“COLLEGE FORM”

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

RGUKT

Rajiv Gandhi University of Knowledge Technologies

R.K.VALLEY

Submitted by

C.Nagamani -- R170538

A.Vanaja -- R170550

Under the Esteemed guidance of
Mr. Satya Nandaram N
RGUKT RK Valley.

DECLARATION

We hereby declare that the report of the B.Tech Minor Project Work entitled " **COLLEGE FORM** " which is being submitted to Rajiv Gandhi University of Knowledge Technologies,RK Valley,in partial fulfillment of the requirements for the award of Degree of Bachelor of Technology in Comput Science and Engineering, is a bonafide report of the work carried out by us. material contained in this report has not been submitted to any university or institution for award of any degree.

A.Vanaja – R170550

C.Nagamani – R170538

Dept. Of Computer Science and Engineering.

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

RGUKT

(A.P.Government Act 18 of 2008)
RGUKT, RK VALLEY
Department of Computer Science and Engineering

CERTIFICATE FOR PROJECT COMPLETION

This is certify that the project entitled " COLLEGE FORM " submitted by **C.Nagamani(R170538) , A.Vanaja(R170550)** under our supervision Partial fulfillment for the degree Bachelor of Technology in Computer Science And Engineering during the academic semester - 2 2021-2022 at RGUKT, RK - VALLEY .To the best of my knowledge , the results embodied in this dissert- -ation work have not been submitted to any University or Institute for the award of any degree or diploma.

Project Internal Guide	**Head of the Department**
Mr.N.Satya Nandaram	Mr.P.Harinath
Assistant Professor	HOD Of CSE
RGUKT, RK Valley	RGUKT, RK Valley

Abstract

A Question and Answer Software is a software that focuses on answering the questions one might put up on an online platform. These are usually implemented by large organizations which aim on implementing platform where users can clear doubts about their respective fields. It varies from smallscale to large scale or from topiic to to topic.Many of these platforms may restrict access to either their employees or make it a public one material.The disadvantages of the existing systems are less security, searching efficiency is less, chances of faking answer. One may access these sites / Applications from anothers system in case they are not public or access the ones which are and post answers which aren't relevant. Sometimes people even lie on such platforms and there might not be any checker to cross - analyze these answers. Now one might also want similar answers to their question which may resolve their query beforehand.

Index

1. Abstraction
2. SRS Document
   Introduction
   2.1 Purpose
   2.2 Intended Audience
   2.3 Product Vision
   2.4 Technologies
3. Requirement Specification
4. html
5. CSS
6. Boot Strap
7. Python Django
8. JQuery
9. SQlite
10. DESIGN AND ANALYSIS
    10.1 Use Case Diagram
    10.2. ER Diagram

11. Agile Development

11.1 Agile

11.2 Agile Methodology

12. How to develop a Project using Agile Methodology

13.Source Code

14. EVALUATION

14.1. Image References

14.2.Home page

14.3.Login Page

14.4.Register Page

14.5.Forum page

14.6.Updates Page

14.7.Admin Page

14.8.Answers Report Page

14.9.Answers Page

14.10.Category Page

14.11.Flagged Answer Page

15. References

SRS DOCUMENT

Introduction:

This document has the requirements of Answering questions and clearing doubts of students.It is an Online network whcih would attract people wanting answers to their respective questions.

1.1:Purpose

The purpose of this document is to gather the requirements that are needed for implementing Th “estudent form” . It also focuses on various key features, the product , product vision and scope, product overview.The main purpose of Student form is to provide a platform to the students to clear their doubts.

1.2:Intended Audience:

The intended audience will be the users who can access the platform to get answers from commoners and experts related to the students doubts.

Users:

1. Students
2. Admin

Product Vision:

Vision Statement:

The product vision is to develop a software hepls people to gain more information and become Knowledgable about important topics.It can be extremely for people of all age groups.

Technologies:

- ➢ HTML,CSS
- ➢ Jquery
- ➢ Bootstrap
- ➢ Django
- ➢ sqlite

REQUIREMENT SPECIFICATION

Hardware Configuration :

Client Side:

Ram	4GB
Harddisk	487.0 GB
Processor	AMD® A9-9420 radeon r5, 5 compute cores 2c+3g × 2

Server Side :

Ram	4GB
Harddisk	487.0 GB
Processor	AMD® A9-9420 radeon r5, 5 compute cores 2c+3g × 2

Software Configuration :

Front end	HTML,CSS,Jquery,Bootsrap
Server side language	Python,Django
Database server	Sqlite
Web browser	Firefox,Google chrome
Operating system	Ubuntu

Functional Requirements :

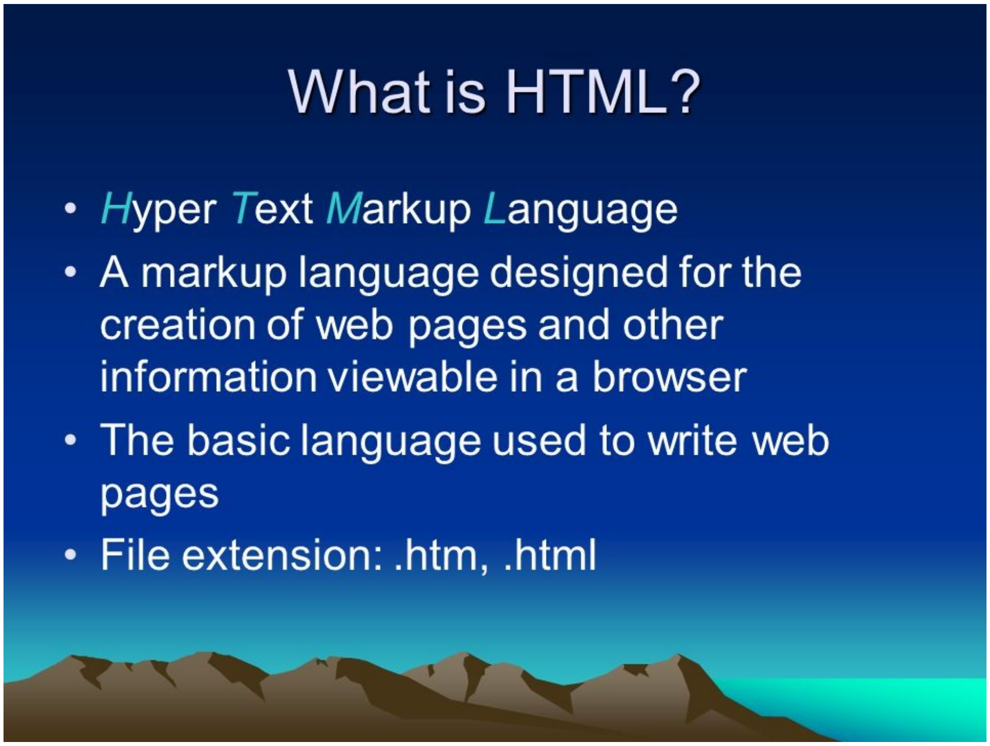
- These requirements include product perspective, features, user characteristics, assumption and dependencies, domain requirements, user requirements.
- They help analyze the reason why the product was created, what all features it has and requires, like private questions, similarity search, rate answers etc.
- The requirements of both the user and domain are analyzed which shows that normal user can answer , post, ask questions and an expert can answer both private and unanswered questions whereas the admin can delete both the users and observe the system they have whole control.

Non-Functional Requirements :

- They help analyze the reason why the product was created, what all features it has and requires, like private questions, similarity search, rate answers etc.
- The requirements of both the user and domain are analyzed which shows that normal user can answer, post, ask questions and an expert can answer both private and unanswered questions whereas the admin can delete both the users and observe the entire system as they have whole control.

HTML :

HTML (HyperText Markup Language) is the code that is used to structure a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables.

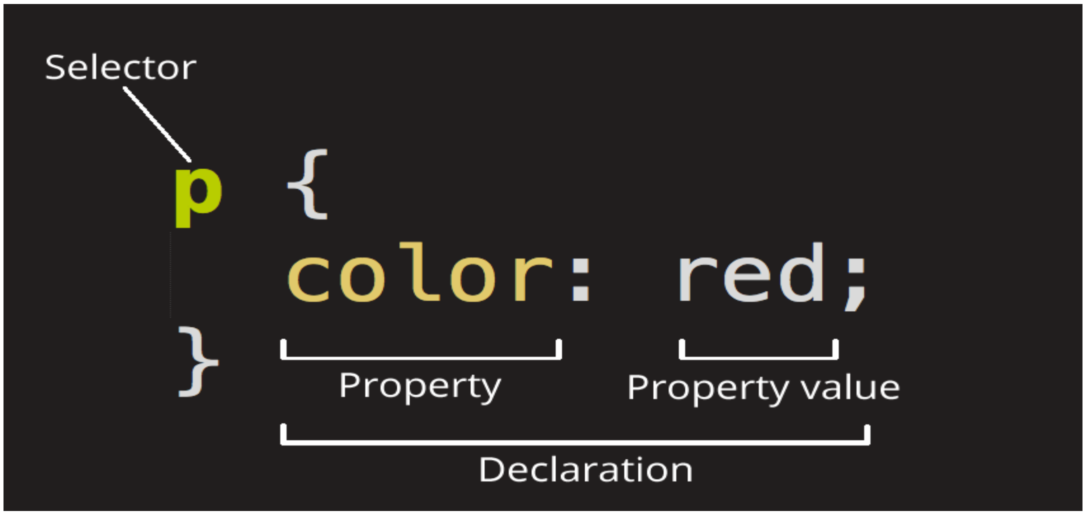


What is HTML?

- *H*yper *T*ext *M*arkup *L*anguage
- A markup language designed for the creation of web pages and other information viewable in a browser
- The basic language used to write web pages
- File extension: .htm, .html

CSS :

CSS (Cascading Style Sheets) is used to style and layout web pages for example, to alter the font, color, size, and spacing of your content, split it into multiple columns , or add animations other decorative features



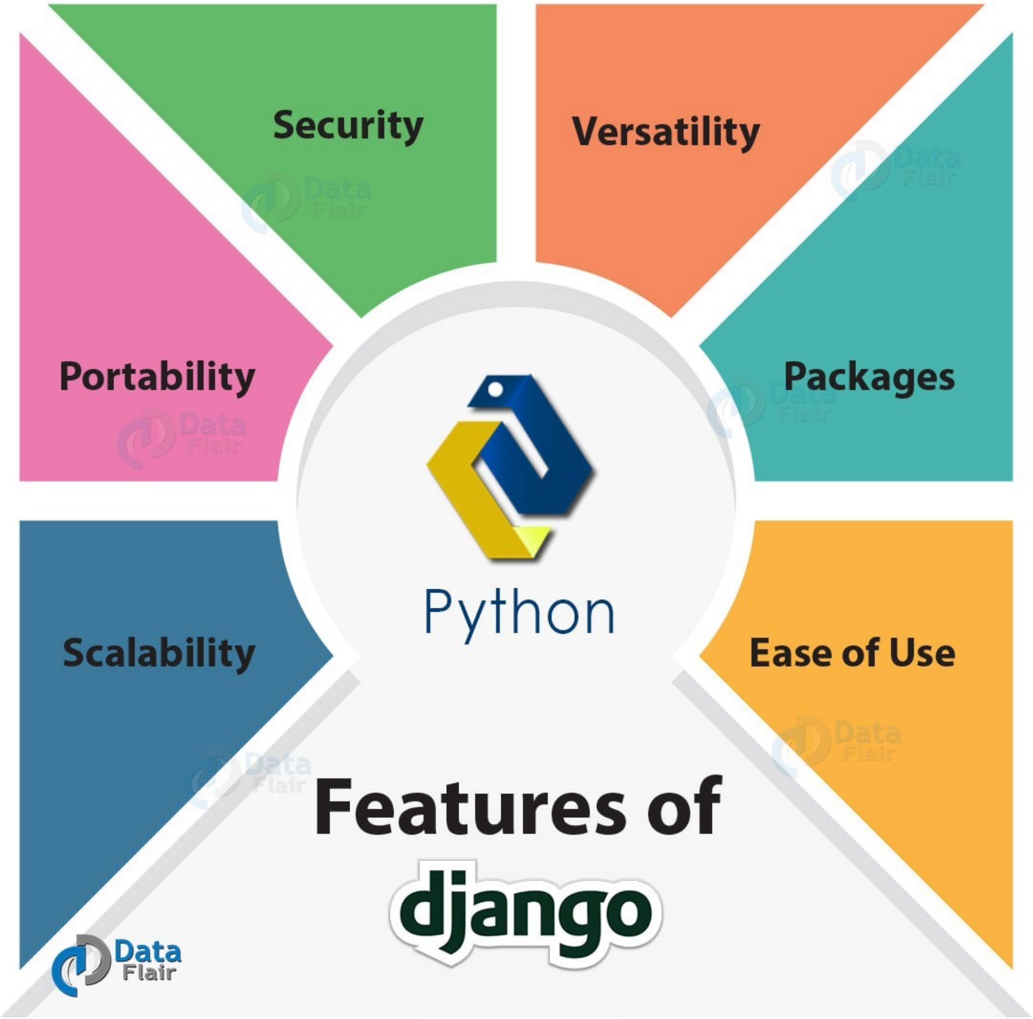
```
p {
    color: red;
}
```

Bootstrap :

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

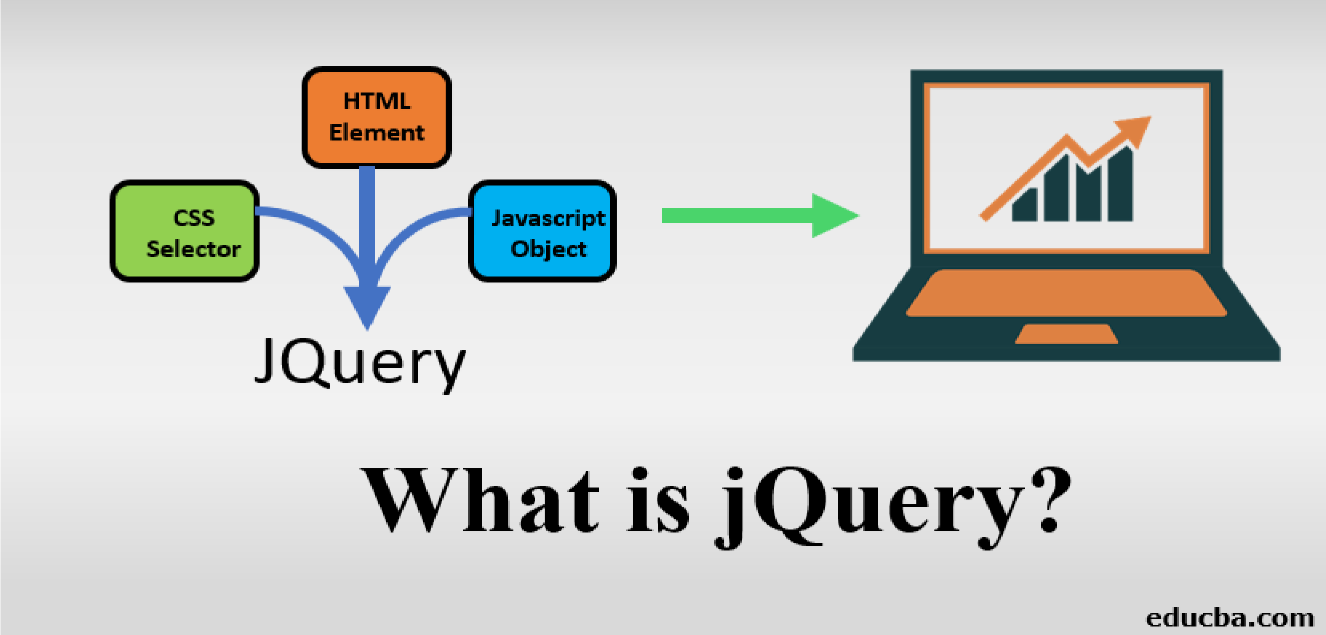
Django :

Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel.



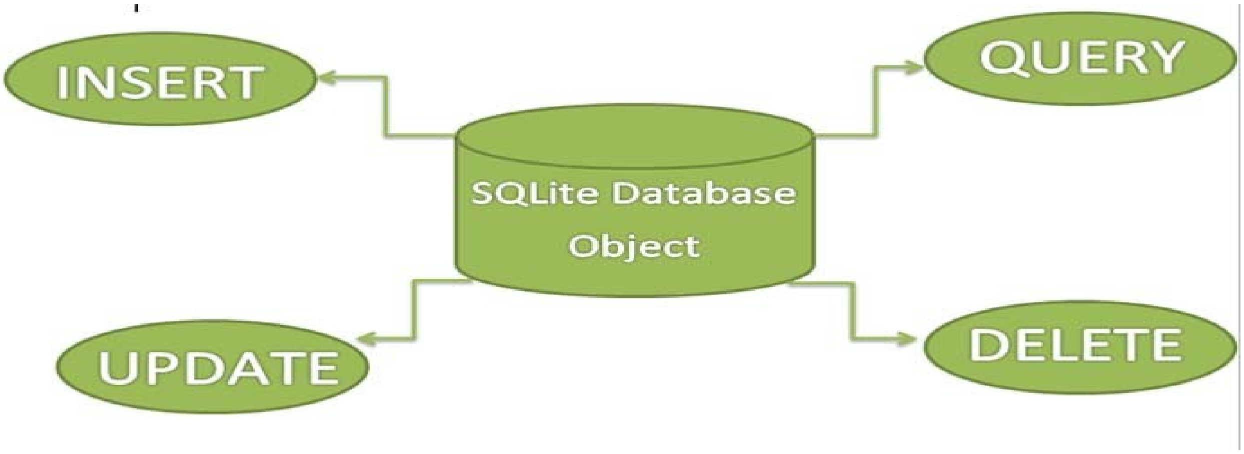
Jquery:

.jQuery is a lightweight, "write less, do more", JavaScript library. The purpose of jQuery is to make it much easier to use JavaScript on your website. jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, wraps them into methods that you can call with a single line of code.



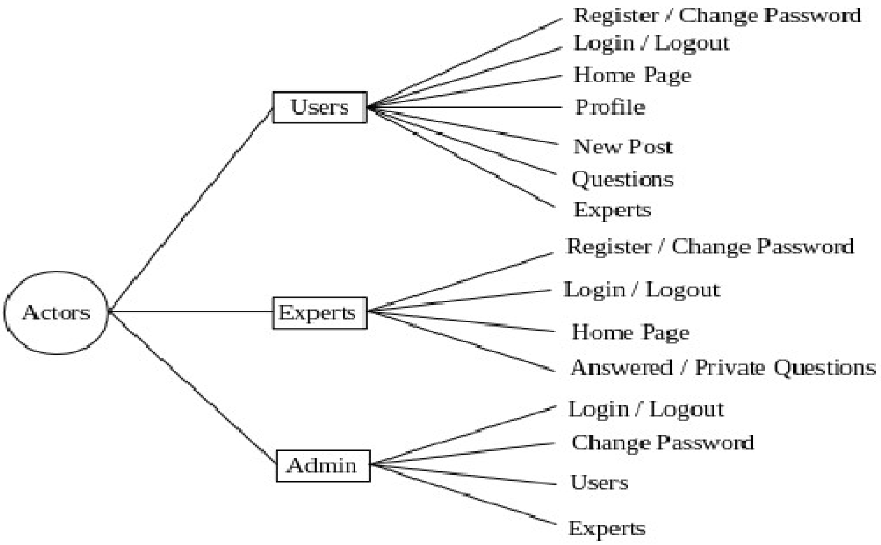
SQlite :

SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine. SQLite is the most used database engine in the world. SQLite is built into all mobile phones and most computers and comes bundled inside countless other applications that people use every day.

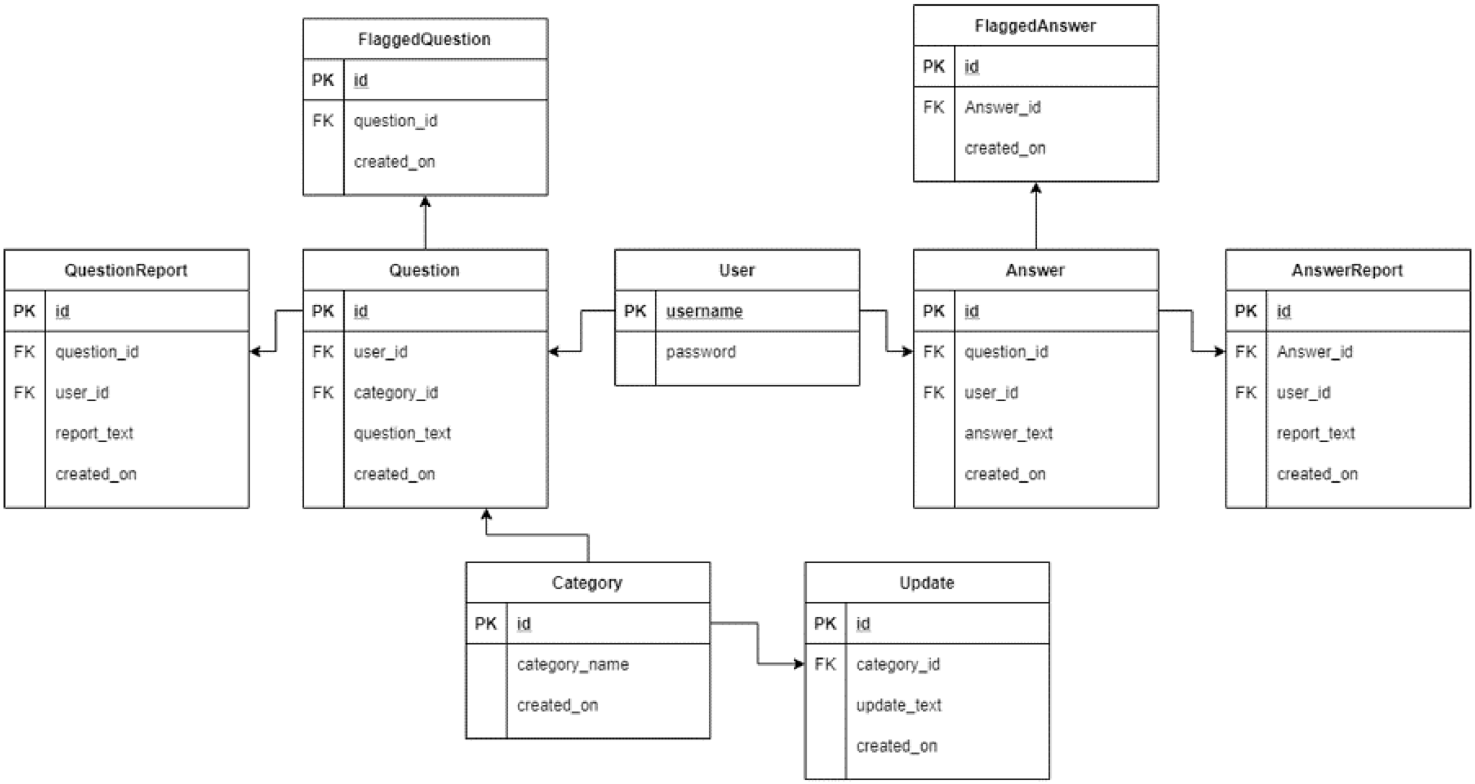


DESIGN AND ANALYSIS

Use case Diagram:



ER Diagram:



FlaggedQuestion	
PK	id
FK	question_id
	created_on

FlaggedAnswer	
PK	id
FK	Answer_id
	created_on

QuestionReport	
PK	id
FK	question_id
FK	user_id
	report_text
	created_on

Question	
PK	id
FK	user_id
FK	category_id
	question_text
	created_on

User	
PK	username
	password

Answer	
PK	id
FK	question_id
FK	user_id
	answer_text
	created_on

AnswerReport	
PK	id
FK	Answer_id
FK	user_id
	report_text
	created_on

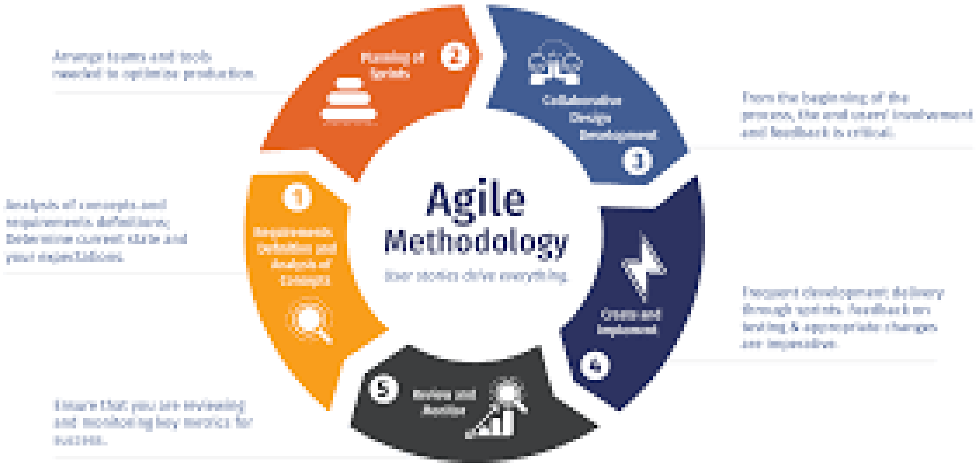
Category	
PK	id
	category_name
	created_on

Update	
PK	id
FK	category_id
	update_text
	created_on

Agile Development

Agile

The Agile methodology is a way to manage a project by breaking it up into several phases. It involves constant collaboration with stakeholders and continuous improvement at every stage.Once the work begins, teams cycle through process of **planning, executing, and evaluating**. Continuous collaboration is vital , both Team members and project stakeholders.


Arrange teams and tools needed to optimize production.
Planning of Sprints
2
Collaborative Design Development
3
From the beginning of the process, the end users' involvement and feedback is critical.
Analysis of concepts and requirements definitions; Determine current state and your expectations.
1
Requirements Definition and Analysis of Concepts
Agile
Methodology
User stories drive everything.
Create and Implement
4
Frequent development delivery through sprints. Feedback on testing & appropriate changes are imperative.
5
Review and Monitor
Ensure that you are reviewing and monitoring key metrics for success.

Agile methodology:

It’s a process for managing a project that involves constant collaboration and working in iterations. Today, the word Agile can refer to these values and the frameworks for implementing them, including Scrum, Kanban, Extreme Programming (XP), and Adaptive Project Framework (APF).

Agile

A project management methodology characterized by building products using short cycles of work that allow for rapid production and constant revision.

Kanban

A visual approach to project management where teams create physical representations of their tasks, often using sticky notes on whiteboards (or online apps). Tasks are moved through predetermined stages to track progress and identify common roadblocks.

Scrum

A PM methodology in which a small team is led by a Scrum master, whose main job is to clear away all obstacles to completing work. Work is done in short cycles called sprints, but the team meets daily to discuss current tasks and roadblocks.

Adaptive Project Framework (APF)

A project management methodology that grew from the idea that most IT projects can't be managed using traditional PM methods. Work is done in stages and evaluated after each one.

Extreme Project Management (XPM)

A PM methodology where the project plan, budget, and final deliverable can be changed to fit evolving needs, no matter how far along the project is.

How to Develop a project using Agile methodology

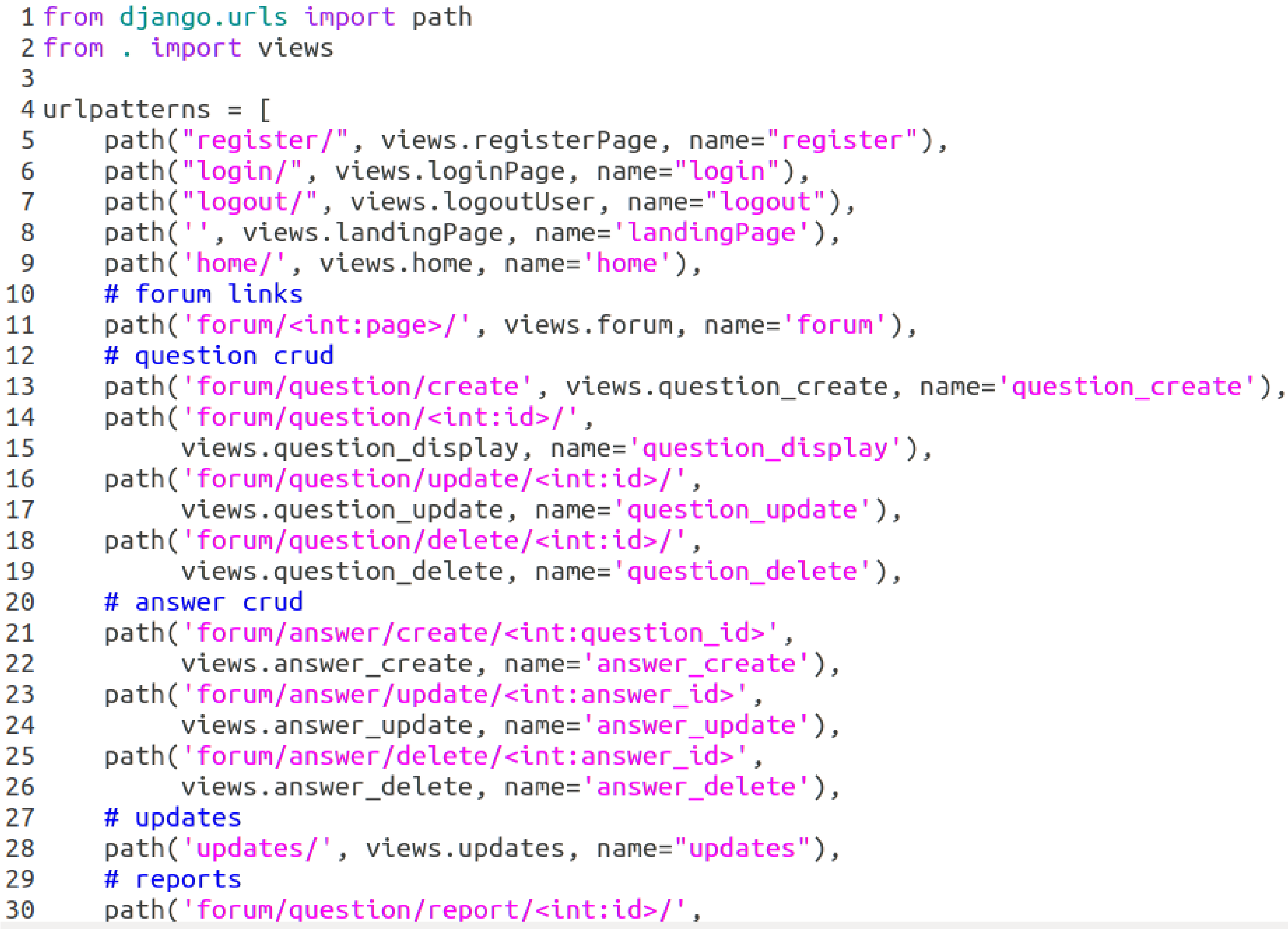
Agile Development is a Continuous Integration(CI) from Requirements gathering to testing the code..

We start the project Development with Requirements analysis and Gathering. In this we collect the data from the project description and draw the UML diagrams like **ER** diagram for database tables and **Use Case** diagram for implementation functionalities.

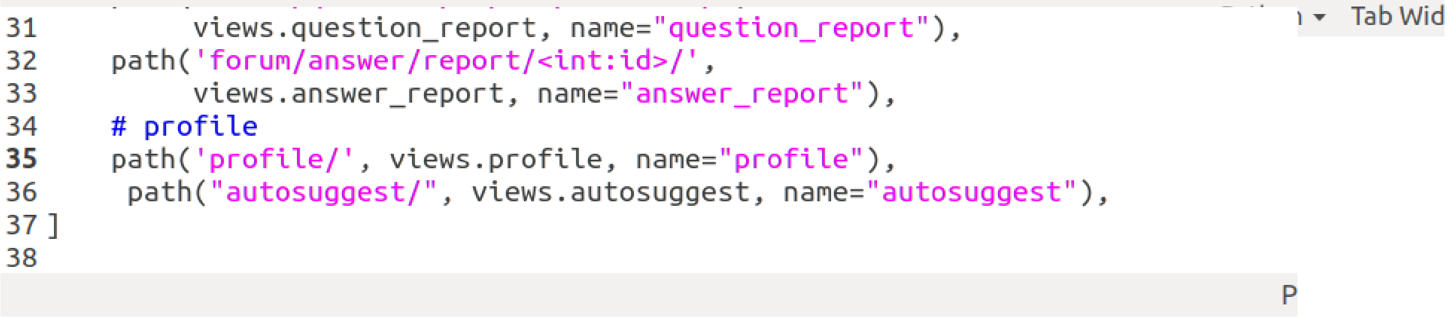
After we create a short stories like Login, Signup, Homepage Design, Database Creation From the requirements file.

SOURCE CODE

url.py

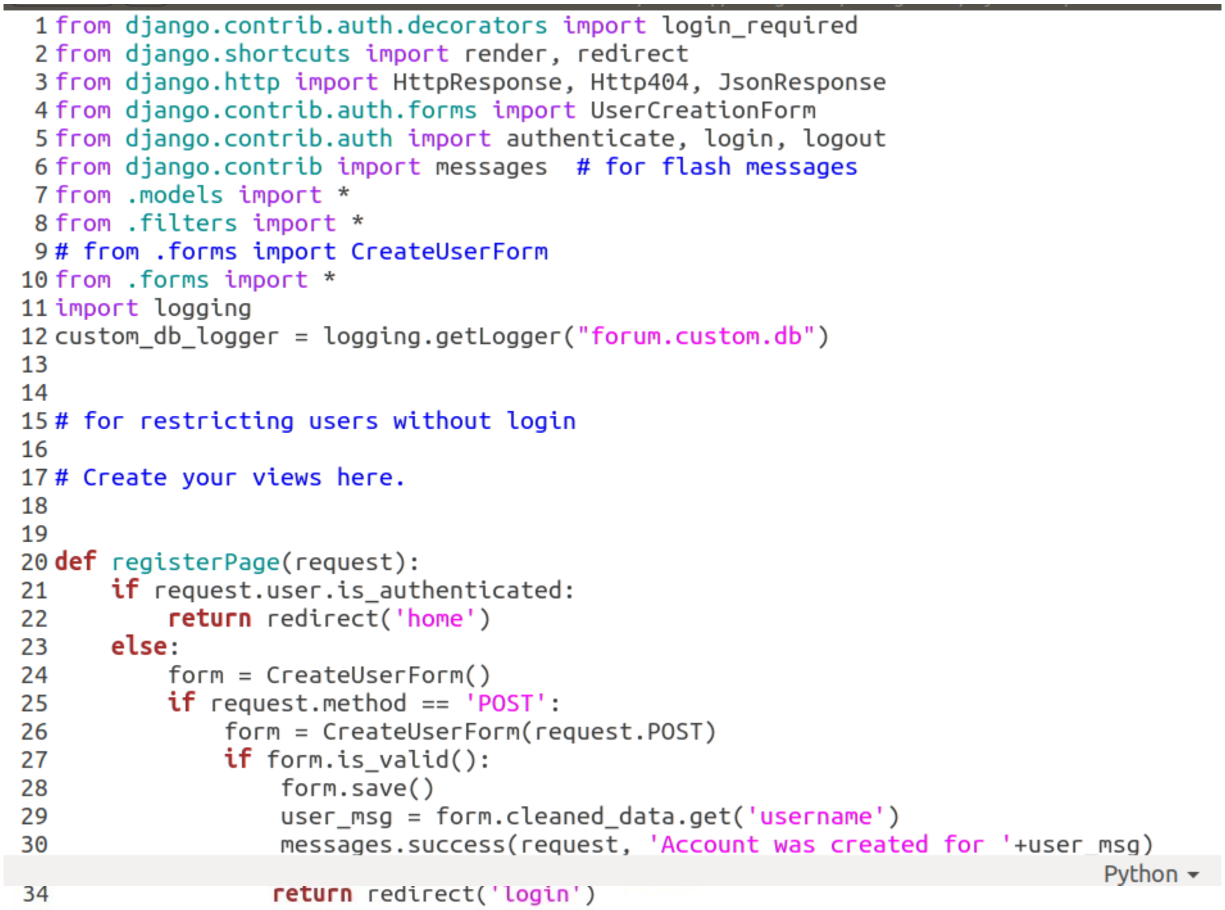


```
from django.urls import path
from . import views

urlpatterns = [
    path("register/", views.registerPage, name="register"),
    path("login/", views.loginPage, name="login"),
    path("logout/", views.logoutUser, name="logout"),
    path('', views.landingPage, name='landingPage'),
    path('home/', views.home, name='home'),
    # forum links
    path('forum/<int:page>/', views.forum, name='forum'),
    # question crud
    path('forum/question/create', views.question_create, name='question_create'),
    path('forum/question/<int:id>/',
         views.question_display, name='question_display'),
    path('forum/question/update/<int:id>/',
         views.question_update, name='question_update'),
    path('forum/question/delete/<int:id>/',
         views.question_delete, name='question_delete'),
    # answer crud
    path('forum/answer/create/<int:question_id>',
         views.answer_create, name='answer_create'),
    path('forum/answer/update/<int:answer_id>',
         views.answer_update, name='answer_update'),
    path('forum/answer/delete/<int:answer_id>',
         views.answer_delete, name='answer_delete'),
    # updates
    path('updates/', views.updates, name="updates"),
    # reports
    path('forum/question/report/<int:id>/',
```

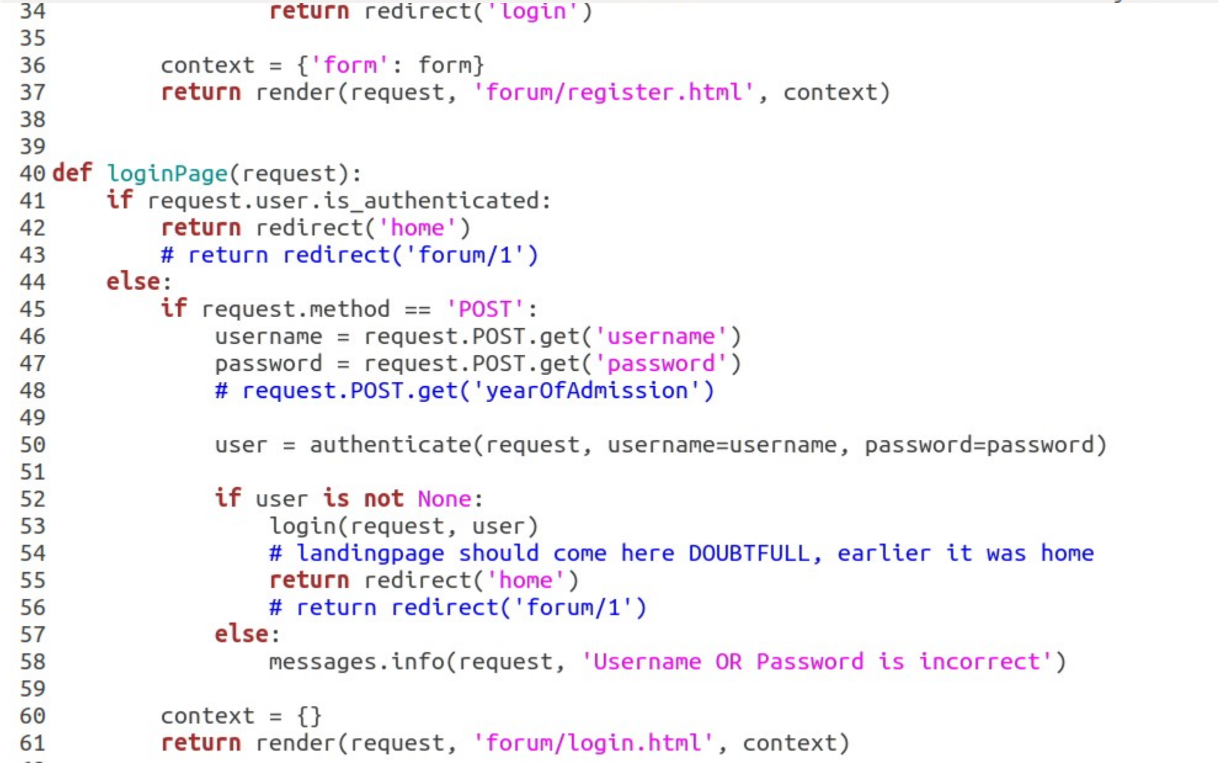


```
         views.question_report, name="question_report"),
    path('forum/answer/report/<int:id>/',
         views.answer_report, name="answer_report"),
    # profile
    path('profile/', views.profile, name="profile"),
     path("autosuggest/", views.autosuggest, name="autosuggest"),
]
```

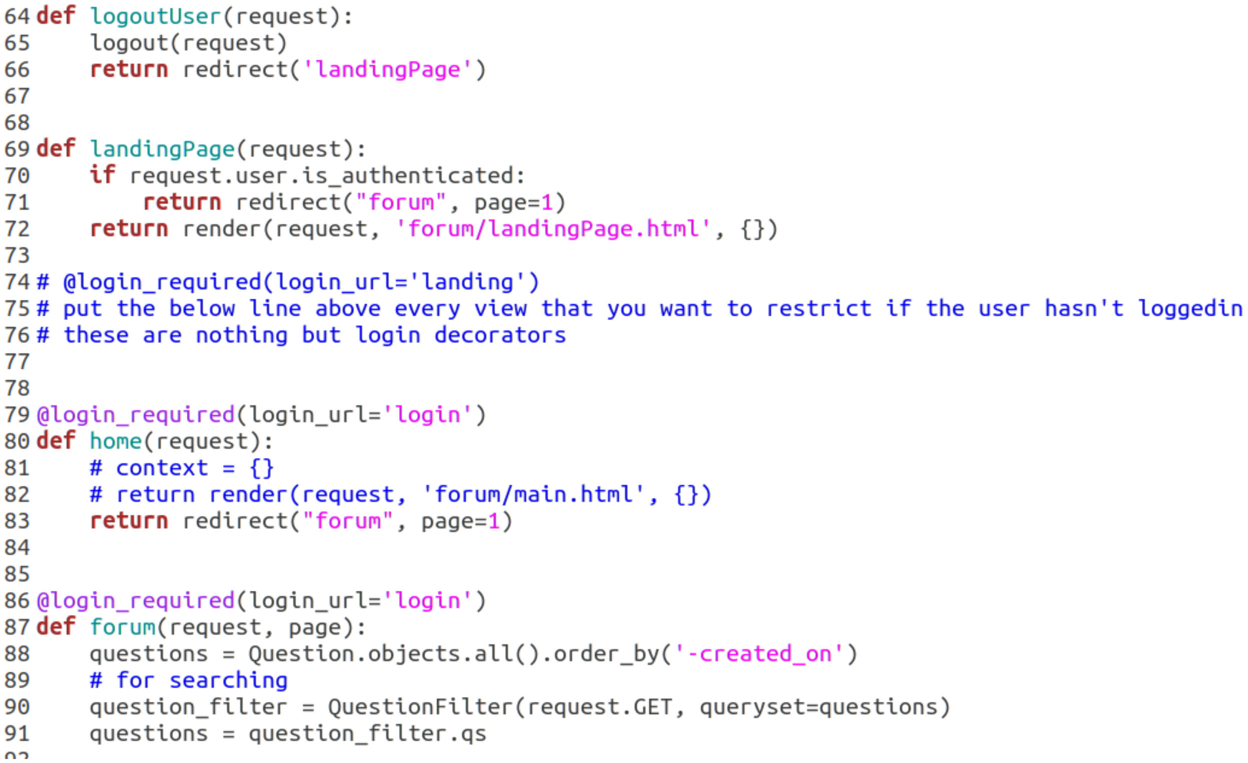
View.py



```python
from django.contrib.auth.decorators import login_required
from django.shortcuts import render, redirect
from django.http import HttpResponse, Http404, JsonResponse
from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth import authenticate, login, logout
from django.contrib import messages  # for flash messages
from .models import *
from .filters import *
# from .forms import CreateUserForm
from .forms import *
import logging
custom_db_logger = logging.getLogger("forum.custom.db")


# for restricting users without login

# Create your views here.


def registerPage(request):
    if request.user.is_authenticated:
        return redirect('home')
    else:
        form = CreateUserForm()
        if request.method == 'POST':
            form = CreateUserForm(request.POST)
            if form.is_valid():
                form.save()
                user_msg = form.cleaned_data.get('username')
                messages.success(request, 'Account was created for '+user_msg)
```

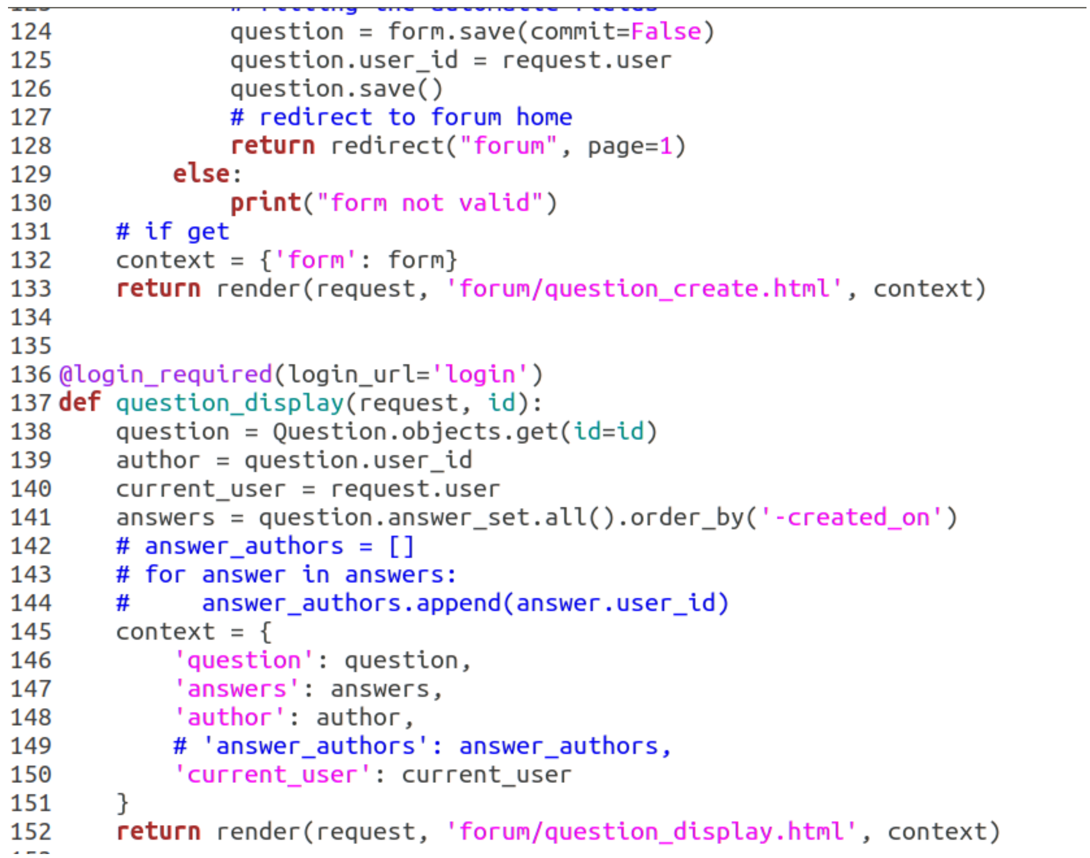


```python
                return redirect('login')

        context = {'form': form}
        return render(request, 'forum/register.html', context)


def loginPage(request):
    if request.user.is_authenticated:
        return redirect('home')
        # return redirect('forum/1')
    else:
        if request.method == 'POST':
            username = request.POST.get('username')
            password = request.POST.get('password')
            # request.POST.get('yearOfAdmission')

            user = authenticate(request, username=username, password=password)

            if user is not None:
                login(request, user)
                # landingpage should come here DOUBTFULL, earlier it was home
                return redirect('home')
                # return redirect('forum/1')
            else:
                messages.info(request, 'Username OR Password is incorrect')

        context = {}
        return render(request, 'forum/login.html', context)
```

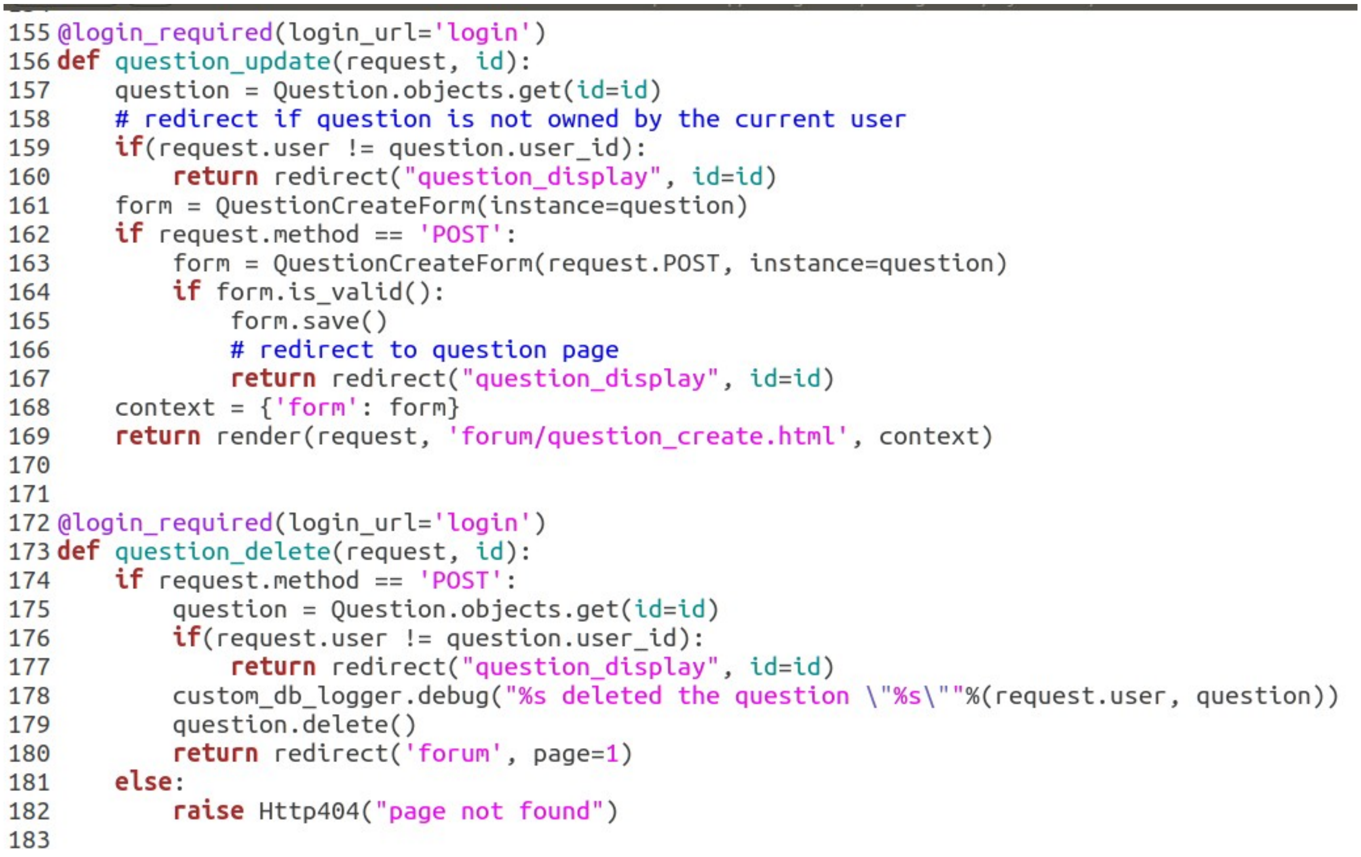


```python
def logoutUser(request):
    logout(request)
    return redirect('landingPage')


def landingPage(request):
    if request.user.is_authenticated:
        return redirect("forum", page=1)
    return render(request, 'forum/landingPage.html', {})

# @login_required(login_url='landing')
# put the below line above every view that you want to restrict if the user hasn't loggedin
# these are nothing but login decorators


@login_required(login_url='login')
def home(request):
    # context = {}
    # return render(request, 'forum/main.html', {})
    return redirect("forum", page=1)


@login_required(login_url='login')
def forum(request, page):
    questions = Question.objects.all().order_by('-created_on')
    # for searching
    question_filter = QuestionFilter(request.GET, queryset=questions)
    questions = question_filter.qs
```

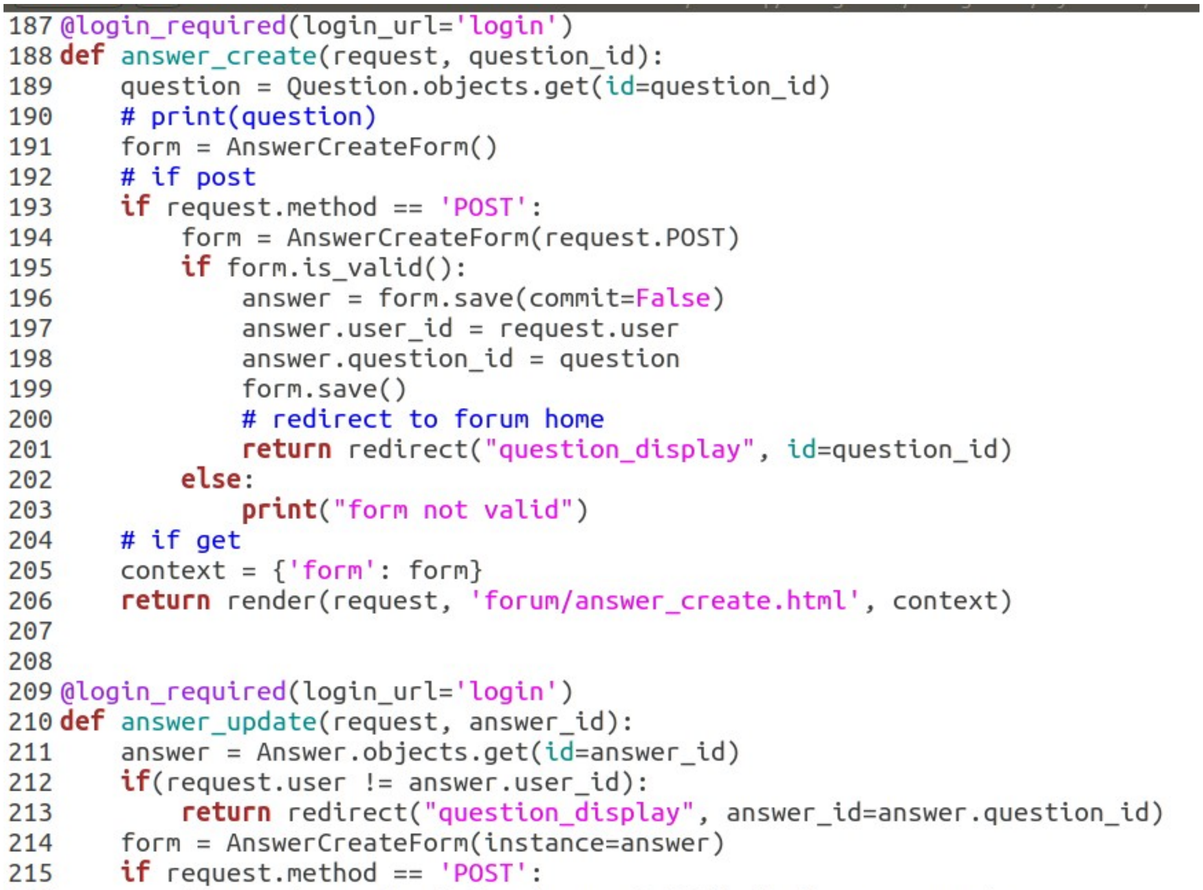
```python
def logoutUser(request):
    logout(request)
    return redirect('landingPage')


def landingPage(request):
    if request.user.is_authenticated:
        return redirect("forum", page=1)
    return render(request, 'forum/landingPage.html', {})

# @login_required(login_url='landing')
# put the below line above every view that you want to restrict if the user hasn't loggedin
# these are nothing but login decorators


@login_required(login_url='login')
def home(request):
    # context = {}
    # return render(request, 'forum/main.html', {})
    return redirect("forum", page=1)


@login_required(login_url='login')
def forum(request, page):
    questions = Question.objects.all().order_by('-created_on')
    # for searching
    question_filter = QuestionFilter(request.GET, queryset=questions)
    questions = question_filter.qs
```



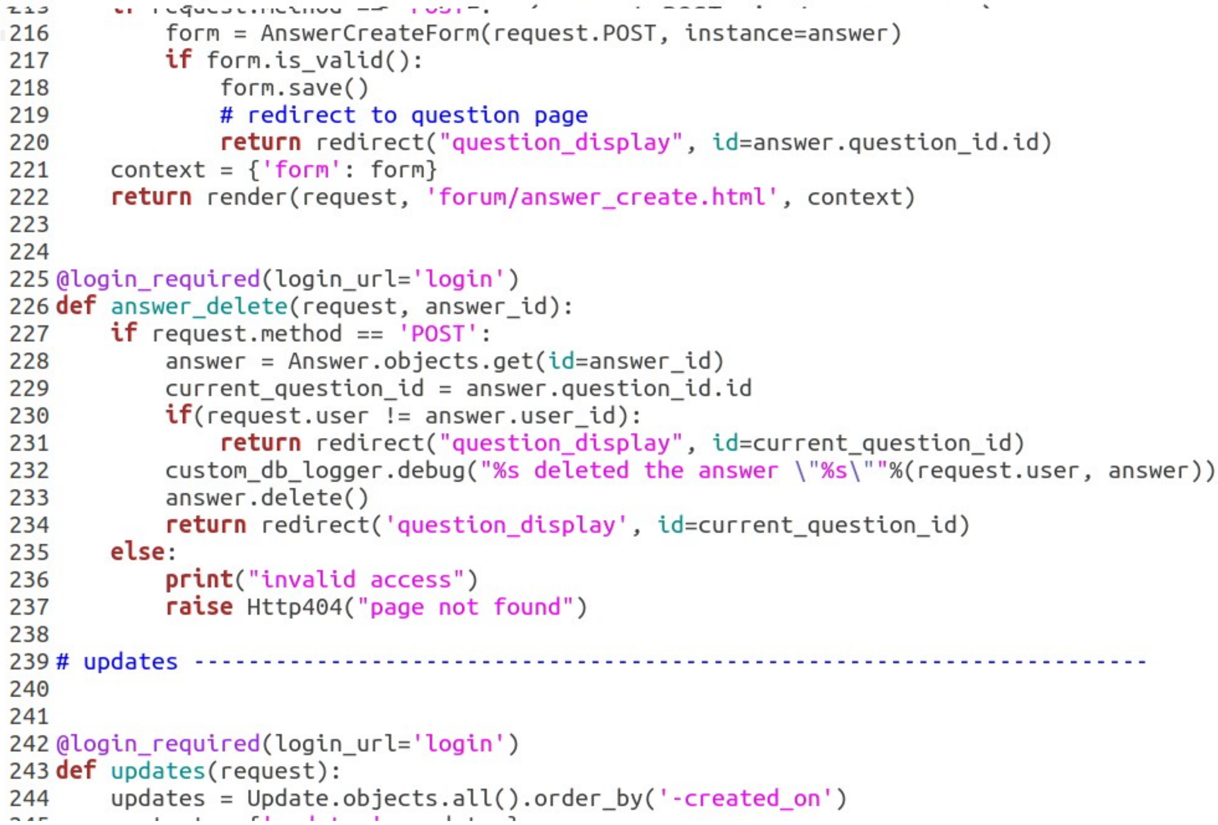
```python
            question = form.save(commit=False)
            question.user_id = request.user
            question.save()
            # redirect to forum home
            return redirect("forum", page=1)
        else:
            print("form not valid")
    # if get
    context = {'form': form}
    return render(request, 'forum/question_create.html', context)


@login_required(login_url='login')
def question_display(request, id):
    question = Question.objects.get(id=id)
    author = question.user_id
    current_user = request.user
    answers = question.answer_set.all().order_by('-created_on')
    # answer_authors = []
    # for answer in answers:
    #     answer_authors.append(answer.user_id)
    context = {
        'question': question,
        'answers': answers,
        'author': author,
        # 'answer_authors': answer_authors,
        'current_user': current_user
    }
    return render(request, 'forum/question_display.html', context)
```



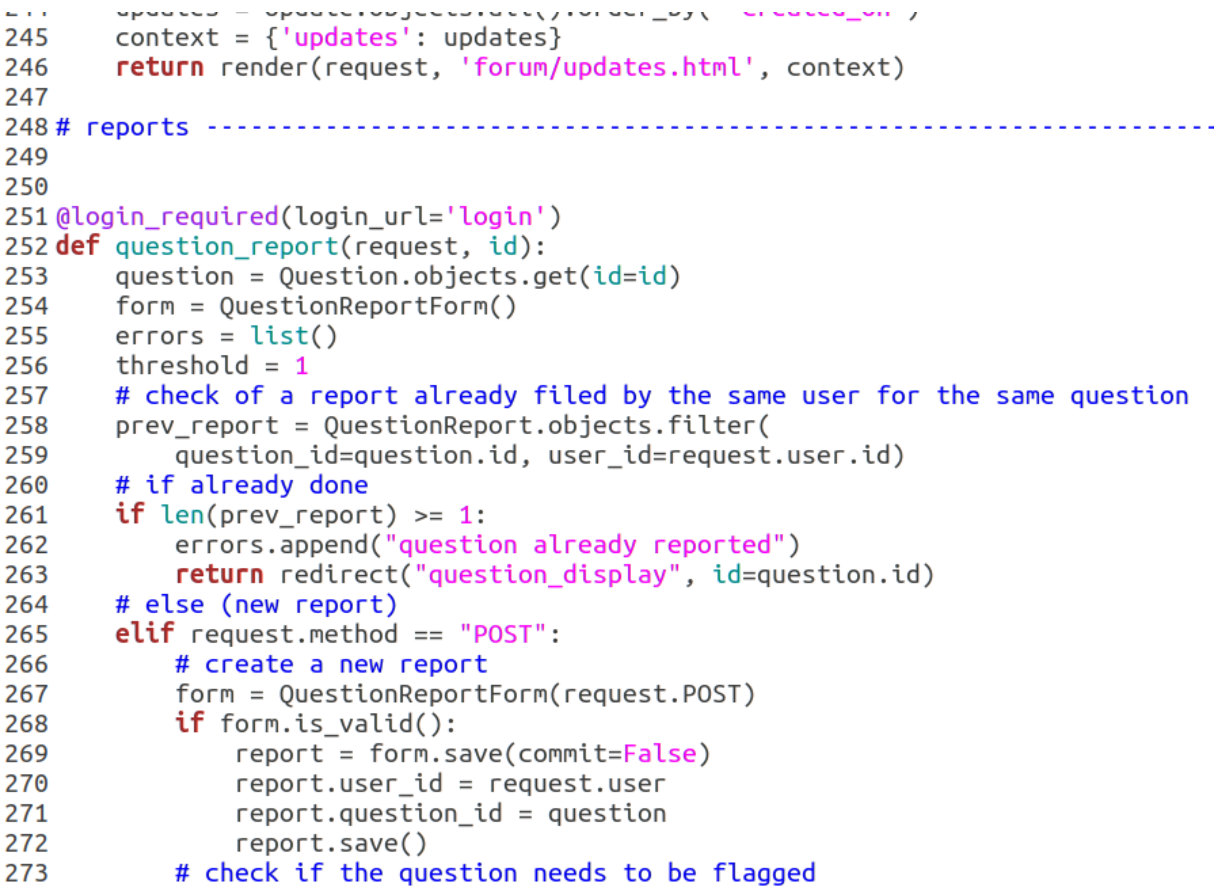
```python
@login_required(login_url='login')
def question_update(request, id):
    question = Question.objects.get(id=id)
    # redirect if question is not owned by the current user
    if(request.user != question.user_id):
        return redirect("question_display", id=id)
    form = QuestionCreateForm(instance=question)
    if request.method == 'POST':
        form = QuestionCreateForm(request.POST, instance=question)
        if form.is_valid():
            form.save()
            # redirect to question page
            return redirect("question_display", id=id)
    context = {'form': form}
    return render(request, 'forum/question_create.html', context)


@login_required(login_url='login')
def question_delete(request, id):
    if request.method == 'POST':
        question = Question.objects.get(id=id)
        if(request.user != question.user_id):
            return redirect("question_display", id=id)
        custom_db_logger.debug("%s deleted the question \"%s\""%(request.user, question))
        question.delete()
        return redirect('forum', page=1)
    else:
        raise Http404("page not found")

```



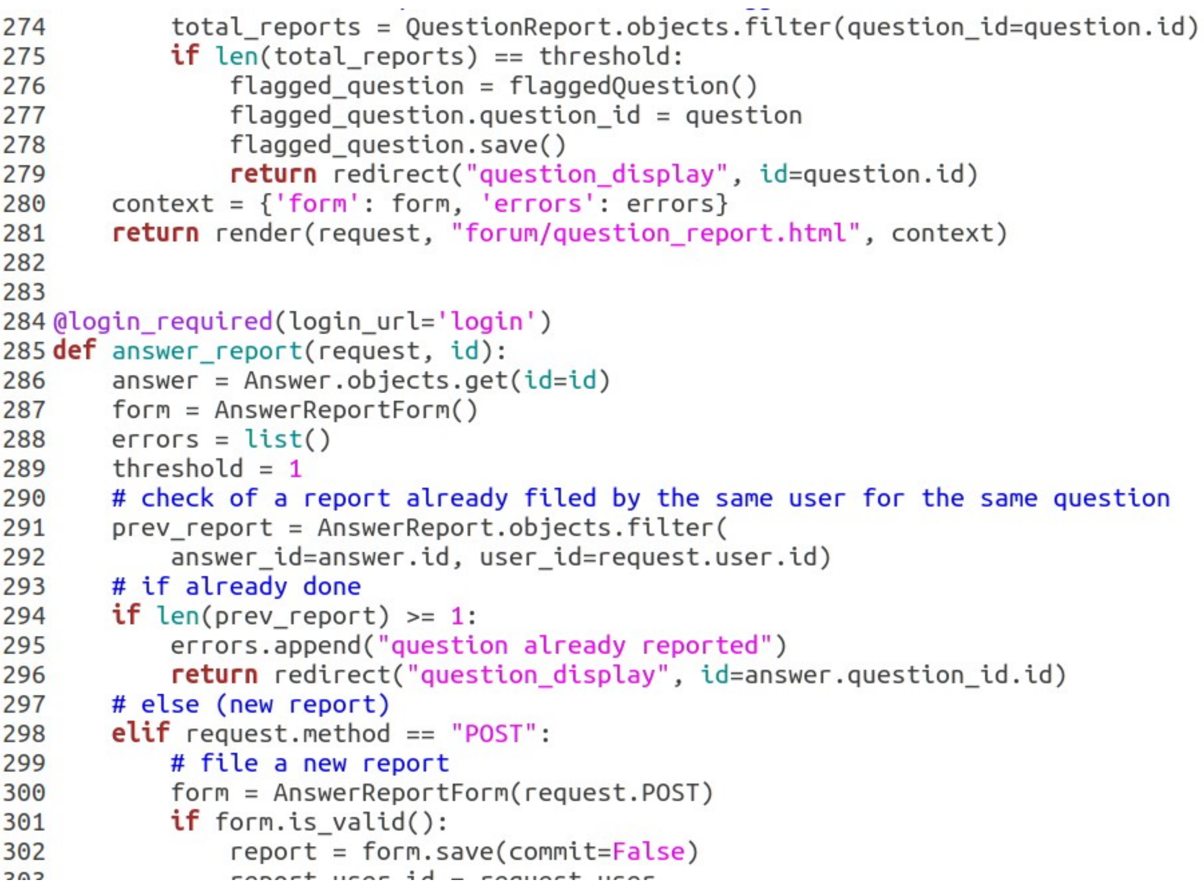
```python
@login_required(login_url='login')
def answer_create(request, question_id):
    question = Question.objects.get(id=question_id)
    # print(question)
    form = AnswerCreateForm()
    # if post
    if request.method == 'POST':
        form = AnswerCreateForm(request.POST)
        if form.is_valid():
            answer = form.save(commit=False)
            answer.user_id = request.user
            answer.question_id = question
            form.save()
            # redirect to forum home
            return redirect("question_display", id=question_id)
        else:
            print("form not valid")
    # if get
    context = {'form': form}
    return render(request, 'forum/answer_create.html', context)


@login_required(login_url='login')
def answer_update(request, answer_id):
    answer = Answer.objects.get(id=answer_id)
    if(request.user != answer.user_id):
        return redirect("question_display", answer_id=answer.question_id)
    form = AnswerCreateForm(instance=answer)
    if request.method == 'POST':
```



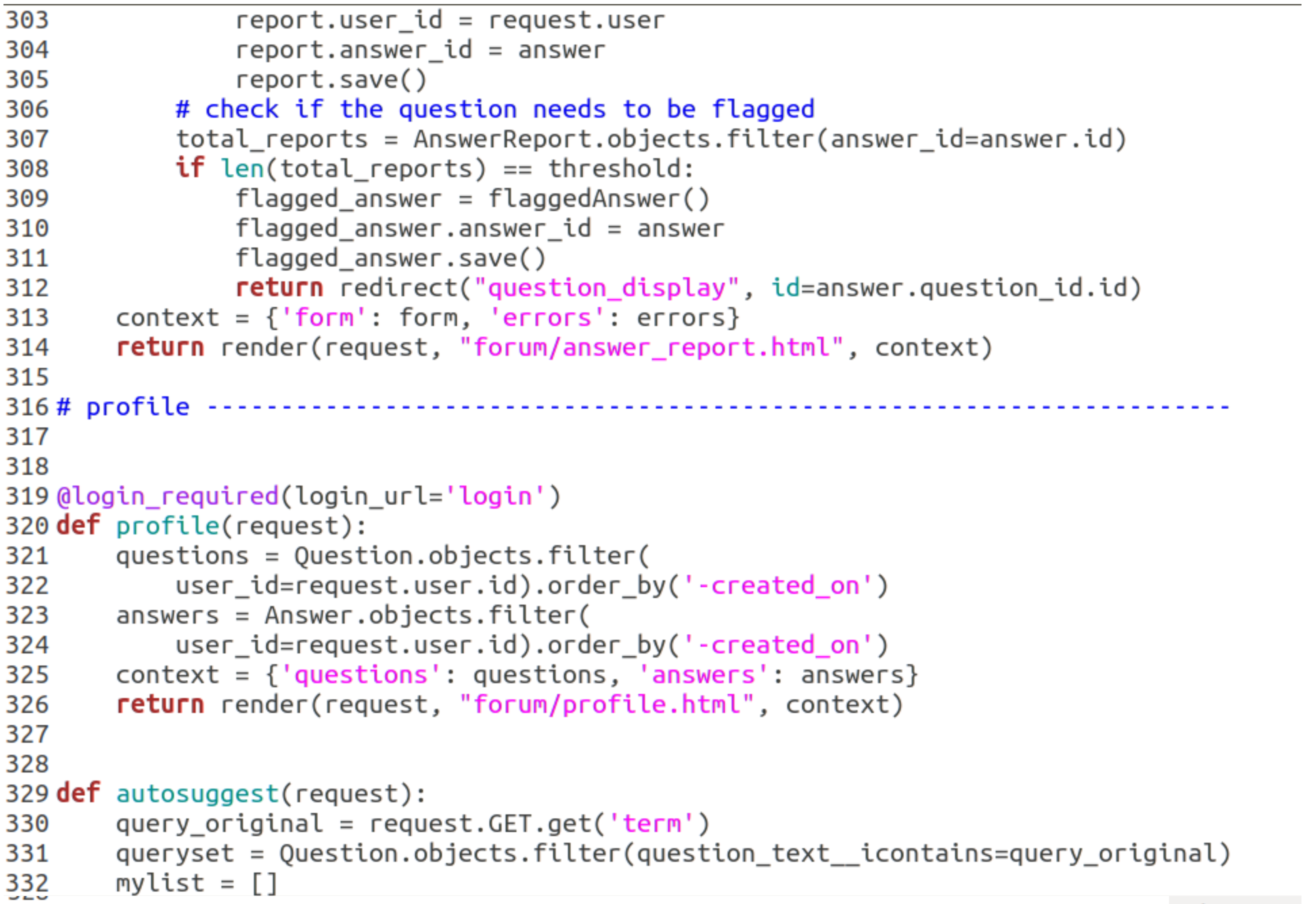
```python
        form = AnswerCreateForm(request.POST, instance=answer)
        if form.is_valid():
            form.save()
            # redirect to question page
            return redirect("question_display", id=answer.question_id.id)
    context = {'form': form}
    return render(request, 'forum/answer_create.html', context)


@login_required(login_url='login')
def answer_delete(request, answer_id):
    if request.method == 'POST':
        answer = Answer.objects.get(id=answer_id)
        current_question_id = answer.question_id.id
        if(request.user != answer.user_id):
            return redirect("question_display", id=current_question_id)
        custom_db_logger.debug("%s deleted the answer \"%s\""%(request.user, answer))
        answer.delete()
        return redirect('question_display', id=current_question_id)
    else:
        print("invalid access")
        raise Http404("page not found")

# updates ----------------------------------------------------------------


@login_required(login_url='login')
def updates(request):
    updates = Update.objects.all().order_by('-created_on')
```



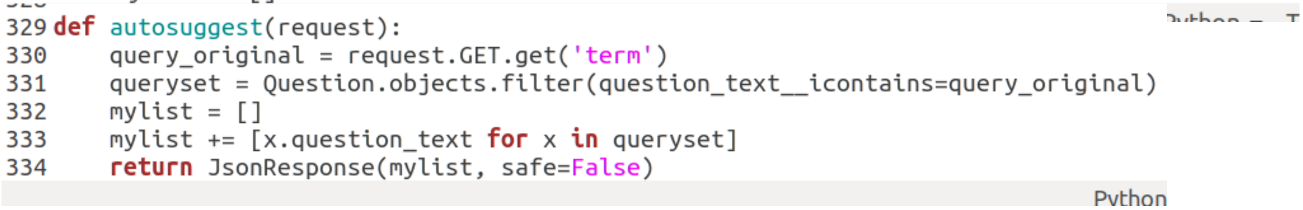
```python
    updates = Update.objects.all().order_by('-created_on')
    context = {'updates': updates}
    return render(request, 'forum/updates.html', context)

# reports -------------------------------------------------------------------


@login_required(login_url='login')
def question_report(request, id):
    question = Question.objects.get(id=id)
    form = QuestionReportForm()
    errors = list()
    threshold = 1
    # check of a report already filed by the same user for the same question
    prev_report = QuestionReport.objects.filter(
        question_id=question.id, user_id=request.user.id)
    # if already done
    if len(prev_report) >= 1:
        errors.append("question already reported")
        return redirect("question_display", id=question.id)
    # else (new report)
    elif request.method == "POST":
        # create a new report
        form = QuestionReportForm(request.POST)
        if form.is_valid():
            report = form.save(commit=False)
            report.user_id = request.user
            report.question_id = question
            report.save()
        # check if the question needs to be flagged
```



```python
        total_reports = QuestionReport.objects.filter(question_id=question.id)
        if len(total_reports) == threshold:
            flagged_question = flaggedQuestion()
            flagged_question.question_id = question
            flagged_question.save()
            return redirect("question_display", id=question.id)
    context = {'form': form, 'errors': errors}
    return render(request, "forum/question_report.html", context)


@login_required(login_url='login')
def answer_report(request, id):
    answer = Answer.objects.get(id=id)
    form = AnswerReportForm()
    errors = list()
    threshold = 1
    # check of a report already filed by the same user for the same question
    prev_report = AnswerReport.objects.filter(
        answer_id=answer.id, user_id=request.user.id)
    # if already done
    if len(prev_report) >= 1:
        errors.append("question already reported")
        return redirect("question_display", id=answer.question_id.id)
    # else (new report)
    elif request.method == "POST":
        # file a new report
        form = AnswerReportForm(request.POST)
        if form.is_valid():
            report = form.save(commit=False)
            report.user_id = request.user
```



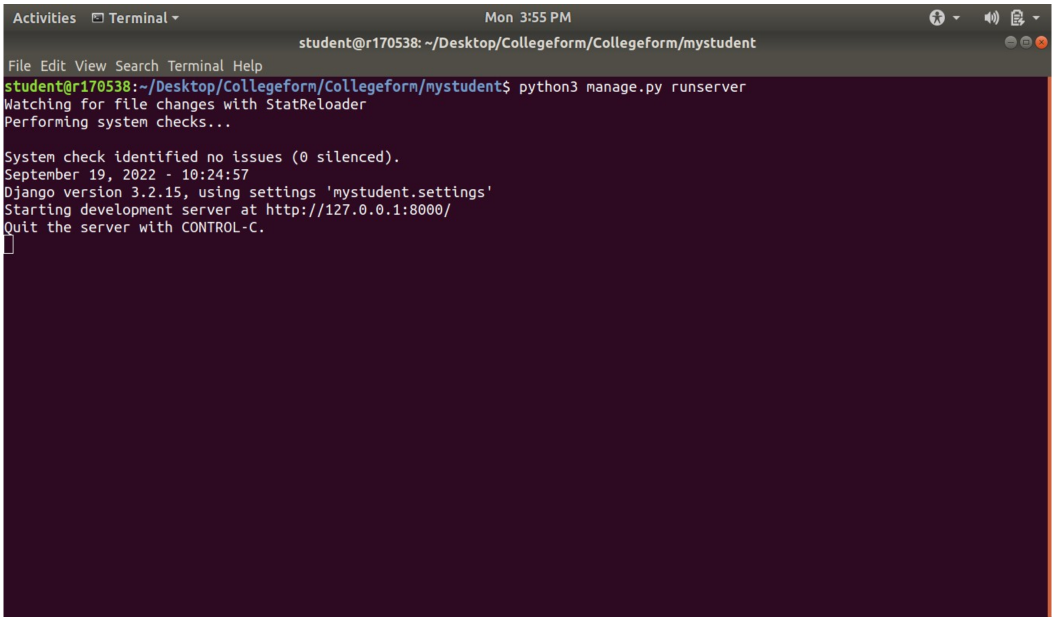
```python
            report.user_id = request.user
            report.answer_id = answer
            report.save()
        # check if the question needs to be flagged
        total_reports = AnswerReport.objects.filter(answer_id=answer.id)
        if len(total_reports) == threshold:
            flagged_answer = flaggedAnswer()
            flagged_answer.answer_id = answer
            flagged_answer.save()
            return redirect("question_display", id=answer.question_id.id)
    context = {'form': form, 'errors': errors}
    return render(request, "forum/answer_report.html", context)

# profile ---------------------------------------------------------------------


@login_required(login_url='login')
def profile(request):
    questions = Question.objects.filter(
        user_id=request.user.id).order_by('-created_on')
    answers = Answer.objects.filter(
        user_id=request.user.id).order_by('-created_on')
    context = {'questions': questions, 'answers': answers}
    return render(request, "forum/profile.html", context)


def autosuggest(request):
    query_original = request.GET.get('term')
    queryset = Question.objects.filter(question_text__icontains=query_original)
    mylist = []
```



```python
def autosuggest(request):
    query_original = request.GET.get('term')
    queryset = Question.objects.filter(question_text__icontains=query_original)
    mylist = []
    mylist += [x.question_text for x in queryset]
    return JsonResponse(mylist, safe=False)
```

EVALUATION

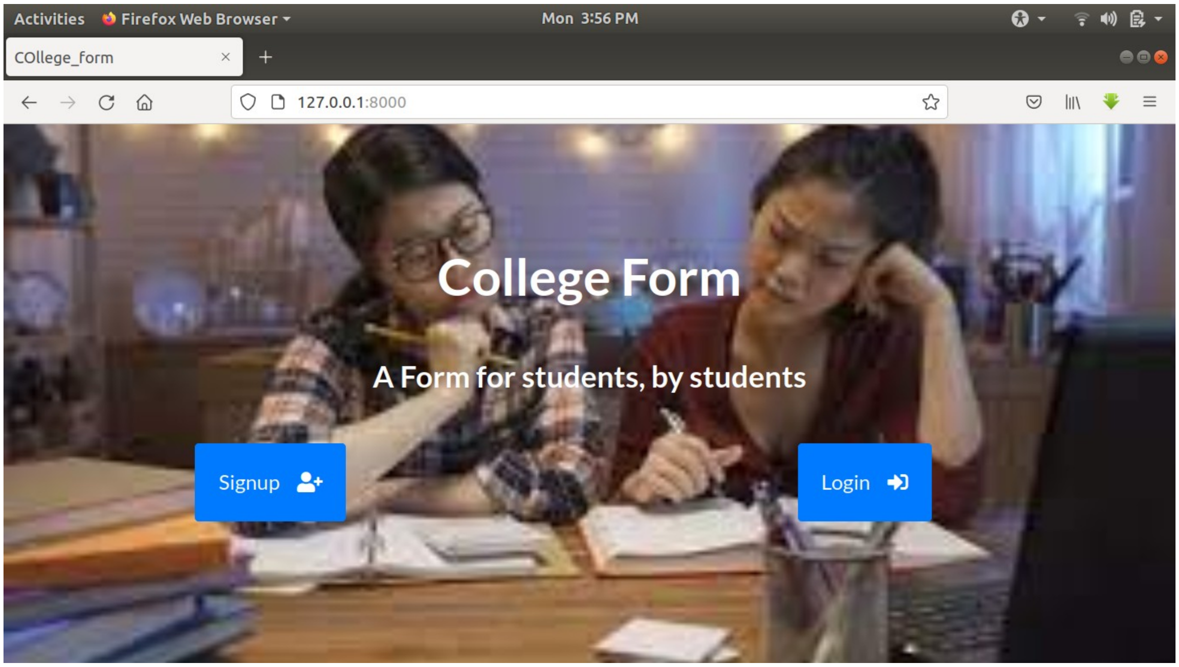
Image references:



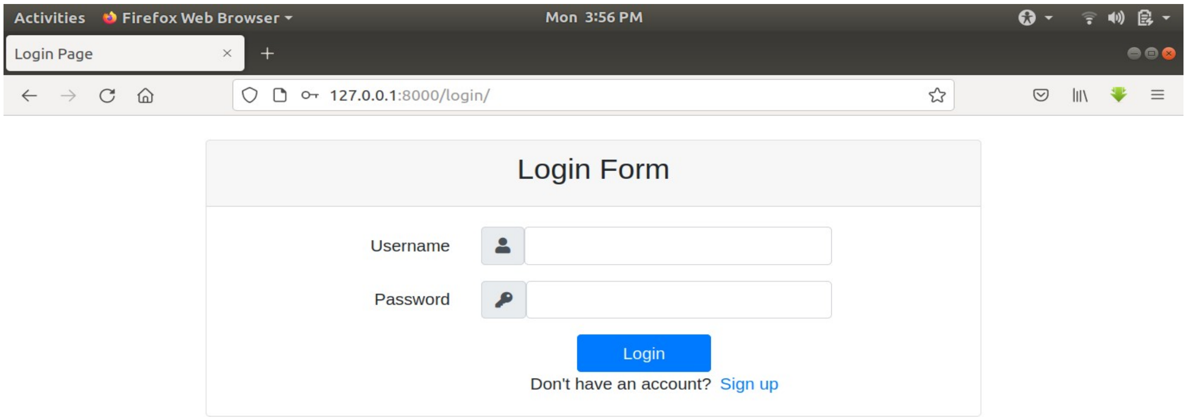
Activities
Terminal
Mon 3:55 PM
student@r170538: ~/Desktop/Collegeform/Collegeform/mystudent
File Edit View Search Terminal Help

```
student@r170538:~/Desktop/Collegeform/Collegeform/mystudent$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
September 19, 2022 - 10:24:57
Django version 3.2.15, using settings 'mystudent.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

HOME PAGE:



Login Page:



Login Form

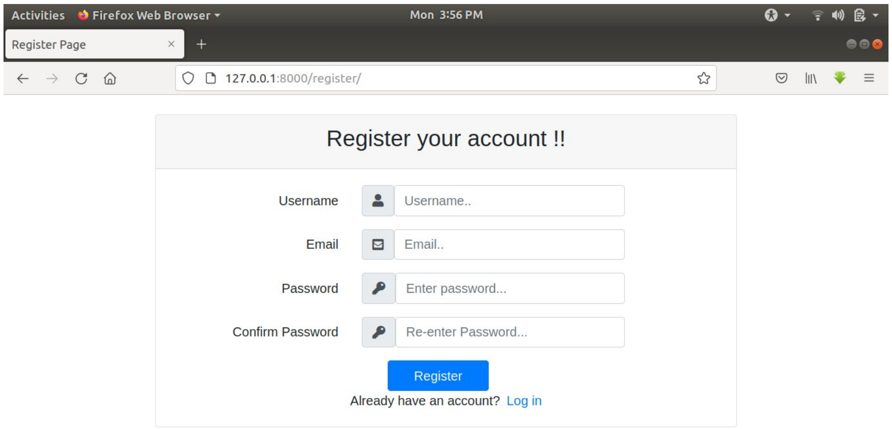
Username

Password

Login

Don't have an account? Sign up

Register Page:



Register your account !!

Username: Username..

Email: Email..

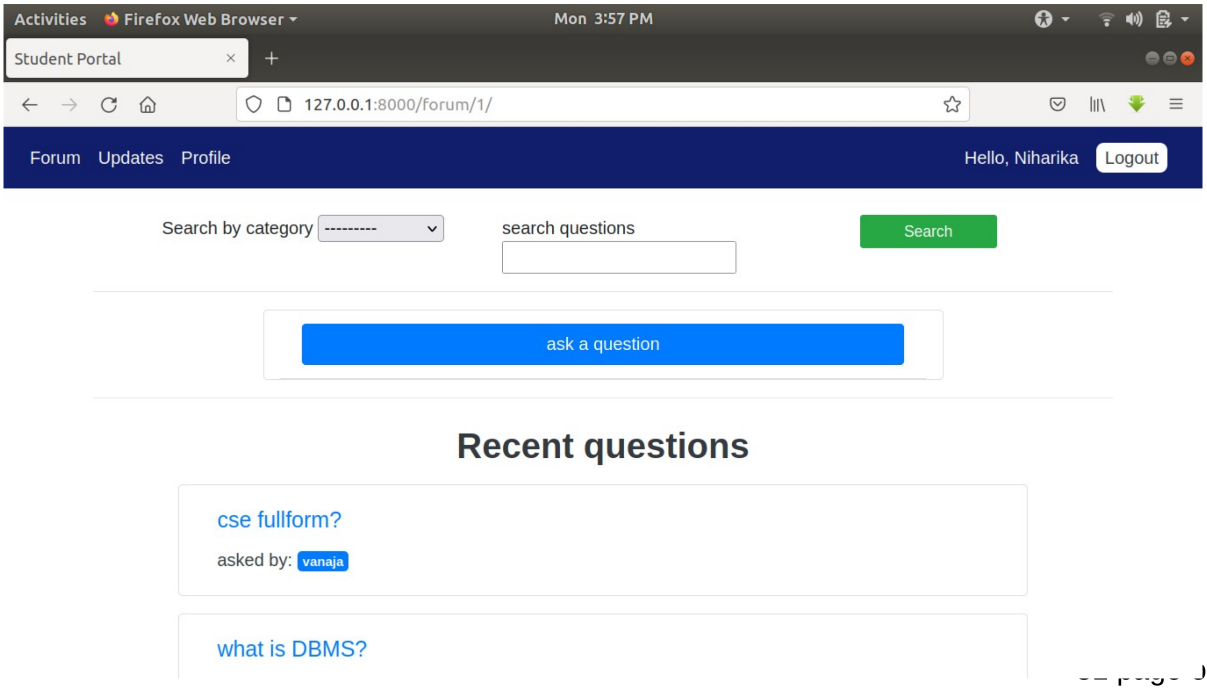
Password: Enter password...

Confirm Password: Re-enter Password...

Register

Already have an account? Log in

Forum Page:



Search by category: ---------

search questions

Search

ask a question

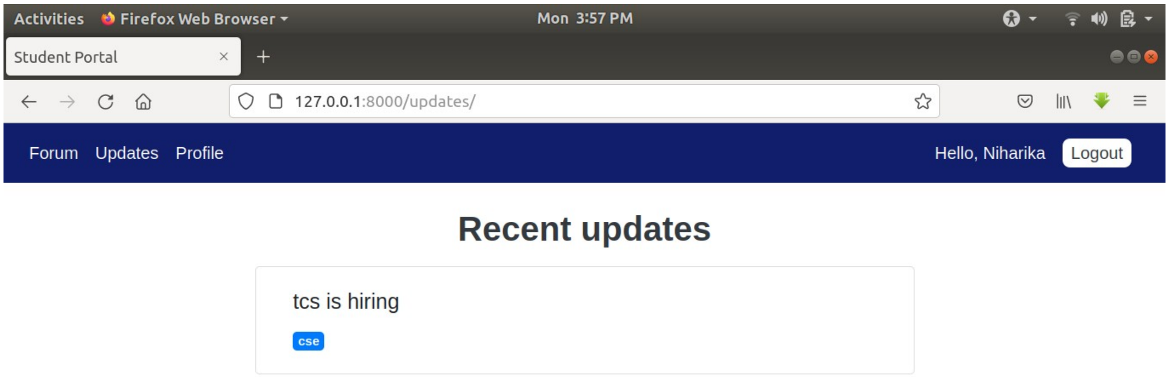
Recent questions

cse fullform?

asked by: vanaja

what is DBMS?

Updates Page:

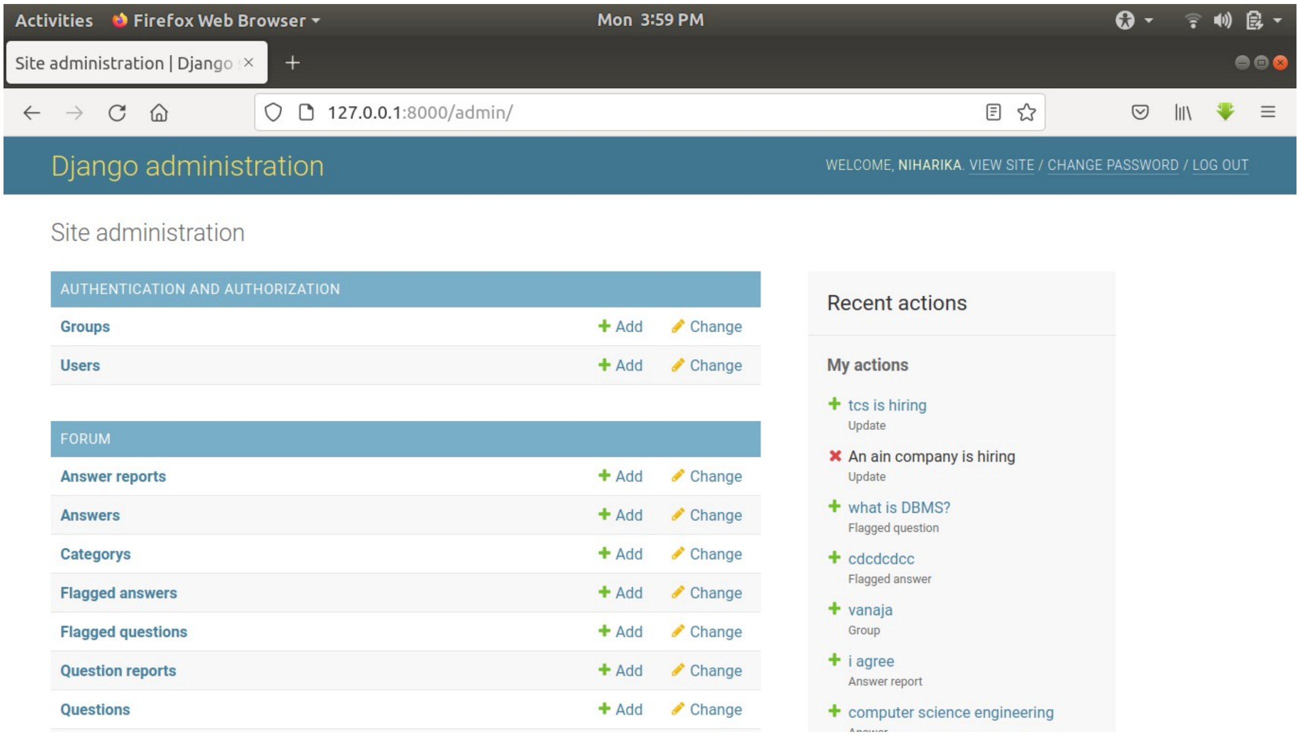


Recent updates

tcs is hiring

cse

Admin Page:


Activities
Firefox Web Browser
Mon 3:59 PM
Site administration | Django
127.0.0.1:8000/admin/
Django administration
WELCOME, NIHARIKA. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

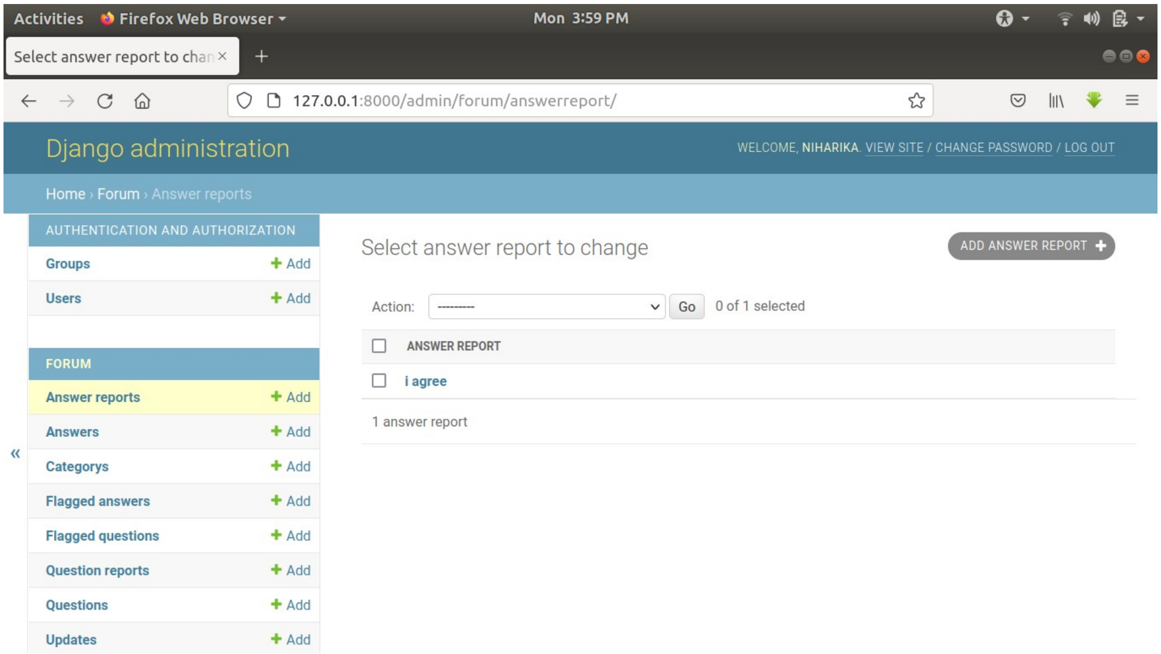
FORUM		
Answer reports	+ Add	Change
Answers	+ Add	Change
Categorys	+ Add	Change
Flagged answers	+ Add	Change
Flagged questions	+ Add	Change
Question reports	+ Add	Change
Questions	+ Add	Change

Recent actions

My actions

- + tcs is hiring
  Update
- ✖ An ain company is hiring
  Update
- + what is DBMS?
  Flagged question
- + cdcdcdcc
  Flagged answer
- + vanaja
  Group
- + i agree
  Answer report
- + computer science engineering

Answers report Page:



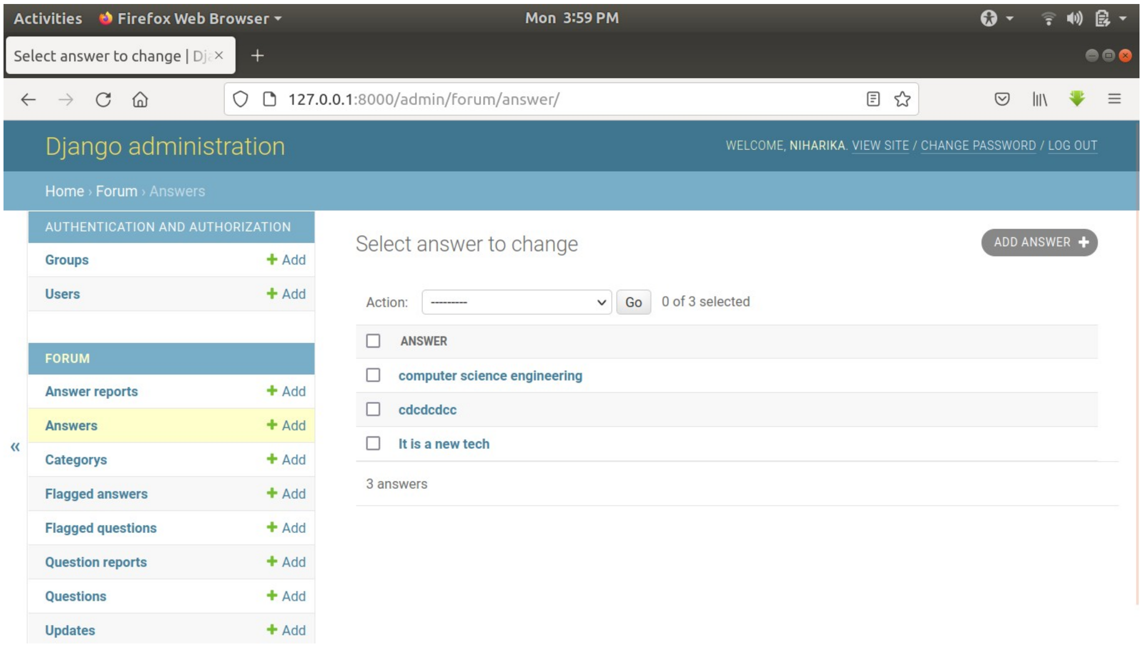
Select answer report to change

Action: --------- Go 0 of 1 selected

□	ANSWER REPORT
□	i agree

1 answer report

Answers Page:



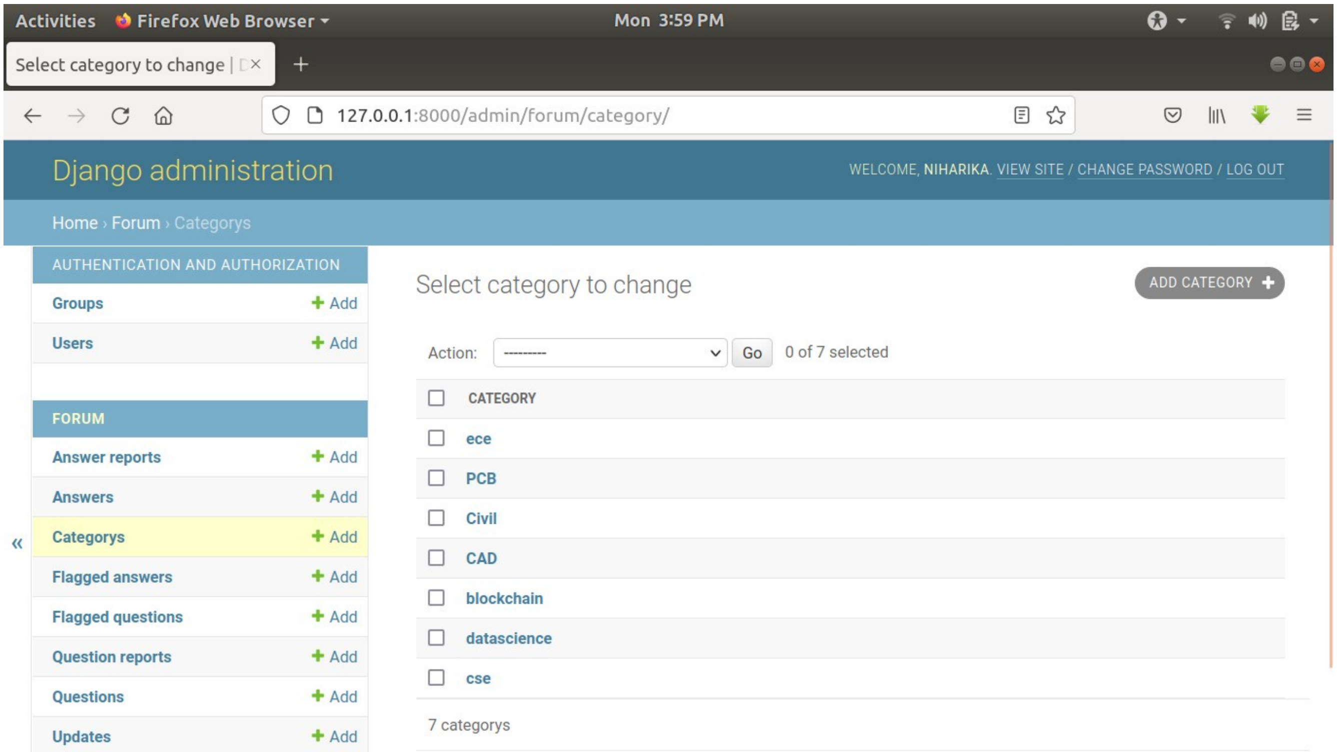
Select answer to change

Action: --------- Go 0 of 3 selected

□	ANSWER
□	computer science engineering
□	cdcdcdcc
□	It is a new tech

3 answers

Category Page:



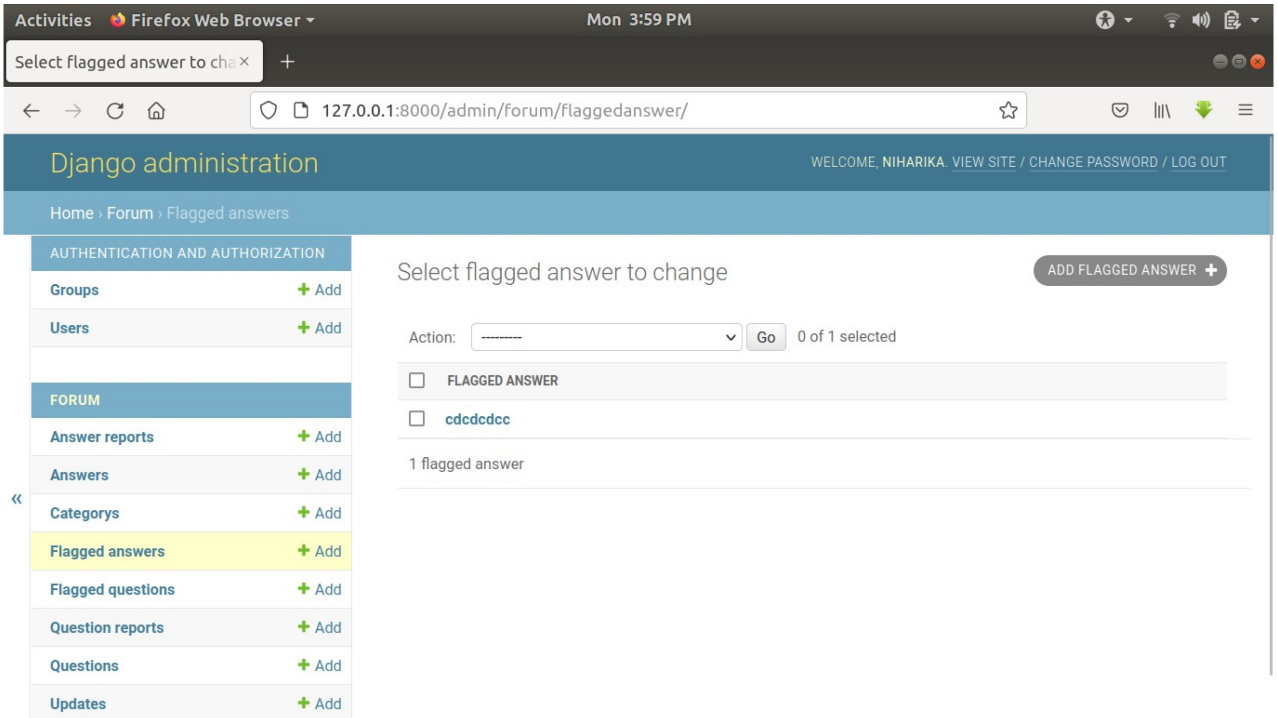
Select category to change

Action: --------- Go 0 of 7 selected

□	CATEGORY
□	ece
□	PCB
□	Civil
□	CAD
□	blockchain
□	datascience
□	cse

7 categorys

Flagged Page:



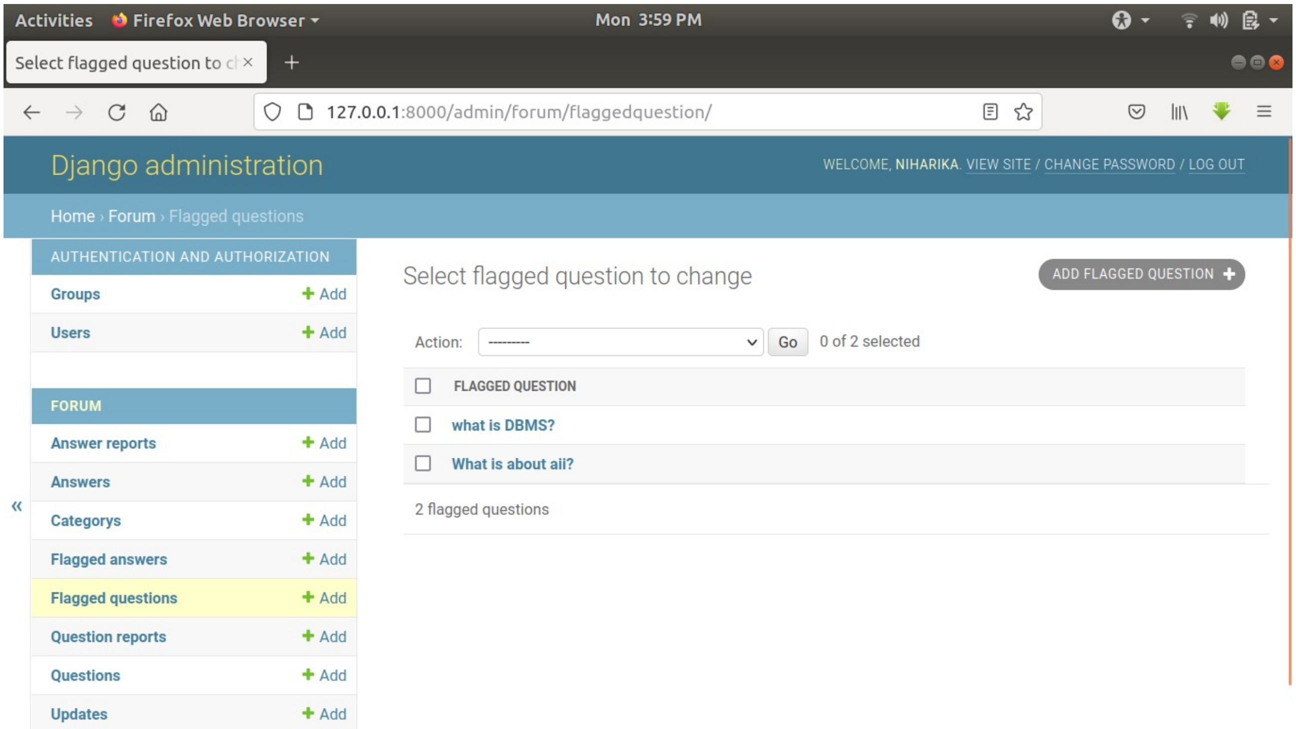
Select flagged answer to change

Action: --------- Go 0 of 1 selected

□	FLAGGED ANSWER
□	cdcdcdcc

1 flagged answer

Flagged question Page:


Activities
Firefox Web Browser
Mon 3:59 PM
Select flagged question to c
127.0.0.1:8000/admin/forum/flaggedquestion/

Django administration

WELCOME, NIHARIKA. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Forum › Flagged questions

AUTHENTICATION AND AUTHORIZATION	
Groups	+ Add
Users	+ Add

FORUM	
Answer reports	+ Add
Answers	+ Add
Categorys	+ Add
Flagged answers	+ Add
Flagged questions	+ Add
Question reports	+ Add
Questions	+ Add
Updates	+ Add

Select flagged question to change

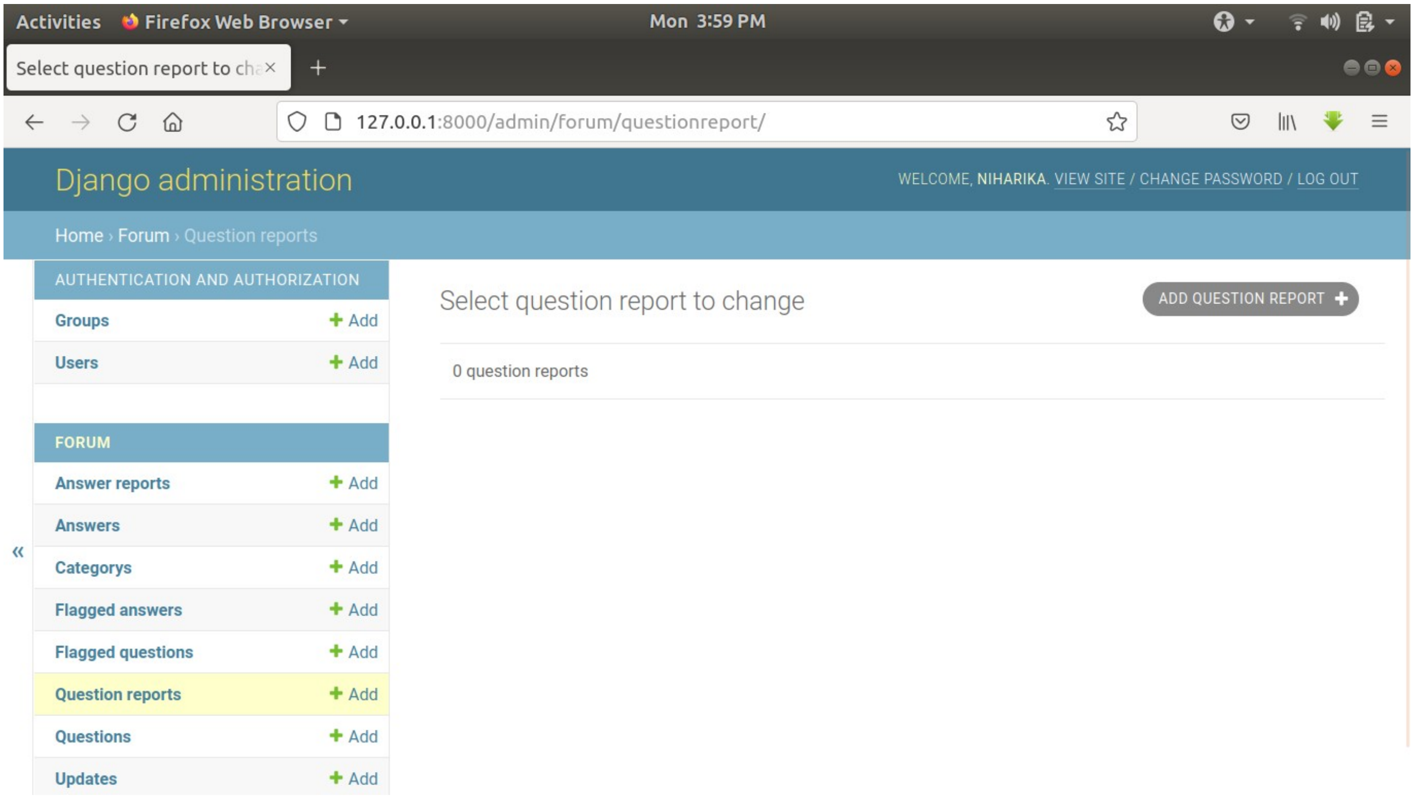
ADD FLAGGED QUESTION +

Action: --------- Go 0 of 2 selected

□	FLAGGED QUESTION
□	what is DBMS?
□	What is about aii?

2 flagged questions

Question Report Page:


Activities
Firefox Web Browser
Mon 3:59 PM
Select question report to cha
127.0.0.1:8000/admin/forum/questionreport/

Django administration

WELCOME, NIHARIKA. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Forum › Question reports

AUTHENTICATION AND AUTHORIZATION	
Groups	+ Add
Users	+ Add

FORUM	
Answer reports	+ Add
Answers	+ Add
Categorys	+ Add
Flagged answers	+ Add
Flagged questions	+ Add
Question reports	+ Add
Questions	+ Add
Updates	+ Add

Select question report to change

ADD QUESTION REPORT +

0 question reports

References

https://docs.djangoproject.com/en/4.1/howto/deployment/asgi/
https://jquery.com/
https://getbootstrap.com/
https://www.sqlite.org/index.html