COMPSOFT TECHNOLOGIES

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI – 590 018.

AN INTERNSHIP REPORT
ON

"QES" (QUORA FOR ENGINEERING STUDENTS)

Bachelor of Engineering

In
Information Science and Engineering

Submitted by:Sangamesh Mulimani(2KD17CS063)

K.L.E. Society's
K.L.E. College of Engineering and Technology, Chikodi – 591201

ABOUT THE COMPANY

They are a digital service provider that aims to provide software, designing and marketing solutions to individuals and businesses, they believe that service and quality is the key to success

They provide all kinds of technological and designing solutions from Billing Software to Web Designs or any custom demand that we may have.
Experience the service like none other!

Some of their services include:

Development- They develop responsive, functional and super-fast websites. They keep User Experience in mind while creating websites. A website should load quickly and should be accessible even on a small view-port and slow internet connection.

Mobile Application- They offer a wide range of professional Android, iOS & Hybrid app development services for our global clients, from a start up to a large enterprise.

Design- They offer professional Graphic design, Brochure design & Logo design. They are experts in crafting visual content to convey the right message to the customers.

Consultancy- They are there to provide you with expert advice on your design and development requirement.

Videos- They create a polished professional video that impresses our audience.

TABLE OF CONTENTS

OVERVIEW OF THE PROJECT

Project Name: QES

Team Members: Shreyanka Kamadhen
Shruti Ballurgi
Sangamesh
Mahaning Nilajagi

This project is based on Web Development And its Applications. The main objective of this project is to learn the implementation of HTML, CSS and JavaScript. The basic webpage of this project is created using HTML and styling of the webpage is done using CSS.

This webpage is about QES. QES is a place to gain and share knowledge.

It's a platform to ask questions and connect with people who contribute unique insights and quality answers.

ABOUT QUORA

QUORA is an American social question and answer website based in Mountain View, California. It was founded on June 25, 2009, and made available to the public on June 21, 2010. Users can collaborate by editing the questions and commenting on answers that have been submitted by other users.

ABOUT QES

QES is a place to gain and share knowledge. It’s a site, where engineering students can use. This site helps the engineering student to gain knowledge. This site enables the student to ask questions and get the answers. It’s a platform to ask questions and connect with people who contribute unique insights and quality answers.

It’s a platform to ask questions and connect with people who contribute unique insights and quality answers. QES is a question and answer website for engineering students.

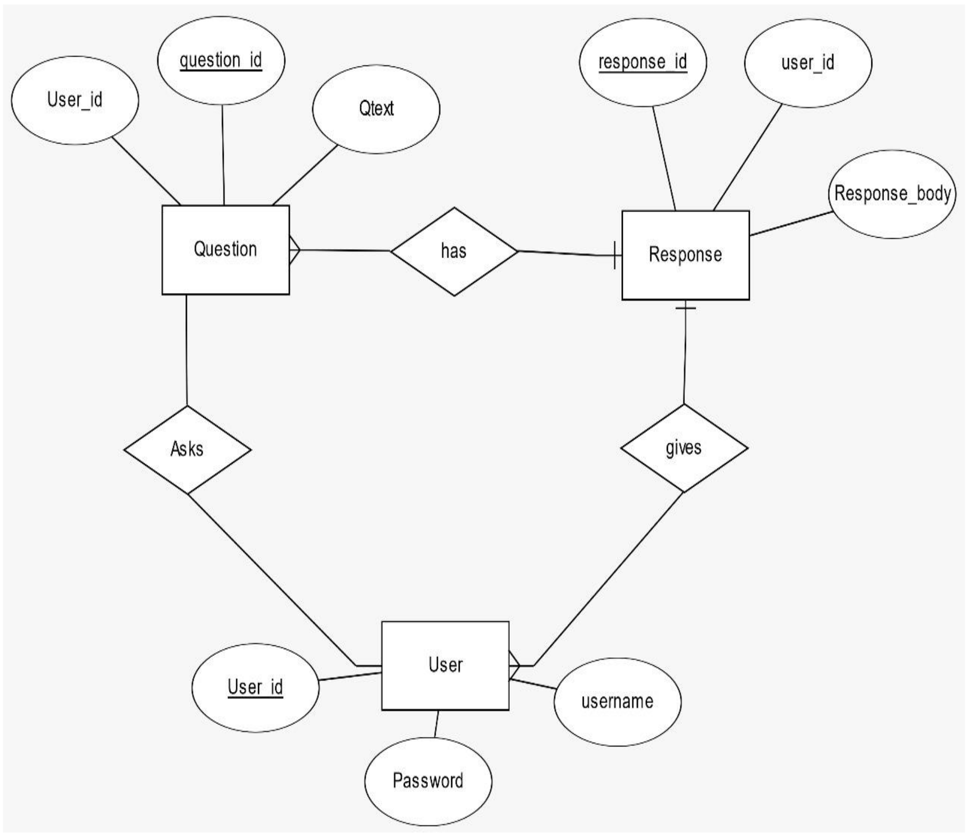
QES features questions and answers on a wide range of topics in academic and non-academic questions and answers would make the engineering community better. It’s a community-based platform which helps bypass the location barrier for Engineering students.

PROPOSED METHOD

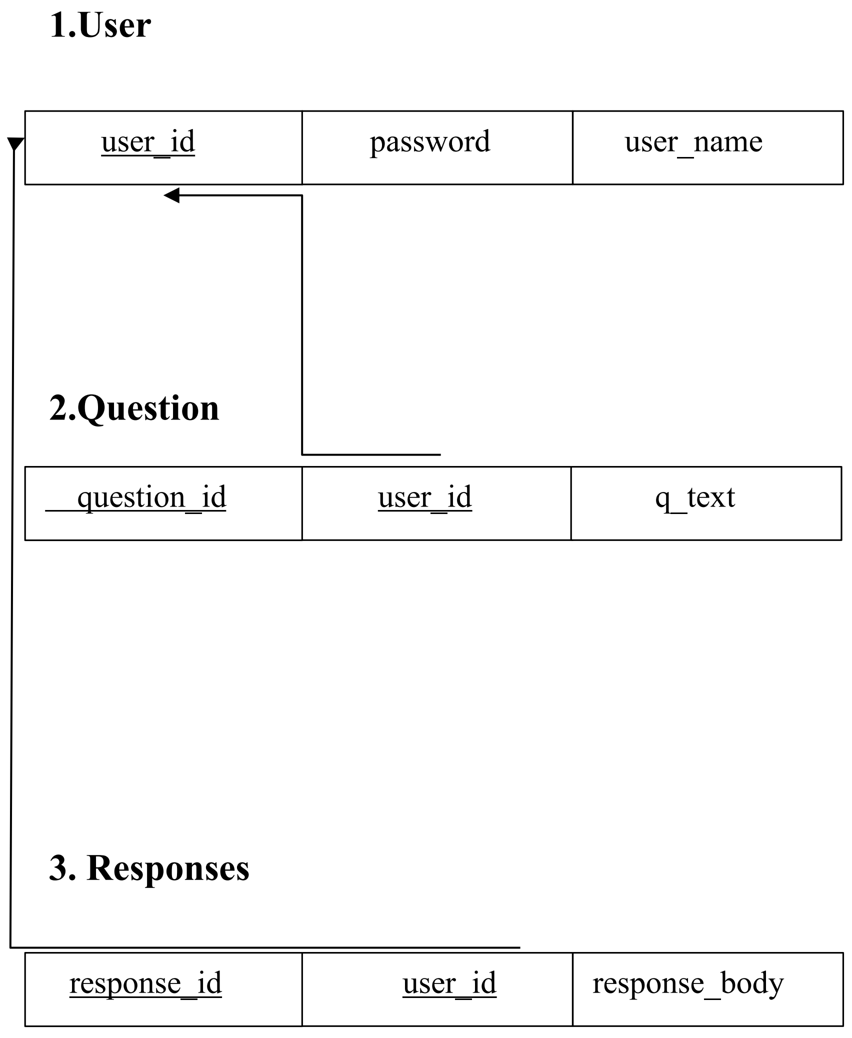
This system is to provide user comfortable environment for users to add or ask questions. Also, for other users to respond or answer the question or to add the comment.

In this, the dashboard will be containing the number of questions asked by the engineering students. Other students can register if he is a new user. After registering they can answer the question based on their knowledge. Other students or the users can upvote or downvote the answers.

ER DIAGRAM


User_id
question_id
Qtext
response_id
user_id
Response_body
Question
has
Response
Asks
gives
User
User_id
username
Password

SCHEMA DIAGRAM



TOOLS USED

Software Requirements

- Visual Studio Code 2019.
- Google Chrome or Microsoft Edge of latest version.
- Front End: HTML, CSS, Django
- Linux 7.1 or Windows XP/7/8/10 OS or Mac OS

Hardware Requirements

- Pentium 200-MHz computer with a minimum of 64 MB of RAM (128 MB of RAM recommended).
- Monitor with a refresh rate of at least 40Hz for a smooth GUI experience (optional).

IMPLEMENTATION

Source Code

response.html:

```
    <blockquote id="{{response.id}}">
  <div class="response-body">     {{response.body}}</div>
  <div class="updates">
    <div class="votes">
    <button class="up-vote"
onclick="handleUpVote({{response.id}})">upvote</button>
    <h6 id="upvotes">0</h6>
    <button class="down-vote"
onclick="handleDownVote({{response.id}})">downvote</button>
    <h6 id="downvotes">0</h6>
    {% if user.is_authenticated %}
      <button class="reply-button"
onclick="handleReplyButton({{response.id}})">reply</button>

    {% endif %}
    </div>
<div class="response-author"> ans by
   {{response.user.username}} at{{response.created_at}}</div>
</div>


    <blockquote id="reply-form-container-{{response.id}}" class="reply-form-
container">
      {% csrf_token %}
      <form class="reply" method="post" action="/reply" >
        {% csrf_token %}
<input type="hidden" name="question" value="{{question.id}}"/>
<input type="hidden" name="parent" value="{{response.id}}"/>

{{response_form.body}}
<button type="button"
onclick="handleCancelReply({{response.id}})"  class="reply-form-cancel-
button">cancel</button>
<input class="reply-form-submit-button" type="submit" value="Reply">
      </form>
    </blockquote>
  {% for children in response.get_responses %}
  {% include 'components/response.html' with response=children %}
  {% endfor %}


    </blockquote>
```

page.html

```
{% load static %}
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta
      name="viewport"
      content="width=device-width, initial-scale=1, shrink-to-fit=no"
    />
    <link
      rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.
css"
      integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
      crossorigin="anonymous"
    />
    <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Plaster&display=swap"
rel="stylesheet">



<link rel="stylesheet"
href="https://fonts.sandbox.google.com/css2?family=Material+Symbols+Outlined:ops
z,wght,FILL,GRAD@48,400,0,0" />    <title>QES</title>
    <link
      rel="stylesheet"
      type="text/css"
      href="{% static 'main/css/styles.css' %}?{% now 'U' %}"
```

5

```
    />
    <script src="{% static 'main/js/main.js' %}?{% now 'U' %}"></script>


    <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Roboto&display=swap"
rel="stylesheet">
  </head>
  <body>
    {% include 'navbar.html' %}

    <div class="container-fluid content" >
        {% block content %}

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



    <script
      src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
      integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
      crossorigin="anonymous"
    ></script>
    <script
      src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js"
      integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
      crossorigin="anonymous"
    ></script>
    <script
      src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js
"
      integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
      crossorigin="anonymous"
    ></script>

    <script defer src="https://use.fontawesome.com/releases/v5.0.8/js/all.js"
integrity="sha384-
SlE991lGASHoBfWbelyBPLsUlwY1GwNDJo3jSJO04KZ33K2bwfV9YBauFfnzvynJ"
crossorigin="anonymous"></script>
```

5

```
  </body>
</html>
```

homepage.html

```
   {% extends 'layout/page.html' %}
{% block content %}
<div class="question-title text-center">
    <h4> All Questions</h4>
</div>
{% for question in questions %}

<div class="card-container" ">
  <div class="card justify-content-between">
    <div class="container question1">
      <a class="qtext" href="{% url 'question' question.id %}">
          {{question.title}}</a>
         <hr style="margin: 2px;">
          <div class="body-ellipse" style="font-size: smaller; color:rgb(66, 
74, 83)">{{question.body}}</div>
          <div style="font-size: smaller; 
color:lightslategrey">{{question.created_at}}</div>
        </div>
    </div>

</div>
<hr class="card-hr">

{% endfor %}

{% endblock %}
```

login.html

```
  {% extends 'layout/page.html' %}

{% block content %}
<hr style="margin: 20px; border-width: 2px; border-color:rgb(133, 133, 133)">

<div class="page login-page">
<h1 class="title">Login</h1>
<hr style="margin: 20px; border-color:rgb(52, 114, 247)">

<form method="POST" action="" class="login-form">
  {% csrf_token %}
<div class="field-wrapper">
  <label for="{{form.username.id_for_label}}">Username:</label>
  {{form.username}}
  <span class="error">{{form.username.errors}}</span>
</div>
<div class="field-wrapper">
  <label for="{{form.password.id_for_label}}">Password:</label>
  {{form.password}}
  <span class="error">{{form.password.errors}}</span>
</div>
{% if form.non_field_errors %}
<div class="non_form_errors">
  <span class="error">{{form.non_field_errors.as_ul}}</span>
</div>
{% endif %}
<input type="submit" value="Login" class="submit-button"/>
</form>

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

navbar.html

```
  {% load static %}



<nav class="navbar navbar-expand-sm fixed-top">
  <div class="container-fluid">
    <div class="navbar-header">
      <a class="navbar-brand" href="#">
        <i class="fa-light fa-magnifying-glass"></i>
        <span class="logo">QES</span></a>
    </div>

    <ul class="nav navbar-nav">

      <li class=" nav-item active">

        <a href="{% url 'index' %}" >All Questions</a>
      </li>
      {% if user.is_authenticated %}

      <li class=" nav-item"><a href="{% url 'new-question' %}">Ask
question</a>
  </li>
  <li class=" nav-item"><a href="{% url 'logout' %}">Logout</a>
    </li>

    {% else %}
    <li class=" nav-item"><a href="{% url 'login' %}">Login</a></li>
    <li class=" nav-item"><a href="{% url 'register' %}">Register</a></li>
  {% endif %}
    </ul>
  </div>
</nav>
```

5

new-question.html

```
   {% extends 'layout/page.html' %}
{% block content %}
<div class="page new-question-page">
  <h1 class="title">Ask question</h1>


<form method="POST" action="" class="new-question-form">
  {% csrf_token %}

  <div class="field-wrapper">
    <label for="{{form.title.id_for_label}}">question</label>
    {{form.title}}
    <span class="error"> {{form.title.errors}}</span>
  </div>
  <div class="field-wrapper">
    <label for="{{form.body.id_for_label}}">Description</label>
    {{form.body}}
    <span class="error"> {{form.body.errors}}</span>
  </div>



  <input type="submit" value="submit" class="submit-button"/>



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

5

question.html

```
        {% extends 'layout/page.html' %}
{% block content %}
<div class="page question-page">
  <h1 class="question-title">{{question.title}}</h1>

<div class="question-body">{{question.body}}</div>
<span class="question-author">asked by {{question.author}}
|  {{question.created_at}}</span>
<hr .card-hr>
<div class="response-container">

    <h4 class="response-container-heading">
        Responses
    </h4>
    {% if not question.responses.all %}
    <div class="no-response-text"> No responses yet</div>
    {% else %}
      {% for response in question.get_responses %}
        {% include 'components\response.html' with response=response %}

      {% endfor %}
    {% endif %}
</div>

{% if user.is_authenticated %}

<form action="" class="response-form" method="post">
    {% csrf_token %}

    <div class="field-wrapper">
        <label for="{{response_form.body.id_for_label}}"> Your answer</label>
        {{response_form.body}}

      </div>
      <input type="submit" value="Send" class="submit-button"/>



</form>
</div>
```

5

```
{% endif %}

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

register.html

```
 {% extends 'layout/page.html' %}
{% block content %}
<div class="page register-page">
  <h1 class="title">Registration</h1>


<form method="POST" action="" class="register-form">
  {% csrf_token %}

  <div class="field-wrapper">
    <label for="{{form.email.id_for_label}}">Email</label>
    {{form.email}}
    <span class="error"> {{form.email.errors}}</span>
  </div>
  <div class="field-wrapper">
    <label for="{{form.username.id_for_label}}">Username</label>
    {{form.username}}
    <span class="error"> {{form.username.errors}}</span>
  </div>
  <div class="field-wrapper">
    <label for="{{form.password1.id_for_label}}">password1</label>
    {{form.password1}}
    <span class="error"> {{form.password1.errors}}</span>
    </div>
    <div class="field-wrapper">
        <label for="{{form.password2.id_for_label}}">password2</label>
        {{form.password2}}
        <span class="error"> {{form.password2.errors}}</span>
        </div>

{% if form.non_field_errors %}
<div class="non_form_errors">
    <span class="error">{{form.non_field_errors.as_ul}}</span>
  </div>
  {% endif %}
  <input type="submit" value="Register" class="submit-button"/>



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

form.py

```
from django.contrib.auth.forms import UserCreationForm, AuthenticationForm
from django.contrib.auth.models import User
from django import forms
from .models import Question,Response
class RegisterUserForm(UserCreationForm):
    class Meta:
        model = User
        fields = ['username', 'email', 'password1', 'password2']
        widgets = {
            'email': forms.EmailInput(attrs={
                'required': True,
                'placeholder': 'joe@example.com',
                'autofocus': True
            }),
            'username': forms.TextInput(attrs={
                'placeholder': 'joe',
            })
        }
    def __init__(self, *args, **kwargs):
        super(RegisterUserForm, self).__init__(*args, **kwargs)
        self.fields['password1'].widget.attrs = {'placeholder': 'password'}
        self.fields['password2'].widget.attrs = {'placeholder': 'confirm
password'}

class LoginForm(AuthenticationForm):
    class Meta:
        fields = '__all__'


class NewQuestionForm(forms.ModelForm):
    class Meta:
        model=Question
        fields=['title','body']
        widgets = {
            'title': forms.TextInput(attrs={
                'required': True,
                'placeholder': 'what is error 404?',
                'autofocus': True
            }),
        }

class ResponseForm(forms.ModelForm):
    class Meta:
        model=Response
        fields=['body']
        widgets = {
            'body': forms.TextInput(attrs={
                'placeholder': 'Your Answer'
            })  }
```

models.py

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


# Create your models here.
class Question(models.Model):
    author = models.ForeignKey(User, null=False, on_delete=models.CASCADE)
    title = models.CharField(max_length=200, null=False)
    body = models.TextField(null=False)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    def __str__(self):
        return str(self.title)

    def get_responses(self):
        return self.responses.filter(parent=None)

class Response(models.Model):
    user = models.ForeignKey(User, null=False, on_delete=models.CASCADE)
    question = models.ForeignKey(Question, null=False, on_delete=models.CASCADE,
related_name='responses')
```

```
    parent = models.ForeignKey('self', null=True, blank=True,
on_delete=models.CASCADE)
    body = models.TextField(null=False)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)


    def __str__(self):
        return self.body

    def get_responses(self):
        return Response.objects.filter(parent=self)
```

url.py

```
from django.urls import path
from django.conf.urls import url
from . import views

urlpatterns=[
    path('register',views.registerPage,name='register'),
    path('login',views.loginPage,name='login'),
    path('logout',views.logoutPage,name='logout'),
    path('new-question',views.newQuestionPage,name='new-question'),
    path('reply',views.replyPage,name='reply'),
    path('',views.homePage,name='index'),
    path('question/<int:id>',views.questionPage,name='question'),
    ]
```

views.py

```

from django.shortcuts import render,redirect
from django.contrib.auth import login,logout
from django.contrib.auth.decorators import login_required

from .models import Question,Response
from .forms import NewQuestionForm,RegisterUserForm, LoginForm,ResponseForm


def registerPage(request):
    form = RegisterUserForm()
    if request.method =='POST':
        try:
            form=RegisterUserForm(request.POST)
            if form.is_valid():
                user=form.save()
                login(request,user)
                return redirect('index')
        except Exception as e:
            print(e)
            raise
    context = {
         'form': form
    }
    return render(request, 'register.html', context)

def loginPage(request):
    form = LoginForm()
    if request.method == 'POST':
        try:
            form = LoginForm(data=request.POST)
            if form.is_valid():
                user = form.get_user()
                login(request, user)
                return redirect('index')
        except Exception as e:
            print(e)
            raise

    context = {'form': form}

    return render(request,'login.html',context)

@login_required(login_url='register')
def logoutPage(request):
    logout(request)
    return redirect('login')

@login_required(login_url='register')
def newQuestionPage(request):
    form=NewQuestionForm()
    if request.method == 'POST':
        try:
            form = NewQuestionForm(data=request.POST)
            if form.is_valid():
                question=form.save(commit=False)
                print(request)
                question.author=request.user
                question.save()
                return redirect('index')
        except Exception as e:
            print(e)
            raise

    context={
        'form':form,
    }
    return render(request,'new-question.html',context)


def homePage(request):
    questions=Question.objects.all().order_by('-created_at')
    context={
        'questions':questions
    }
    return render(request,'homepage.html',context)


def questionPage(request, id):
    question=Question.objects.get(id=id)
    response_form=ResponseForm()

    if request.method =='POST':
        try:
            response_form=ResponseForm(data=request.POST)
            if response_form.is_valid():
                response= response_form.save(commit=False)
                response.user=request.user
                response.question=Question(id=id)
                response.save()
                return
redirect('/question/'+str(id)+'#'+str(response.id))

        except Exception as e:
            print(e)
            raise
    context={
        'question':question,
        'response_form':response_form,

    }
    return render(request,"question.html",context)

@login_required(login_url='register')
def replyPage(request):
    if request.method == 'POST':
        try:
            form = ResponseForm(request.POST)
            if form.is_valid():
                question_id = request.POST.get('question')
                parent_id = request.POST.get('parent')
                reply = form.save(commit=False)
                reply.user = request.user
                reply.question = Question(id=question_id)
                reply.parent = Response(id=parent_id)
                reply.save()
                return redirect('/question/'+str(question_id)+'#'+str(reply.id))
        except Exception as e:
            print(e)
            raise

    return redirect('index')
```

urls.py

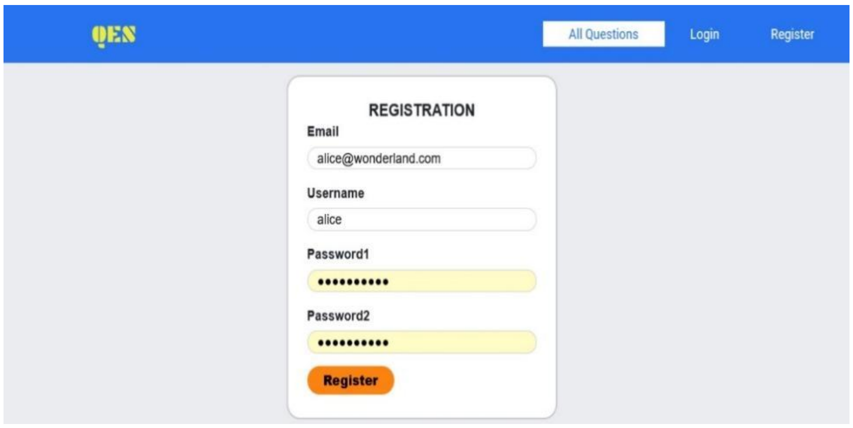
```
 """mywebsite URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/3.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path,include
```

5

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('main.urls'))
]
```

SNAPSHOTS OF IMPLEMENTATION

1.Registration form



QES All Questions Login Register

REGISTRATION

Email: alice@wonderland.com

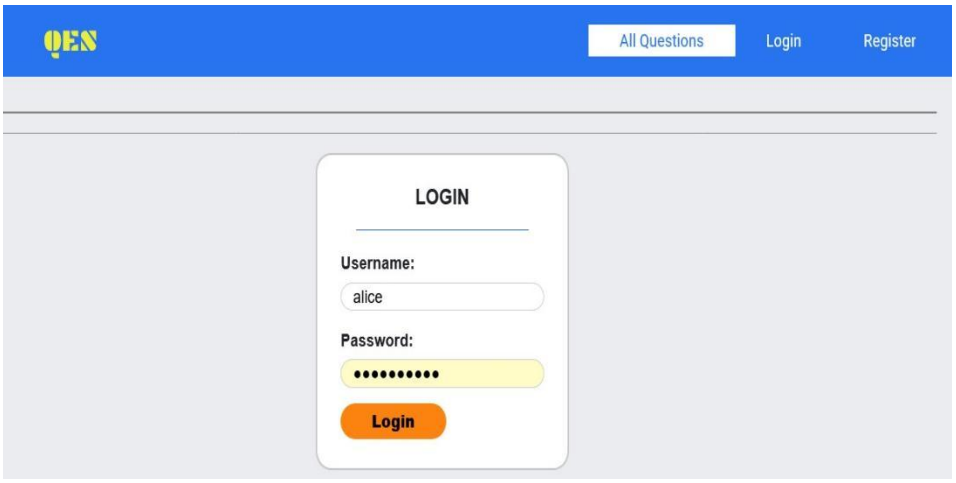
Username: alice

Password1: ••••••••••

Password2: ••••••••••

Register

2.Login form



QES All Questions Login Register

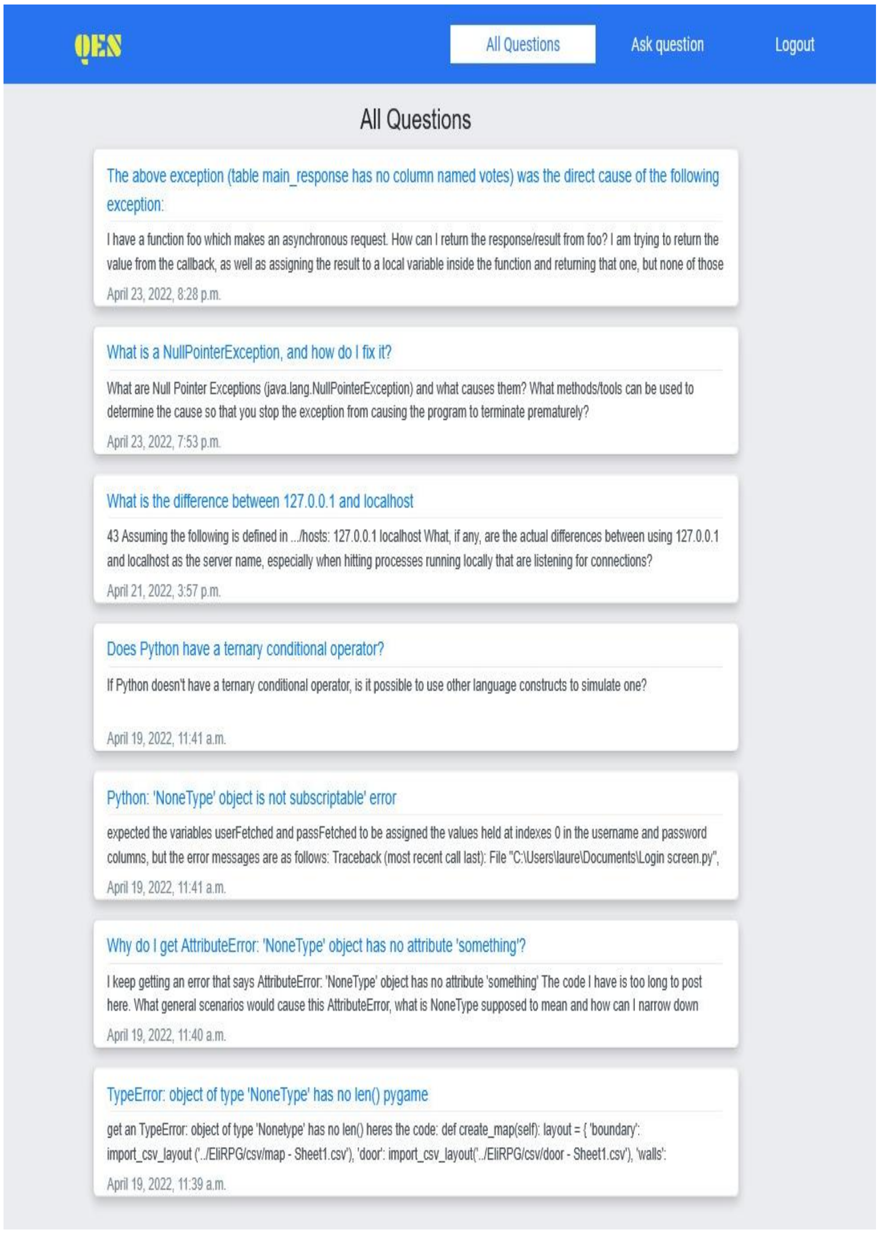
LOGIN

Username: alice

Password: ••••••••••

Login

3.All questions



All Questions

The above exception (table main_response has no column named votes) was the direct cause of the following exception:

I have a function foo which makes an asynchronous request. How can I return the response/result from foo? I am trying to return the value from the callback, as well as assigning the result to a local variable inside the function and returning that one, but none of those

April 23, 2022, 8:28 p.m.

What is a NullPointerException, and how do I fix it?

What are Null Pointer Exceptions (java.lang.NullPointerException) and what causes them? What methods/tools can be used to determine the cause so that you stop the exception from causing the program to terminate prematurely?

April 23, 2022, 7:53 p.m.

What is the difference between 127.0.0.1 and localhost

43 Assuming the following is defined in .../hosts: 127.0.0.1 localhost What, if any, are the actual differences between using 127.0.0.1 and localhost as the server name, especially when hitting processes running locally that are listening for connections?

April 21, 2022, 3:57 p.m.

Does Python have a ternary conditional operator?

If Python doesn't have a ternary conditional operator, is it possible to use other language constructs to simulate one?

April 19, 2022, 11:41 a.m.

Python: 'NoneType' object is not subscriptable' error

expected the variables userFetched and passFetched to be assigned the values held at indexes 0 in the username and password columns, but the error messages are as follows: Traceback (most recent call last): File "C:\Users\laure\Documents\Login screen.py",

April 19, 2022, 11:41 a.m.

Why do I get AttributeError: 'NoneType' object has no attribute 'something'?

I keep getting an error that says AttributeError: 'NoneType' object has no attribute 'something' The code I have is too long to post here. What general scenarios would cause this AttributeError, what is NoneType supposed to mean and how can I narrow down

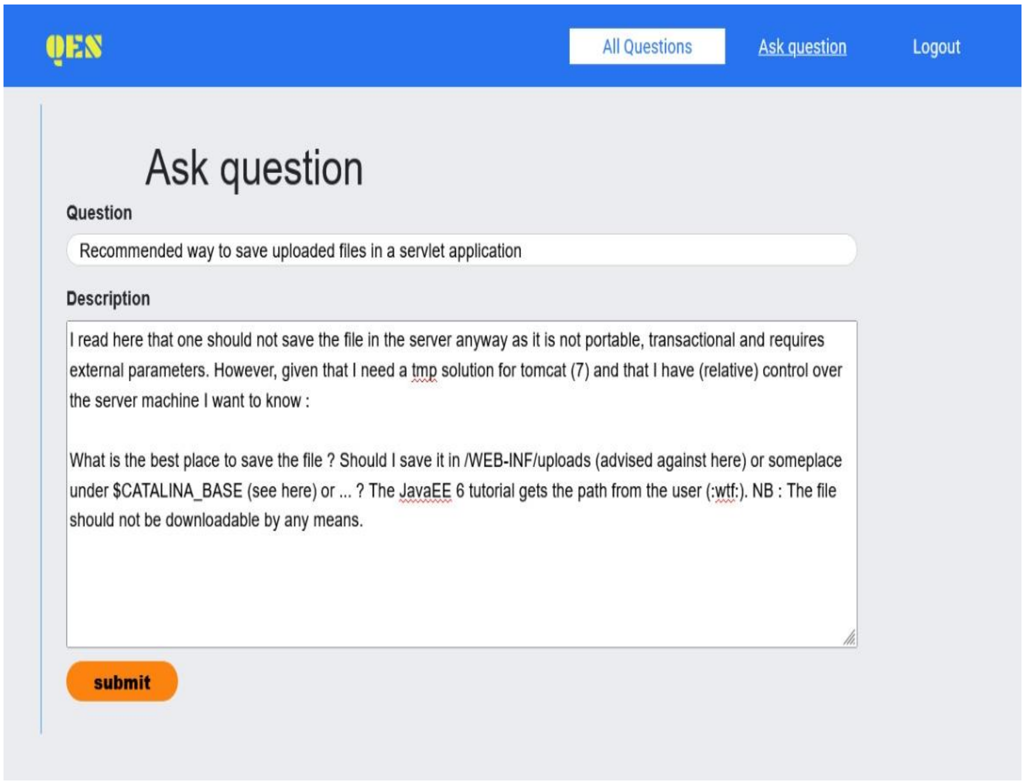
April 19, 2022, 11:40 a.m.

TypeError: object of type 'NoneType' has no len() pygame

get an TypeError: object of type 'Nonetype' has no len() heres the code: def create_map(self): layout = { 'boundary': import_csv_layout ('../EliRPG/csv/map - Sheet1.csv'), 'door': import_csv_layout('../EliRPG/csv/door - Sheet1.csv'), 'walls':

April 19, 2022, 11:39 a.m.

4.Ask question



Ask question

Question

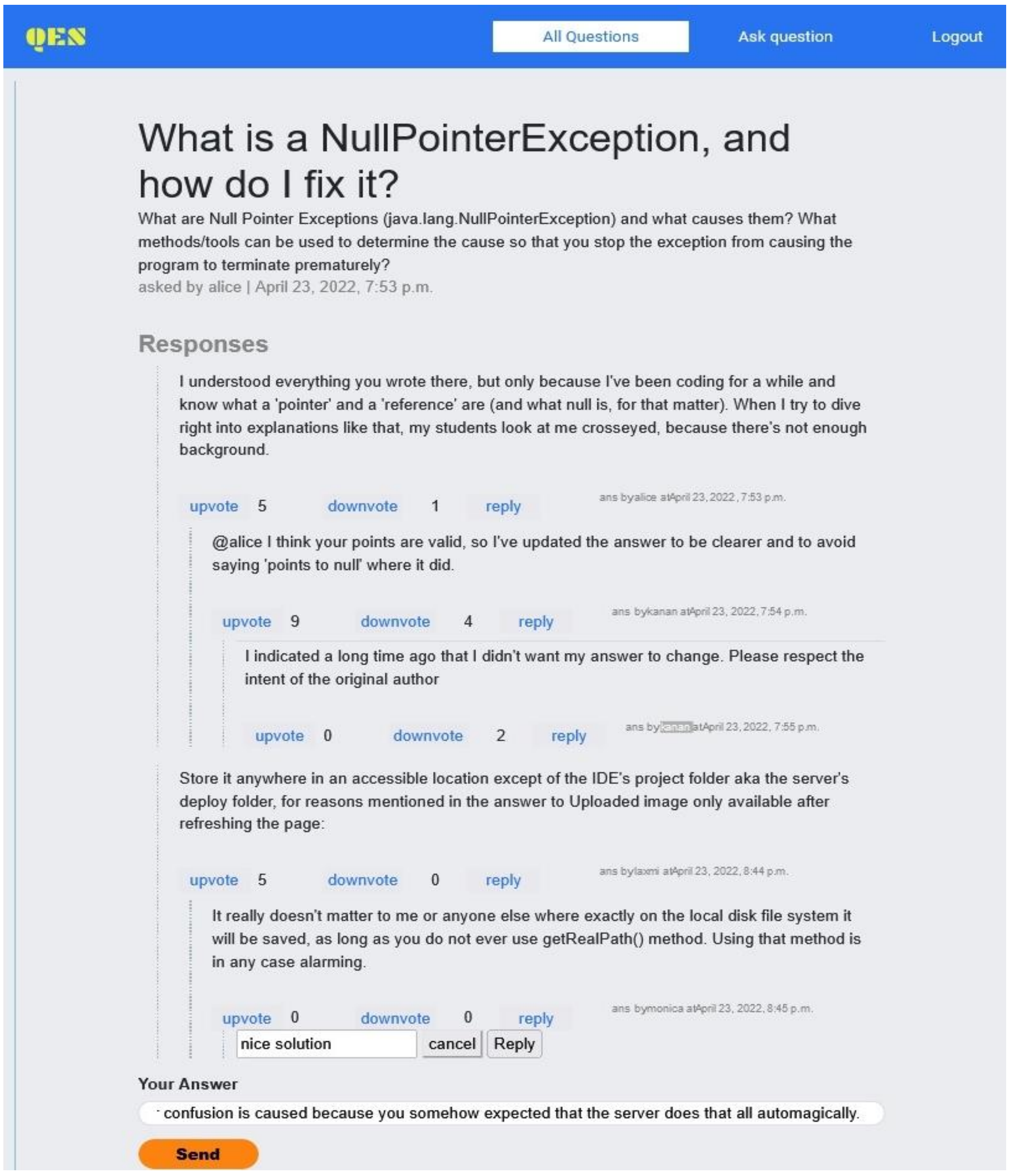
Recommended way to save uploaded files in a servlet application

Description

I read here that one should not save the file in the server anyway as it is not portable, transactional and requires external parameters. However, given that I need a tmp solution for tomcat (7) and that I have (relative) control over the server machine I want to know :

What is the best place to save the file ? Should I save it in /WEB-INF/uploads (advised against here) or someplace under $CATALINA_BASE (see here) or ... ? The JavaEE 6 tutorial gets the path from the user (:wtf:). NB : The file should not be downloadable by any means.

5.Responses



What is a NullPointerException, and how do I fix it?

What are Null Pointer Exceptions (java.lang.NullPointerException) and what causes them? What methods/tools can be used to determine the cause so that you stop the exception from causing the program to terminate prematurely?

asked by alice | April 23, 2022, 7:53 p.m.

Responses

I understood everything you wrote there, but only because I've been coding for a while and know what a 'pointer' and a 'reference' are (and what null is, for that matter). When I try to dive right into explanations like that, my students look at me crosseyed, because there's not enough background.

upvote 5 downvote 1 reply ans byalice atApril 23, 2022 ,7:53 p.m.

@alice I think your points are valid, so I've updated the answer to be clearer and to avoid saying 'points to null' where it did.

upvote 9 downvote 4 reply ans bykanan atApril 23, 2022, 7:54 p.m.

I indicated a long time ago that I didn't want my answer to change. Please respect the intent of the original author

upvote 0 downvote 2 reply ans bykanan atApril 23, 2022, 7:55 p.m.

Store it anywhere in an accessible location except of the IDE's project folder aka the server's deploy folder, for reasons mentioned in the answer to Uploaded image only available after refreshing the page:

upvote 5 downvote 0 reply ans bylaxmi atApril 23, 2022, 8:44 p.m.

It really doesn't matter to me or anyone else where exactly on the local disk file system it will be saved, as long as you do not ever use getRealPath() method. Using that method is in any case alarming.

upvote 0 downvote 0 reply ans bymonica atApril 23, 2022, 8:45 p.m.

nice solution cancel Reply

Your Answer

confusion is caused because you somehow expected that the server does that all automagically.

Send

BIBLIOGRAPHY

- https://docs.djangoproject.com/en/4.0/topics/
- https://docs.djangoproject.com/en/4.0/ref/databases/
- https://getbootstrap.com/docs/4.0/layout/grid/
- https://docs.djangoproject.com/en/4.0/intro/tutorial01/
- https://www.geeksforgeeks.org/college-management-system-using-django-python-project/
- https://www.geeksforgeeks.org/django-basics/

ABOUT MY TEAM

My team members are:

Shreyanka Kamadhen,Shruti Ballurgi,Sangamesh,Mahaning Nilajagi.

Role played by team:

Shreyanka Kamadhen: Frontend, Backend

Shruti Ballurgi: Frontend

Sangamesh: Login and Registration form frontend

Mahaning Nilajagi: Database and Report