

DevEstate

Adeeb Khan, John Chandler III, Jason Lei, Samuel Li



Design Document Specifications:

- Project name.
 - A list of applications you will implement.
 - The components and actions that will be developed in each application.
 - Explanation of how each component relates to the others.
 - Database design. Diagram showing tables, fields, PKs, FKs, indexes, and relationships between tables.
 - Tentative site map for front-end diagram (See info and example here: <https://www.outofsightdesigns.com/what-is-a-sitemap/>)
 - Represent each page you envision for your site.
 - Show linkages conveying all possible pathways for a user traversing your site.
 - A breakdown of the different tasks required to complete this project.
- Include assignments of each task to each group member.
- Clearly labeled section delineating APIs you will use.
 - This document should be a PDF file. You will add it to your GitHub repo in the designated location (see below), and you must hand in a hard copy (see checkpoints deliverables)

- CReate, Update, Delete (CRUD)
- Implement Zillow API to provide accurate and meaningful information regarding home ownership
- Users will be able to keep track of Properties which they've liked
- [SITEMAP](#)
- Our Service will feature five main tables. They are as follows:
 - Property Table
 - Realtor Table
 - Property/Agent joined Table
 - Property/School joined table

1. Property Table

This table will store the main details about each property. It will have the following fields:

- PropertyID: INTEGER (Primary Key)
- StreetAddress: VARCHAR
- City: VARCHAR
- State: VARCHAR
- Zipcode: VARCHAR
- Country: VARCHAR
- Latitude: DECIMAL
- Longitude: DECIMAL
- LivingArea: INTEGER
- Bedrooms: INTEGER
- Bathrooms: INTEGER
- YearBuilt: INTEGER
- Price: DECIMAL
- DatePosted: TIMESTAMP
- DateSold: TIMESTAMP
- HomeType: VARCHAR
- PropertyTaxRate: DECIMAL
- TimeOnZillow: INTEGER
- HomeStatus: VARCHAR
- AnnualHomeownersInsurance: DECIMAL
- RentZestimate: DECIMAL
- BrokerageName: VARCHAR
- PageViewCount: INTEGER
- Description: TEXT

2. PropertyFeatures Table

Stores specific features of each property. For example, some fields could be:

- FeatureID: INTEGER (Primary Key)
- PropertyID: INTEGER (Foreign Key)
- Flooring: VARCHAR
- FoundationDetails: VARCHAR
- AccessibilityFeatures: VARCHAR
- GarageSpaces: INTEGER
- ParkingSpaces: INTEGER
- ViewType: VARCHAR
- WaterView: BOOLEAN
- Heating: VARCHAR
- Cooling: VARCHAR
- ConstructionMaterials: VARCHAR
- RoofType: VARCHAR
- LotSize: DECIMAL
- HOAFee: DECIMAL

3. Agent Table

Stores information about real estate agents. Some features will be:

- AgentID: INTEGER (Primary Key)
- DisplayName: VARCHAR
- ReviewCount: INTEGER
- RatingAverage: DECIMAL
- PhoneNumber: VARCHAR
- ImageURL: VARCHAR
- BadgeType: VARCHAR

4. PropertyAgent Table

This is a junction table to handle the many-to-many relationship between properties and agents.

Fields:

- PropertyAgentID: INTEGER (Primary Key)
- PropertyID: INTEGER (Foreign Key)
- AgentID: INTEGER (Foreign Key)

5. School Table

Stores information about schools in the vicinity of the property.

Fields:

- SchoolID: INTEGER (Primary Key)
- Name: VARCHAR
- Rating: INTEGER
- StudentsPerTeacher: INTEGER
- Size: INTEGER
- Level: VARCHAR
- Grades: VARCHAR
- Type: VARCHAR
- Distance: DECIMAL

6. **PropertySchool Table**

This is a junction table for the many-to-many relationship between properties and schools.

Fields:

- PropertySchoolID: INTEGER (Primary Key)
- PropertyID: INTEGER (Foreign Key)
- SchoolID: INTEGER (Foreign Key)

7. **PriceHistory Table**

Stores historical pricing information for each property.

Fields:

- PriceHistoryID: INTEGER (Primary Key)
- PropertyID: INTEGER (Foreign Key)
- Date: TIMESTAMP
- Event: VARCHAR
- Price: DECIMAL
- PricePerSquareFoot: DECIMAL

8. **NearbyHomes Table**

Information about homes nearby the property.

Fields:

- NearbyHomeID: INTEGER (Primary Key)
- PropertyID: INTEGER (Foreign Key)

- ZPID: INTEGER
- Longitude: DECIMAL
- Latitude: DECIMAL
- Price: DECIMAL
- HomeType: VARCHAR
- HomeStatus: VARCHAR

Relationships

Each Property has many PropertyFeatures.

- Real estate properties can have a wide array of features, such as different types of flooring (e.g., hardwood, tile), various amenities (e.g., pool, fireplace), and diverse utility setups (e.g., types of heating and cooling systems). These features are too numerous and varied to be efficiently stored in a single column of a property table.

Each Property can be associated with multiple Agents and vice versa.

Each Property can be associated with multiple Schools and vice versa.

Each Property has many entries in the PriceHistory table.

Each Property has many NearbyHomes.

Primary keys (PK) are unique identifiers for each table.

Foreign keys (FK) establish relationships between tables.

Junction tables are used to handle many-to-many relationships.

SQL ER Diagram

