**Real Estate Property Listing Platform- Design**

Here is outline of the core components to develop for the property listing platform:

**1. Core Features**

1. **User Management**
   * **Signup/Login**: Users can register, log in, and manage their profiles.
   * **Roles**: Various roles like buyer, seller, and admin with specific permissions.
   * **Saved Searches & Favourites**: User can save or bookmark property listings.
2. **Property Listings**
   * **Operations**: Sellers can create, update, and delete property listings.
   * **Attributes**: Each property includes fields like title, description, price, location, size, images and amenities.
   * **Images**: Allow multiple high-resolution images per listing with image optimization.
3. **Search Functionality**
   * **Searching Capabilities**: Search by keywords, location, or property type.
   * **Filter Criteria** : Price range, number of rooms, size, etc.
   * **Sorting Methods**: Sort by price, area, or date added.
   * **Mapping**: Integration with mapping APIs (e.g., Google Maps) for spatial searches.
4. **Interaction Features**
   * **Contact Forms**: Buyers can contact sellers directly from the listing page.
   * **Reviews and Ratings**: Users can rate properties.
   * **Chat feature**: In-app messaging for buyer-seller communication.
5. **Admin Panel**
   * **Moderation**: Review and approve listings.
   * **Analytics**: Track user activity, popular searches, and other metrics.
   * **Reports**: Handle user-submitted issues or spam reports.

**2. Backend Architecture**

1. **Tech Stack**
   * **Framework**: Django
   * **Database**: PostgreSQL for its spatial data capabilities.
   * **Search Engine**: Elastic Search/Semantic Search for efficient search.
2. **APIs**
   * RESTful APIs for interaction between the frontend and backend.
   * External integrations (e.g., payment gateways, Map APIs).
3. **Task Queue**
   * Celery + Redis for asynchronous tasks like sending emails or processing images.

**3. Frontend Architecture**

1. **Framework**:
   * Streamlit for rapid prototyping or React for a robust UI.
2. **Key Pages**:
   * Homepage: Featured listings and search bar.
   * Search Results: List and map view of properties.
   * Property Details: Full property description, images, and seller details.
   * User Dashboard: Manage favourites, saved searches, and listings.

**4. Features for Scalability**

1. **Caching**: Use Redis or Aws DynamoDB accelerator(DAX) for caching frequent queries.
2. **File Storage**: Use AWS S3 or another object storage for uploaded images.
3. **Load Balancing**: Aws Load Balancer for handling high traffic.

**5. Enhancements**

1. **Authentication**: Add authentication using django rest framework-JWT.
2. Replace JSONField for amenities with a relational model if required .
3. **Searching**: Use Elasticsearch or Postgres full-text search for advanced querying.