REPORT – 3

INTEGRATION OF DJANGO

WITH AN API

An **Application Programming Interface (API)** is a [computing interface](https://en.wikipedia.org/wiki/Interface_(computing)) which defines interactions between multiple software intermediaries.

**Steps :**

1. Set up Django
2. Create a model in the database that the Django ORM will manage
3. Set up the Django REST Framework
4. Serialize the model from step 2
5. Create the URI endpoints to view the serialized data

**1.Set up Django**

1.1 Create a new virtual environment for your project so you can manage your dependencies separately.

1.2 Install Django and start a new Django project

1.3 Create an API app

1.4 Register the app with the project : We need to tell Django to recognize this new app that we just created.So, we edit the settings.py

Add the following in the installed apps section : ‘api\_app.apps.api\_appConfig’

1.5 Migrate the database

**2. Create a model in the database that Django ORM will manage**

2.1Make your models

2.2 Make migrations : Whenever we define or change a model, we need to tell Django to migrate those changes.

3. Set up Django REST Framework

Pip install djangorestframework

Now, tell Django that we installed the REST Framework in settings.py by finding the installed apps section and add ‘rest\_framework’

**4. Serialize the model**

create a new file — api\_app/serializers.py

In this file, we need to:

1. Import the model
2. Import the REST Framework serializer
3. Create a new class that links the model with its serializer

5. Display the data

Now, all that’s left to do is wire up the URLs and views to display the data!

5.1 Views

1. Query the database
2. Pass that database queryset into the serializer we just created, so that it gets converted into JSON and rendered

5.2 Site URLs : Point a URL at the viewset we just created.

5.3 API URLs : Route the ‘api\_app.urls’

The REST Framework router will make sure our requests end up at the right resource dynamically. If we add or delete items from the database, the URLs will update to match.

INTEGRATION OF DJANGO

WITH A DATABASE

1. Create a Django project by running django-admin.py startproject mysite (where mysite is your project’s name).
2. Edit the settings file in that project, mysite/settings.py, to tell Django what your database connection parameters are and what the name of the database is. Specifically, provide the DATABASE\_NAME, DATABASE\_ENGINE, DATABASE\_USER, DATABASE\_PASSWORD, DATABASE\_HOST, and DATABASE\_PORT settings. (Some of these settings are optional)
3. Create a Django application within your project by running python mysite/manage.py startapp myapp (where myapp is your application’s name).
4. Run the command python mysite/manage.py inspectdb. This will examine the tables in the DATABASE\_NAME database and print the generated model class for each table.
5. Save the output to the models.py file within your application by using standard shell output redirection:

python mysite/manage.py inspectdb > mysite/myapp/models.py

1. Edit the mysite/myapp/models.py file to clean up the generated models and make any necessary customizations.