Developed by: **Kaiwalya Kshirsagar** LinkedIn: https://www.linkedin.com/in/kaiwalyakshirsagar/

# Project: Currency Exchange

This project is a backend system for managing currency exchange rates. It provides APIs for CRUD operations, currency conversion, and historical rate management, with asynchronous task handling using Celery.

---

**Features**

- **Currency Management**: CRUD operations for currencies.
- **Exchange Rate Management**: Fetch and manage exchange rates dynamically.
- **Currency Conversion**: Real-time conversion with caching support.
- **Historical Rate Management**: Fetch historical exchange rates using external APIs.
- **Asynchronous Processing**: Celery integration for background tasks.
- **Dockerized Deployment**: Run the project with Docker Compose.

---

**Setup Instructions**

**Using Docker**

1. Build and start the services:
2. docker-compose up --build
3. Access the application:
   - App: http://localhost:8000
   - Admin: http://localhost:8000/admin
4. Start a Celery worker:
5. docker-compose exec celery celery -A mycurrency worker --loglevel=info

---

**Without Docker**

1. Clone the repository or extract the zip file.
2. Create a virtual environment and install dependencies:
3. python -m venv venv
4. source venv/bin/activate # On Windows: venv\\Scripts\\activate
5. pip install -r requirements.txt
6. Set up environment variables for sensitive data:
7. export CURRENCYBEACON_API_KEY=your_api_key

8. Run migrations and start the server:
9. python manage.py migrate
10. python manage.py runserver
11. Access the admin interface at /admin (create a superuser using python manage.py createsuperuser).

---

**API Endpoints**

| **Endpoint** | **HTTP Method** | **Description** |
|---|---|---|
| /api/currency/ | GET | List all currencies. |
| /api/currency/ | POST | Create a new currency. |
| /api/currency/<int:pk>/ | GET | Retrieve details of a specific currency by ID. |
| /api/currency/<int:pk>/ | PUT/PATCH | Update a specific currency by ID. |
| /api/currency/<int:pk>/ | DELETE | Delete a specific currency by ID. |
| /api/exchange-rate/ | GET | Fetch exchange rates by source currency and date range. |
| /api/convert/ | GET | Convert an amount between two currencies. |

---

**Example Requests**

1. **Fetch Exchange Rates**
2. GET /api/exchange-rate/?source_currency=USD&date_from=2023-01-01&date_to=2023-01-31
3. **Convert Currency**
4. GET /api/convert/?source_currency=USD&target_currency=EUR&amount=100

---

**File Structure and Functions**

**Core Application Files**

| **File** | **Purpose** |
|---|---|
| models.py | Defines database models (Currency and CurrencyExchangeRate). |
| serializers.py | Serializes models into JSON and validates API inputs. |
| views.py | Implements API endpoints for currency management, exchange rate retrieval, and conversion. |

| **File** | **Purpose** |
|---|---|
| urls.py | Maps URLs to views for routing. |
| tasks.py | Defines asynchronous tasks for fetching historical rates. |
| adapters.py | Handles integration with external APIs (e.g., CurrencyBeacon) for exchange rate data. |
| admin.py | Customizes the admin interface for managing currencies and exchange rates. |
| celery.py | Configures Celery for task queue handling. |

---

**Supporting Files**

| **File** | **Purpose** |
|---|---|
| .gitignore | Specifies files and directories to ignore in version control. |
| Dockerfile | Defines the environment for running the Django app in Docker. |
| docker-compose.yml | Orchestrates multiple services, including the Django app, Redis, and Celery workers. |
| manage.py | Entry point for running Django commands. |
| README.md | Documentation for setup, usage, and project details. |

---

**Environment Variables**

| **Variable** | **Description** |
|---|---|
| CURRENCYBEACON_API_KEY | API key for the CurrencyBeacon provider. |
| SECRET_KEY | Django secret key for encryption. |

---