***DJANGO DEVELOPER***

# **ASSESSMENT**

# TASK 1:

1. To create a new Django project named "myproject", you can run the following command in the command prompt:

**django-admin startproject myproject**

2. Inside the project, create an app named "myapp" inside the project, you can run the following command 8 ,9. You can create a function in the views.py file that queries the Issue table and assigns mechanics based on the availability of mechanics and the priority of the issue. You can then update the status of the issue to reflect the assigned mechanic. If there are 10 mechanics available, you can assign them based on the priority of the issue. To randomly assign an agent with the least requests in his queue, you can create a function in the views.py file that queries the Agents table and selects the agent with the lowest queue value. You can create a model form in the forms.py file to create a new issue where the user can send a request stating the problem, location, time, and userID.

Here's an algorithm for handling the queue and assigning mechanics:

1. Query the Issue table and assign mechanics based on the availability of mechanics and the priority of the issue.

2. Update the status of the issue to reflect the assigned mechanic.

3. If there are 10 mechanics available, assign them based on the priority of the issue.

4. Create a function in the views.py file that queries the Agents table and selects the agent with the lowest queue value.

5. Create a model form in the forms.py file to create a new issue where the user can send a request stating the problem, location, time, and userID.

**cd myproject**

**python manage.py startapp myapp**

3. To define a model named "Issue" in "myapp" with the fields issueID, userID, location, problem, time, and status, you can add the following code to the models.py file inside the "myapp" directory:

**python**

**class Issue(models.Model):**

**issueID = models.AutoField(primary\_key=True)**

**userID = models.IntegerField()**

**location = models.CharField(max\_length=100)**

**problem = models.TextField()**

**time = models.DateTimeField()**

**status = models.CharField(max\_length=100)**

4. To define another model named "Agents" in "myapp" with the fields AgentID and Queue, you can add the following code to the models.py file:

**python**

**class Agents(models.Model):**

**agentID = models.AutoField(primary\_key=True)**

**queue = models.IntegerField()**

5. To define another model named "Mechanic" in "myapp" with the fields mechanicID and availability, you can add the following code to the models.py file:

**python**

**class Mechanic(models.Model):**

**mechanicID = models.AutoField(primary\_key=True)**

**availability = models.BooleanField()**

6. New Issue: for the user to send a request stating the problem, his location, time, and userID

**You can create a form in `forms.py` for the user to submit a new issue:**

from django import forms

**from .models import Issue**

**class IssueForm(forms.ModelForm):**

**class Meta:**

**model = Issue**

**fields = ['userID', 'location', 'problem']**

7. Once the issue is created you have to randomly assign an agent with the least requests in his queue.

You can create a function in [`views.py`](http://views.py/) to assign an agent:

**from.models import Issue, Agents**

**import random**

**def assign\_agent():**

**agents = Agents.objects.all()**

**min\_queue = min([agent.queue for agent in agents])**

**agents\_with\_min\_queue = [agent for agent in agents if agent.queue == min\_queue]**

**return random.choice(agents\_with\_min\_queue)**

8 ,9. You can create a function in the views.py file that queries the Issue table and assigns mechanics based on the availability of mechanics and the priority of the issue. You can then update the status of the issue to reflect the assigned mechanic. If there are 10 mechanics available, you can assign them based on the priority of the issue. To randomly assign an agent with the least requests in his queue, you can create a function in the views.py file that queries the Agents table and selects the agent with the lowest queue value. You can create a model form in the forms.py file to create a new issue where the user can send a request stating the problem, location, time, and userID.

Here's an algorithm for handling the queue and assigning mechanics:

1. Query the Issue table and assign mechanics based on the availability of mechanics and the priority of the issue.

2. Update the status of the issue to reflect the assigned mechanic.

3. If there are 10 mechanics available, assign them based on the priority of the issue.

4. Create a function in the views.py file that queries the Agents table and selects the agent with the lowest queue value.

5. Create a model form in the forms.py file to create a new issue where the user can send a request stating the problem, location, time, and userID.