif %}
<script>
$.ajax({
          type: "POST",
          url: "/getcert_id",
          data:JSON.stringify({"c_id":cid}),
          contentType: "application/json",
          dataType: 'json',
          success: function(result) {
             // Location.reload()
             let id  = result['id']
             window.location.href = "/viewcertificate/"+id
          }
       });
}
```

```
$("#myVideo").bind("ended", function() {
  location.reload()
});
</script>
</div>
{% endblock %}
{% block javascript%}
<script src="/static/js/modules.js"></script>
{% endblock %}
```

Payment Page

```
{% extends 'base.html'%}
{% block title%}
Payment
{% endblock %}
{% block cssblock%}
<link rel="stylesheet" href="{{url_for('static',filename = 'css/payment_page.css')}}" />
{% endblock %}

{% block Bodyblock %}
<div class="payment_container">
   <div class="heading">
      <h1>
         Checkout
      </h1>
      <div class="payment_details">
         <div class="user_details">
            <h3>
               Purchased By.
               <hr>
               <br>
               {{session['username']}}
            </h3>
            <div>
```

```
        Details
    </div>
</div>
<div class="product_details">
    <div>
        <h4>
            Order Summary
        </h4>
        <hr>
        </div>
    </div>
</div>
{%set t = i.price | int %}
{% set tax = (t*10) /100 %}
<form action="/purchase/{{i.id}}" method="POST">
    <div class="payment">
        <h3>
            Payment
        </h3>
        <hr><div class="card__front card__part">
                    <div class="card__space-75">
                        <img class="card__back-square card__square" src="/static/images/chip.png">
                        <img class="card__back-logo card__logo" src="/static/images/visa_logo.png">
                    </div>
                </div>
            </div>
            {% endfor%}
        {% endif %}
        <div class="add-new">
            <a href="{{ url_for( 'addcard',course_id = i.id ) }}"><button type="button"
                    class="Symbol">+</button></a>
            Add new cards
        </div>
```

```
            </div>
        </div>
        <div>
            <input type="hidden" value="{{t}}" name="price"/>
            <input type="submit" class="btn btn-success btn-lg btn-block" value="Pay Now">
        </div>
    </form>
    <div>
        {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}
            {% for category, message in messages %}
            <div  class="alert alert-warning" role="alert">{{ message }}</div>
            {% endfor %}
        {% endif %}
        {% endwith %}
    </div>
    </div>
    <div class="bill">
        <div class="details">
            <p>Price</p>
            <b>
                <p> ₹{{i.price}}</p>
            </b>
            <p>Service Tax 10%</p>
            <p>₹{{tax}}</p>
            <h5>Sub-Total</h5>
            <h5> ₹{{t + tax}}</h5>
            <p> Discount</p>
            <p>₹{{tax}}</p>
            <h2 class="alert alert-success" role="alert">Total</h3>
                <h2 class="alert alert-success" role="alert">₹{{i.price}}</h3>
        </div>
    </div>
```

```
    </div>
</div>
{% endblock %}
```

Admin Home

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <link rel = "shortcut icon" href="/static/images/favicon.ico"/>
    <title>Educator | Admin</title>
</head>
<body>
    <nav class="navbar">
        <a class="navbar-brand" href="#">
            <h1 class="Logo"> EDUCATOR</h1>
            <span class="slogan">An Online Learning App</span>
         </a>
    </nav>
    <div class="Message">
        {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}
            {% for category, message in messages %}
            <div  class="alert alert-warning" role="alert">{{ message }}
                <button type="button" class="close" data-dismiss="alert" aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            {% endfor %}
        {% endif %}
        {% endwith %}
```

```
</div>
<section class="Options">
    <img src="/static/images/courseimage/bf2.jpg"/>
    </div>
    <div>
      <h4>
        Welcome Admin
      </h4>
    </div>
  </div>
  <div class="menuoptions">
      <button  id ="Home" class="active-menu"><i class='fas fa-home' style='font-size:36px'></i>   Home</button>
        <button  id ="Course" ><i class='fas fa-book-open' style='font-size:36px'></i>   Course</button>
       <button  id ="Modules" ><i class='fas fa-puzzle-piece' style='font-size:36px'></i>   Modules</button>
          <button  id ="Analize" ><i class='fas fa-poll' style='font-size:36px'></i>   Insights</button>
       <button  id ="Modules" onclick=" window.location.href = '{{url_for('admin_login')}}'" > <i class="fa fa-sign-out" style="font-size:36px"></i>   Logout</button>
  </div>
</section>
<section class="Container">
</section>
</body>
var course_list = []
const course_fcn = (options)=>{
  if (options == 'add'){
    addpage(`{% include "admin/addcourse.html"%}`,"#Course")
    $.ajax({
      type: "GET",
      url: "/getcoursecount",
      success: function(result) {
```

```
            let c_id = parseInt(result['c_id'])
            if (isNaN(c_id)){
                c_id = 0
            }
            $('#c_id').val(c_id)
        }
    });
} if (options == 'del'){
    addpage(`{% include "admin/deletecourse.html"%}`,"#Course")
    $.ajax({
        type: "POST",
if (options == 'update'){
    addpage(`{% include "admin/updatecourse.html"%}`,"#Course")
    $.ajax({
        type: "POST",
        url: "/getcouorseno",
        success: function(result) {
            course_list = result['result']
            for (let i in course_list)
            {
                $('#c_id').append('<option value="'+course_list[i][0]+'">'+course_list[i][0]+'</option>')
            }
        }
    });
}
}
var course_list = []
const Modules_fcn = (options)=> {
if (options =='Add'){
    addpage(`{% include "admin/admin_addmodules.html"%}`,"#Modules")
    $.ajax({
        type: "POST",
        url: "/getcouorseno",
```

```
            success: function(result) {
                course_list = result['result']
                for (let i in course_list)
                {
                    $('#c_id').append('<option value="'+course_list[i][0]+'">'+course_list[i][0]+'</option>')
                }
            }
        });
    }
    if (options =='del'){
        addpage(`{% include "admin/admin_deletemodules.html"%}`,"#Modules")
        $.ajax({
            type: "POST",
            url: "/getcouorseno",
            success: function(result) {
                course_list = result['result']
                for (let i in course_list)
                {
                    $('#c_id').append('<option value="'+course_list[i][0]+'">'+course_list[i][0]+'</option>')
                }

            }
        });
    }
}
     $('.question-paper').append(template)
     times++;
    $('#no_of_question').val(times)
}
const remove_question = ()=>{
    times -= 1
    $('#no_of_question').val(times)
    $('.question-paper').children().last().remove();
```

```
}</script>
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<script src="/static/js/chart2.js"></script>
</html>
```

Admin Course page

```
<h3 class="heading"> Courses </h3>
<div class="Course-options">
   <button class="opt" onclick="course_fcn('add')">
      <h4>
         Add Course
      </h4>
   </button>
   <button class="opt" onclick="course_fcn('update')" >
      <h4>
         Update Course
      </h4>
   </button>
   <button class="opt"  onclick="course_fcn('del')">
      <h4>
         Delete Course
      </h4>
   </button>
</div>
```

Admin Modules

```
<h3 class="heading"> Modules</h3>
<div class="Course-options">
   <button class="opt" id="AddModule" onclick="Modules_fcn('Add')">
      <h4>
         Add Module
      </h4>
   </button>
   </a>
   <button class="opt" id="DelModule" onclick="Modules_fcn('del')">
      <h4>
         Delete Module
      </h4>
   </button>
</div>
```

Make Test Page

```
<h3 class="heading">Make Test </h3>
<div class="modules-Details">
  <form action="/addtest" method="POST" enctype="multipart/form-data">
```

```
    <label>Lecture ID</label>
    <input type="text" name="c_id"  id='c_id' readonly required style="display: none;"/>
    <input type="text" name="l_id"  id='l_id' readonly required/>
    <label>Question Paper name </label>
    <input type="text" name="q_name"  required />
    <input type="text" name="no_of_question" value="1" id="no_of_question" />
    <div>
      <label>Select Paper type</label>
      <select name="papertype">
          <option value="test">Pre Course</option>
          <option value="Finaltest">Final Test</option>
          <option value="test">Other (Mid Term, etc)</option>
      </select>
    </div>
    <div class="question-paper" name = "q-paper">
      <div class="question">
       <label>Write Question</label>
       <input type="text" name="q0" id="q0" style="background-color: #015a7d;" required />
       <label>Weights</label>
       <select name="l0">
         <option value="0">Easy</option>
         <option value="1"> Average</option>
         <option value="2">Difficult</option>
       </select>
         <div class="options-div">
           <label>D:</label>
           <input type="text" name="answer0" id="opt1" />
           <input type="checkbox" value ='3' name='check0'/>
         </div>
       </div>

      </div>

    </div>
     <button type="button" onclick="create_questiontemplate()" class="btn btn-block btn-outline-
primary">Add Question</button>

     <button type="submit" class="btn btn-block btn-outline-danger">Add Test</button>
  </form>
</div>
```

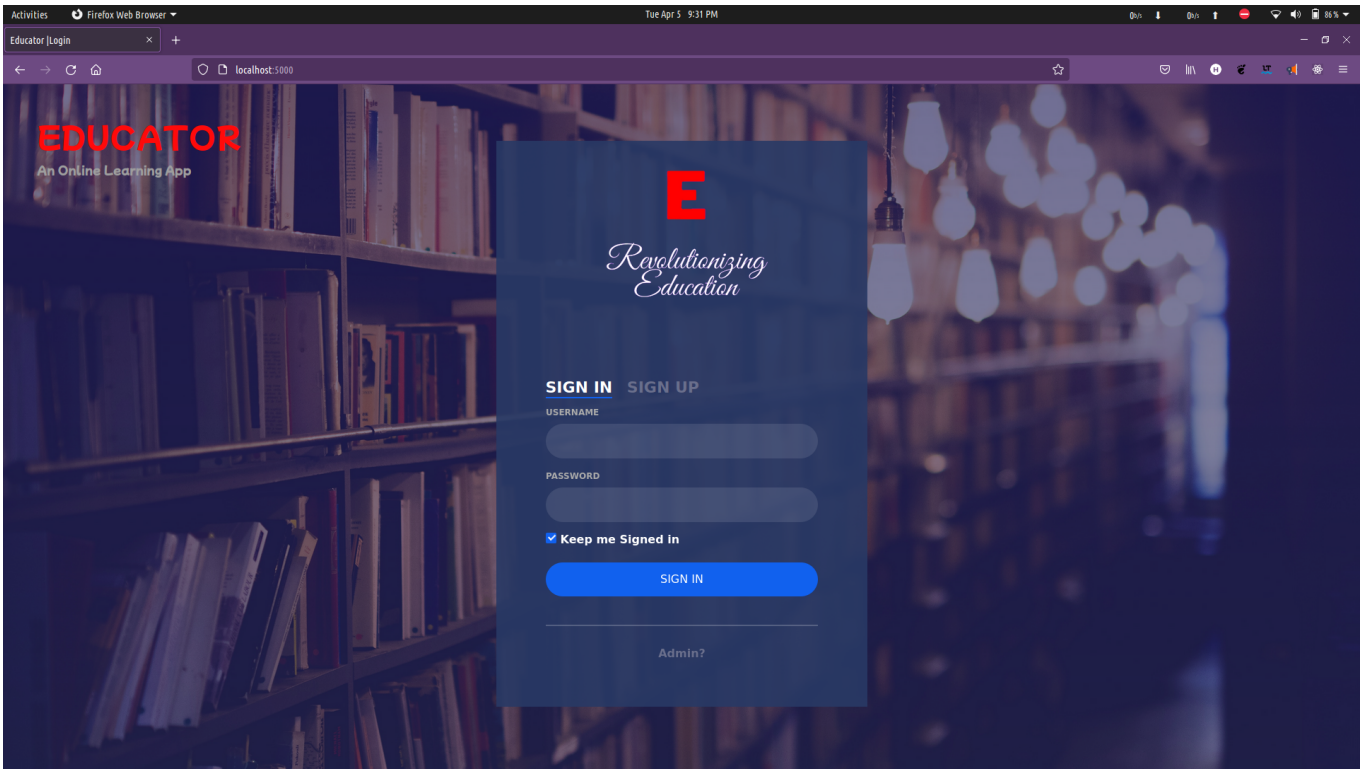
Insight Page

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

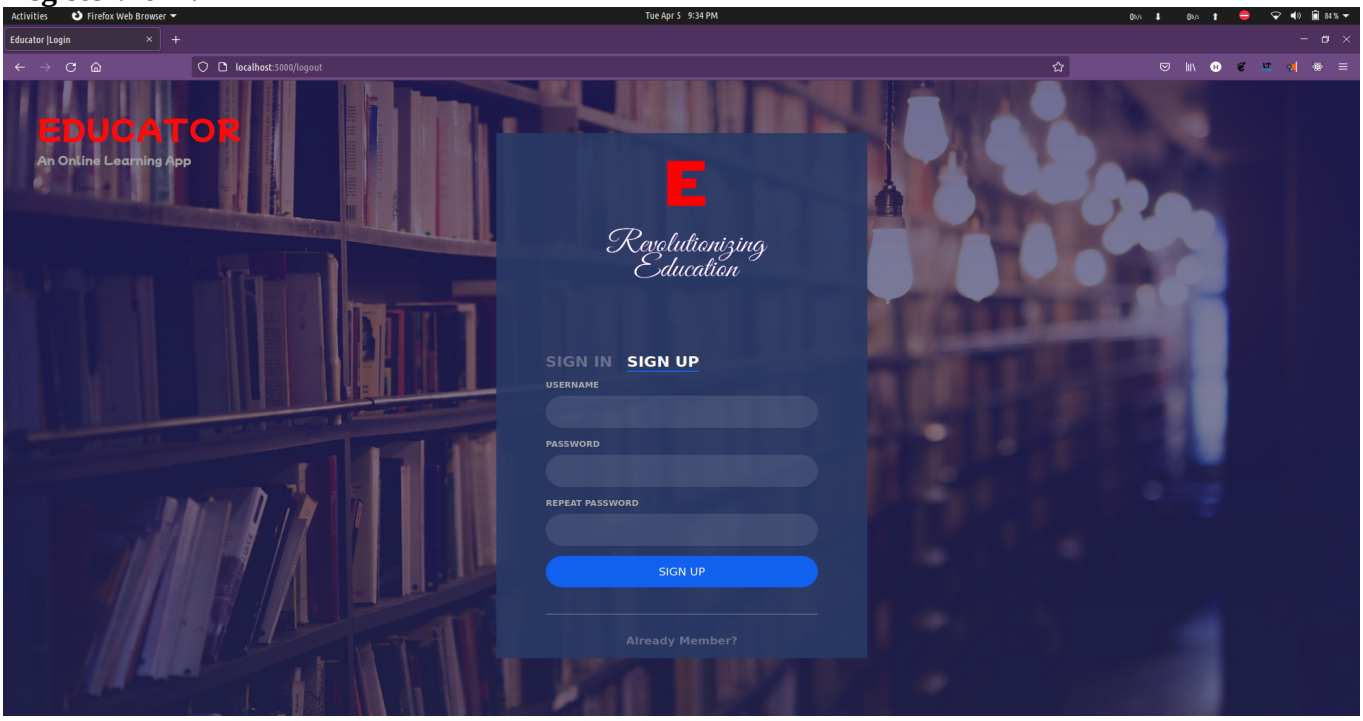
```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel = "shortcut icon" href="/static/images/favicon.ico"/>
    <title>Educator | Admin</title>
</head>
<body>
    <nav class="navbar">
        <a class="navbar-brand" href="#">
            <h1 class="Logo"> EDUCATOR</h1>
            <span class="slogan">An Online Learning App</span>
          </a>
    </nav>
    <div class="Message">
        {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}

            {% for category, message in messages %}
            <div  class="alert alert-warning" role="alert">{{ message }}
                <button type="button" class="close" data-dismiss="alert" aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            {% endfor %}
        {% endif %}
        {% endwith %}
    </div>
            <div class="row">
                <div class="col">
                    <div class="chart ">
                        <h5> Details </h5>
                        <div id="failed">
                            <div class="label-vals">
                                <label class="text-secondary">Color </label>
                                <label class="text-secondary"> lecture_id</label>
                            </div>
                        </div>
                    </div>
                </div>
        </div>
        </div>
    </section>
</body>
</html>
```

Outputs:

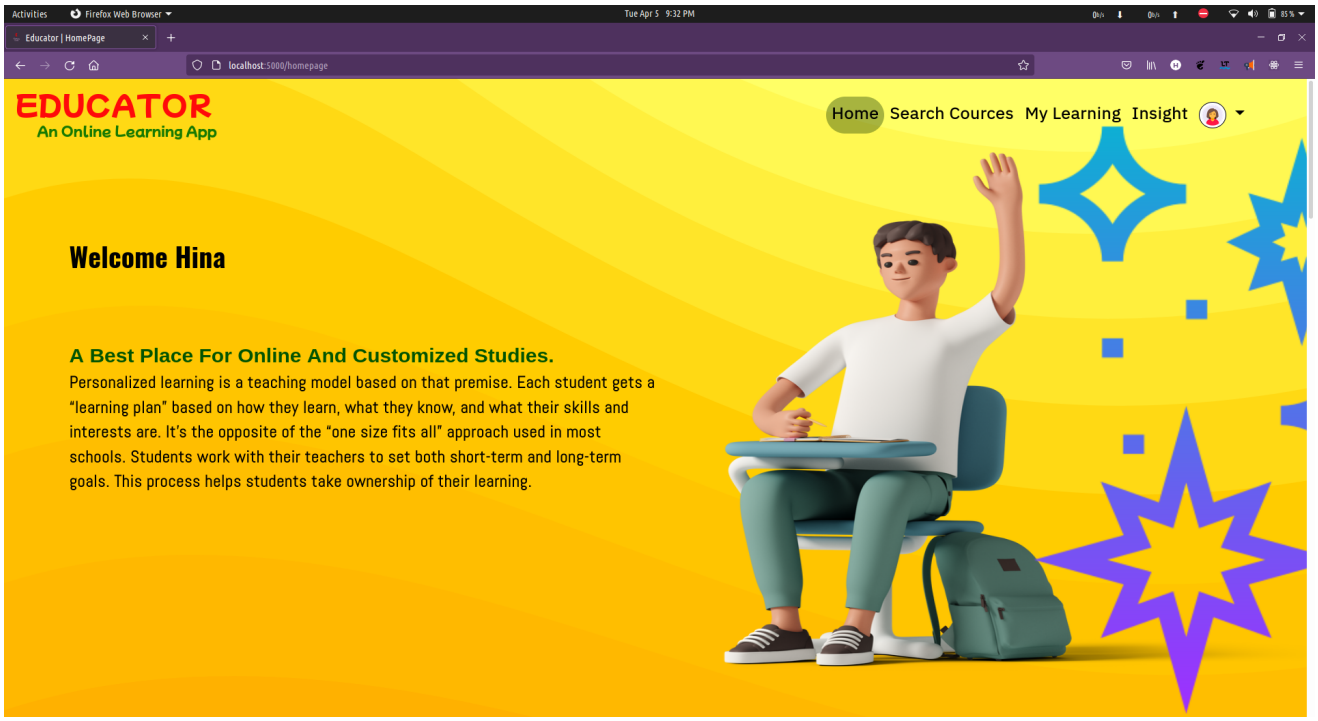
Login.html



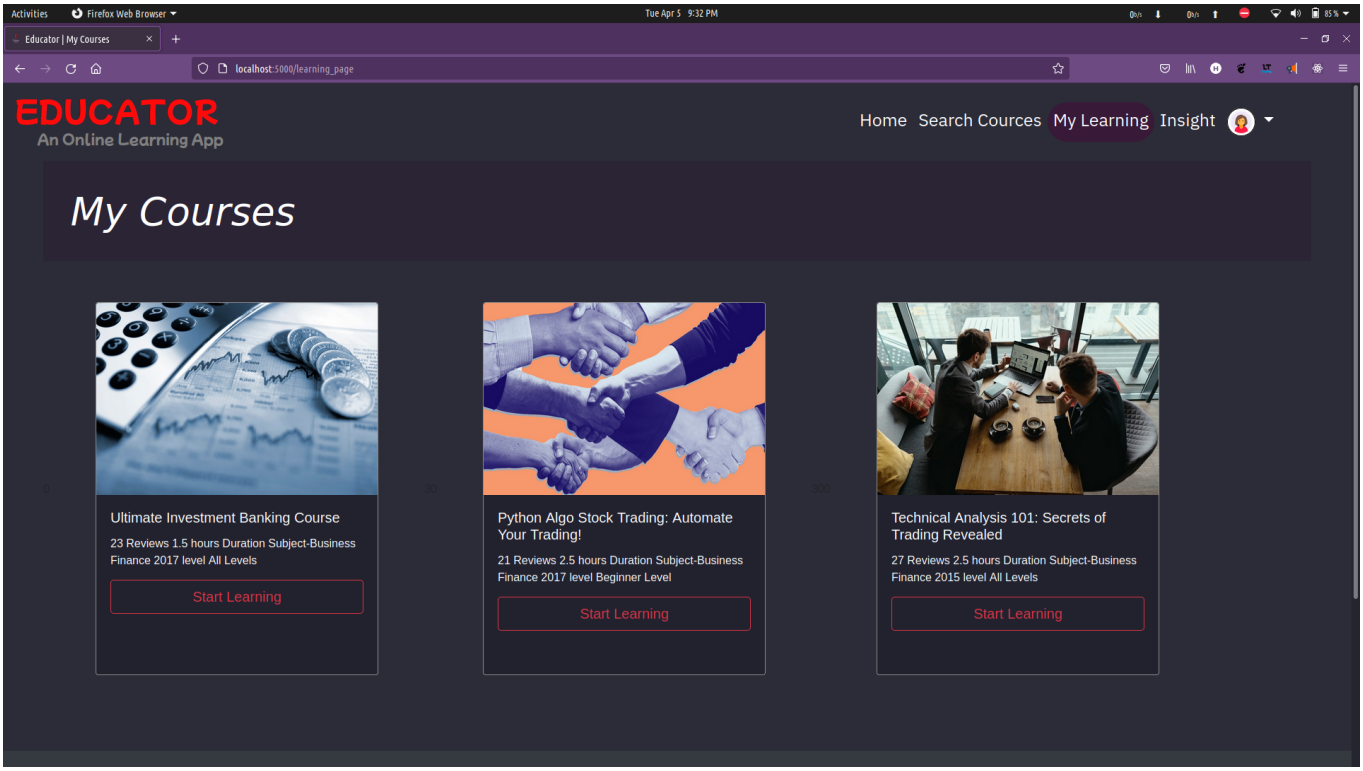
Register.html:



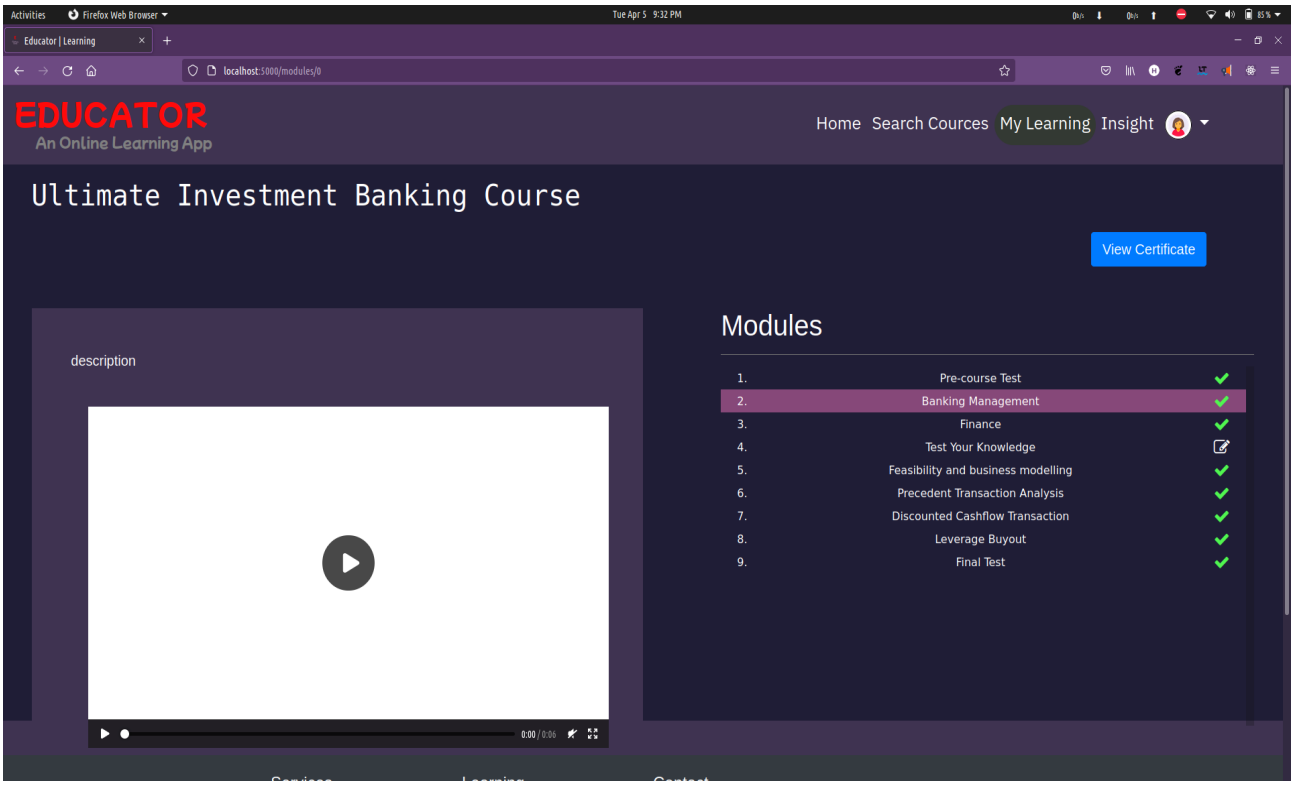
Home.html



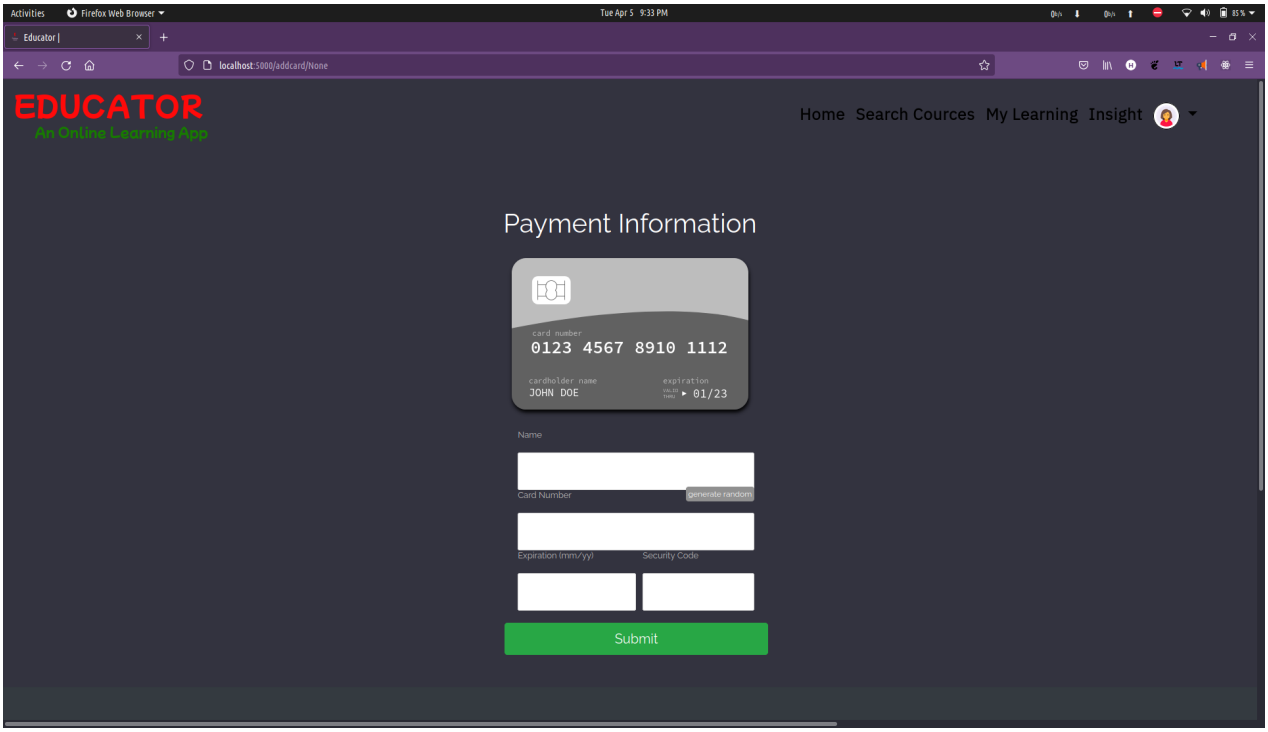
Learning.html



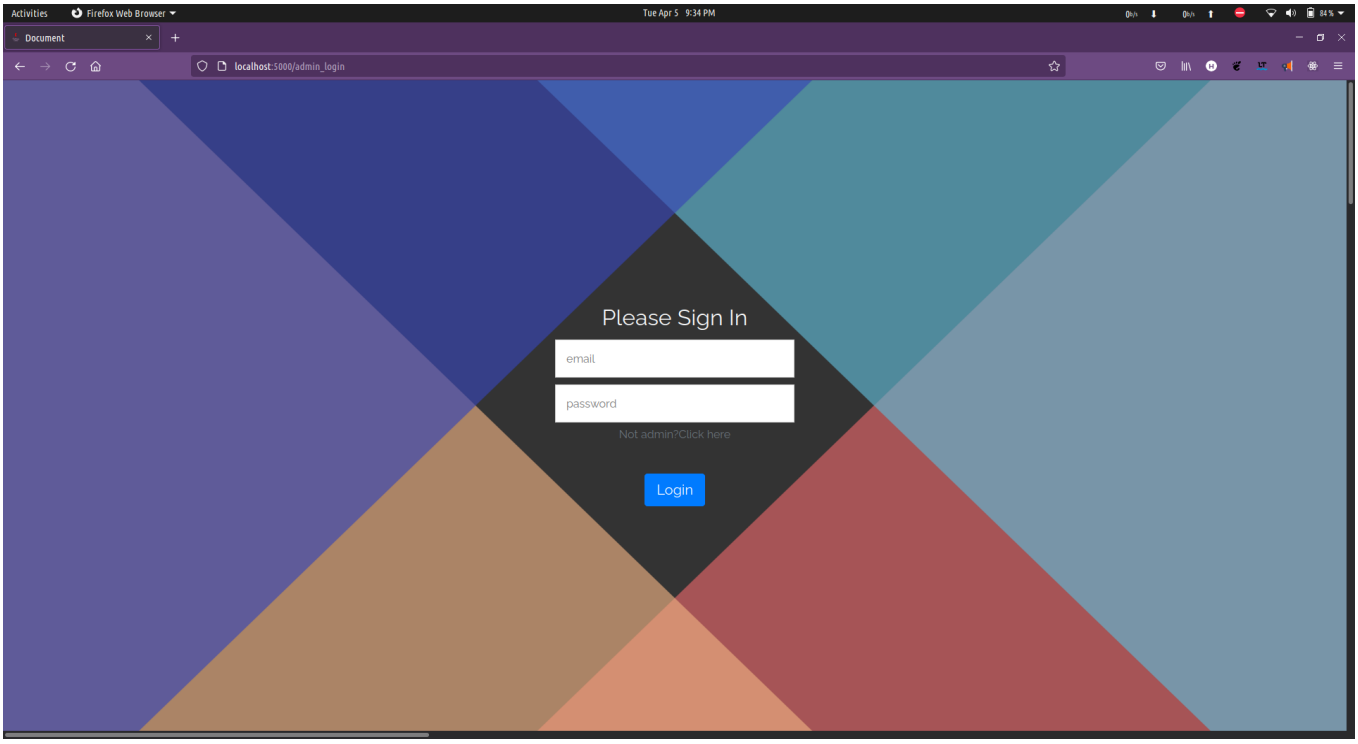
Modules.html



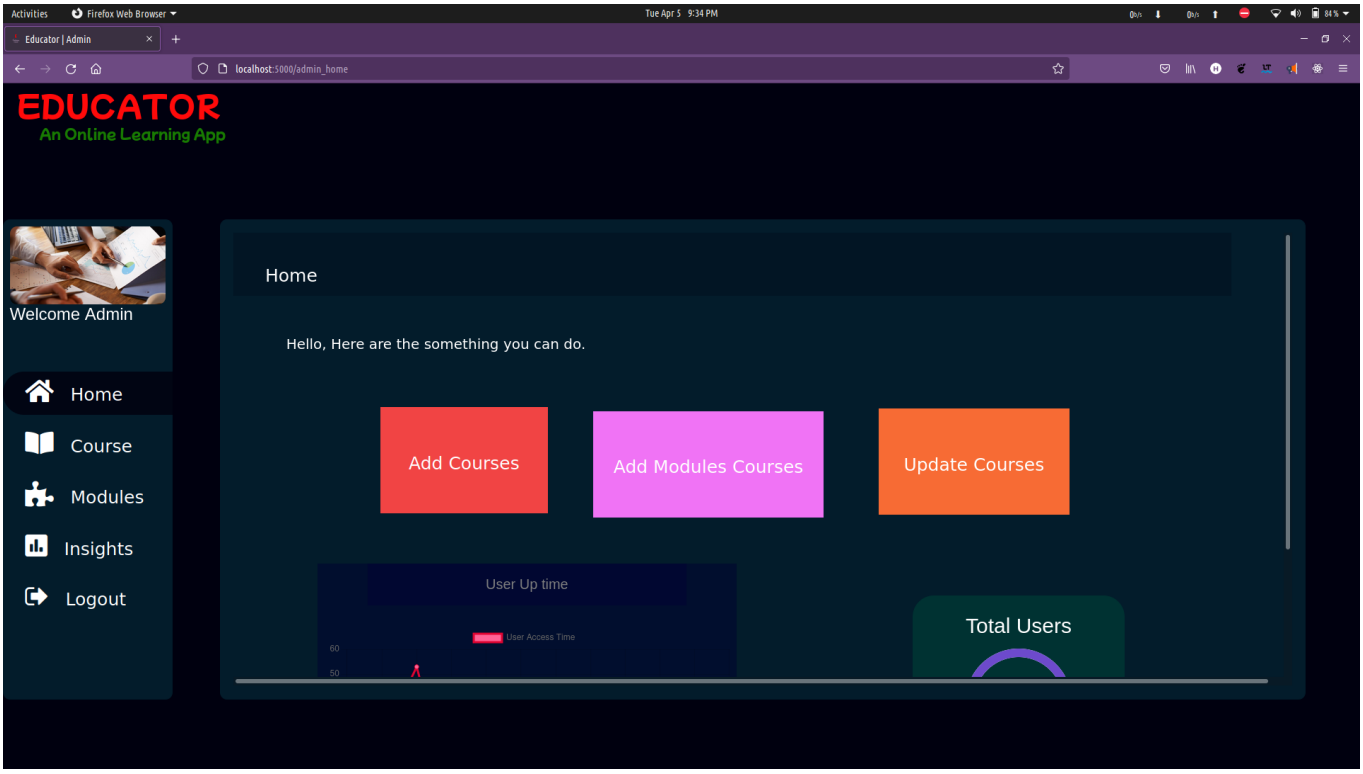
Payment.html



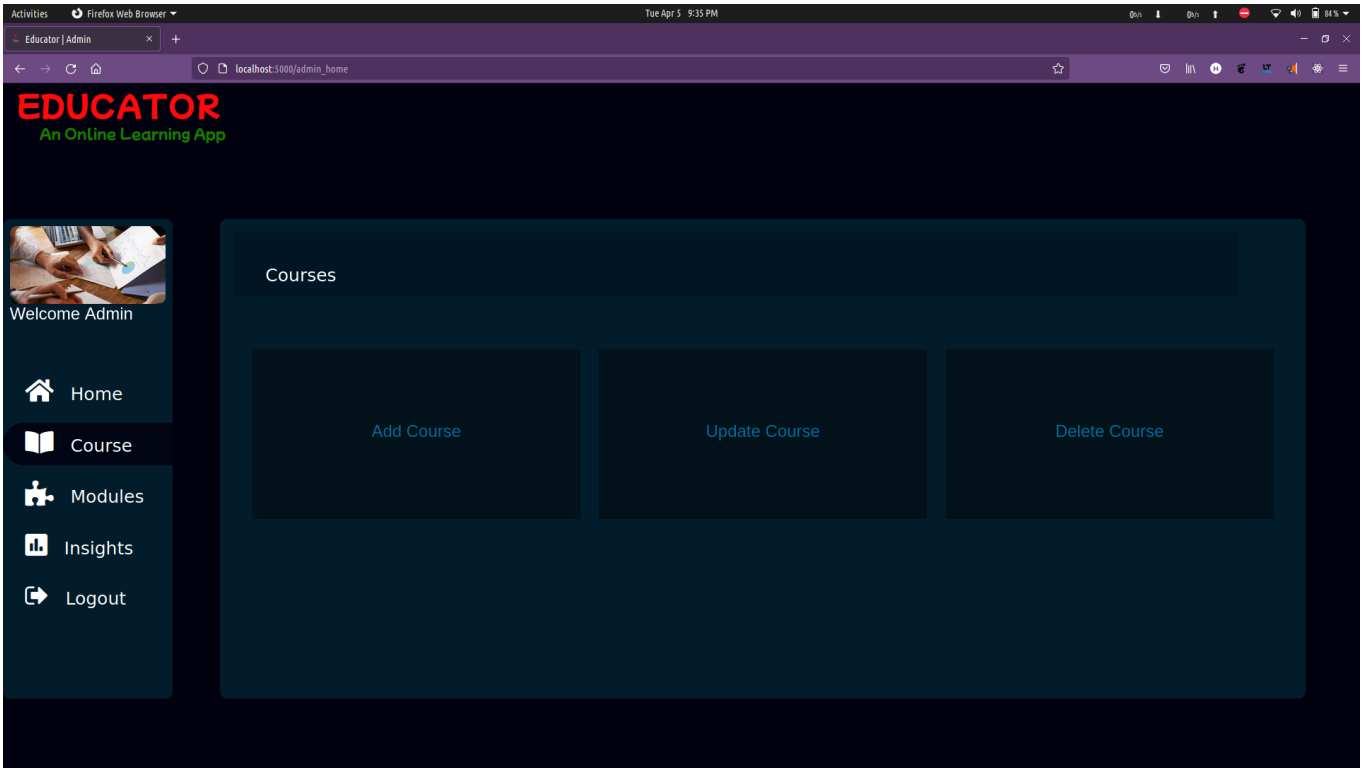
Admin.html



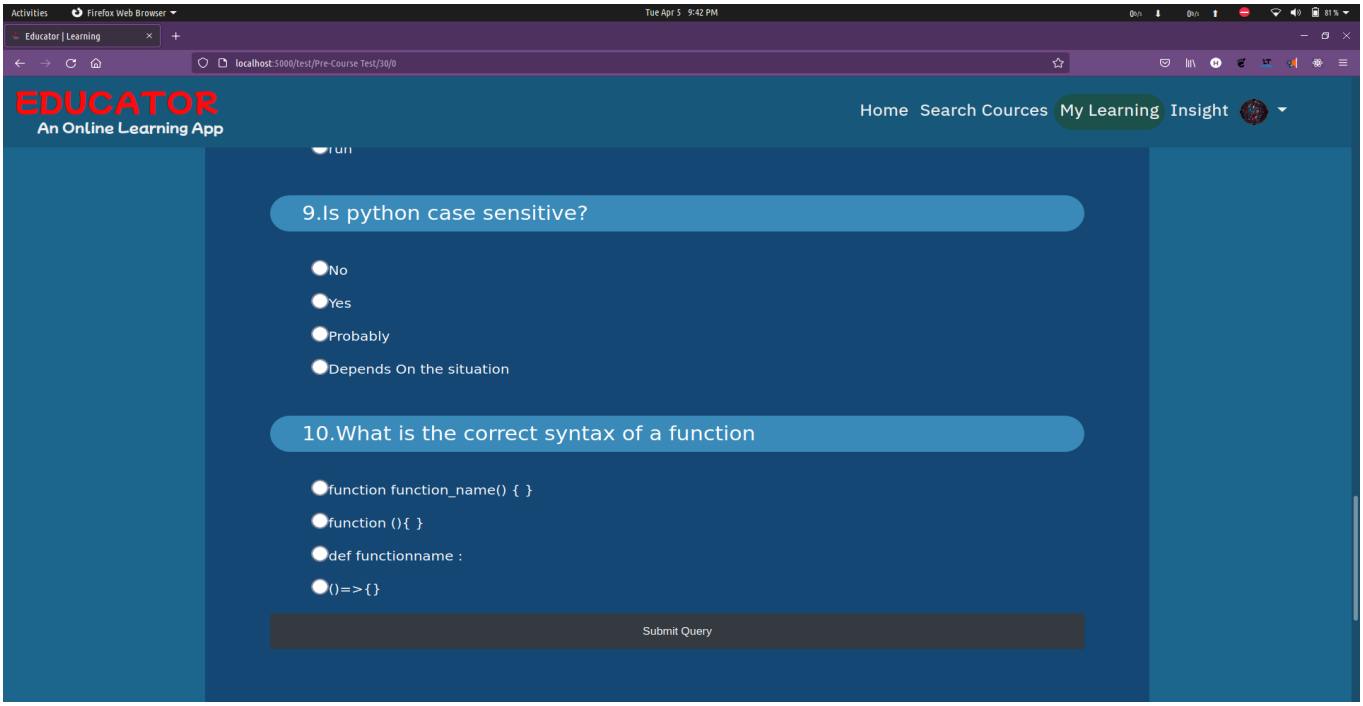
Admin Home.html



AdminCourse.html



AdminMakeTest.html



4.SYSTEM CONFIGURATION

4.1 HARDWARE REQUIREMENTS

RAM	4.00 GB AND ABOVE
HARD DISK	256 GB AND ABOVE
PROCESSOR	INTEL i3 AND ABOVE
PROCESSOR SPEED	2.06 GHz

4.2 SOFTWARE REQUIREMENTS

OPERATING SYSTEM	WINDOWS 10 , UBUNTU OS
PROGRAMMING LANGAUGE BACKEND FRAMEWORK	PYTHON FLASK,
FRONTEND FRAMEWORK	JINJA2,HTML,BOOTSTRAP
DATABASE	SQL SERVER
TOOL(S)	MICROSOFT PROJECT PLANNER, TEAM GANTT
DOCUMENTATION	MS WORD

5. DETAILS OF SOFTWARE

5.1 OVERVIEW OF FRONT END

JINJA2

Jinja is a fast, expressive, extensible templating engine. Special placeholders in the template allow writing code similar to Python syntax. Then the template is passed data to render the final document. code and managed code.It includes:

1. Template inheritance and inclusion.
2. Define and import macros within templates.
3. HTML templates can use autoescaping to prevent XSS from untrusted user input.
4. A sandboxed environment can safely render untrusted templates.
5. Async support for generating templates that automatically handle sync and async functions without extra syntax.
6. I18N support with Babel.
7. Templates are compiled to optimized Python code just-in-time and cached, or can be compiled ahead-of-time.
8. Exceptions point to the correct line in templates to make debugging easier.
9. Extensible filters, tests, functions, and even syntax.

Jinja uses a central object called the template Environment. Instances of this class are used to store the configuration and global objects, and are used to load templates from the file system or other locations. Even if you are creating templates from strings by using the constructor of Template class, an environment is created automatically for you, albeit a shared one.Most applications will create one Environment object on application initialization and use that to load templates. In some cases however, it’s useful to have multiple environments side by side, if different configurations are in use.

The simplest way to configure Jinja to load templates for your application is to use PackageLoader.

BOOTSTRAP

Quickly design and customize responsive mobile-first sites with Bootstrap, the world's most popular front-end open source toolkit, featuring Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful JavaScript plugins.

Originally created by a designer and a developer at Twitter, Bootstrap has become one of the most popular front-end frameworks and open source projects in the world.Bootstrap was created at Twitter in mid-2010 by @mdo and @fat. Prior to being an open-sourced framework, Bootstrap was known as Twitter Blueprint. A few months into development, Twitter held its first Hack Week and the project exploded as developers of all skill levels jumped in without any external guidance. It served as the style guide for internal tools development at the company for over a year before its public release, and continues to do so today.Originally released on Friday, August 19, 2011, we've since had over twenty releases, including two major rewrites with v2 and v3. With Bootstrap 2, we added responsive functionality to the entire framework as an optional stylesheet. Building on that with Bootstrap 3, we rewrote the library once more to make it responsive by default with a mobile first approach.

With Bootstrap 4, we once again rewrote the project to account for two key architectural changes: a migration to Sass and the move to CSS's flexbox. Our intention is to help in a small way to move the web development community forward by pushing for newer CSS properties, fewer dependencies, and new technologies across more modern browsers.

Bootstrap is maintained by a small team of developers on GitHub. We're actively looking to grow this team and would love to hear from you if you're excited about CSS at scale, writing and maintaining vanilla JavaScript plugins, and improving build tooling processes for frontend code.

5.2 OVERVIEW OF BACK-END

MICROSOFT SOL SERVER:

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications—which may run either on the same computer or on another computer across a network (including the Internet).

Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users. The protocol layer implements the external interface to SQL Server. All operations that can be invoked on SQL Server are communicated to it via a Microsoft-defined format, called Tabular Data Stream (TDS).

- Enables you to implement a database with tables, columns and indexes.
- Guarantees the Referential Integrity between rows of various tables.
- Updates the indexes automatically.
- Interprets an SQL query and combines information from various tables.

FLASK FRAMEWORK

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

Flask was created by Armin Ronacher of Pocoo, an international group of Python enthusiasts formed in 2004.According to Ronacher, the idea was originally an April Fool's joke that was popular enough to make into a serious application.The name is a play on the earlier Bottle framework.When Ronacher and Georg Brandl created a bulletin board system written in Python in 2004, the Pocoo projects Werkzeug and Jinja were developed.In April 2016, the Pocoo team was disbanded and development of Flask and related libraries passed to the newly formed Pallets project.

5.3 ABOUT THE PLATFORM

WINDOWS

Windows is a series of Operating Systems developed by Microsoft. Each version of Windows includes a Graphical User Interface, with a desktop that allows users to view files and folders in Windows. For the past two decades, Windows has been the most widely used operating system for personal computers PCs. Microsoft Windows is designed for both home computing and professional purposes. Past versions of Windows home editions include Windows 3.0 (1990), Windows 3.1 (1992), Windows 95 (1995), Windows 98 (1998), Windows Me (2000), Windows XP (2001), and Windows Vista (2006). The current version, Windows 7, was released in 2009. The first business-oriented version of Windows, called Windows NT 3.1, was in 1993.

UBUNTU-OS

Ubuntu is a complete Linux operating system, freely available with both community and professional support. The Ubuntu community is built on the ideas enshrined in the Ubuntu Manifesto: that software should be available free of charge, that software tools should be usable by people in their local language and despite any disabilities, and that people should have the freedom to customize and alter their software in whatever way they see fit.Ubuntu will always be free of charge, and there is no extra fee for the “enterprise edition”, we make our very best work available to everyone on the same Free terms.Ubuntu includes the very best in translations and accessibility infrastructure that the Free Software community has to offer.

6. TESTING

Testing is a vital part of software development, and it is important to start it as early as possible, and to make testing a part of the process of deciding requirements. To get the most useful perspective on your development project, it is worthwhile devoting some thought to the entire lifecycle including how feedback from users will influence the future of the application. The tools and techniques we've discussed in this book should help your team to be more responsive to changes without extra cost, despite the necessarily wide variety of different development processes. Nevertheless, new tools and process improvements should be adopted gradually, assessing the results after each step.

Software development life cycle

Testing is a proxy for the customer. You could conceivably do your testing by releasing it into the wild and waiting for the complaints and compliments to come back. Some companies have been accused of having such a strategy as their business model even before it became fashionable:

• Tests represent requirements. Whether you write user stories on sticky notes on the wall, or use cases in a big thick document, your tests should be derived from and linked to those requirements. And as we've said, devising tests is a good vehicle for discussing the requirements.

• We're not done till the tests pass. The only useful measure of completion is when tests have been performed successfully.

Those principles apply no matter how you develop your software.

Software Testing Types:

Unit testing - Testing of individual software components or modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code, may require developing test driver modules or test harnesses.

Integration testing - Testing of integrated modules to verify combined functionality after integration. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

Functional testing - This type of testing ignores the internal parts and focus on the output is as per requirement or not. Black-box type testing geared to functional requirements of an application. System

testing - Entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system.

Load testing - Its a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system's response time degrades or fails.

Stress testing - System is stressed beyond its specifications to check how and when it fails. Perform under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load. Performance testing - Term often used interchangeably with 'stress' and 'load' testing. To check whether system meets performance requirements. Used different performance and load tools to do this.

Install/uninstall testing - Tested for full, partial, or upgrade install/uninstall processes on different operating systems under different hardware, software environment. Recovery testing - Testing how well a system recovers from crashes, hardware failures, or other catastrophic problems.

During The Progress Of The Application , Unit Testing ,Regression Testing , Performance Testing were used.

Black box testing - Internal system design is not considered in this type of testing. Tests are based on requirements and functionality.

White box testing - This testing is based on knowledge of the internal logic of an application's code. Also known as Glass box Testing. Internal software and code working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions.

7. CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION

This is an online learning platform in which students can learn any course at their own pace. It is different from other online learning platform. That it not only customize and recommend courses, but also it customizes modules / syllabus to each and every student with the help of Deep/Machine learning algorithms. In other online app. The content provided was vague and general to all student. It was not specific to a particular student group. But our app “Educator” Gives a customized learning experience which Increases productivity in learning by providing a learning roadmap to it.

FUTURE ENHANCEMENT

This project can provide beneficial impact on student behavior and their learning pattern. So eventually this enables us to provide a better roadmap to the student to study the course effectively.

The ML model use in this will assist to classify the students based on their check result and deliver them higher experience in learning. Student will finally voluntarily take Initiative to learn and enhance themselves.

8.BIBLOGRAPHY

Book References:

- **Flask:** Basics of Flask And Python To Develop Web App by Michelle Arman, Published by Sunrise Publisher
  Jinja2: Essentials of Jinja by Sanjay Kumar, Published by Packt Publishing
- **SQL:** Introduction to SQL by Ramshaw Mask, Published by Dawn Publisher

Web References:

https://www.tutorialspoint.com
https://www.javatpoint.com
https://www.geeksforgeeks.com

9.APPENDICES A-TABLE STRUCTURE

User_login

Name	Datatype	Constrants	Description
Username	varchar(50)	Primary key	Identifies User
Password	varchar(30)	Not null	User password
Phone No	varchar(10)	nullable	User Phone no
Interest	varchar(60)	nullable	User interest
User_Image	BLOB(500)	nullable	User image

Course_list

Name	Datatype	Constrants	Description
course_id	varchar(50)	Primary key	Unique course_id
Course title	varchar(30)	Not null	Course Title
Description	varchar(50)	Not Null	Course Description
Price	int	Not Null	Course Price
Image	BLOB(500)	Not Null	Course Image
Lecture_count	int	Not Null	Total Tecture
level	int	Not Null	Level of course

Modules

Name	Datatype	Constrants	Description
course_id	varchar(50)	Primary key	Unique module ID
lecture_id	varchar(50)	Primary Key	Unique Lecture ID
content	varchar(50)	Not Null	Course Content
video	varchar(500)	Not Null	Video Location

Student_History

Name	Datatype	Constrants	Description
username	varchar(50)	Foreign key	Unique User ID
course_id	varchar(50)	Not Null	Unique Course id
Lecture_id	varchar(50)	Foreign Key	Unique lecture id
Is_Completed	int	Not Null	Completed modules
Result	varchar(10)	Not Null	Result of lecture

Purchased Courses

Name	Datatype	Constrants	Description
username	varchar(50)	Foreign key	Unique user id
course_id	varchar(50)	Not Null	Unique course id
Is_Completed	int	Not Null	Completed Course

TestQuestion

Name	Datatype	Constrants	Description
course_id	varchar(50)	Primary Key	Unique course id
lecture_id	varchar(50)	Primary Key	Unique lecture id
Question	varchar(50)	Not Null	Question name
Answer	varchar(10)	Not Null	Answer of queston
Weights	int	Not Null	Weights of Question
Options	varchar(50)	Nullable	Options of question

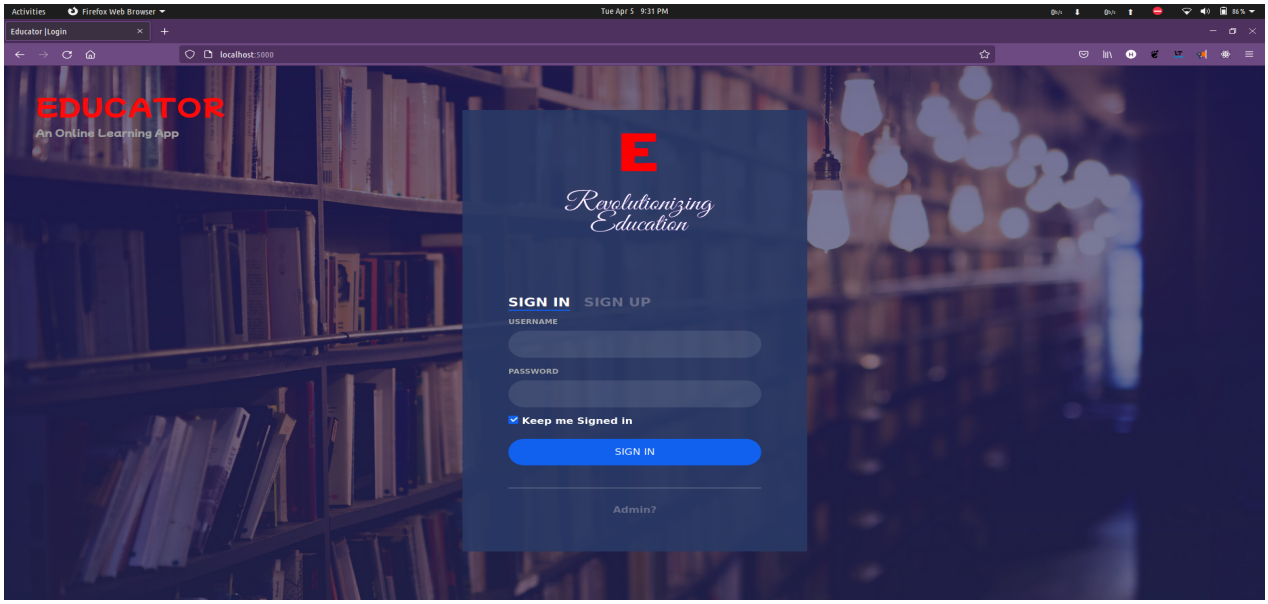
Bank_Details

Name	Datatype	Constrants	Description
Username	varchar(50)	Primary Key	Unique user id
Card_NO	varchar(15)	Primary Key	User Card No
Cardholder_name	varchar(50)	Not Null	User Card Holder name
CVV	int(4)	Not Null	Card CVV number
Expire	Date	Not Null	Card Expiry

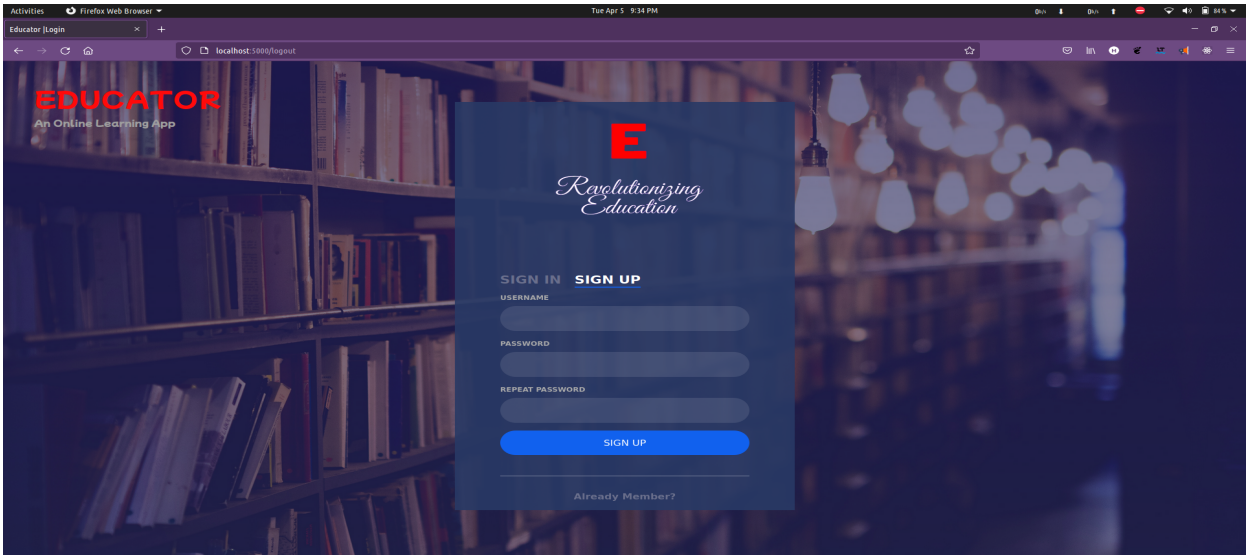
10.APPENDICES B-SCREENSHOTS

User Validation Module:

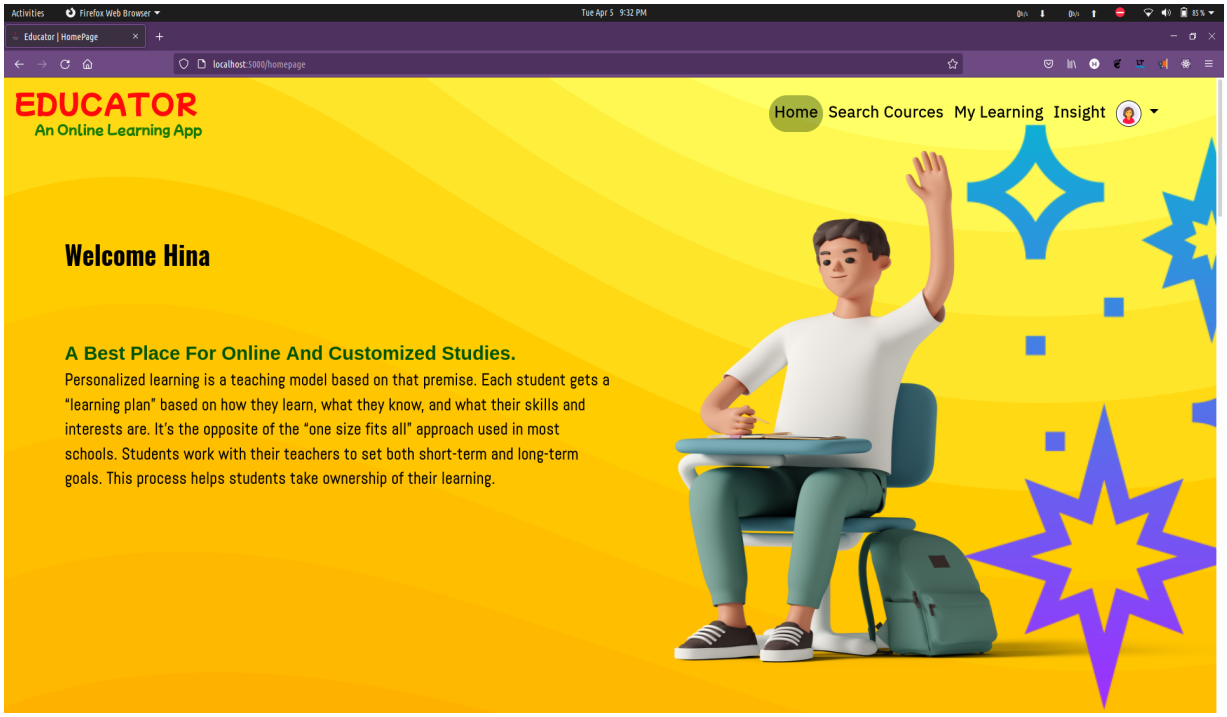
Login Page



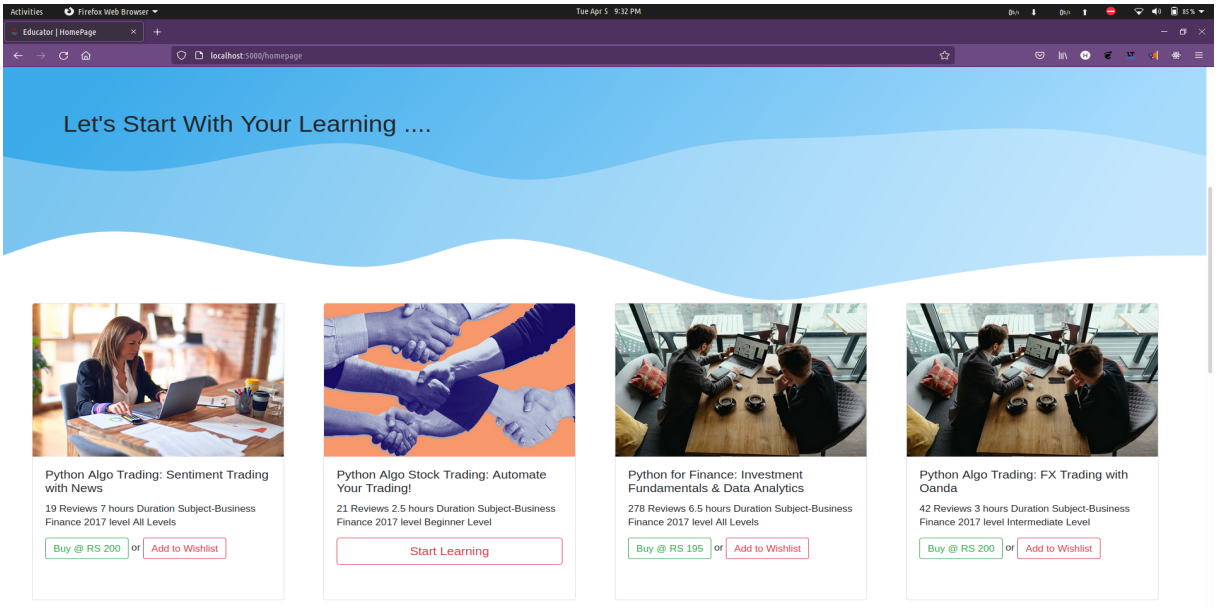
Registration Page



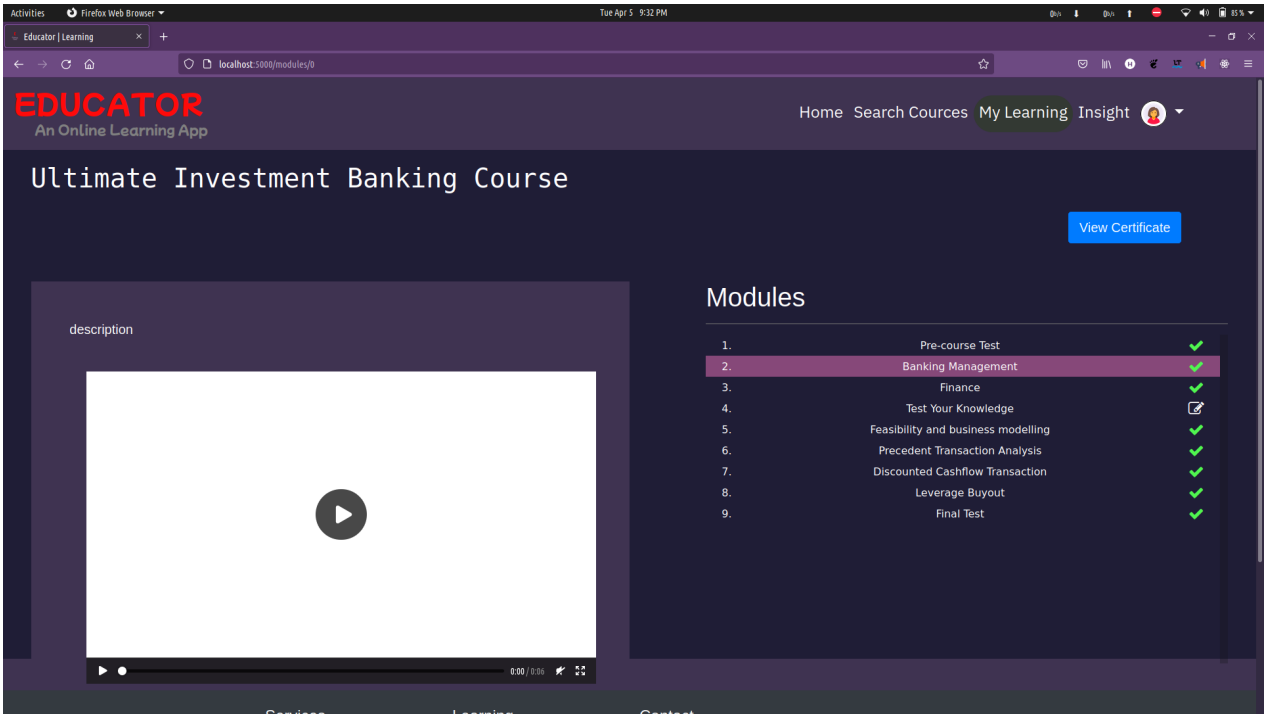
Home Page



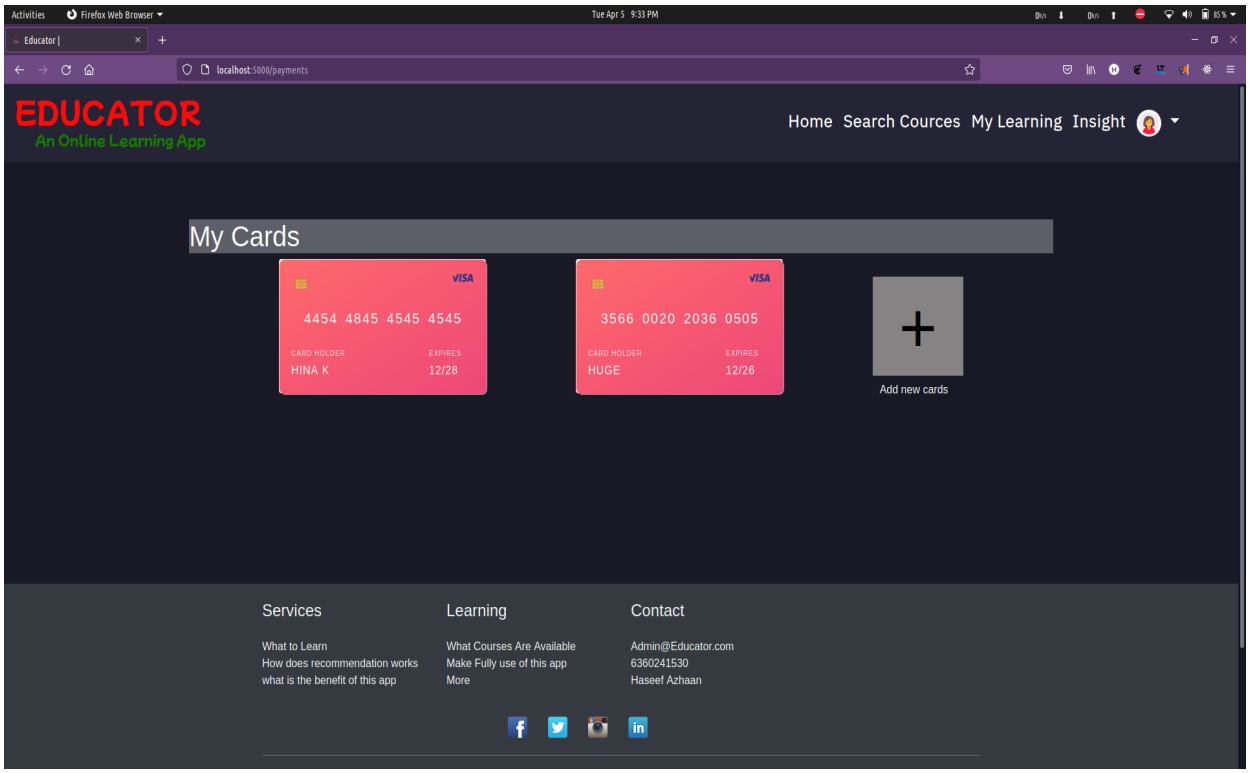
Course Page



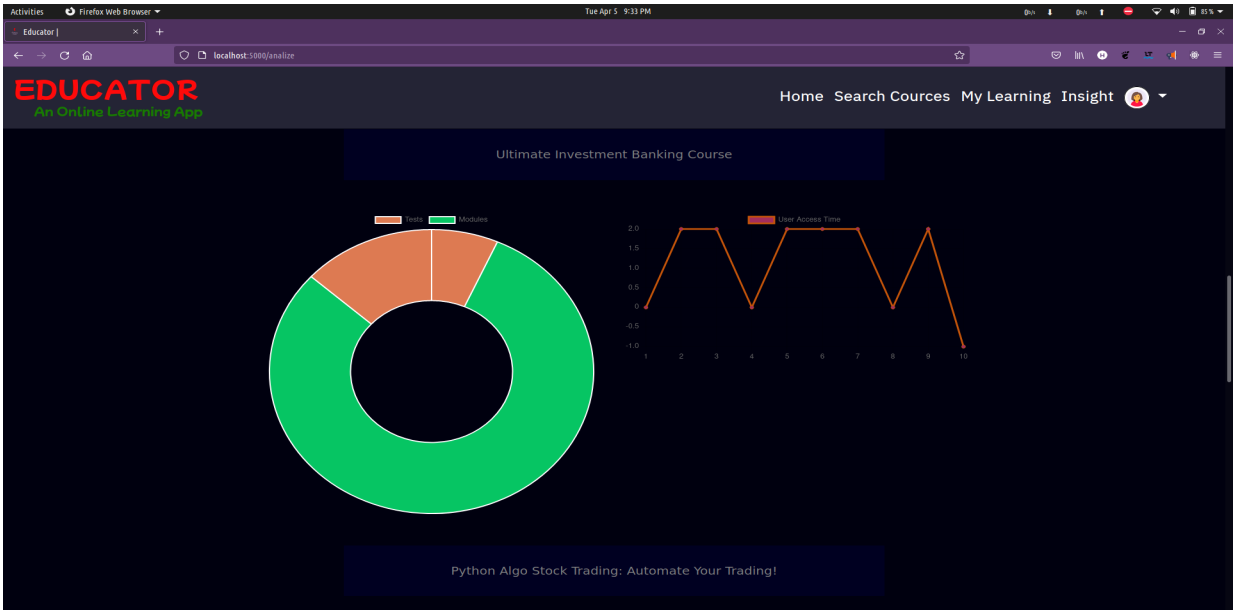
Modules Page



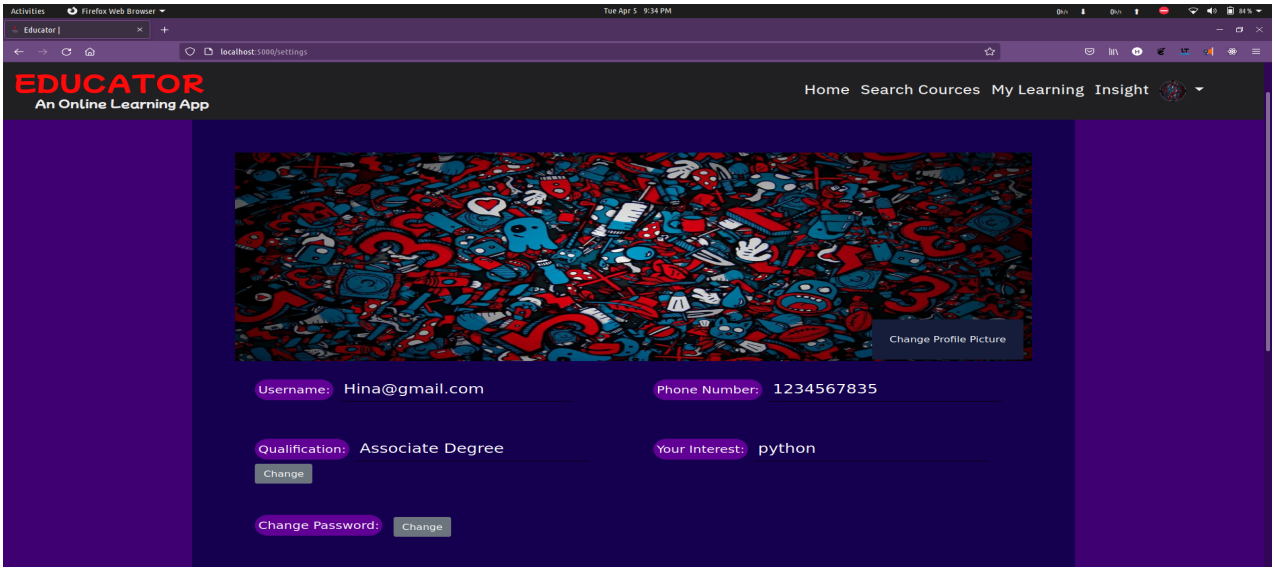
Payment And Card Page



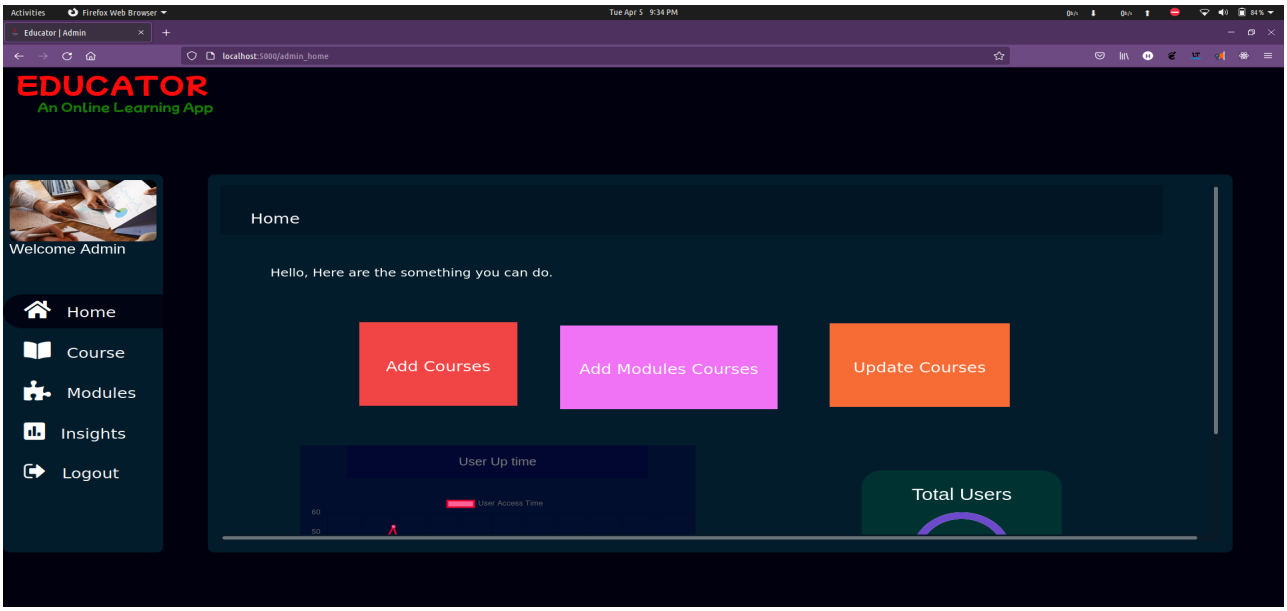
Insight Page



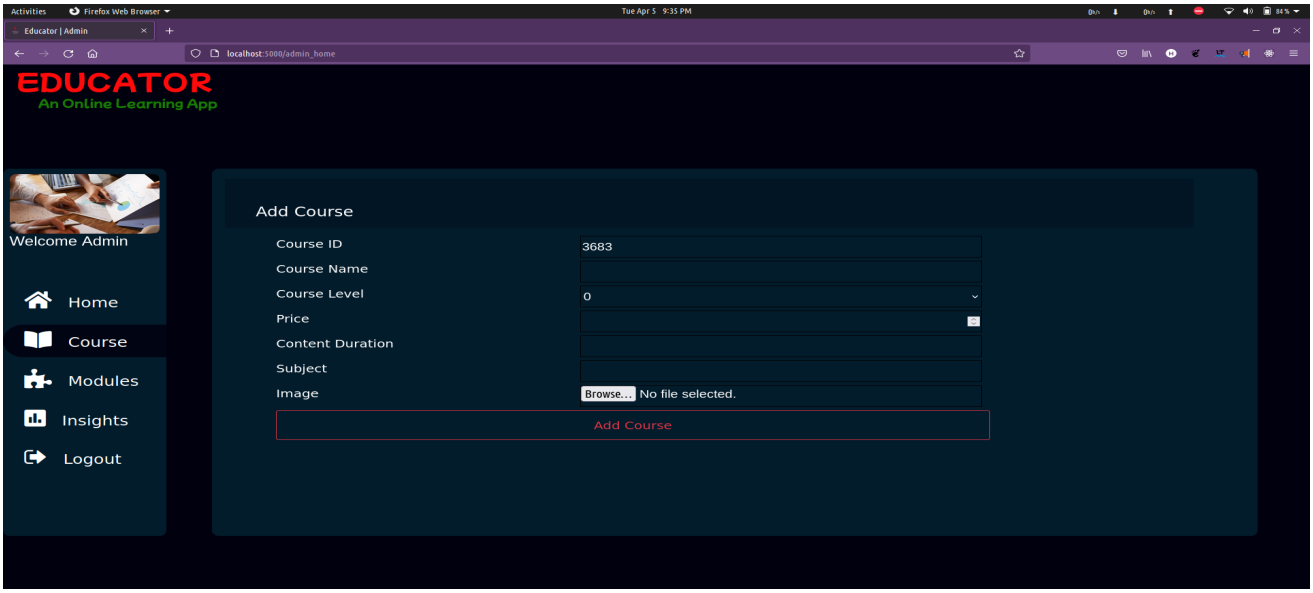
Account Setting Page



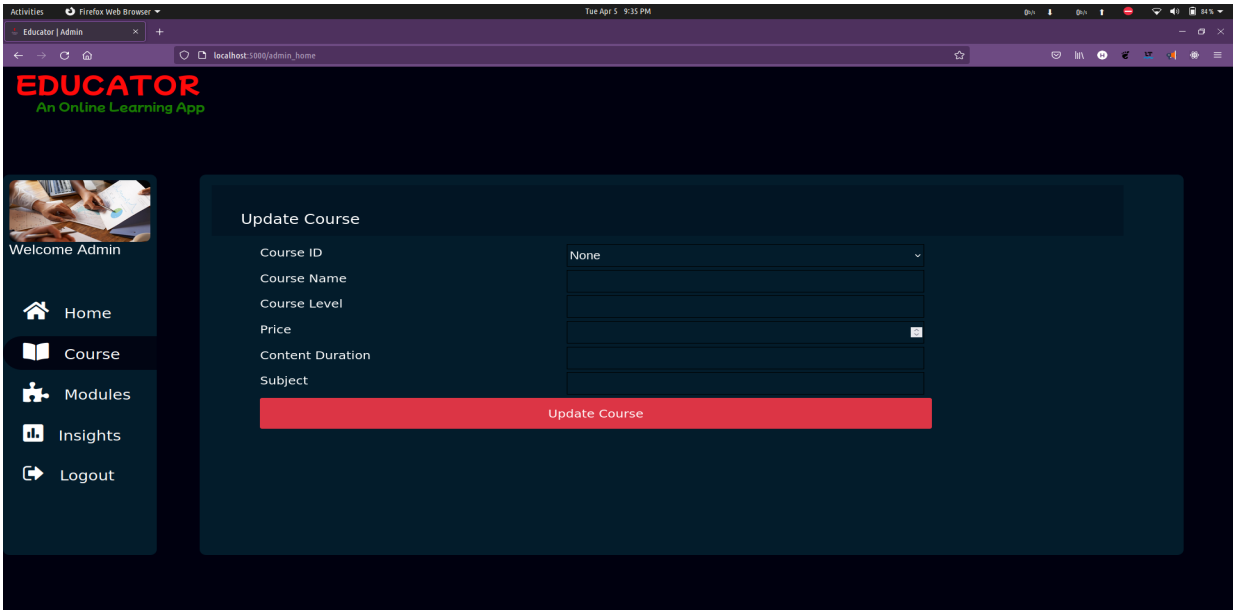
Admin Home Page



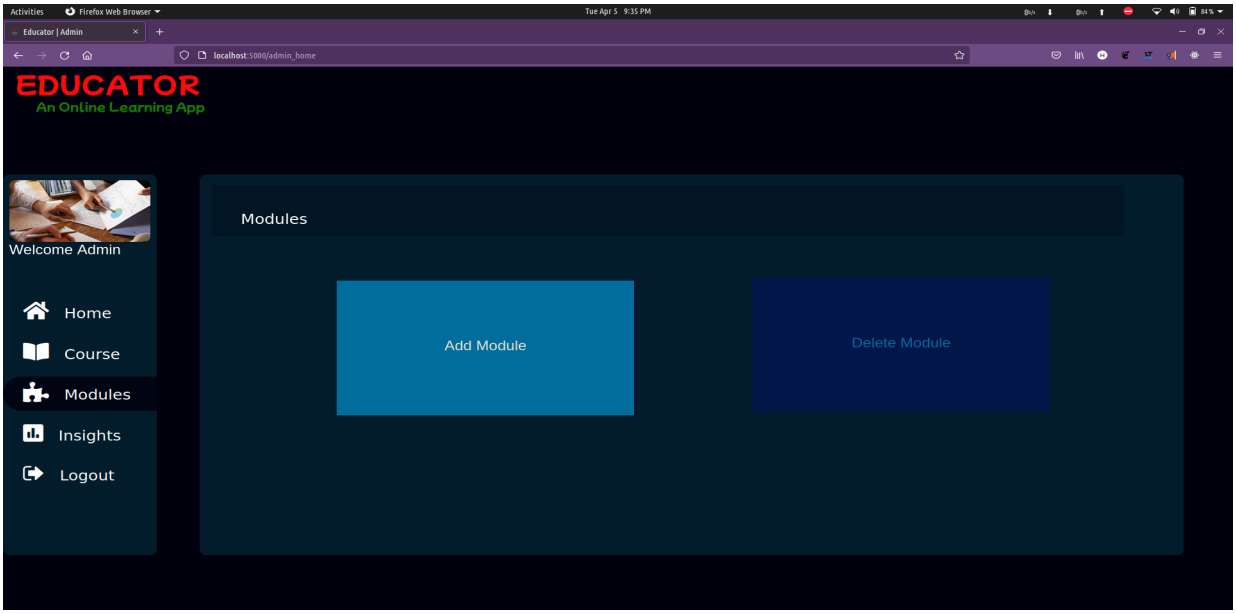
Admin Add Course



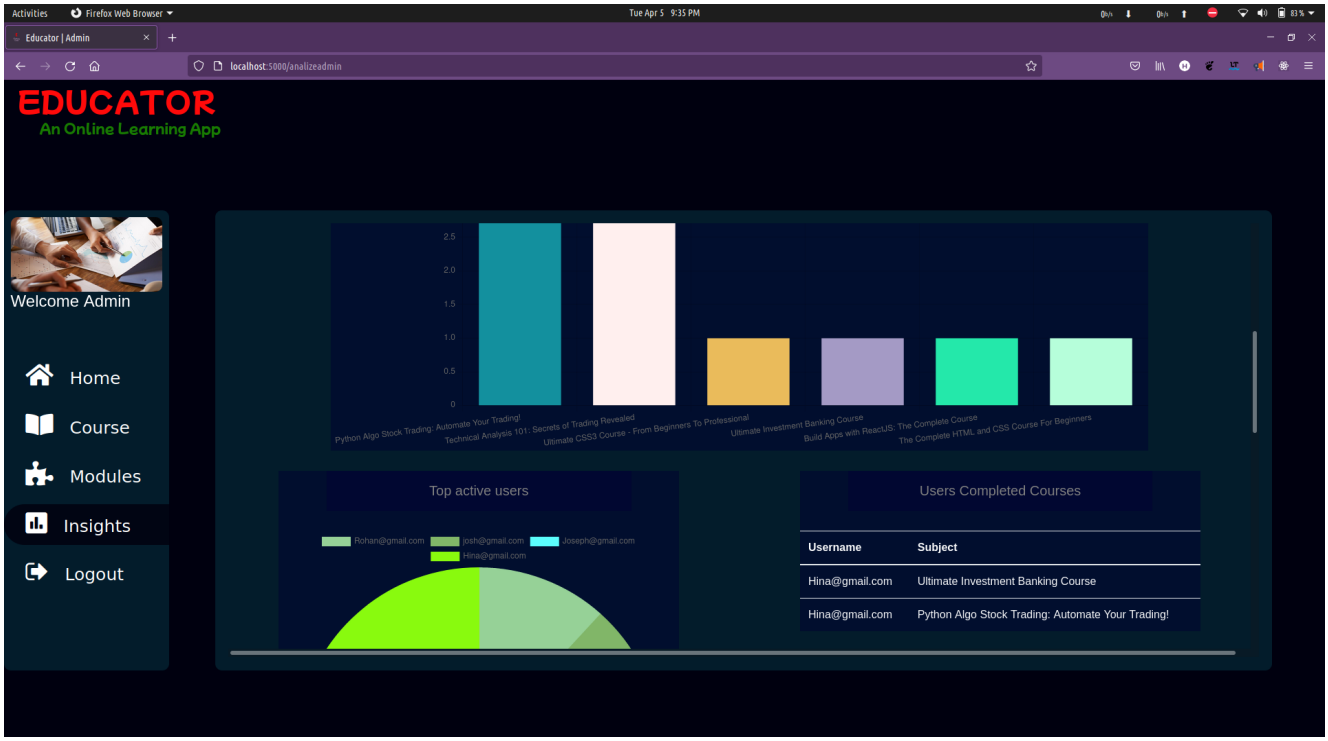
Admin Update Course



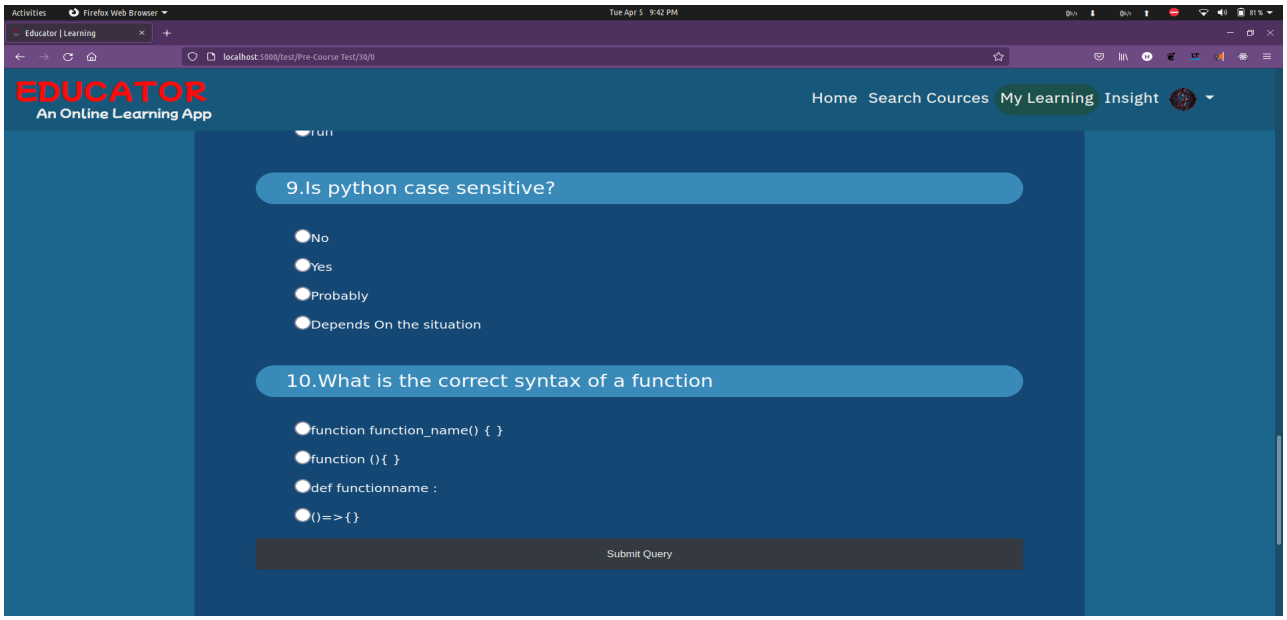
Add Modules



Admin Insight Page



User Test Module



11.APPENDINCES C-SAMPLE REPORTS OF TEST CASES

sl.no	Test Case Id	Test Description	Steps to Execute	Test Data	Expected Result	Actual Result	Status
01	Test case id 1	Correct username and password	Enters username and password in login form and clicks enter	Correct username and password	Successful login	Successful login	pass
02	Test case id 2	Correct username and Wrong password	Enters username and password in login form and clicks enter	Correct username and Wrong password	Login Failed-Invalid Credentials	Login Failed-invalid Credentials	Pass
03	Test case id 3	wrong username and correct password	Enters username and password in login form and clicks enter	Wrong username and Correct password	Login Failed-Invalid Credentials	Login Failed-invalid Credentials	Pass
04	Test case id 4	wrong username and wrong password	Enters username and password in login form and clicks enter	Wrong username and Wrong password	Login Failed-Invalid Credentials	Login Failed-Invalid Credentials	Pass
05	Test case id 5	Register New user	Enters Users Details	New User	registered Successfully	registered Successfully	Pass
06	Test case id 6	Register Existing User	Enters Users Details	Existing User	User Already Exist – Recheck username	User Already Exist – Recheck username	Pass
07	Test case id 7	Access Course List	Access a particular course	Click on start learning button	Display course modules	Display course modules	Pass
08	Test case id 8	Purchase a course	Clicks on buy option	Enter correct card details	Course Purchased	Course Purchased	Pass
09	Test case id 9	Purchase a course	Clicks on buy option	Enter Incorrect card details	Course-Not Purchased	Course-Not Purchased	Pass
10	Test Case id 10	View Modules	Click on learning	Click on start learning button	View modules	View Modules	Pass

12.APPENDINCES D-Source Code

app.py

```
from werkzeug.utils import secure_filename

from pathlib import Path

import os

from flaskext.mysql import MySQL

from flask import (Flask,render_template,request,flash, url_for, redirect, session, abort, jsonify )

import json

from datetime import date, datetime

from csv import writer

import csv

import uuid

from operator import itemgetter

# derived modules

from MlModels.course_recommendation import recommend

from backend import searchcourse

from backend.get_data import coursedata, create_course_data, create_cards_details, deduct_balance, mycourses

app = Flask(__name__)

app.config['MYSQL_DATABASE_USER'] = 'root'

# app.config['MYSQL_DATABASE_PASSWORD'] = 'root'

app.config['MYSQL_DATABASE_DB'] = 'project'

app.config['MYSQL_DATABASE_HOST'] = 'localhost'

app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'
```

```
app.config['UPLOAD_FOLDER'] = 'static/modules'
app.config['UPLOAD_FOLDER_IMAGE'] = 'static/images/courseimage'
app.config['UPLOAD_EXTENSIONS'] = ['.MP4', '.mp4', '.jpg', '.png']
app.config['CSVFILE'] = 'dataset'
mysql = MySQL(app)
conn = mysql.connect()
cursor = conn.cursor()
@app.route('/', methods=('GET', 'POST'))
def indexpage():
    return render_template("index.html")
@app.route("/homepage", methods=['POST', 'GET'])
def homepage():
    try:
        if session['username']:
            return render_template("home.html", courselist=create_course_data(conn, session['username']))
        else:
            return redirect(url_for("indexpage"))
    except KeyError as k:
        return redirect(url_for("indexpage"))
@app.route('/logout', methods=['POST', 'GET'])
def logout():
    session.pop("username", None)
    return render_template("index.html")
@app.route('/login', methods=('POST', 'GET'))
```

```
def login_creds():
    if request.method == 'POST':
        uname = request.form.get('username')
        password = request.form.get('password')
        cursor.execute(
                f"SELECT count(*) FROM userlogin WHERE username = '{uname}'AND password = '{password}' ")
        row = cursor.fetchone()
        if row[0] == 0:
            flash("Invalid Username/ Password")
            return render_template("index.html")
        session['username'] = uname
           return
@app.route('/signup', methods=['POST', 'GET'])
def createuser():
    result = False
    if request.method == 'POST':
        new_user = request.form.get('new_username')
        password = request.form.get('new_pass')
        conf_pass = request.form.get('conf_pass')
    if result:
        session["username"] = None
        session['username'] = new_user
        return render_template("signup.html")
    else:
```

```
        return "Something went Wrong"
@app.route("/savedata", methods=['POST', 'GET'])
def saveData():
       Interested = ",".join(List_interested)
        try:
                sql = f"UPDATE userlogin SET phoneno = '{Phoneno}', qual = '{Qualification}', interest = '{Interested}',image_url = '/static/users/default.png' WHERE username = '{session['username']}'"
            cursor.execute(sql)
            conn.commit()
        except Exception as e:
            print("Problem inserting into db: " + str(e))
            return "An Error Occured Please Try Again..."
    return redirect(url_for("homepage"))
@app.route("/Continue_", methods=['POST', 'GET'])
def Continue():
    return redirect(url_for("homepage"))
#  search course
# payment
@app.route('/addcard/<course_id>', defaults={'course_id': None})
@app.route('/addcard/<course_id>')
def learning_page():
    courselist = mycourses(conn, session['username'])
    return(render_template('learning_dash_page.html', courselist=courselist))
@app.route('/modules/<course_id>')
def modules(course_id):
```

```
    #Selecting Course
    q = f"SELECT course_title FROM course_list_1 WHERE course_index = '{course_id}'"
    cursor.execute(q)
        return(render_template('module.html', subjectname=subject[0], data=res))
    #getting userlevel
    level = getuserlevel(course_id)
         conn.commit()
        return("Done")
@app.route('/moduledata/<modulename>')
def moduledata(modulename):
     # query    = f"SELECT * FROM MODULES WHERE modulename = '{modulename}' GROUP BY subtype ORDER BY lecture_id"
    # cursor.execute(query)
    # conn.commit()
    # data = cursor.fetchall()
    pass
@app.route('/test/<testname>/<c_id>/<l_id>',methods=['POST','GET'])
def test(testname,c_id,l_id):
     sqlquery = f"SELECT * FROM  testquestion WHERE course_id = '{c_id}' AND lecture_id = '{l_id}' ORDER BY question_id"
    cursor.execute(sqlquery)
    conn.commit()
    data = cursor.fetchall()

    return (render_template('testuser.html',data = data,cid=c_id,lid=l_id))
```

```
@app.route('/settings',methods = ['POST','GET'])
def settings():
    try:
        username = session['username']
        query =  f"SELECT * FROM userlogin  WHERE username = '{username}'"
        cursor.execute(query)
        conn.commit()
        data = cursor.fetchall()[0]
        print(data)
    except Exception as e:
        print(str(e))
        flash("Error. Please Try Later ...")
    return(render_template('setting.html',data = data))
@app.route('/setuserimage',methods=['POST','GET'])
def setuserimage():
    username = session['username']
    image = request.get_json();
    try:
            query =  f"UPDATE userlogin SET image_url = '{image['img']}' WHERE username = '{username}'"
        cursor.execute(query)
        conn.commit()
        flash("Successfully Updated...")
        session['image'] = image['img']
    except Exception as e:
```

```
        print(str(e))
        flash("Error. Please Try Later ...")
    return jsonify(success=True)
@app.route('/changepass',methods=['POST'])
def changepass():
    if request.method == 'POST':
        username = session['username']
        data = request.get_json();
    try:
            query =  f"UPDATE userlogin SET password = '{data['password']}' WHERE username = '{username}'"
        cursor.execute(query)
        conn.commit()
        flash("Successfully Updated...")
    except Exception as e:
        print(str(e))
        flash("Error. Please Try Later ...")
    return jsonify(success=True)
@app.route('/changedetails',methods=['POST'])
def changedetails():
    if request.method == 'POST':
        phoneno = request.form.get('phno');
        qual = request.form.get('qual')
        interest = request.form.get('interest')
    try:
```

```
@app.route('/timeslogin',methods=['POST','GET'])
def timeslogin():
    if request.method =='POST':
        val =[1]
        try:
                q = f"SELECT sum(times),MONTH(cdate) FROM timeslogin  WHERE username = '{session['username']}'  GROUP BY MONTH(cdate)"
            cursor.execute(q)
            res = cursor.fetchall()
            val = [i[0] for i in res]
        except Exception as e:
            print(str(e))
        return jsonify(data = val)
    else:
        return "No"
@app.route('/usercourse',methods = ['POST','GET'])
def usercourse():
    coursedata = [0]
     sql1 = f"SELECT count(course_id),course_id FROM student_learning_history WHERE username = '{session['username']}' GROUP BY course_id"
    cursor.execute(sql1)
    conn.commit()
    res = cursor.fetchall()
    if request.method =='POST':
        try:
```

```
        data1 = cursor.fetchall()
@app.route('/alluseraccess',methods = ['POST','GET'])
def alluseraccess():
    except Exception as e:
        print(str(e))
    return jsonify(data = [val,no])
  else:
    return "No"
@app.route('/analizeadmin')
def analizeadmin():
  return render_template("admin/analize.html")
@app.route('/top10course',methods=['POST'])
def top10course():
  sql = f"SELECT course_id from student_course ORDER BY course_id"
  cursor.execute(sql)
  no = cursor.fetchall()
  courseno = []
  for i in no:
    courseno.append(i[0])
  val = [ [l, courseno.count(l)] for l in set(courseno)]
  top10key = sorted(val, key=itemgetter(1),reverse=True)
  toplabel = []
  topval = []
@app.route('/topuser',methods=['POST'])
```

```
    try:
sqlquery = f"INSERT INTO course_list_1(`course_index`, `course_title`, `price`, `level`, `content_duration`, `subject`, `year`, `image_url`) VALUES({c_id},'{cname}','{price}','{diff}','{duration}','{subject}','{year}','{filename}')"
            cursor.execute(sqlquery)
            conn.commit()
            flash("Course Added Sucessfully..")
            with open(app.config['CSVFILE']+'/course_list.csv', 'a', newline='') as f_object:
                writer_object = writer(f_object)
                writer_object.writerow(data)
                f_object.close()
                print("Added")
        except Exception as e:
            flash("Error Occured ..")
            print(str(e))
    return(redirect(url_for('admin_home')))
@app.route('/getmoduleindex',methods=['POST'])
def getmoduleindex():
    c_id = request.get_json()
    query = f"SELECT lecture_id FROM modules WHERE course_id ='{c_id['c_id']}'"
    cursor.execute(query)
    row = cursor.fetchall()
    return jsonify(result=row)


@app.route('/getlecturename',methods=['POST'])
```

```
def getlecturename():
    value = request.get_json()
     query = f"SELECT modulename,video FROM modules WHERE course_id ='{value['c_id']}' AND lecture_id='{value['l_id']}'"
    cursor.execute(query)
    row = cursor.fetchall()
    print(row)
    return jsonify(result=row)
@app.route('/deletemodules',methods=['POST'])
def deletemodules():
    if request.method == 'POST':
        c_id = request.form.get('c_id')
        l_id = request.form.get('l_id')
        try:
        for i in level:
            if i== '0':
                weights.append(0.3)
            elif i == '1':
                weights.append(0.5)
            else:
                weights.append(0.9)
        totalmarks = sum(weights)
        obtained = 0
        for i in zip(answer_user,realanswer,weights):
            if i[0]==i[1]:
```

```
            obtained += i[2]
        percentage = obtained/totalmarks
        summary= "Pass"
        if percentage > 0.85:
            level = 2
        elif percentage < 0.85 and percentage >0.65:
            level =1
        elif percentage < 0.65 and percentage > 0.45:
            level = 0
        else:
            level =-1
            summary= "Failed"
                query = f"INSERT INTO test_history VALUES ('{session['username']}',
'{c_id}','{l_id}','{obtained}','{summary}')"
        cursor.execute(query)
        conn.commit()
        result = [round(totalmarks,1),round(obtained,1), summary,level,c_id,l_id]
                completedmodule(c_id,l_id,level,1)
                return(redirect(url_for('finalcomplete',id = cert_id)))


    return(render_template('testresult.html',data = result)
@app.route('/viewcertificate/<id>',methods = ['POST','GET'])
def viewcertificate(id):
    print(id)
    sql = f"SELECT  * FROM student_course WHERE cert_id = '{id}' "
```

```
    cursor.execute(sql)
    conn.commit()
    data = cursor.fetchall()[0]
    print(data)
    course_id = data[1]
    name = data[0].split("@")[0]
    sql2  = f"SELECT course_title FROM course_list_1 WHERE course_index = '{course_id}'"
    return render_template('certificate.html',data = details)
@app.route('/getcert_id',methods=['POST'])
def getcert_id():
    data = request.get_json()
    username = session['username']
    print(data['c_id'])
    return(redirect('admin_home'))
if __name__ == "__main__":
    app.run()
```

getdata.py

```
from backend import searchcourse
def get_user_interest(cursor, conn, username):
    select_interest = f"SELECT  interest FROM userlogin WHERE username = '{username}'"
    cursor.execute(select_interest)
    conn.commit()
    row1 = cursor.fetchone()
    session_user_interests = " ".join(list(row1))
```

```
    return session_user_interested
def checkifregistered(cursor, username, id):
     sql = f"SELECT count(course_id) FROM student_course WHERE username = '{username}' AND course_id = {id}"
    cursor.execute(sql)
    row = cursor.fetchone()
    if row[0] == 0:
        registered = False
    else:
        registered = True
    return registered
def create_course_data(conn, username, keyword="interest", recommend=True, course_id=-1):
    cursor = conn.cursor()
    courselist = []
    if recommend:
        if keyword == "interest":
            user_interest = get_user_interest(cursor, conn, username)
        else:
            user_interest = keyword
        recommend_list = searchcourse.recommend_course_data(
            conn, user_interest)
    return courselist
def create_cards_details(username, cursor, conn):
    try:
        sqlquery = f"SELECT * FROM payment_cards WHERE username = '{username}'"
```

```
        cursor.execute(sqlquery)
        conn.commit()
        cards = cursor.fetchall()
    except:
        return("Error")
card_list = []
            card_list.append(cards_dict)
    else:
        card_list = False
    return card_list
def deduct_balance(cursor, conn, course_price, cardno):
    res = False
    try:
            sqlquery = f"UPDATE payment_cards SET balance = balance - {int(course_price)}  WHERE cardnumber = '{cardno}'"
        cursor.execute(sqlquery)
        conn.commit()
        res = True
    except:
        return("something Went Wrong Try Later")
    return(res)
def coursedata(conn,course_index):
    try:
        cursor = conn.cursor()
        sqlquery = f"SELECT * FROM course_list_1 WHERE course_index IN ({course_index})"
```

```
        cursor.execute(sqlquery)
        conn.commit()
        my_course = cursor.fetchall()
        cursor.close()
    except Exception as e:
        print(str(e))
        return [None]
    if my_course[0] != None:
        courselist = []
            courselist.append(dictionery_course)
    else:
        courselist = False
    return courselist
def mycourses(conn, username):
    cursor = conn.cursor()
    sqlquery = f"SELECT course_id FROM student_course WHERE username = '{username}'"
    cursor.execute(sqlquery)
    conn.commit()
    course_index = cursor.fetchall()
    l = []
    for i in course_index:
        l.append(i[0])
    l = list(set(l))
    strlist = ",".join(l)
```

```
    cursor.close()
    course = coursedata(conn,strlist)
    return course
```

ML_LDA.py

```
import numpy  as np
import pandas as pd
from rake_nltk import Rake
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import CountVectorizer
import re
def load_data():
    dataset = pd.read_csv("dataset/course_list.csv")
    print(dataset)
    return dataset
# def getKeywords(sentence):
#     Keywords = []
#     r = Rake()
#     r.extract_keywords_from_text(sentence)
#     Keywords.append(list(r.get_word_degrees()))
#     return Keywords
#     BoW = []
#     for i in bag:
#         BoW.append(" ".join(i))
```

```
#     return BoW
# def cosinevectors():
#     count = CountVectorizer()
#     count_matrix = count.fit_transform(Bag_Of_Words)
#     cosine_sim = cosine_similarity(count_matrix, count_matrix)
#     dataset['Words_Vector'] = ""
#     dataset['Words_Vector'] = cosine_sim
 def recommendindex(dataset,kw):
    coursename = kw
    r = Rake()
    r.extract_keywords_from_text(coursename)
    cname = " ".join(r.get_word_degrees())
    recom_course_index = dataset[dataset['Bag_Of_Words'].str.contains(
        cname,regex=True
        )].sort_values('Words_Vector',ascending = False)[0:10].index
    return recom_course_index
def recommend(val):
    dataset = load_data()
    res = recommendindex(dataset,val)
    listofindex = []
    for i in res:
        listofindex.append(str(i))
    indexs = ",".join(listofindex)
    return indexs
```

ML_RecommendCourse.py

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn.cluster import KMeans
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow import keras
from MlModels.course_recommendation import recommend
from distutils.log import debug
from flaskext.mysql import MySQL
from flask import (
    Flask,
    render_template,
    request,
    flash,
    url_for,
    redirect,
```

```
    session
)
alg3 = LDA(solver='lsqr', shrinkage=0.8).fit(X,y)
ypred = alg3.predict(x_test)
accuracy_score(ypred,y_test)
def recommend_course_data(conn,session_user_interests):
    cursor = conn.cursor()
    indexs = recommend(session_user_interests)
    try:
        sqlquery = f"SELECT * FROM course_list_1 WHERE  course_index IN ({indexs})"
        cursor.execute(sqlquery)
        conn.commit()
        row = cursor.fetchall()
    except:
        row = [None]
    cursor.close()
```