**Step 1: Setup the Project Structure Create the folders as per the given structure:**

UI: C:\Git\ModelPredictApp\view

API: C:\Git\ModelPredictApp\controller\api

Model (Database scripts): C:\Git\ModelPredictApp\model

**Step 2: Setup the UI with React**

cd C:\Git\ModelPredictApp\view

npx create-react-app <UI APP name>

Create a new component RegisterForm.js in the src folder.

Implement the form with the required fields in RegisterForm.js// C:\Git\ModelPredictApp\view\src\RegisterForm.js

---------------------------------------

import React, { useState } from 'react';

function RegisterForm() {

const [formData, setFormData] = useState({

firstTime: '',

lastTime: '',

userId: '',

phoneNumber: '',

password: '',

confirmPassword: ''

});

const handleChange = (e) => {

setFormData({ ...formData, [e.target.name]: e.target.value });

};

const handleSubmit = async (e) => {

e.preventDefault();

// Call API to submit the form data

const response = await fetch('http://localhost:8000/register', {

method: 'POST',

headers: {

'Content-Type': 'application/json',

},

body: JSON.stringify(formData),

});

if (response.ok) {

console.log('Registration successful');

} else {

console.log('Registration failed');

}

};

return (

<form onSubmit={handleSubmit}>

<input name="firstTime" value={formData.firstTime} onChange={handleChange} placeholder="First Time" />

<input name="lastTime" value={formData.lastTime} onChange={handleChange} placeholder="Last Time" />

<input name="userId" value={formData.userId} onChange={handleChange} placeholder="User ID" />

<input name="phoneNumber" value={formData.phoneNumber} onChange={handleChange} placeholder="Phone Number" />

<input type="password" name="password" value={formData.password} onChange={handleChange} placeholder="Password" />

<input type="password" name="confirmPassword" value={formData.confirmPassword} onChange={handleChange} placeholder="Confirm Password" />

<button type="submit">Register</button>

</form>

);

}

export default RegisterForm;.

---------------------------------

**Step 3: Setup the API with Django**

1. Navigate to the api folder.
2. Create a Django project and app

----------

django-admin startproject ModelPredictApp .

django-admin startapp api

------------

1. Define your model in api/models.py.

------------

# C:\Git\ModelPredictApp\controller\api\api\models.py

from django.db import models

class User(models.Model):

first\_time = models.CharField(max\_length=100)

last\_time = models.CharField(max\_length=100)

user\_id = models.CharField(max\_length=100, unique=True)

phone\_number = models.CharField(max\_length=100)

password = models.CharField(max\_length=100)

---------

1. Create a view in api/views.py to handle registration.

----------------------

# C:\Git\ModelPredictApp\controller\api\api\views.py

from django.http import JsonResponse

from .models import User

from django.views.decorators.csrf import csrf\_exempt

import json

@csrf\_exempt

def register(request):

if request.method == 'POST':

data = json.loads(request.body)

user = User.objects.create(

first\_time=data['firstTime'],

last\_time=data['lastTime'],

user\_id=data['userId'],

phone\_number=data['phoneNumber'],

password=data['password'] # In a real app, ensure to hash passwords

)

user.save()

return JsonResponse({"message": "User registered successfully"}, status=200)

---------------------------------

1. Add the route in ModelPredictApp/urls.py.

----------------

# C:\Git\ModelPredictApp\controller\api\ModelPredictApp\urls.py

from django.contrib import admin

from django.urls import path

from api.views import register

urlpatterns = [

path('admin/', admin.site.urls),

path('register/', register),

]

----------------

**Step 4: Setup the Model with PostgreSQL**

First, you need to install psycopg2, which is a PostgreSQL adapter for Python. You can install it using pip:

pip install psycopg2

1. Update settings.py

Next, update the DATABASES setting in your settings.py file to use PostgreSQL. You will find this file at controller/api/ModelPredictApi/ModelPredictApi/settings.py. Replace the current DATABASES configuration with the following, making sure to substitute your\_database\_name, your\_database\_user, and your\_database\_password with your actual database name, user, and password:

----------------

DATABASES = {

    'default': {

        'ENGINE': 'django.db.backends.postgresql',

        'NAME': 'your\_database\_name',

        'USER': 'your\_database\_user',

        'PASSWORD': 'your\_database\_password',

        'HOST': 'localhost',  # Or an IP Address that your DB is hosted on

        'PORT': '5432',

    }

}

---------------

2: Run Migrations

After configuring your database, you need to run migrations to create the database schema. You can do this by running the following commands in your terminal:

---------------

python manage.py makemigrations

python manage.py migrate

---------------------

**Step5 After installing PostgreSQL, you can create a new database and user for your Django project. Here's a quick way to do that:**

1.Switch to the postgres user

sudo -i -u postgres

2.Open the PostgreSQL interactive terminal

psql

3.Create a database:

CREATE DATABASE your\_database\_name;

4.Create a user and set a password

CREATE USER your\_database\_user WITH PASSWORD 'your\_password';

5.Grant all privileges on your database to your user

GRANT ALL PRIVILEGES ON DATABASE your\_database\_name TO your\_database\_user;

6.Exit the PostgreSQL terminal

\q