Project Title

Employee Management API using Django REST Framework

Problem Statement

Managing employee data manually or using unstructured systems leads to inefficiency, data inconsistency, and security issues.
This project provides a **secure, RESTful backend API** to manage employee records efficiently with authentication and CRUD operations.

Tech Stack

- **Backend Framework:** Django
- **API Framework:** Django REST Framework (DRF)
- **Authentication:** JWT (Simple JWT)
- **Database:** SQLite (default)
- **Language:** Python 3
- **Tools:** Git, GitHub, VS Code, Postman

Features

- Employee CRUD operations (Create, Read, Update, Delete)
- Secure authentication using JWT
- Token refresh mechanism
- RESTful API structure
- Clean and scalable project architecture
- Ready for frontend or mobile app integration

API Endpoints

- POST /api/employees/ – Create employee
- GET /api/employees/ – List employees
- GET /api/employees/{id}/ – Retrieve employee
- PUT /api/employees/{id}/ – Update employee

- DELETE /api/employees/{id}/ – Delete employee
- POST /api/token/ – Get JWT token
- POST /api/token/refresh/ – Refresh JWT token

Setup Steps

1. **Clone the repository**

git clone https://github.com/Thamodharans/employee-management-api.git

2. **Navigate to project folder**

cd employee-management-api

3. **Create virtual environment**

python -m venv venv

4. **Activate virtual environment**

venv\Scripts\activate

5. **Install dependencies**

pip install -r requirements.txt

6. **Run migrations**

python manage.py migrate

7. **Start development server**

python manage.py runserver

🔗 GitHub Repository Link

👉 https://github.com/Thamodharans/employee-management-api

Conclusion

This Employee Management API provides a secure and scalable backend solution for managing employee data and can be easily extended with frontend applications or additional features.