# Setup and Running Instructions for IELTS Coaching Project

A Step-by-Step Guide for First-Time Setup

Generated on May 23, 2025 at 10:11 PM IST

For assistance, contact the project maintainer.

## Contents

1 Introduction 2

2 Prerequisites 2

3 Backend Setup (`ielts_coaching`) 2

3.1 Navigate to the Backend Directory . . . 2

3.2 Create and Activate a Virtual Environment . . . 2

3.3 Generate and Install `requirements.txt` . . . 3

3.4 Configure the Backend Settings . . . 3

3.5 Run Migrations . . . 4

3.6 Create a Default User for Testing . . . 4

3.7 Start the Backend Server . . . 5

4 Frontend Setup (`ielts-frontend`) 5

4.1 Navigate to the Frontend Directory . . . 5

4.2 Install Node.js Dependencies . . . 5

4.3 Configure the Frontend . . . 5

4.4 Start the Frontend Server . . . 6

5 Running the Project 6

5.1 Start the Backend . . . 6

5.2 Start the Frontend . . . 6

5.3 Test the Application . . . 7

6 Troubleshooting 7

6.1 Backend Issues . . . 7

6.2 Frontend Issues . . . 7

7 Project Structure Overview 8

8 Notes 8

## 1 Introduction

This document provides detailed instructions to set up and run the IELTS Coaching project, which consists of a Django backend (`ielts_coaching`) and a React/Vite frontend (`ielts-frontend`). These steps are designed for first-time setup on a new system, including creating a virtual environment, installing dependencies, and configuring both the backend and frontend.

## 2 Prerequisites

Before proceeding, ensure the following are installed on your system:

- **Python** (version 3.9 or higher): For the backend.
- **Node.js** (version 16 or higher) and **npm** (version 8 or higher): For the frontend.
- **Git**: To clone the project repository (if applicable).
- **PostgreSQL** (optional): If using PostgreSQL instead of SQLite.
- A terminal or command-line interface (e.g., PowerShell on Windows, Terminal on macOS/Linux).
- A code editor (e.g., VS Code).

## 3 Backend Setup (`ielts_coaching`)

The backend is a Django REST Framework application located in the `C:\Ielts Website 1.0\ielts_coaching` directory.

### 3.1 Navigate to the Backend Directory

1. Open a terminal.
2. Navigate to the backend directory:

```
cd "C:\Ielts Website 1.0\ielts_coaching"
```

### 3.2 Create and Activate a Virtual Environment

A virtual environment isolates Python dependencies for the project.

1. **Create a virtual environment**:

```
python -m venv venv
```

   This creates a virtual environment named `venv` in the `ielts_coaching` directory.

2. **Activate the virtual environment**:
   - On Windows (PowerShell):

```
.\venv\Scripts\activate
```

   - On macOS/Linux:

```
source venv/bin/activate
```

After activation, your terminal prompt should change (e.g., `(venv) PS C:\Ielts Website 1.0\ielts_coaching>`).

### 3.3 Generate and Install `requirements.txt`

The backend requires several Python packages.

1. **Create `requirements.txt`**: Create a file named `requirements.txt` in the `C:\Ielts Website 1.0\ielts_coaching` directory with the following content:

   ```
   Django==4.2.11
   djangorestframework==3.15.1
   django-cors-headers==4.3.1
   djangorestframework-simplejwt==5.3.1
   psycopg2-binary==2.9.9  # For PostgreSQL (optional, remove if using SQLite)
   python-decouple==3.8  # For environment variables (optional)
   ```

2. **Install the dependencies**: With the virtual environment activated, run:

   ```
   pip install -r requirements.txt
   ```

### 3.4 Configure the Backend Settings

1. **Check `settings.py`**: Open `C:\Ielts Website 1.0\ielts_coaching\ielts_coaching\settings.py` and ensure the following configurations:

   - **CORS Settings**:

     ```
     INSTALLED_APPS = [
         ...
         'corsheaders',
         'rest_framework',
         'api',
     ]

     MIDDLEWARE = [
         ...
         'corsheaders.middleware.CorsMiddleware',
         'django.middleware.common.CommonMiddleware',
         ...
     ]

     CORS_ALLOWED_ORIGINS = [
         "http://localhost:3000",
         "http://localhost:5173",  # Vite's default port
     ]
     ```

   - **Database Settings**: For SQLite (default), no changes are needed. For PostgreSQL, update:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'ielts_db',
        'USER': 'your_postgres_user',
        'PASSWORD': 'your_postgres_password',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

If using PostgreSQL, create the database:

```
psql -U your_postgres_user -c "CREATE DATABASE ielts_db;"
```

### 3.5 Run Migrations

1. **Apply migrations**:

   ```
   python manage.py makemigrations
   python manage.py migrate
   ```

2. **Create a Superuser (Admin)**:

   ```
   python manage.py createsuperuser
   ```

   Follow the prompts to set a username, email, and password (e.g., `username: admin`, `email: admin@example.com`, `password: admin123`).

### 3.6 Create a Default User for Testing

1. **Using the Django Shell**:

   ```
   python manage.py shell
   ```

   In the shell, run:

   ```
   from api.models import User
   user = User.objects.create_user(username='gunjans', password='gunjans123', role='studen
   user.save()
   exit()
   ```

2. **Using the Admin Panel (Alternative)**:

   - Start the Django server:

     ```
     python manage.py runserver
     ```

   - Open `http://127.0.0.1:8000/admin` in a browser.
   - Log in with the superuser credentials.
   - Add a new user with `username: gunjans`, `password: gunjans123`, `role: student`.

### 3.7 Start the Backend Server

```
python manage.py runserver
```

The server should start at http://127.0.0.1:8000/. Keep this terminal running.

## 4 Frontend Setup (ielts-frontend)

The frontend is a React application using Vite, located in the C:\Ielts Website 1.0\ielts-frontend directory.

### 4.1 Navigate to the Frontend Directory

1. Open a new terminal.
2. Navigate to the frontend directory:

   ```
   cd "C:\Ielts Website 1.0\ielts-frontend"
   ```

### 4.2 Install Node.js Dependencies

1. **Install dependencies**:

   ```
   npm install
   ```

   This installs packages like react, react-router-dom, axios, vite, and tailwindcss.

### 4.3 Configure the Frontend

1. **Check Tailwind CSS Setup**: Ensure C:\Ielts Website 1.0\ielts-frontend\tailwind.config.js is configured:

   ```
   /** @type {import('tailwindcss').Config} */
   export default {
     content: [
       "./index.html",
       "./src/**/*.{js,ts,jsx,tsx}",
     ],
     theme: {
       extend: {
         colors: {
           brandRed: '#FF0000',
           brandCream: '#FFFDD0',
         },
       },
     },
     plugins: [],
   }
   ```

   Ensure src/index.css includes:

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

2. **Verify API Base URL**: Ensure API calls match the backend server (e.g., in `LoginPage.jsx`):

```
const response = await axios.post("http://127.0.0.1:8000/api/login/", { username, passw
```

### 4.4 Start the Frontend Server

```
npm start
```

The frontend should start at http://localhost:3000/ or http://localhost:5173/. Open the URL in a browser.

## 5 Running the Project

### 5.1 Start the Backend

1. Navigate to the backend directory:

```
cd "C:\Ielts Website 1.0\ielts_coaching"
```

2. Activate the virtual environment:

    - On Windows:

    ```
    .\venv\Scripts\activate
    ```

    - On macOS/Linux:

    ```
    source venv/bin/activate
    ```

3. Start the Django server:

```
python manage.py runserver
```

### 5.2 Start the Frontend

1. Open a second terminal and navigate to the frontend directory:

```
cd "C:\Ielts Website 1.0\ielts-frontend"
```

2. Start the Vite server:

```
npm start
```

### 5.3 Test the Application

1. **Access the Homepage**: Open `http://localhost:3000/` (or `http://localhost:5173/`). You should see the homepage with a "Login | Sign up" button.

2. **Log In**:

   - Click "Login | Sign up" to go to `/login`.
   - Log in with:
     - Username: `gunjans`
     - Password: `gunjans123`
   - You should be redirected to `/student-dashboard`.

3. **Test Other Features**:

   - Register a new user at `/register`.
   - Access the admin dashboard by logging in with the superuser credentials at `/admin-dashboard`.
   - Test teacher and student dashboards by creating users with appropriate roles.

## 6 Troubleshooting

### 6.1 Backend Issues

- **Port Already in Use**: Start the Django server on a different port:

  ```
  python manage.py runserver 8001
  ```

  Update the frontend API URLs to match (e.g., `http://127.0.0.1:8001/api/login/`).

- **CORS Errors**: Ensure `CORS_ALLOWED_ORIGINS` includes the frontend URL:

  ```
  CORS_ALLOWED_ORIGINS = [
      "http://localhost:3000",
      "http://localhost:5173",
  ]
  ```

- **Database Errors**: If migrations fail, delete the database file (e.g., `db.sqlite3`) or drop the PostgreSQL database, then rerun:

  ```
  python manage.py makemigrations
  python manage.py migrate
  ```

### 6.2 Frontend Issues

- **Port Already in Use**: Specify a different port:

  ```
  npm start -- --port 3001
  ```

  Update `CORS_ALLOWED_ORIGINS` in the backend to include the new port.

- **API Request Failures**: Ensure the backend server is running and API URLs match the backends address.
- **Styling Issues**: If Tailwind CSS styles arent applying, run:

  ```
  npx tailwindcss -i ./src/input.css -o ./src/output.css --watch
  ```

  Update `index.html` to include `output.css`.

## 7 Project Structure Overview

- **Backend (`C:\Ielts Website 1.0\ielts_coaching`)**:
  - `manage.py`: Django management script.
  - `ielts_coaching/`: Django project settings.
  - `api/`: Django app with models, views, and serializers.
  - `venv/`: Virtual environment directory.
  - `requirements.txt`: Python dependencies.
- **Frontend (`C:\Ielts Website 1.0\ielts-frontend`)**:
  - `src/`: React components, pages, and styles.
  - `package.json`: Node.js dependencies and scripts.
  - `node_modules/`: Installed Node.js packages.
  - `tailwind.config.js`: Tailwind CSS configuration.

## 8 Notes

- Always activate the virtual environment before running the backend.
- Keep both the backend and frontend servers running in separate terminals.
- If you make changes to the backend models, rerun migrations:

  ```
  python manage.py makemigrations
  python manage.py migrate
  ```

- Check terminal output and browser console for error messages.