House Renting Platform Documentation

Table of Contents

1. Key Features

1. Secure User Authentication and Role Handling: Each user can register and act as both a property owner and a renter.

2. Property Listings Management: Allows property owners to post detailed property listings including type, description, price, and image.

3. Rental Request System: Enables users to send rental requests to property owners with a custom message and budget proposal.

4. Request Review and Decision: Owners can accept or reject rental requests. Status updates reflect the availability of the property.

5. Integrated Payment System: Payments are processed using Chapa for property posting and transaction finalization.

6. Property Search & Filtering: Renters can use multiple filters to find properties matching their requirements.

7. Feedback and Messaging: Users can submit feedback or questions to the platform admin for assistance.

2. Technologies Used

- Backend: Django (Python Framework)
- Frontend: HTML5, CSS3, JavaScript, Bootstrap
- Database: SQLite (local dev)
- Payment Integration: Chapa API

3. Installation Instructions

1. Clone the Repository:
git clone https://github.com/ABELMATHIOS/House-Renting-Platform.git
cd House-Renting-Platform

2.Create python environment

Python -m venv env

3. activate the env

env/Scripts/activate

4.install Dependencies:
pip install -r requirements.txt

5. Change directory to house renting platform folder

cd House_Renting_Platform

6.Start the development server

python manage.py runserver

7.Go to local host port 8000

https://localhost/8000

4. Code Structure Overview

The project uses Django architecture. Controllers manage logic, models define database schemas. Routing is defined in `urls.py`, while configuration files are stored in the `settings.py` Files.

5. Detailed File/Directory Descriptions

- **app/models.py: Contains models like `User`, `Property`, and `Reviews and Rating`. These represent tables in the database and define relationships.**

- **app/urls.py: Defines the routes used by the web interface (e.g., '/', '/properties', '/about-us'). Routes are grouped and often middleware-protected.**

- **app/views.py: Html templates for rendering UI. Pages like login, registration, property list, and payment live here.**

- **settings.py: Application config files, including payment keys if mapped in `.env`.**

- **static: Static assets like images, CSS, and JS files.**

- **.env: Stores environment-specific configurations such as database connection and Chapa keys.**

6. User Guide

A. Authentication

Users can register using the signup form which collects their name, email, and password. After registration, users log in to access property posting or browsing functionality.

B. Posting and Managing Property Listings

Owners can post new properties by navigating to the 'Add Property' section. Required fields include title, description, type (e.g., apartment), price, location, and an image. Owners can edit or delete their listings as needed.

C. Searching and Filtering Listings

Users can view the full list of available properties on the homepage. The search feature allows filtering by location, type, and price range.

D. Sending and Managing Rental Requests

Interested renters can send a request via a Phone number on the property's page. The request includes an optional message and a proposed rent amount. Property owners receive these requests in their dashboard and can accept or reject them.

E. Request Tracking and Deletion

Users can track all their sent requests and delete any pending ones from their dashboard.

F. Recommendation system

When users view a property they like other similar listed properties will be recommended to the user. This feature is built by using langchain python framework.

7. Payment Integration Flow

The payment system allows property owners to pay a listing fee before their property goes live. The system integrates Chapa for secure transactions.

Flow:
1. Owner submits property listing form, if it is valid.
2. Redirected to Chapa payment page.
3. Payment verified via callback/webhook.
4. On success, will be redirected to the listing and submits the form,then property is listed.

Security:
- All sensitive payment keys are stored in the `.env` file.
- Django validation and middleware protect payment endpoints.
- Payment status is logged in the database for traceability.