# QUIZ APPLICATION

*Project Based Lab Report Submitted in partial fulfilment of the requirements for the award of the degree of*

## BACHELOR OF TECHNOLOGY

## IN ECM

*Submitted by*

LAKSHMI DURGA 190050052

*Under the*

*esteemed guidance of*

Faculty Name: **Mrs. SRI DEVI SAKHAMURI**

Designation: **ASSISTANT PROFESSOR**

**Department of Electronics & Computer Engineering**
**K L Deemed to be UNIVERSITY**
**Green Fields, Vaddeswaram**
**2021-22**

# KL Deemed to be UNIVERSITY

**Green Fields, VADDESWARAM**

## CERTIFICATE

This is to certify that the **Web Programming with Python and Django** project based lab report entitled “QUIZ APPLICATION” submitted by 190050052 Puppala Lakshmi Durga Branch is Electronics **and Computer Engineering in partial fulfillment of the requirements for the award of**
the Degree **Bachelor of Technology** in “**Electronics & Computer Engineering**” is a bonafied record of the work carried out under our guidance and supervision at KL Deemed to be University during the academic year 2021-2022.

**Signature of Faculty In charge**

**Faculty: Mrs. SRI DEVI SAKHAMURI**

**Designation: ASSISTANT. PROFESSOR**

**Head of The Department**

**DR. M.SIVA GANGA PRASAD**

# ACKNOWLEGDEMENT

We are greatly indebted to our KLDeemed to be University that has provided a healthy environment to drive us to achieve our ambitions and goals. We would like to express our sincere thanks to our project Incharge **Mrs. Sri Devi Sakhamuri** for the guidance, support and assistance they have provided in completing this project.

We are thankful to our Head of the Department **Dr. M. Siva Ganga Prasad, Professor, Dept. of ECM**, who modeled us both technically and morally for achieving greater success in life.

We are very much glad for having the support given by our principal, **Dr.K.Subba Ro** who inspired us with his words filled with dedication and discipline towards work.

Finally, we owe a lot to the teaching and non-teaching staff of the **Dept. of ECM** for their direct or indirect support in doing our Lab based project work.

# INDEX

## Content

# ABSTRACT

Quiz App Project in Django and let others attempt and test their knowledge by giving the Quiz. Every one of us likes to attempt a Quiz and check our score at the end. Have you ever thought of building a quiz using some of the programming knowledge and which other people can also be able to play. If you have thought about this, you have come to the right place to learn to make a beautiful quiz on various topics. The system is built fully in Django Framework in back-end and HTML, CSS in front-end. It has similar features as any quiz app where there is a list of quiz questions in the home screen and user answer all questions. After user answers the question, the system will automatically perform check and provide total marks gained. It will also point out the wrong answers. This app provides an admin panel through which questions can be added along with its four options and at last the answer. It is also used to add new users to the system using which new users can enter to the system and attend the quiz examination.

# PROJECT DESCRIPTION:

The Quiz App Project using Django Framework is created using Python Django Framework. The system is built fully in Django Framework in back-end and HTML, CSS in front-end. This project is a basic quiz app that offers features such as adding as many questions as you like, registering and taking the quiz with a user on the system. A  Quiz App Project using Django Framework It has similar characteristics to any quiz app where the home screen has a list of quiz questions and all questions are answered by the user. After the user answers the query, the system conducts the search automatically and provides the total marks obtained. It will also figured out the wrong answers as well**.**

## SOFTWARE REQUIREMENT:

### PYCHARM:

**PyCharm** is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains (formerly known as IntelliJ). It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features – released under a proprietary license.

## TECHNOLOGIES USED(HTML, CSS) :

## HTML:

HTML is used to create electronic documents (called pages) that are displayed on the World Wide Web. Each page contains a series of connections to other pages called hyperlinks. Every web page you see on the Internet is written using one version of HTML code or another. HTML code ensures the proper formatting of text and images so that your Internet browser may display them as they are intended to look. Without HTML, a browser would not know how to display text as elements or load images or other elements.

HTML5 is the update made to HTML from HTML4 (XHTML follows a different version numbering scheme). It uses the same basic rules as HTML4, but adds some new tags and attributes which allow for better semantics and for dynamic elements that are activated using JavaScript. New elements include section, , , , , , , . There are also new input types for forms, which include tel, search, url, email, datetime, date, month, week, time, number, range and colour.

### CSS:

Cascading style sheets are used to format the layout of Web pages. They can be used to define text styles, table sizes, and other aspects of Web pages that previously could only be defined in a page's HTML.

CSS helps Web developers create a uniform look across several pages of a Web site. Instead of defining the style of each table and each block of text within a page's HTML, commonly used styles need to be defined only once in a CSS document. Once the style is defined in cascading style sheet, it can be used by any page that references the CSS file. Plus, CSS makes it easy to change styles across several pages at once. If the pages all reference the same style sheet, the text size only needs to be changed on the style sheet and all the pages will show the larger text.

While CSS is great for creating text styles, it is helpful for formatting other aspects of Web page layout as well. For example, CSS can be used to define the cell padding of table cells, the style, thickness, and color of a table's border, and the padding around images or other objects. CSS gives Web developers more exact control over how Web pages will look than HTML does. Therefore,most Web pages today incorporate cascading style sheets.

## DATABASE DESIGN:

### PostgreSQL:

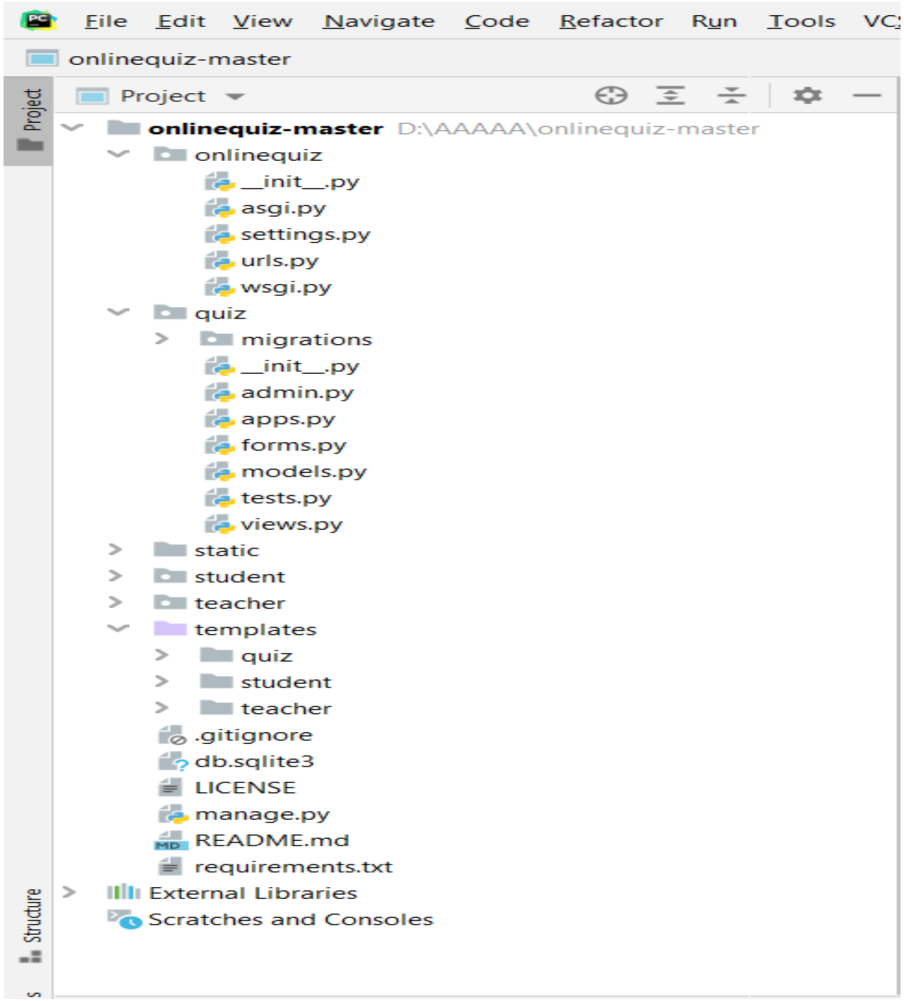
PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads. PostgreSQL comes with many features aimed to help developers build applications, administrators to protect data integrity and build fault-tolerant environments, and help you manage your data no matter how big or small the dataset. In addition to being free and open source, PostgreSQL is highly extensible. For example, you can define your own data types, build out custom functions, even write code from different programming languages without recompiling your database.

PostgreSQL tries to conform with the SQL standard where such conformance does not contradict traditional features or could lead to poor architectural decisions. Many of the features required by the SQL standard are supported, though sometimes with slightly differing syntax or function. Further moves towards conformance can be expected over time.

- Data Types
  - Primitives: Integer, Numeric, String, Boolean
  - Structured: Date/Time, Array, Range / Multirange, UUID
  - Document: JSON/JSONB, XML, Key-value (Hstore)
  - Geometry: Point, Line, Circle, Polygon
  - Customizations: Composite, Custom Types
- Data Integrity
  - UNIQUE, NOT NULL
  - Primary Keys
  - Foreign Keys
  - Exclusion Constraints
  - Explicit Locks, Advisory Locks
- Concurrency, Performance
  - Indexing: B-tree, Multicolumn, Expressions, Partial
  - Advanced Indexing: GiST, SP-Gist, KNN Gist, GIN, BRIN, Covering indexes, Bloom filters
  - Sophisticated query planner / optimizer, index-only scans, multicolumn statistics
  - Transactions, Nested Transactions (via savepoints)
  - Multi-Version concurrency Control (MVCC)
  - Parallelization of read queries and building B-tree indexes
  - Table partitioning
  - All transaction isolation levels defined in the SQL standard, including Serializable
  - Just-in-time (JIT) compilation of expressions
- Reliability, Disaster Recovery
  - Write-ahead Logging (WAL)
  - Replication: Asynchronous, Synchronous, Logical
  - Point-in-time-recovery (PITR), active standbys
  - Tablespaces

- Security
    - Authentication: GSSAPI, SSPI, LDAP, SCRAM-SHA-256, Certificate, and more
    - Robust access-control system
    - Column and row-level security
    - Multi-factor authentication with certificates and an additional method
- Extensibility
    - Stored functions and procedures
    - Procedural Languages: PL/PGSQL, Perl, Python (and many more)
    - SQL/JSON path expressions
    - Foreign data wrappers: connect to other databases or streams with a standard SQL interface
    - Customizable storage interface for tables
    - Many extensions that provide additional functionality, including PostGIS
- Internationalisation, Text Search
    - Support for international character sets, e.g. through ICU collations
    - Case-insensitive and accent-insensitive collations
    - Full-text search

## PROJECT STRUCTURE:



```
onlinequiz-master  D:\AAAAA\onlinequiz-master
  onlinequiz
    __init__.py
    asgi.py
    settings.py
    urls.py
    wsgi.py
  quiz
    migrations
    __init__.py
    admin.py
    apps.py
    forms.py
    models.py
    tests.py
    views.py
  static
  student
  teacher
  templates
    quiz
    student
    teacher
  .gitignore
  db.sqlite3
  LICENSE
  manage.py
  README.md
  requirements.txt
External Libraries
Scratches and Consoles
```

# Code:

## index.html:

```
<!DOCTYPE html>
{% load static %}
<html lang="en" dir="ltr">

<style>
  .jumbotron{
    margin-bottom: 0px;
  }
</style>
<body>
  {% include "quiz/navbar.html" %}
<br>
<section id="section-jumbotron" class="jumbotron jumbotron-fluid d-flex justify-
content-center align-items-center">
  <div class="container text-center">
    <h1 class="display-1 text-info">Let's Quiz</h1>
    <p class="display-4 d-none d-sm-block">Test your skills and become a
master.</p>
    <p class="lead">We organize quizzes on various topics.</p>



    <p class="lead">Sign up if you haven't already and get access to millions of
quizzes on the topic of your interest.</p>
    <p><strong> Start Your Journey Here:</strong></p>
    <a href="/student/studentsignup" class="btn btn-lg btn-info" style="padding-
right: 35px;"><i class="fa fa-user-plus" aria-hidden="true"></i> Sign Up</a>

    <br><br><br>
  </div>
</section>
  {% include "quiz/footer.html" %}


</body>

</html>
```

## logout.html:

```
{%load static%}
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title></title>
  </head>
  <body>
    {% include "quiz/navbar.html" %}
    <br><br><br><br><br>
<img src="{% static "image/logout_img.png" %}"  style="width:800px;
```

```
height:100px; padding-left:400px;">

<br><br><br><br><br>
<br><br><br><br><br>
<br><br>
{% include "quiz/footer.html" %}

  </body>
</html>
```

## Admin_add_course.html:

```
{% extends 'quiz/adminbase.html' %}
{% load widget_tweaks %}
{% block content %}
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></scri
pt>
  <style>
      <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></scri
pt>
  a:link {
        text-decoration: none;
      }

  </style>
</head>
<h2 style="text-align:center; color:blue;">ADD COURSE</h2>
<form method="POST" autocomplete="off" style="margin:100px;margin-top: 0px;">
    {%csrf_token%}
    <div class="form-group">
      <label for="course_name">Course Name</label>
      {% render_field courseForm.course_name class="form-control"
placeholder="Java" %}

      <label for="question_number">Total Question</label>
      {% render_field courseForm.question_number class="form-control"
placeholder="10" %}

      <label for="total_marks">Total Marks</label>
      {% render_field courseForm.total_marks class="form-control"
placeholder="50" %}
    </div>
    <button type="submit" class="btn btn-primary">ADD</button>
  </form>
<br><br><br>
{% endblock content %}
```

### Admin_add_questions.html:

```
{% extends 'quiz/adminbase.html' %}
{% load widget_tweaks %}
{% block content %}
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></scri
pt>
  <style>
       <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></scri
pt>
  a:link {
        text-decoration: none;
      }

  </style>
</head>
<h2 style="text-align:center; color:blue;">ADD QUESTION</h2>
<form method="POST" autocomplete="off" style="margin:100px;margin-top: 0px;">
    {%csrf_token%}
    <div class="form-group">
        <label for="question">Course</label>
      {% render_field questionForm.courseID|attr:'required:true' class="form-
control"  %}
        <br>

      <label for="question">Question</label>
      {% render_field questionForm.question|attr:'required:true' class="form-
control" placeholder="What is the currency of India ?" %}
        <br>
      <label for="question_number">Marks</label>
      {% render_field questionForm.marks|attr:'required:true' class="form-
control" placeholder="10" %}
        <br>
      <label for="option1">Option 1</label>
      {% render_field questionForm.option1|attr:'required:true' class="form-
control" placeholder="Rupees" %}
        <br>
      <label for="option2">Option 2</label>
      {% render_field questionForm.option2|attr:'required:true' class="form-
control" placeholder="Dollar" %}
        <br>
      <label for="option3">Option 3</label>
      {% render_field questionForm.option3|attr:'required:true' class="form-
control" placeholder="Taka" %}
        <br>
      <label for="option4">Option 4</label>
      {% render_field questionForm.option4|attr:'required:true' class="form-
control" placeholder="Euro" %}
        <br>
      <label for="answer">Answer</label>
      {% render_field questionForm.answer|attr:'required:true' class="form-
control" %}
    </div>


    <button type="submit" class="btn btn-primary">ADD</button>
  </form>
<br><br><br>
{% endblock content %}
```

### Admin_check_marks.html:

```
{% extends 'quiz/adminbase.html' %}
{% block content %}
{%load static%}

<head>
  <link href="//netdna.bootstrapcdn.com/bootstrap/3.0.0/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">
  <script
src="//netdna.bootstrapcdn.com/bootstrap/3.0.0/js/bootstrap.min.js"></script>
  <script src="//code.jquery.com/jquery-1.11.1.min.js"></script>

  <style media="screen">
    a:link {
      text-decoration: none;
    }

    h6 {
      text-align: center;
    }

    .row {
      margin: 100px;
    }
  </style>
</head>
<br><br>
<div class="container">
  <div class="panel panel-primary">
    <div class="panel-heading">
      <h6 class="panel-title">View Marks</h6>
    </div>
    <table class="table table-hover" id="dev-table">
      <thead>
```

```
        <tr>

          <th>Exam Name</th>

          <th>Total Marks</th>
          <th>Attempt Number</th>
          <th>Exam Date</th>
        </tr>
      </thead>
      {% for t in results %}
      <tr>

        <td> {{t.exam}}</td>
        <td> {{t.marks}}</td>
        <td>Attemp {{ forloop.counter }}</td>
        <td> {{t.date}}</td>


      </tr>
      {% endfor %}
    </table>
  </div>
</div>

<br><br><br><br><br><br>
{% endblock content %}
```

**Admin_dashboard.html:**

```
{% extends 'quiz/adminbase.html' %}
{% load widget_tweaks %}
{% block content %}


<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no">

    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script
>
    <link href="http://netdna.bootstrapcdn.com/bootstrap/4.0.0-
beta/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
    <link href="https://maxcdn.bootstrapcdn.com/font-awesome/4.3.0/css/font-
awesome.min.css" rel="stylesheet">
    <style type="text/css">
      a:link {
        text-decoration: none;
      }

      .order-card {
        color: rgb(255, 255, 255);
      }
      .bg-c-blue {
        background: #04868f;
      }

      .bg-c-green {
        background:#4C51BF;
      }

      .bg-c-yellow {
        background: #F56565;
      }

      .bg-c-pink {
        background: #663a30;
      }


      .card {

        -webkit-box-shadow: 0 1px 2.94px 0.06px rgba(4, 26, 55, 0.16);
        box-shadow: 0 1px 2.94px 0.06px rgba(4, 26, 55, 0.16);
        border: 1px solid black;
        margin-bottom: 30px;
        -webkit-transition: all 0.3s ease-in-out;
        transition: all 0.3s ease-in-out;
      }

      .card .card-block {
        padding: 25px;
      }

      .order-card i {
        font-size: 26px;
      }

      .f-left {
        float: left;
      }

      .f-right {
        float: right;
      }
      header {
      left: 0px;
      right: 0px;
    }
    </style>
  </head>
<br><br>

  <div class="container">
    <div class="row">
      <div class="col-md-4 col-xl-3">
        <div class="card bg-c-blue order-card">
          <div class="card-block">
            <h6 class="m-b-20"> <a href="admin-view-student" style="text-
decoration: none;color:white;">Total Students</a> </h6>
```

```
            <h2 class="text-right"><i class="fas fa-user-graduate f-
left"></i><span>{{total_student}}</span></h2>

          </div>
        </div>
      </div>

      <div class="col-md-4 col-xl-3">
        <div class="card bg-c-green order-card">
          <div class="card-block">
            <h6 class="m-b-20"><a href="admin-view-teacher" style="text-
decoration: none;color:white;">Total Teacher</a> </h6>
            <h2 class="text-right"><i class="fas fa-chalkboard-teacher f-
left"></i><span>{{total_teacher}}</span></h2>

          </div>
        </div>
      </div>

      <div class="col-md-4 col-xl-3">
        <div class="card bg-c-yellow order-card">
          <div class="card-block">
            <h6 class="m-b-20"><a href="admin-view-course" style="text-
decoration: none;color:white;">Total Courses</a></h6>
           <h2 class="text-right"><i class="fas fa-book f-
left"></i><span>{{total_course}}</span></h2>

          </div>
        </div>
      </div>

      <div class="col-md-4 col-xl-3">
        <div class="card bg-c-pink order-card">
          <div class="card-block">
            <h6 class="m-b-20"><a href="admin-view-question" style="text-
decoration: none;color:white;">Total Questions</a></h6>
            <h2 class="text-right"><i class="fas fa-question-circle f-
left"></i><span>{{total_question}}</span></h2>

          </div>
        </div>
      </div>


    </div>
  </div>


<br><br><br><br><br><br><br><br><br><br>

<script src="http://netdna.bootstrapcdn.com/bootstrap/4.0.0-
beta/js/bootstrap.min.js"></script>


{% endblock content %}
```

### Start_exam.html:

```
{% extends 'student/studentbase.html' %}
{% block content %}
{%load static%}

<head>
  <link href="//netdna.bootstrapcdn.com/bootstrap/3.0.0/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">
  <script
src="//netdna.bootstrapcdn.com/bootstrap/3.0.0/js/bootstrap.min.js"></script>
  <script src="//code.jquery.com/jquery-1.11.1.min.js"></script>


</head>


<div class="jumbotron my-4">

  <form class="form" autocomplete="off" onsubmit="return saveAns()"
action="/student/calculate-marks" method="POST">
    {% csrf_token %}
    <h1 style="text-align: center;">{{course.course_name}}</h1>
    {% for q in questions%}
    <h3 class="text-info">{{ forloop.counter }}. {{q.question}}</h3><h4
style="text-align: right;">[Marks {{q.marks}}]</h4>

        <input type="hidden" name="csrfmiddlewaretoken"
value="C24rUotmdHawVQJL3KrqiWxvti8UffOFYUc8TRbZtLt36AVLdP3jbkzUVe3beRAa">


          <div class="form-check mx-4">
            <input class="form-check-input" type="radio" name="{{
forloop.counter }}" id="{{q.option1}}" value="Option1">
            <label class="form-check-label" for="option1">
              {{q.option1}}
            </label>
          </div>


          <div class="form-check mx-4">
            <input class="form-check-input" type="radio" name="{{
forloop.counter }}" id="{{q.option2}}" value="Option2">
            <label class="form-check-label" for="option2">
              {{q.option2}}
            </label>
          </div>


          <div class="form-check mx-4">
            <input class="form-check-input" type="radio" name="{{
forloop.counter }}" id="{{q.option3}}" value="Option3">
            <label class="form-check-label" for="option3">
              {{q.option3}}
            </label>
          </div>
          <div class="form-check mx-4">
            <input class="form-check-input" type="radio" name="{{
forloop.counter }}" id="{{q.option4}}" value="Option4">
            <label class="form-check-label" for="option4">
              {{q.option4}}
            </label>
          </div>

        {% endfor %}
        <input class="btn btn-info btn-lg"   type="submit" value="Submit">
    </form>
  </div>

<script>
    function saveAns(){

        var ele = document.getElementsByTagName('input');
        for(i = 0; i < ele.length; i++) {
            if(ele[i].type="radio") {
                if(ele[i].checked){
                  setCookie(ele[i].name,ele[i].value,3)
                }
            }
        }

    }

    function setCookie(cname, cvalue, exdays) {
  var d = new Date();
  d.setTime(d.getTime() + (exdays*24*60*60*1000));
  var expires = "expires="+ d.toUTCString();
  document.cookie = cname + "=" + cvalue + ";" + expires + ";path=/";
}

</script>

<br><br><br><br><br><br>
{% endblock content %}
```

**studentdashboard.html:**

```
{% extends 'student/studentbase.html' %}
{% load widget_tweaks %}
{% block content %}


<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no">

    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script
>
    <link href="http://netdna.bootstrapcdn.com/bootstrap/4.0.0-
```

```
beta/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
    <link href="https://maxcdn.bootstrapcdn.com/font-awesome/4.3.0/css/font-
awesome.min.css" rel="stylesheet">
    <style type="text/css">
      a:link {
        text-decoration: none;
      }

      .order-card {
        color: rgb(255, 255, 255);
      }

      .bg-c-blue {
        background: #04868f;
      }

      .bg-c-green {
        background:#4C51BF;
      }

      .bg-c-yellow {
        background: #F56565;
      }

      .bg-c-pink {
        background: #663a30;
      }


      .card {

        -webkit-box-shadow: 0 1px 2.94px 0.06px rgba(4, 26, 55, 0.16);
        box-shadow: 0 1px 2.94px 0.06px rgba(4, 26, 55, 0.16);
        border: 1px solid black;
        margin-bottom: 30px;
        -webkit-transition: all 0.3s ease-in-out;
        transition: all 0.3s ease-in-out;
      }

      .card .card-block {
        padding: 25px;
      }

      .order-card i {
        font-size: 26px;
      }

      .f-left {
        float: left;
      }

      .f-right {
        float: right;
      }
      header {
      left: 0px;
```

```
   right: 0px;
  }
  </style>
 </head>
<br><br>

 <div class="container">
  <div class="row">


   <div class="col-md-4 col-xl-6">
    <div class="card bg-c-yellow order-card">
     <div class="card-block">
      <h6 class="m-b-20">Total Exams Available</h6>
     <h2 class="text-right"><i class="fas fa-book f-
left"></i><span>{{total_course}}</span></h2>

     </div>
    </div>
   </div>

   <div class="col-md-4 col-xl-6">
    <div class="card bg-c-pink order-card">
     <div class="card-block">
      <h6 class="m-b-20">Total Questions</h6>
      <h2 class="text-right"><i class="fas fa-question-circle f-
left"></i><span>{{total_question}}</span></h2>

     </div>
    </div>
   </div>
  </div>
 </div>


<br><br><br><br><br><br><br><br><br><br>

<script src="http://netdna.bootstrapcdn.com/bootstrap/4.0.0-
beta/js/bootstrap.min.js"></script>


{% endblock content %}
```

## Student_marks.html:

```
{% extends 'student/studentbase.html' %}
{% block content %}
{%load static%}

<head>
 <link href="//netdna.bootstrapcdn.com/bootstrap/3.0.0/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">
 <script
src="//netdna.bootstrapcdn.com/bootstrap/3.0.0/js/bootstrap.min.js"></script>
 <script src="//code.jquery.com/jquery-1.11.1.min.js"></script>

 <style media="screen">
```

```
    a:link {
      text-decoration: none;
    }

    h6 {
      text-align: center;
    }

    .row {
      margin: 100px;
    }
  </style>
</head>
<br><br>
<div class="container">
  <div class="panel panel-primary">
    <div class="panel-heading">
      <h6 class="panel-title">View Marks</h6>
    </div>
    <table class="table table-hover" id="dev-table">
      <thead>
        <tr>

          <th>Exam Name</th>

          <th>View Marks</th>
        </tr>
      </thead>
      {% for t in courses %}
      <tr>

        <td> {{t.course_name}}</td>

        <td><a class="btn btn-danger btn-xs" href="{% url 'check-marks' t.id
%}"><span class="glyphicon glyphicon-eye-open"></span></a></td>
      </tr>
      {% endfor %}
    </table>
  </div>
</div>

<br><br><br><br><br><br>
{% endblock content %}
```

### Studentlogin.html:

```
<!DOCTYPE html>
{% load widget_tweaks %}
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title></title>
    <style media="screen">
      body {
    margin: 0;
    padding: 0;
    font-family: sans-serif;
    background: linear-gradient(to right, #b92b27, #1565c0)
}

.box {
    width: 500px;
    padding: 40px;
    position: absolute;
    top: 50%;
    left: 50%;
    background: #191919;

    text-align: center;
    transition: 0.25s;
    margin-top: 100px
}

.box input[type="text"],
.box input[type="password"] {
    border: 0;
    background: none;
    display: block;
    margin: 20px auto;
    text-align: center;
    border: 2px solid #3498db;
    padding: 10px 10px;
    width: 250px;
    outline: none;
    color: white;
    border-radius: 24px;
    transition: 0.25s
}

.box h1 {
    color: white;
    text-transform: uppercase;
    font-weight: 500
}

.box input[type="text"]:focus,
.box input[type="password"]:focus {
    width: 300px;
    border-color: #2ecc71
}

.box input[type="submit"] {
    border: 0;
    background: none;
    display: block;
    margin: 20px auto;
    text-align: center;
    border: 2px solid #2ecc71;
    padding: 14px 40px;
    outline: none;
    color: white;
    border-radius: 24px;
    transition: 0.25s;
    cursor: pointer
```

```
}

.box input[type="submit"]:hover {
    background: #2ecc71
}


    </style>
  </head>
  <body>
    {% include "quiz/navbar.html" %}
    <div class="container">
    <div class="row">
        <div class="col-md-6">
            <div class="card">
                <form class="box" method="post">
                    {% csrf_token %}
                    <h1>Student Login</h1>
                    <p class="text-muted"> Please enter your login and
password!</p>

                     {% render_field form.username class="form-control"
placeholder="Username" %}
                     {% render_field form.password class="form-control"
placeholder="Password" %}

                     <input type="submit" name="" value="Login">

                </form>
            </div>
        </div>
    </div>
</div>
<br><br><br><br><br><br><br><br><br>
<br><br><br><br><br><br><br><br><br>
<br><br><br><br><br>
{% include "quiz/footer.html" %}
  </body>
</html>
```

### Teacher_approval.html:

```
<!DOCTYPE html>
{% load widget_tweaks %}


  <style media="screen">
    .jumbotron {
      margin-top: 0px;
      margin-bottom: 0px;
    }

    .jumbotron h1 {
      text-align: center;
    }

    .alert {
      margin: 0px;
  }
 </style>
 <body>
  {% include "quiz/navbar.html" %}

   <div class="jumbotron" style="margin-top: 0px;
    margin-bottom: 0px;">
    <h1 class="display-4">Hello {{request.user.first_name}}</h1>
    <p class="lead">Your Account is not approved till now <br><br>Our Team
is checking your profile <br><br> Soon Your Account Will Be Approved</p>
    <hr class="my-4">
    <p>Check Later</p>
    <p class="lead">
     <a class="btn btn-primary btn-lg" href="/logout" role="button">Logout
For Now</a>
    </p>
   </div>

{% include "quiz/footer.html" %}
 </body>
</html>
```

### Teacher_base.html:

```
< body>

 <input type="checkbox" id="check">
 <!--header area start-->
 <header>
  <label for="check">
   <i style="left: 200px;margin-top: 17px;"class="fas fa-bars"
id="sidebar_btn"></i>
  </label>
  <div class="left_area">
   <h3>Online Quiz</h3>
  </div>
  <div class="right_area">
   <a href="/logout" class="logout_btn">Logout</a>
  </div>
 </header>
 <!--header area end-->
 <!--sidebar start-->
 <div class="sidebar">
  <center>

   <img src="{% static "image/teacher.png" %}" class="profile_image" alt="">
   <h4>{{request.user.first_name}}</h4>
   <h6 style="color: rgb(255, 255, 255);">( Teacher )</h6>

  </center>

  <a href="/teacher/teacher-dashboard"><i class="fas fa-tachometer-
alt"></i><span>Dashboard</span></a>
  <a href="/teacher/teacher-exam"><i class="fas fa-
book"></i><span>Exam</span></a>
  <a href="/teacher/teacher-question"><i class="fas fa-question-
```

```
circle"></i><span>Questions</span></a>
  </div>
  <!--sidebar end-->


  <!--content start-->
  <div class="content">
    <br><br><br><br><br>
    {% block content %}

    {% endblock content %}


    <br><br><br>
    {% include "quiz/footer.html" %}
  </div>
  <!--content end-->
</body>
```

### student\urls.py:

```
from django.urls import path
from student import views
from django.contrib.auth.views import LoginView

urlpatterns = [
path('studentclick', views.studentclick_view),
path('studentlogin',
LoginView.as_view(template_name='student/studentlogin.html'),name='studentlogin'
),
path('studentsignup', views.student_signup_view,name='studentsignup'),
path('student-dashboard', views.student_dashboard_view,name='student-
dashboard'),
path('student-exam', views.student_exam_view,name='student-exam'),
path('take-exam/<int:pk>', views.take_exam_view,name='take-exam'),
path('start-exam/<int:pk>', views.start_exam_view,name='start-exam'),

path('calculate-marks', views.calculate_marks_view,name='calculate-marks'),
path('view-result', views.view_result_view,name='view-result'),
path('check-marks/<int:pk>', views.check_marks_view,name='check-marks'),
path('student-marks', views.student_marks_view,name='student-marks'),
]
```

### student\models.py:

```
from django.db import models
from django.contrib.auth.models import User

class Student(models.Model):
    user=models.OneToOneField(User,on_delete=models.CASCADE)
    profile_pic=
models.ImageField(upload_to='profile_pic/Student/',null=True,blank=True)
    address = models.CharField(max_length=40)
    mobile = models.CharField(max_length=20,null=False)
    @property
    def get_name(self):
        return self.user.first_name+" "+self.user.last_name
    @property
    def get_instance(self):
        return self
    def __str__(self):
        return self.user.first_name
```

**quiz\forms.py:**

```
from django import forms
from django.contrib.auth.models import User
from . import models

class ContactusForm(forms.Form):
    Name = forms.CharField(max_length=30)
    Email = forms.EmailField()
    Message =
forms.CharField(max_length=500,widget=forms.Textarea(attrs={'rows': 3, 'cols':
30}))

class TeacherSalaryForm(forms.Form):
    salary=forms.IntegerField()

class CourseForm(forms.ModelForm):
    class Meta:
        model=models.Course
        fields=['course_name','question_number','total_marks']

class QuestionForm(forms.ModelForm):

courseID=forms.ModelChoiceField(queryset=models.Course.objects.all(),empty_label
="Course Name", to_field_name="id")
    class Meta:
        model=models.Question

fields=['marks','question','option1','option2','option3','option4','answer']
        widgets = {
            'question': forms.Textarea(attrs={'rows': 3, 'cols': 50})
        }
```

**quiz\models.py:**

```
from django.db import models

from student.models import Student
class Course(models.Model):
   course_name = models.CharField(max_length=50)
   question_number = models.PositiveIntegerField()
   total_marks = models.PositiveIntegerField()
   def __str__(self):
        return self.course_name

class Question(models.Model):
    course=models.ForeignKey(Course,on_delete=models.CASCADE)
    marks=models.PositiveIntegerField()
    question=models.CharField(max_length=600)
    option1=models.CharField(max_length=200)
    option2=models.CharField(max_length=200)
    option3=models.CharField(max_length=200)
    option4=models.CharField(max_length=200)

cat=(('Option1','Option1'),('Option2','Option2'),('Option3','Option3'),('Option4
','Option4'))
    answer=models.CharField(max_length=200,choices=cat)

class Result(models.Model):
    student = models.ForeignKey(Student,on_delete=models.CASCADE)
    exam = models.ForeignKey(Course,on_delete=models.CASCADE)
    marks = models.PositiveIntegerField()
    date = models.DateTimeField(auto_now=True)
```

**onlinequiz\urls.py:**

```
from django.urls import path,include
from django.contrib import admin
from quiz import views
from django.contrib.auth.views import LogoutView,LoginView
urlpatterns = [

    path('admin/', admin.site.urls),
    path('teacher/',include('teacher.urls')),
    path('student/',include('student.urls')),



    path('',views.home_view,name=''),
    path('logout',
LogoutView.as_view(template_name='quiz/logout.html'),name='logout'),
    path('aboutus', views.aboutus_view),
    path('contactus', views.contactus_view),
    path('afterlogin', views.afterlogin_view,name='afterlogin'),



    path('adminclick', views.adminclick_view),
    path('adminlogin',
LoginView.as_view(template_name='quiz/adminlogin.html'),name='adminlogin'),
    path('admin-dashboard', views.admin_dashboard_view,name='admin-dashboard'),
    path('admin-teacher', views.admin_teacher_view,name='admin-teacher'),
    path('admin-view-teacher', views.admin_view_teacher_view,name='admin-view-
teacher'),
    path('update-teacher/<int:pk>', views.update_teacher_view,name='update-
teacher'),
    path('delete-teacher/<int:pk>', views.delete_teacher_view,name='delete-
teacher'),
    path('admin-view-pending-teacher',
views.admin_view_pending_teacher_view,name='admin-view-pending-teacher'),
    path('admin-view-teacher-salary',
views.admin_view_teacher_salary_view,name='admin-view-teacher-salary'),
    path('approve-teacher/<int:pk>', views.approve_teacher_view,name='approve-
teacher'),
    path('reject-teacher/<int:pk>', views.reject_teacher_view,name='reject-
teacher'),
    path('admin-student', views.admin_student_view,name='admin-student'),
    path('admin-view-student', views.admin_view_student_view,name='admin-view-
student'),
    path('admin-view-student-marks',
views.admin_view_student_marks_view,name='admin-view-student-marks'),
    path('admin-view-marks/<int:pk>', views.admin_view_marks_view,name='admin-
view-marks'),
    path('admin-check-marks/<int:pk>', views.admin_check_marks_view,name='admin-
check-marks'),
    path('update-student/<int:pk>', views.update_student_view,name='update-
student'),
    path('delete-student/<int:pk>', views.delete_student_view,name='delete-
student'),

    path('admin-course', views.admin_course_view,name='admin-course'),
    path('admin-add-course', views.admin_add_course_view,name='admin-add-
course'),
    path('admin-view-course', views.admin_view_course_view,name='admin-view-
course'),
    path('delete-course/<int:pk>', views.delete_course_view,name='delete-
course'),

    path('admin-question', views.admin_question_view,name='admin-question'),
    path('admin-add-question', views.admin_add_question_view,name='admin-add-
question'),
    path('admin-view-question', views.admin_view_question_view,name='admin-view-
question'),
    path('view-question/<int:pk>', views.view_question_view,name='view-
question'),
    path('delete-question/<int:pk>', views.delete_question_view,name='delete-
question'),


]
```

]

**student\forms.py:**

```
from django import forms
from django.contrib.auth.models import User
from . import models
from quiz import models as QMODEL

class StudentUserForm(forms.ModelForm):
    class Meta:
        model=User
        fields=['first_name','last_name','username','password']
        widgets = {
        'password': forms.PasswordInput()
        }

class StudentForm(forms.ModelForm):
    class Meta:
        model=models.Student
        fields=['address','mobile','profile_pic']
```

**student\views.py:**

```
from django.shortcuts import render,redirect,reverse
from . import forms,models
from django.db.models import Sum
from django.contrib.auth.models import Group
from django.http import HttpResponseRedirect
from django.contrib.auth.decorators import login_required,user_passes_test
from django.conf import settings
from datetime import date, timedelta
from quiz import models as QMODEL
from teacher import models as TMODEL


#for showing signup/login button for student
def studentclick_view(request):
    if request.user.is_authenticated:
        return HttpResponseRedirect('afterlogin')
    return render(request,'student/studentclick.html')

def student_signup_view(request):
    userForm=forms.StudentUserForm()
    studentForm=forms.StudentForm()
    mydict={'userForm':userForm,'studentForm':studentForm}
    if request.method=='POST':
        userForm=forms.StudentUserForm(request.POST)
        studentForm=forms.StudentForm(request.POST,request.FILES)
        if userForm.is_valid() and studentForm.is_valid():
            user=userForm.save()
            user.set_password(user.password)
            user.save()
            student=studentForm.save(commit=False)
            student.user=user
            student.save()
            my_student_group = Group.objects.get_or_create(name='STUDENT')
            my_student_group[0].user_set.add(user)
        return HttpResponseRedirect('studentlogin')
    return render(request,'student/studentsignup.html',context=mydict)

def is_student(user):
    return user.groups.filter(name='STUDENT').exists()

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def student_dashboard_view(request):
    dict={

    'total_course':QMODEL.Course.objects.all().count(),
    'total_question':QMODEL.Question.objects.all().count(),
    }
    return render(request,'student/student_dashboard.html',context=dict)

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def student_exam_view(request):
    courses=QMODEL.Course.objects.all()
    return render(request,'student/student_exam.html',{'courses':courses})
```

```
@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def take_exam_view(request,pk):
    course=QMODEL.Course.objects.get(id=pk)
    total_questions=QMODEL.Question.objects.all().filter(course=course).count()
    questions=QMODEL.Question.objects.all().filter(course=course)
    total_marks=0
    for q in questions:
        total_marks=total_marks + q.marks

    return
render(request,'student/take_exam.html',{'course':course,'total_questions':total
_questions,'total_marks':total_marks})

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def start_exam_view(request,pk):
    course=QMODEL.Course.objects.get(id=pk)
    questions=QMODEL.Question.objects.all().filter(course=course)
    if request.method=='POST':
        pass
    response=
render(request,'student/start_exam.html',{'course':course,'questions':questions}
)
    response.set_cookie('course_id',course.id)
    return response


@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def calculate_marks_view(request):
    if request.COOKIES.get('course_id') is not None:
        course_id = request.COOKIES.get('course_id')
        course=QMODEL.Course.objects.get(id=course_id)

        total_marks=0
        questions=QMODEL.Question.objects.all().filter(course=course)
        for i in range(len(questions)):

            selected_ans = request.COOKIES.get(str(i+1))
            actual_answer = questions[i].answer
            if selected_ans == actual_answer:
                total_marks = total_marks + questions[i].marks
        student = models.Student.objects.get(user_id=request.user.id)
        result = QMODEL.Result()
        result.marks=total_marks
        result.exam=course
        result.student=student
        result.save()

        return HttpResponseRedirect('view-result')



@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def view_result_view(request):
    courses=QMODEL.Course.objects.all()
    return render(request,'student/view_result.html',{'courses':courses})


@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def check_marks_view(request,pk):
    course=QMODEL.Course.objects.get(id=pk)
    student = models.Student.objects.get(user_id=request.user.id)
    results=
QMODEL.Result.objects.all().filter(exam=course).filter(student=student)
    return render(request,'student/check_marks.html',{'results':results})

@login_required(login_url='studentlogin')
@user_passes_test(is_student)
def student_marks_view(request):
    courses=QMODEL.Course.objects.all()
    return render(request,'student/student_marks.html',{'courses':courses})
```

**quiz\views.py:**

```
from django.shortcuts import render,redirect,reverse
from . import forms,models
from django.db.models import Sum
from django.contrib.auth.models import Group
from django.http import HttpResponseRedirect
from django.contrib.auth.decorators import login_required,user_passes_test
from django.conf import settings
from datetime import date, timedelta
from django.db.models import Q
from django.core.mail import send_mail
from teacher import models as TMODEL
from student import models as SMODEL
from teacher import forms as TFORM
from student import forms as SFORM
from django.contrib.auth.models import User



def home_view(request):
    if request.user.is_authenticated:
        return HttpResponseRedirect('afterlogin')
    return render(request,'quiz/index.html')


def is_teacher(user):
    return user.groups.filter(name='TEACHER').exists()

def is_student(user):
    return user.groups.filter(name='STUDENT').exists()

def afterlogin_view(request):
    if is_student(request.user):
        return redirect('student/student-dashboard')

    elif is_teacher(request.user):
```

```
accountapproval=TMODEL.Teacher.objects.all().filter(user_id=request.user.id,stat
us=True)
        if accountapproval:
            return redirect('teacher/teacher-dashboard')
        else:
            return render(request,'teacher/teacher_wait_for_approval.html')
    else:
        return redirect('admin-dashboard')



def adminclick_view(request):
    if request.user.is_authenticated:
        return HttpResponseRedirect('afterlogin')
    return HttpResponseRedirect('adminlogin')


@login_required(login_url='adminlogin')
def admin_dashboard_view(request):
    dict={
    'total_student':SMODEL.Student.objects.all().count(),
    'total_teacher':TMODEL.Teacher.objects.all().filter(status=True).count(),
    'total_course':models.Course.objects.all().count(),
    'total_question':models.Question.objects.all().count(),
    }
    return render(request,'quiz/admin_dashboard.html',context=dict)

@login_required(login_url='adminlogin')
def admin_teacher_view(request):
    dict={
    'total_teacher':TMODEL.Teacher.objects.all().filter(status=True).count(),
    'pending_teacher':TMODEL.Teacher.objects.all().filter(status=False).count(),

'salary':TMODEL.Teacher.objects.all().filter(status=True).aggregate(Sum('salary'
))['salary__sum'],
    }
    return render(request,'quiz/admin_teacher.html',context=dict)

@login_required(login_url='adminlogin')
def admin_view_teacher_view(request):
    teachers= TMODEL.Teacher.objects.all().filter(status=True)
    return render(request,'quiz/admin_view_teacher.html',{'teachers':teachers})


@login_required(login_url='adminlogin')
def update_teacher_view(request,pk):
    teacher=TMODEL.Teacher.objects.get(id=pk)
    user=TMODEL.User.objects.get(id=teacher.user_id)
    userForm=TFORM.TeacherUserForm(instance=user)
    teacherForm=TFORM.TeacherForm(request.FILES,instance=teacher)
    mydict={'userForm':userForm,'teacherForm':teacherForm}
    if request.method=='POST':
        userForm=TFORM.TeacherUserForm(request.POST,instance=user)

teacherForm=TFORM.TeacherForm(request.POST,request.FILES,instance=teacher)
        if userForm.is_valid() and teacherForm.is_valid():
            user=userForm.save()
            user.set_password(user.password)
            user.save()
            teacherForm.save()
            return redirect('admin-view-teacher')
    return render(request,'quiz/update_teacher.html',context=mydict)



@login_required(login_url='adminlogin')
def delete_teacher_view(request,pk):
    teacher=TMODEL.Teacher.objects.get(id=pk)
    user=User.objects.get(id=teacher.user_id)
    user.delete()
    teacher.delete()
    return HttpResponseRedirect('/admin-view-teacher')




@login_required(login_url='adminlogin')
def admin_view_pending_teacher_view(request):
    teachers= TMODEL.Teacher.objects.all().filter(status=False)
    return
render(request,'quiz/admin_view_pending_teacher.html',{'teachers':teachers})


@login_required(login_url='adminlogin')
def approve_teacher_view(request,pk):
    teacherSalary=forms.TeacherSalaryForm()
    if request.method=='POST':
        teacherSalary=forms.TeacherSalaryForm(request.POST)
        if teacherSalary.is_valid():
            teacher=TMODEL.Teacher.objects.get(id=pk)
            teacher.salary=teacherSalary.cleaned_data['salary']
            teacher.status=True
            teacher.save()
        else:
            print("form is invalid")
        return HttpResponseRedirect('/admin-view-pending-teacher')
    return
render(request,'quiz/salary_form.html',{'teacherSalary':teacherSalary})

@login_required(login_url='adminlogin')
def reject_teacher_view(request,pk):
    teacher=TMODEL.Teacher.objects.get(id=pk)
    user=User.objects.get(id=teacher.user_id)
    user.delete()
    teacher.delete()
    return HttpResponseRedirect('/admin-view-pending-teacher')

@login_required(login_url='adminlogin')
def admin_view_teacher_salary_view(request):
    teachers= TMODEL.Teacher.objects.all().filter(status=True)
    return
render(request,'quiz/admin_view_teacher_salary.html',{'teachers':teachers})
```

```
@login_required(login_url='adminlogin')
def admin_student_view(request):
    dict={
    'total_student':SMODEL.Student.objects.all().count(),
    }
    return render(request,'quiz/admin_student.html',context=dict)

@login_required(login_url='adminlogin')
def admin_view_student_view(request):
    students= SMODEL.Student.objects.all()
    return render(request,'quiz/admin_view_student.html',{'students':students})



@login_required(login_url='adminlogin')
def update_student_view(request,pk):
    student=SMODEL.Student.objects.get(id=pk)
    user=SMODEL.User.objects.get(id=student.user_id)
    userForm=SFORM.StudentUserForm(instance=user)
    studentForm=SFORM.StudentForm(request.FILES,instance=student)
    mydict={'userForm':userForm,'studentForm':studentForm}
    if request.method=='POST':
        userForm=SFORM.StudentUserForm(request.POST,instance=user)

studentForm=SFORM.StudentForm(request.POST,request.FILES,instance=student)
        if userForm.is_valid() and studentForm.is_valid():
            user=userForm.save()
            user.set_password(user.password)
            user.save()
            studentForm.save()
            return redirect('admin-view-student')
    return render(request,'quiz/update_student.html',context=mydict)



@login_required(login_url='adminlogin')
def delete_student_view(request,pk):
    student=SMODEL.Student.objects.get(id=pk)
    user=User.objects.get(id=student.user_id)
    user.delete()
    student.delete()
    return HttpResponseRedirect('/admin-view-student')


@login_required(login_url='adminlogin')
def admin_course_view(request):
    return render(request,'quiz/admin_course.html')


@login_required(login_url='adminlogin')
def admin_add_course_view(request):
    courseForm=forms.CourseForm()
    if request.method=='POST':
        courseForm=forms.CourseForm(request.POST)
        if courseForm.is_valid():
            courseForm.save()
        else:
            print("form is invalid")
        return HttpResponseRedirect('/admin-view-course')
    return
render(request,'quiz/admin_add_course.html',{'courseForm':courseForm})


@login_required(login_url='adminlogin')
def admin_view_course_view(request):
    courses = models.Course.objects.all()
    return render(request,'quiz/admin_view_course.html',{'courses':courses})

@login_required(login_url='adminlogin')
def delete_course_view(request,pk):
    course=models.Course.objects.get(id=pk)
    course.delete()
    return HttpResponseRedirect('/admin-view-course')



@login_required(login_url='adminlogin')
def admin_question_view(request):
    return render(request,'quiz/admin_question.html')


@login_required(login_url='adminlogin')
def admin_add_question_view(request):
    questionForm=forms.QuestionForm()
    if request.method=='POST':
        questionForm=forms.QuestionForm(request.POST)
        if questionForm.is_valid():
            question=questionForm.save(commit=False)
            course=models.Course.objects.get(id=request.POST.get('courseID'))
            question.course=course
            question.save()
        else:
            print("form is invalid")
        return HttpResponseRedirect('/admin-view-question')
    return
render(request,'quiz/admin_add_question.html',{'questionForm':questionForm})


@login_required(login_url='adminlogin')
def admin_view_question_view(request):
    courses= models.Course.objects.all()
    return render(request,'quiz/admin_view_question.html',{'courses':courses})

@login_required(login_url='adminlogin')
def view_question_view(request,pk):
    questions=models.Question.objects.all().filter(course_id=pk)
    return render(request,'quiz/view_question.html',{'questions':questions})

@login_required(login_url='adminlogin')
def delete_question_view(request,pk):
    question=models.Question.objects.get(id=pk)
    question.delete()
    return HttpResponseRedirect('/admin-view-question')

@login_required(login_url='adminlogin')
def admin_view_student_marks_view(request):
    students= SMODEL.Student.objects.all()
    return
render(request,'quiz/admin_view_student_marks.html',{'students':students})

@login_required(login_url='adminlogin')
def admin_view_marks_view(request,pk):
    courses = models.Course.objects.all()
    response =  render(request,'quiz/admin_view_marks.html',{'courses':courses})
    response.set_cookie('student_id',str(pk))
    return response

@login_required(login_url='adminlogin')
def admin_check_marks_view(request,pk):
    course = models.Course.objects.get(id=pk)
    student_id = request.COOKIES.get('student_id')
    student= SMODEL.Student.objects.get(id=student_id)

    results=
models.Result.objects.all().filter(exam=course).filter(student=student)
    return render(request,'quiz/admin_check_marks.html',{'results':results})




def aboutus_view(request):
    return render(request,'quiz/aboutus.html')

def contactus_view(request):
    sub = forms.ContactusForm()
    if request.method == 'POST':
        sub = forms.ContactusForm(request.POST)
        if sub.is_valid():
            email = sub.cleaned_data['Email']
            name=sub.cleaned_data['Name']
            message = sub.cleaned_data['Message']
            send_mail(str(name)+' ||
'+str(email),message,settings.EMAIL_HOST_USER, settings.EMAIL_RECEIVING_USER,
fail_silently = False)
            return render(request, 'quiz/contactussuccess.html')
    return render(request, 'quiz/contactus.html', {'form':sub})
```

**teacher\urls.py:**

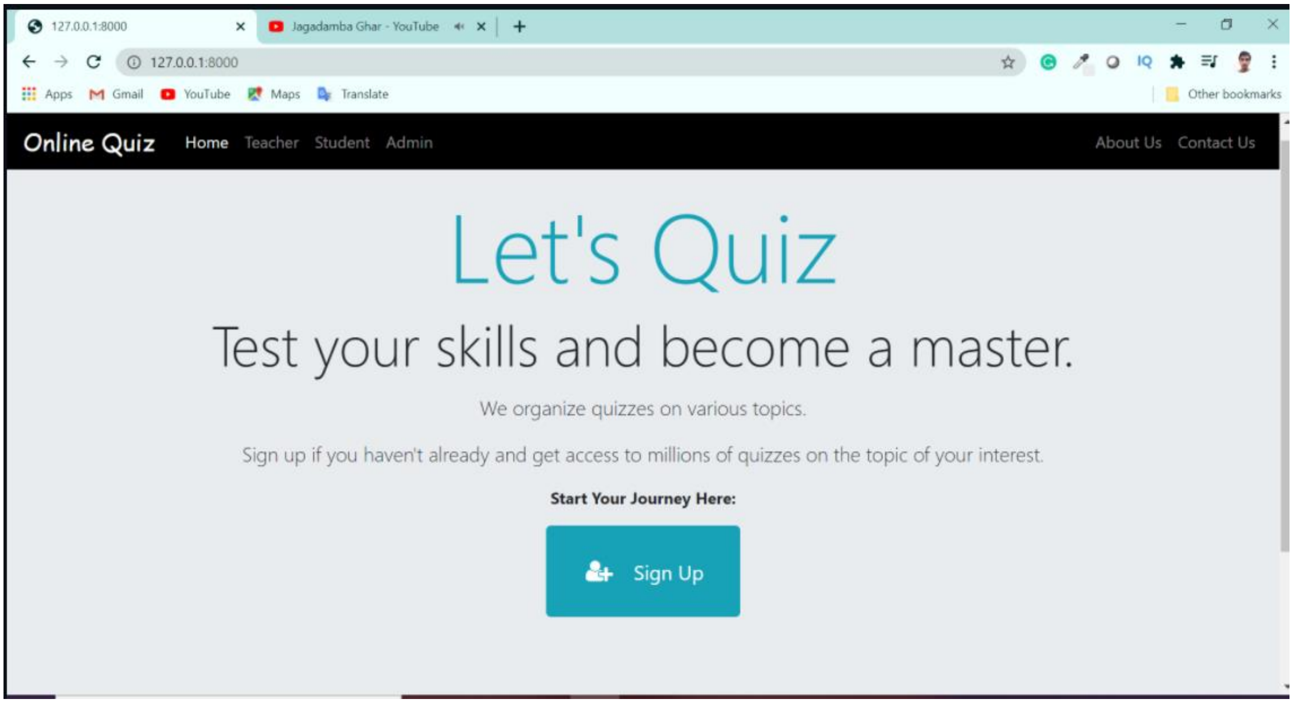
```
from django.urls import path
from teacher import views
from django.contrib.auth.views import LoginView

urlpatterns = [
path('teacherclick', views.teacherclick_view),
path('teacherlogin',
LoginView.as_view(template_name='teacher/teacherlogin.html'),name='teacherlogin'
),
path('teachersignup', views.teacher_signup_view,name='teachersignup'),
path('teacher-dashboard', views.teacher_dashboard_view,name='teacher-
dashboard'),
path('teacher-exam', views.teacher_exam_view,name='teacher-exam'),
path('teacher-add-exam', views.teacher_add_exam_view,name='teacher-add-exam'),
path('teacher-view-exam', views.teacher_view_exam_view,name='teacher-view-exam'),
path('delete-exam/<int:pk>', views.delete_exam_view,name='delete-exam'),]
```

**Output:**

**Screen Shots:**

**Home Page:**



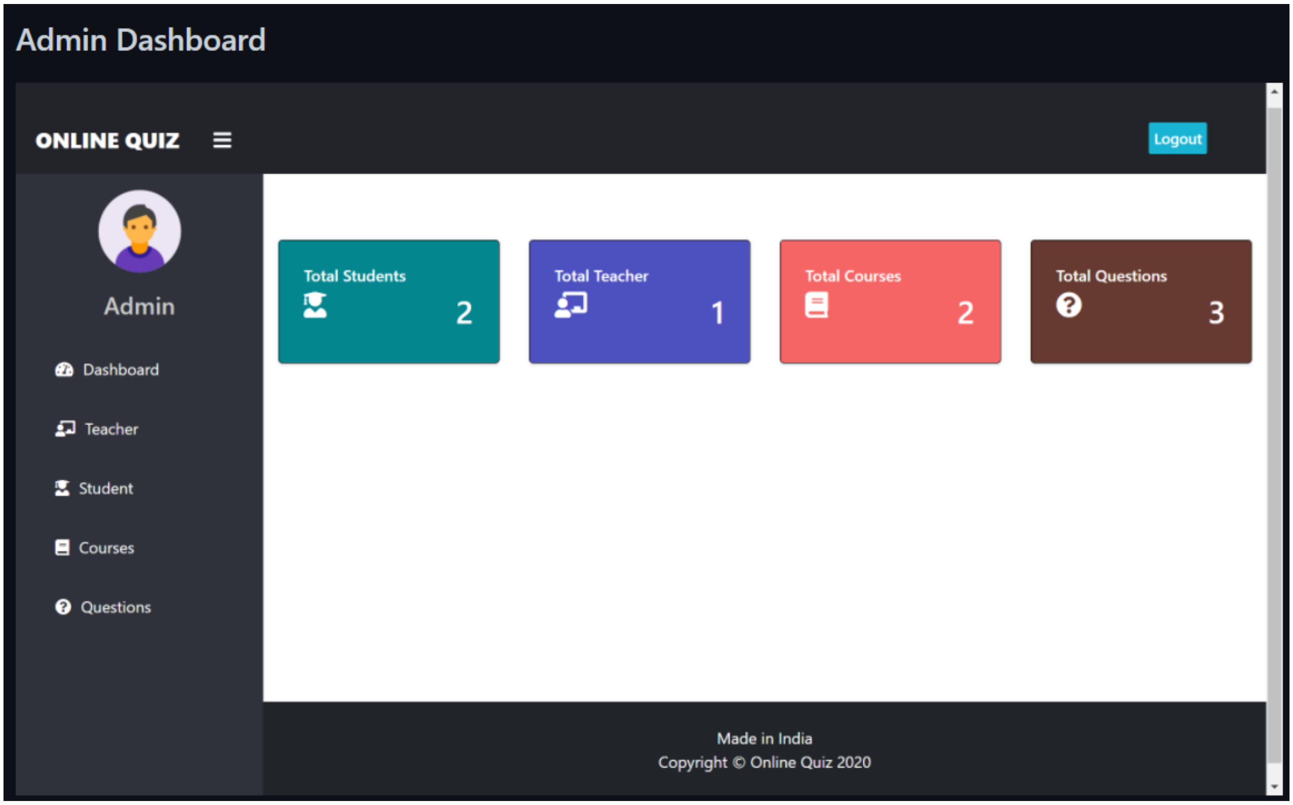
Let's Quiz

Test your skills and become a master.

We organize quizzes on various topics.

Sign up if you haven't already and get access to millions of quizzes on the topic of your interest.

**Start Your Journey Here:**



Admin Dashboard

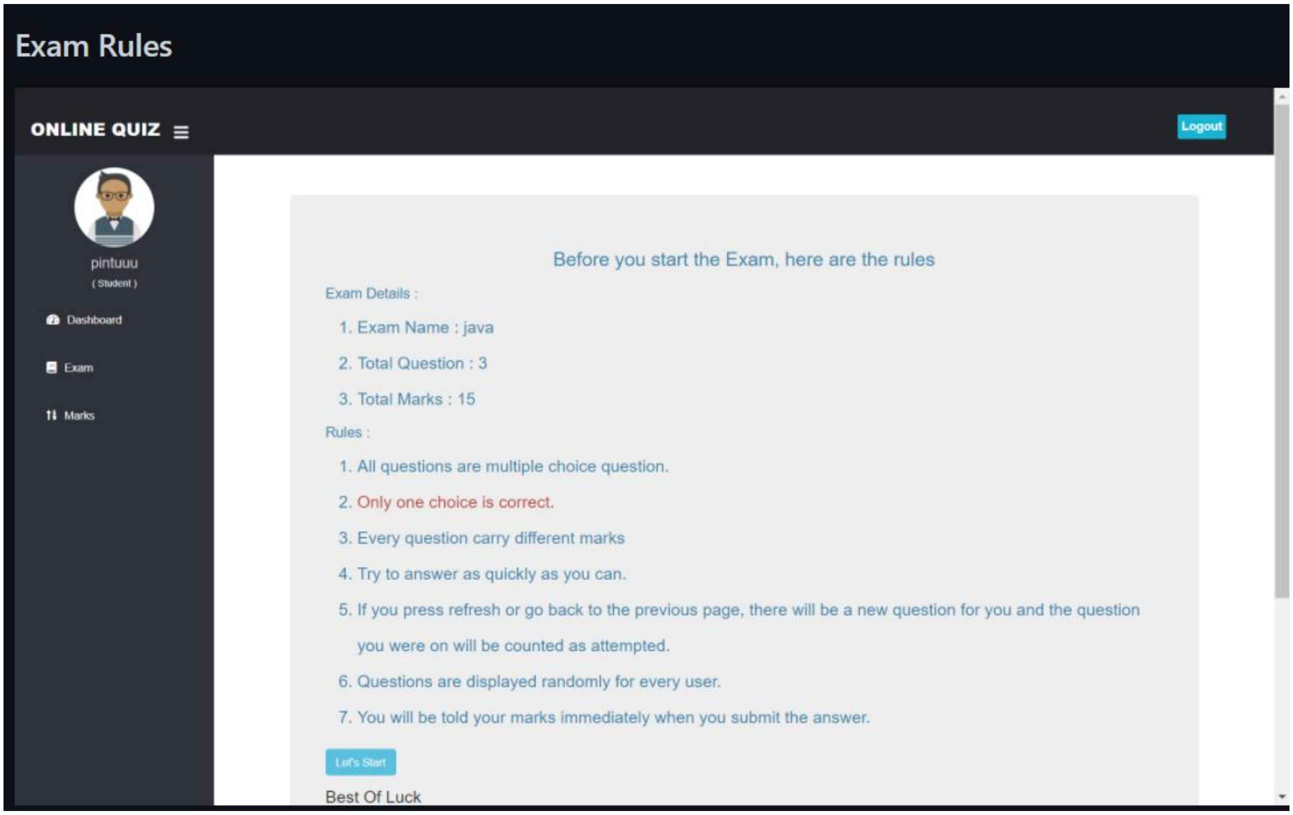
Total Students 2

Total Teacher 1

Total Courses 2

Total Questions 3

Made in India
Copyright © Online Quiz 2020



Exam Rules

Before you start the Exam, here are the rules

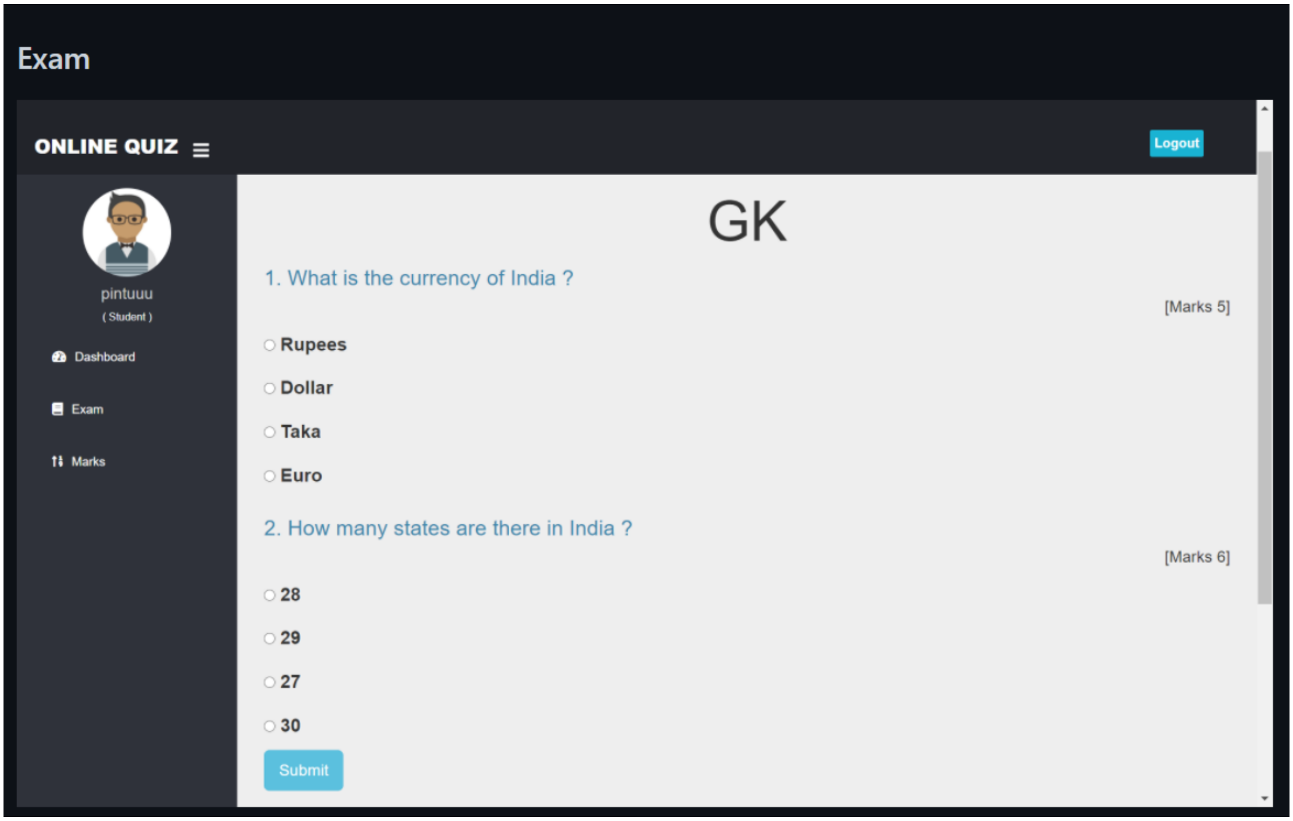
Exam Details :

1. Exam Name : java
2. Total Question : 3
3. Total Marks : 15

Rules :

1. All questions are multiple choice question.
2. Only one choice is correct.
3. Every question carry different marks
4. Try to answer as quickly as you can.
5. If you press refresh or go back to the previous page, there will be a new question for you and the question you were on will be counted as attempted.
6. Questions are displayed randomly for every user.
7. You will be told your marks immediately when you submit the answer.

Best Of Luck



Exam

ONLINE QUIZ ≡

Logout

pintuuu
( Student )

Dashboard

Exam

Marks

GK

1. What is the currency of India ?

[Marks 5]

○ Rupees

○ Dollar

○ Taka

○ Euro

2. How many states are there in India ?

[Marks 6]

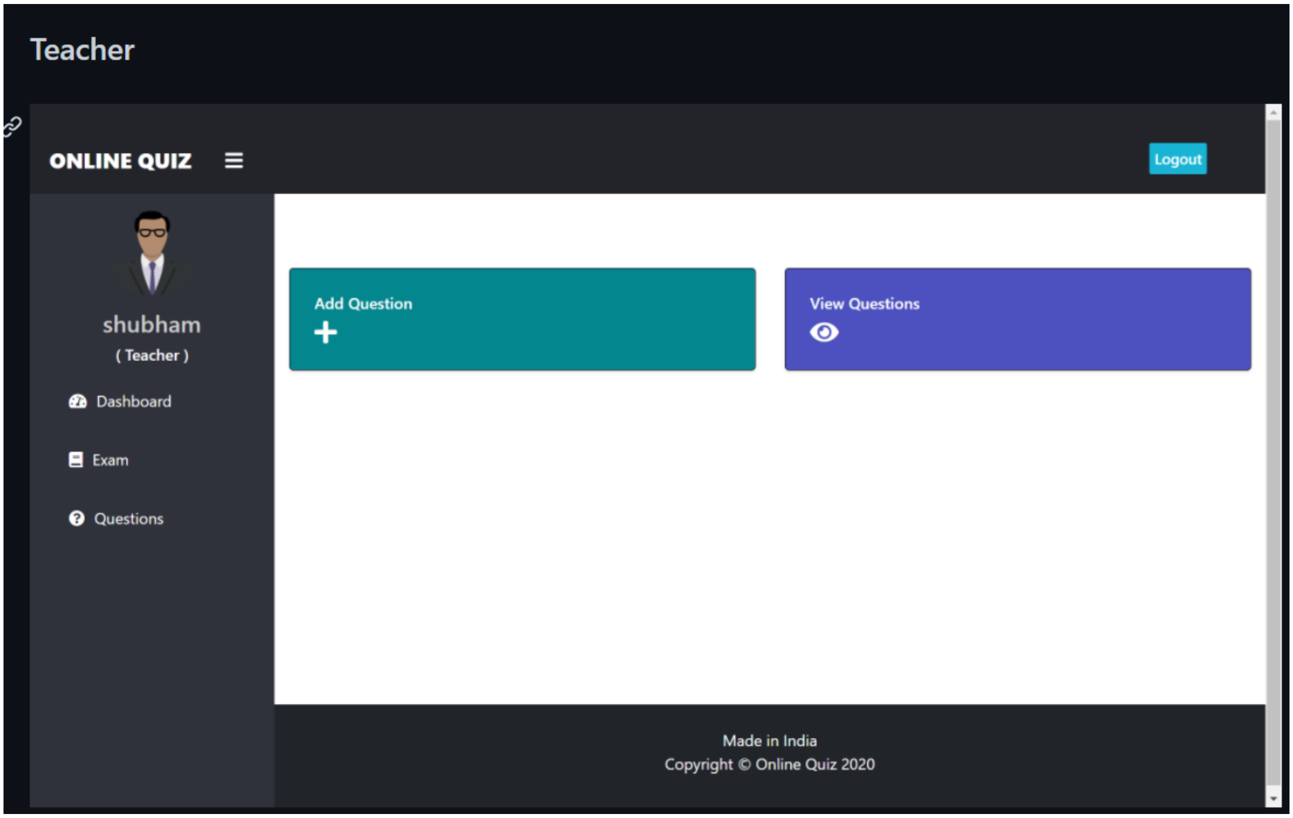
○ 28

○ 29

○ 27

○ 30

Submit



Teacher

ONLINE QUIZ ≡

Logout

shubham
( Teacher )

Dashboard

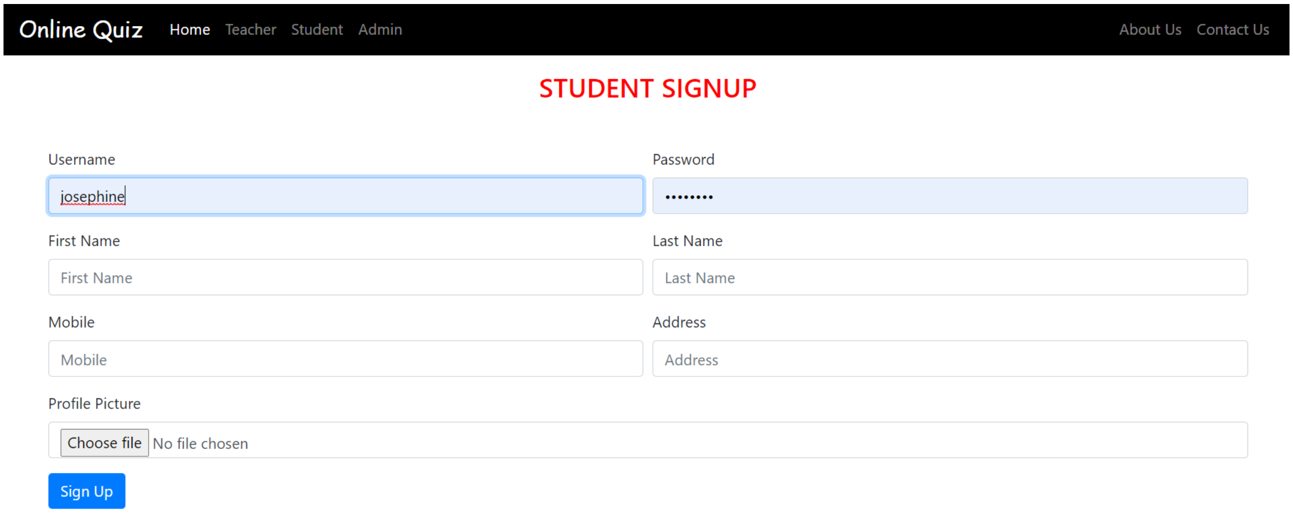
Exam

Questions

Add Question
+

View Questions

Made in India
Copyright © Online Quiz 2020



STUDENT SIGNUP

Username: josephine

Password: ••••••••

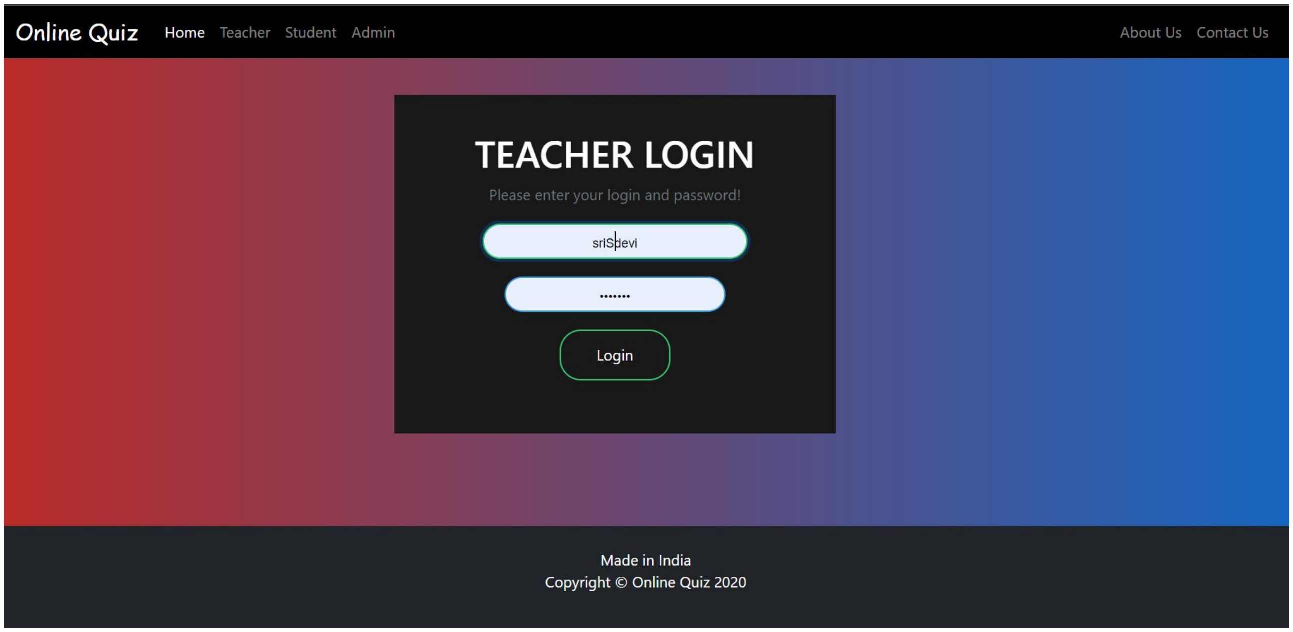
First Name: First Name

Last Name: Last Name

Mobile: Mobile

Address: Address

Profile Picture: No file chosen



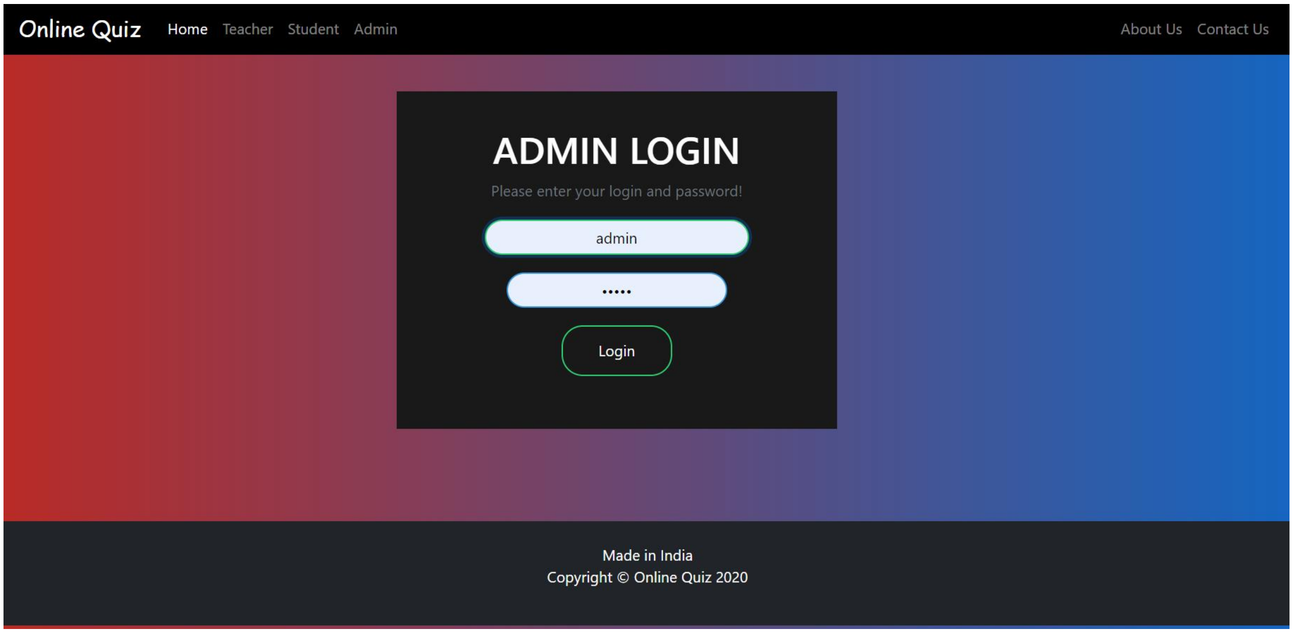
TEACHER LOGIN

Please enter your login and password!

sriSdevi

•••••••

Made in India
Copyright © Online Quiz 2020



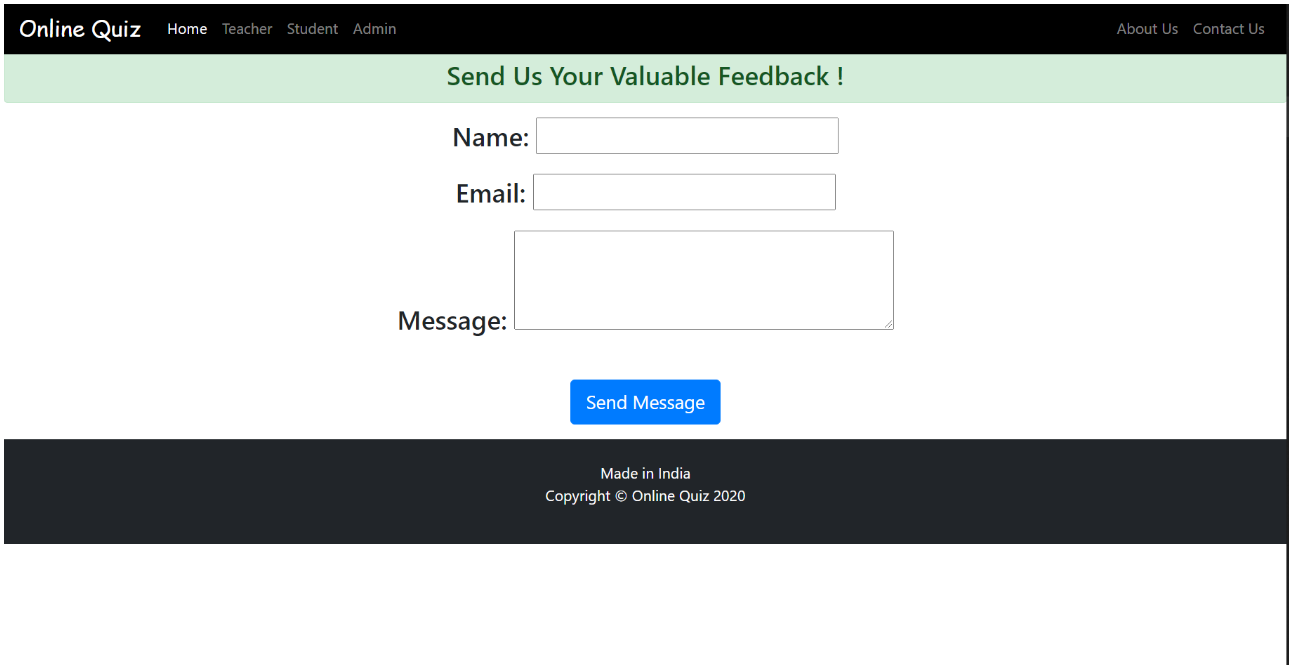
ADMIN LOGIN

Please enter your login and password!

admin

•••••

Made in India
Copyright © Online Quiz 2020



Send Us Your Valuable Feedback !

Name:

Email:

Message:

Made in India
Copyright © Online Quiz 2020

## CONCLUSION:

In our project, we have included as many features as we can make the site viable and usable and also web application very friendly and mobile application very easy to use for all user to attend quiz and test knowledge on a particular topic. Our proposed system is online and real time based on Internet, which is very much ahead to go so, user of the system would like to use this.

## REFERENCES:

https://www.tutorialspoint.com/django/index.htm
https://techvidvan.com/tutorials/django-cookies-handling/

## FUTURE SCOPE:

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:

- We can give more advance software for quiz application system including more facilities
- We will host the platform on online servers to make it accessible worldwide
- Integrate multiple load balancers to distribute the loads of the system
- Create the master and slave database structure to reduce the overload of the database queries