Weather Dashboard Application

# Features

- Current Weather Information: Displays temperature, weather condition, humidity, wind speed, and precipitation for the selected city.  
- 5-Hour Forecast: Provides a forecast for the next 5 hours with temperature, weather condition, precipitation, and feels-like temperature.  
- Temperature Units: Users can select between Celsius, Fahrenheit, or Kelvin for temperature display.  
- Responsive Design: Works well on both desktop and mobile screens.  
- Dynamic Background: Changes the background based on the current weather condition.

# Design Choices

## Frontend Design

- Dynamic Weather Icons: Weather icons change according to the weather conditions (e.g., sun, clouds, rain).  
- Responsive Layout: Uses flexbox and grid systems for a fluid design that adjusts well to different screen sizes.  
- Elegant UI Components: Transparent panels with hover effects are used to create a modern and smooth user interface.  
- Background Image: The application uses dynamic background images to match the weather condition for a visually pleasing experience.

## Backend Design

- API Integration: The application uses the OpenWeatherMap API for fetching weather data, including current conditions and 5-hour forecasts.  
- Django Framework: The application is built on Django, leveraging Django's robust features for web development.  
- Scheduled Tasks: Background tasks are used for periodically fetching weather data and updating the database every 60 seconds.

# Dependencies

Before running the project, ensure that you have the following dependencies installed:  
  
1. Python 3.x: The application is built with Python 3, so make sure you have Python 3 installed.  
2. Django: Install Django to run the web application: pip install django  
3. Requests Library: Required to fetch weather data from the OpenWeatherMap API: pip install requests  
4. Background Task: For scheduling periodic tasks to fetch weather data: pip install django-background-tasks  
5. Weather Icons: The application uses weather icons (already included in the static folder), but you can update the icons by downloading the latest version from Weather Icons.

# Setup Instructions

## Step 1: Clone the Repository

Clone the GitHub repository to your local machine:  
git clone https://github.com/yourusername/weather-dashboard.git  
cd weather-dashboard

## Step 2: Set Up a Virtual Environment (Optional but Recommended)

Create a virtual environment to isolate dependencies:  
python -m venv myenv  
source myenv/bin/activate # For Windows: myenv\Scripts\activate

## Step 3: Install the Dependencies

Install all the required Python packages:  
pip install -r requirements.txt

## Step 4: Obtain an API Key

Sign up at OpenWeatherMap and obtain an API key.  
In views.py, update the API\_KEY variable with your OpenWeatherMap API key.

## Step 5: Run Database Migrations

Run the following command to create the necessary database tables:  
python manage.py migrate

## Step 6: Run the Application

Start the Django development server:  
python manage.py runserver

# How to Use

1. Select City: Use the dropdown in the top right to select a city (Delhi, Mumbai, Bangalore, etc.).  
2. Select Temperature Unit: Choose between Celsius, Fahrenheit, or Kelvin using the dropdown next to the city selector.  
3. View Current Weather: The current temperature, weather condition, humidity, wind speed, and precipitation will be displayed.  
4. View Forecast: The next 5 hours of forecast will be shown with temperature, weather condition, and feels-like temperature.

# Additional Notes

1. Dynamic Backgrounds: The background changes based on the current weather condition.  
2. Scheduled Tasks: The background tasks for fetching weather data run every 60 seconds. To start background tasks, run:  
**python manage.py process\_tasks**

# Troubleshooting

1. API Key Issues: If the weather data isn't loading, ensure your OpenWeatherMap API key is valid.  
2. Database Issues: If you encounter database errors, try clearing the existing data and re-running the migrations.  
3. Icons Not Displaying: Ensure that the static files (CSS and icons) are correctly linked by running:  
python manage.py collectstatic

**Historical Data and Chart Display**

An effort was made to implement a feature that retrieves and displays historical weather data using APIs, along with visualizing the information in charts for enhanced user experience. However, due to the limitations imposed by API pricing models and request rate limits, it was decided to refrain from fully integrating this functionality at this time. Future enhancements could include revisiting this feature when more cost-effective API options are available or if budgetary constraints allow for higher usage tiers.

# Future Enhancements

1. Extend Forecast: Add a 7-day forecast.  
2. User Alerts: Allow users to set weather alerts for specific conditions.  
3. Historical Data: Display historical weather data trends.