****

**Web Development Project – A Real Estate Website**



**MILESTONE #2 DOCUMENT**

(Detailed Requirements, UI Mock-Ups, Architecture)

**Local Group #22**

**Team Members:**

Akhila Cyriac Panackal

Aleksandar Mitkovski

Mahdis Lohrasebynejad

Mahbubur Rahman

Swetaketu Majumder

Rohan Deo

**Winter Semester 2017, Fulda**

# Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Version Number** | **Revision Date** | **Revision Description** |
| 1 | Milestone2, Version 1.0 | 06-11-2017 | Added Milestone2 document sections. |
| 2 | Milestone2, Version 1.1 | 07-11-2017 | Added Data Definitions V2. |
| 3 | Milestone2, Version 1.2 | 08-11-2017 | Added Functional Requirements V2 |
| 4 | Milestone2, Version 1.3 | 10-11-2017 | Added UI Mockup and Storyboards |
| 5 | Milestone2, Version 1.4 | 13-11-2017 | Added High Level Architecture and UML Diagrams. |
| 6 | Milestone2, Version 1.5 | 14-11-2017 | Added Key Risks |
| 7 | Milestone2, Version 2.0 | 16-11-2017 | Milestone 2 ready for submission |

Table of Contents

[Version Control 2](#_Toc498611769)

[1. Use Cases: 4](#_Toc498611770)

[2. Data Definition: 6](#_Toc498611771)

[3. Functional Requirements: 11](#_Toc498611772)

[4. UI Mock Ups and Storyboards: 14](#_Toc498611773)

[5. High Level Architecture and Database Organization 27](#_Toc498611774)

[6. High Level UML Diagrams 11](#_Toc498611775)

[7. Key Risks 16](#_Toc498611776)

# 1. Use Cases:

**1.1.** ***Listing Agent posts properties for sale***

Company SFStateHomes recently completed construction of new apartments near the university and would like to **sell** these apartments with their **Listing Agents**.

Major steps identified are:

1. Company admin **creates an account** and the **company profile** if company is new to the application.
2. Company admin **creates listing agents** and listing agents log in to their accounts and create new **properties** in the application and **add** all new property details.
3. Properties are available to all users.

|  |  |
| --- | --- |
| **Use Case:** | Listing Agent posts listing for sale on website |
| **Actor**: | User (Listing Agent) |
| **Description:** | Company SFStateHomes Listing Agents wants to sell their apartments by creating a property advertisement and post them on the website and make it available to all users |
| **Preconditions:** | Listing Agents must be created by Company admin |
| **Normal flow:** | System receives request and processes it. Records will be inserted to the Database. Property will be displayed on website. |

**1.2. *Customer wants to buy their preferred apartment***

An elderly couple (Mr. and Mrs. Müller) decided to move to a different city with better healthcare facilities. They would like to **buy an apartment** that is suitably located and elderly friendly. Major steps identified are:

1. Mrs.Müller **creates an account** (if she doesn't have any) or **logs in to the system.**
2. From all the properties available she would narrow down her **search results** to find one apartment near a hospital.
3. Once she finds the right apartment for her, she would **contact the listing agent** using the **messaging service** of the application to finalize the deal.

|  |  |
| --- | --- |
| **Use Case:** | Listing Agent posts listing for sale |
| **Actor**: | User (Customer) |
| **Description:** | User searches with his predefined conditions for an apartment and creates an account to be able to buy it. To buy the property a listing agents has to be contacted to finalize the deal. |
| **Preconditions:** | User need an account or log in via Google or Facebook |
| **Normal flow:** | Enter keywords and search. Request goes to the database and the database returns the requested list of properties. User clicks on preferred property. Database return all the details for the property. Property opens in a different window for display. User can use the contact form (provided in the detailed property page). This message goes to listing agents and can also be seen via the user’s dashboard. |

**1.3.** ***User browses, searches and filters among all listed properties for buying***

Aleks moves to a city for his job. He wants to **buy an apartment** and wants to be sure that the apartment has **stable internet**. Major steps identified are:

1. Aleks **creates** an account (if he doesn't have any) or logs in to the system.
2. From all the **properties**, available he would narrow down his **search results** with all the **amenities** he needed also **sorts** in such a way it's giving cheapest one first.

|  |  |
| --- | --- |
| **Use Case:** | Search for a property with specific requirements |
| **Actor**: | User (Visitor) |
| **Description:** | User wants to find place of interests by his chosen predefined conditions and sort them |
| **Preconditions:** | Must visit our website |
| **Normal flow:** | Enter search keyword for property, choose filters, click search. Records will be displayed to the database. Now can apply additional filters. |

**1.4.** ***Company wants to track sales for the last quarter***

**Company** wants to **track the sale** for the last quarter.

1. Company’s Admin **logs** in to our application.
2. In **the dashboard**, the **administrator** will be able to see a **visual representation** of the sales for the last quarter.

|  |  |
| --- | --- |
| **Use Case:** | Tracking sales for companies |
| **Actor**: | User (Company Admin) |
| **Description:** | Company Admin logs to our website to get access to his dashboard. Previous sale view is available for him to see. |
| **Preconditions:** | Company Admin has to register and log in |
| **Normal flow:** | Dashboard showing sales view which has been saved in the database. |

# 2. Data Definition:

1. **Real Estate Property:**

Real estate property is the main data element of this web application. Almost every data flow is related to this element. Property has 03(Three) categories. They are:

* 1. **Residential:**

Residential property is used for living purpose only. There are few types of residential property that are given bellow:

* + 1. **House:**

The house is an abstract object and describes the relevant structural characteristics of the offered property. Maybe also features like images and custom properties for further object presentation (Example: room sizes, number of rooms, interior features, front and back yard facilities etc)

* + 1. **Apartment:**

The apartment or its specialization as a shared room is an abstract object and describes the relevant structural characteristics of the offered property. Maybe also features like images and custom properties for further object presentation (Example: Internet availability and animal friendly etc.)

* + 1. **Town home:**

The town home is an abstract object and describes the relevant structural characteristics of the offered property that located in city area or in the city centre.

* + 1. **Land / Lots:**

The land / lots is an abstract object and describes the relevant structural characteristics of the offered property that is undeveloped and free.

* 1. **Commercial:**

Commercial property is used for business purpose. There are few types of commercial property that are given below:

* + 1. **Office buildings:**

The office buildings are an abstract object and describes the relevant structural characteristics of the offered property that are used as office and business meeting, conference places.

* + 1. **Retail store buildings:**

The retail store buildings are an abstract object and describes the relevant structural characteristics of the offered property that are used super markets, shopping mall, shops, show rooms etc.

* 1. **Industrial:**

Industrial property is used for production purpose. There are few types of industrial property that are given bellow:

* + 1. **Factories:**

The factories are an abstract object and describes the relevant structural characteristics of the offered property that are used to produce different products etc.

* + 1. **Mines:**

The mines are an abstract object and describes the relevant structural characteristics of the offered property that are used collect different types of metal from ground.

* + 1. **Farms:**

The mines are an abstract object and describes the relevant structural characteristics of the offered property that are used collect different types of metal from ground.

1. **Real Estate Professional:**

Real estate professionals are one of the main actors of this web application. They are some skilled and experienced persons who are waiting to help their clients. There are some types of real estate professionals given bellow:

* 1. **Property Agent:**

Property agent is skilled broker who helps customers in buying or selling their any kind of real estate properties. This agent has some unique specialties. There are three types of agents in our marketplace. They are:

* + 1. **Individual:**

Individual agents have their own license from government authority.

**2.1.2 Team:**   
Team is consisting of two or more agents where team is operated by team lead and each team is situated within a company. Company has license and teams are using it.

**2.1.3 Both:**

Some agents can be act as individual and a team member.

* 1. **Property Improvement:**

Property improvement professionals are ready to help their clients in improving their properties.

* 1. **Property Manager:**

Property manager professionals are ready to help their clients in renting, mortgage properties.

* 1. **Property Builder:**

Property builder professionals are ready to help their clients in giving innovative building ideas and pathways to build new constructions in land/lots.

* 1. **Property Inspector:**

Property inspector professionals are ready to help their clients in inspecting their properties to evaluate loan and mortgage value.

* 1. **Property Photographer:**

Property photographer professionals are ready to help their clients in taking clear 2D, 3D and 360-degree views of their properties.

1. **Customer:**

Customer is one of main actors of this web application. There are several types of customer. They are as follows:

* 1. **Buyer:**

Buyer is willing to buy real estate properties.

* 1. **Owner / Landlord:**

Owner / Landlord is such kind of customers who wants to sell or mortgage or give rent their properties through real estate professionals.

**3.3 Renter:**

Renter is willing to take rent properties from owner / landlord. Renter is able to give feedbacks on rental properties so that next renter can get a overview on the property's actual condition and owner attitude.

* 1. **Lessee:**

Lessee can take lease of any properties from owner / landlord through real estate professionals. Same as renter, lessee also can give feedbacks on real estate properties.

Here renter is associated residential real estate properties and on the other hand lessee is associated with commercial and industrial real estate properties.

Any kind of customer has their own profile, account settings, subscription list. After successful registration into the website, a customer can edit his / her profile information, account settings and also subscription list.

In the profile, the following fields should be provided:

Profile picture

Title

First Name

Last name

Gender

Email address

Phone number

Postal Address

House / Flat / Room Number

Street

Plz

City

State

Country

Here, for profile picture, we will prefer JPEG format for better result, the maximum image upload size should be 2 MB and the profile picture area should be 160px - 180px square. So, image must be uploaded at least 180px square and. Customer can also upload a larger image, preferably in a multiple of 180, such as 360, 540, or 720, and our system will resize it for customer. That means the system will always convert the raw image into thumbnail image and save the thumbnail image into server. For customer profile picture, system will always use thumbnail image.

The saving image in database is expensive in terms of space and performance than saving in file system. So, the image will be saved in server file system and image path will be saved in database.

Postal Address is necessary for future communication such as to send hard copy of land documents, rental / lease agreement, buy / sell deeds etc.

In the account settings section, customer can change his / her password, security question and answer, some account related settings such as “allow other user to contact with me”, “allow only agents to contact with me” etc.

In the subscription list, there will be a mailing list of different offers and newsletters, notifications, listing reports etc.

1. **Favourites:**

A customer or a listing agent can add a property to their “Favourites” list, which is a list of properties in which the customer/agent is interested. This list is saved with their account and can be retrieved later.

1. **Property Review / Feedback:**

A customer can provide a feedback and can rate a specific property. It contains customer id and property id as database table column.

# 3. Functional Requirements:

**Priority 1 (must have):**

**2.1. Property Finder (Every user type)**

2.1.1 Every User shall be able to search through the given

property.

2.1.2 Search can be done by a free text search filter.

2.1.3 User can open detailed view of a property by clicking on

it.

**2.2. Registration:**

2.2.1 Customer Registration Version1: will be based on Username, Email Secret Question, and Secret Answer (for security matters).

2.2.2 Customer Registration Version 2: will be based on existing

Google or Facebook Accounts

2.2.2 Every visitor can register and become a registered user.

2.2.3 Customers must submit all necessary information for registration.

2.5.6 Company Admin creates company profile.

2.5.6 Listing Agents must be created be Company Administrator.

**2.3 User Login:**

2.3.1 After successful registration, the user can log in by using his email address or username and chosen password.

2.3.2 Customers can also use Google and Facebook API to log in

* 1. **User (All types) Logout:**

2.4.1 When the user wants to leave our application he can press the log

out button and will be logged out.

* 1. **Registered Customer can add and manage his favourite properties:**

2.5.1 Registered Customer can add desired properties to his favourites.

2.5.2 Selected favourites can be seen on customer dashboard and

also deleted if desired

* 1. **Messaging between Buyers and Sellers:**

2.6.1 Buyers and sellers can communicate within their dashboard, using a messaging system.

* 1. **Property tracking**

2.7.1 Enable companies and listing agents to track their sales

* 1. **Manage Profile**

2.8.1 User can see his profile data and edit it if needed

* 1. **Task of Site Administrator**

2.9.2 Site Administrator can delete users permanently. The Users data

will be completely deleted in the database.

2.9.3 Site Administrator can remove properties form the platform

for legitimate reason.

**3.0 Manage Dashboard**

Depending on the type of user, dashboard provides specific functionalities

3.0.1 For Customers:

Messaging listing agents, viewing his favourite

Properties, compare his favourite properties and edit the Customers profile.

3.0.2 For Listing Agents:

Messaging customers, viewing his sales, create property listing and edit the Listing Agents profile.

Property listing can be viewed with Google

Map API.

3.0.3 For Site Administrator:

Monitor and remove property listings and users.

**Priority 2 (desired):**

* 1. **Properties for rent**
     1. Listing Agents will be able to list properties for rent, which can be selected by customers
  2. **Feedback and Rate**

3.2.1 User can give feedback and Rate on property rent

* 1. **Rating on Listing Agents**

3.3.1 User can give Ratings on Listing Agents

**3.4 Compare User Favourite**

3.4.1 User can compare his chosen favourite properties

**Priority 3 (opportunistic):**

* 1. **Include support for commercial and industrial properties**
     1. Customer and Listing Agents can buy/sell/rent commercial and industrial properties
  2. **Include support for multiple types of Real Estate Agents**
     1. Companies can list buying agents and specialized agents for lease or mortgage.

# 4. UI Mock Ups and Storyboards:

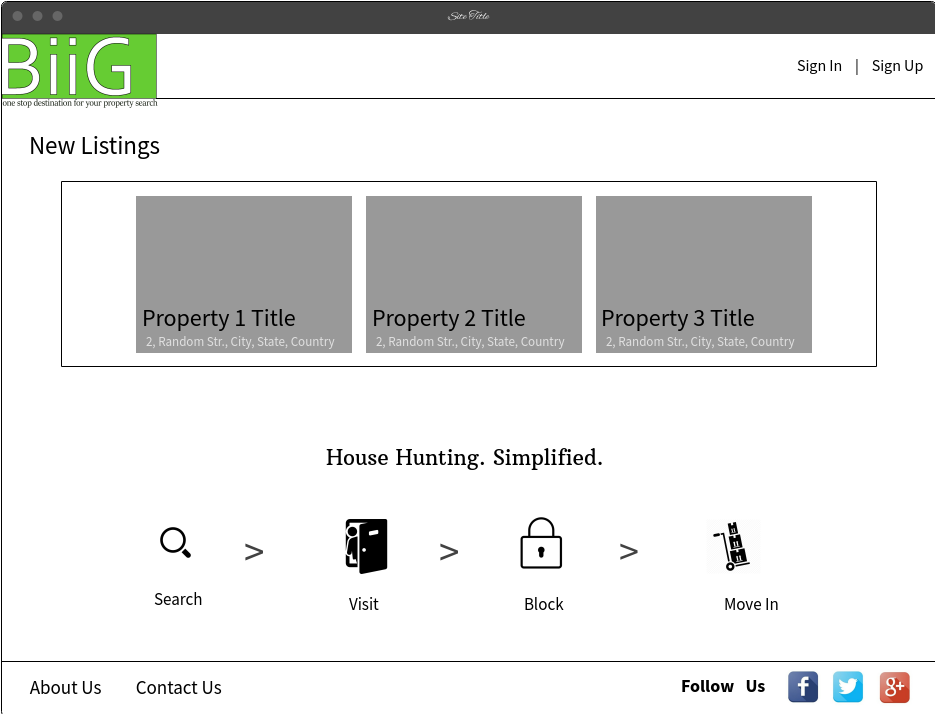
**4.1 Homepage**

For any user landing page of the application is Home Page. Part 1 of Home page is initially displayed to the user and once the user scrolls down Part2 of Home page is displayed.

4.1.1 Homepage Part1



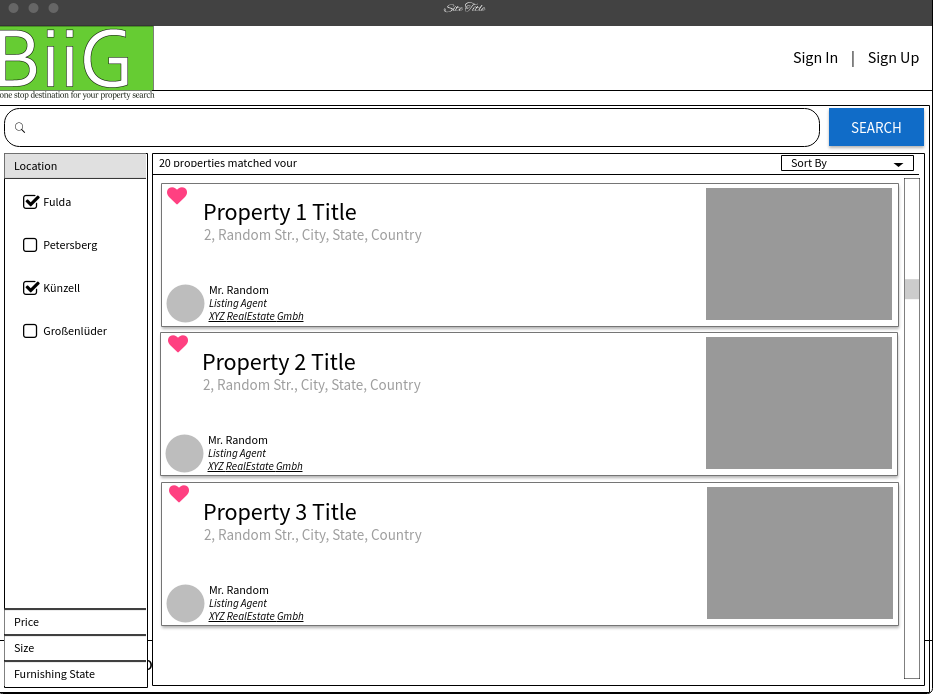
4.1.2 Homepage Part 2



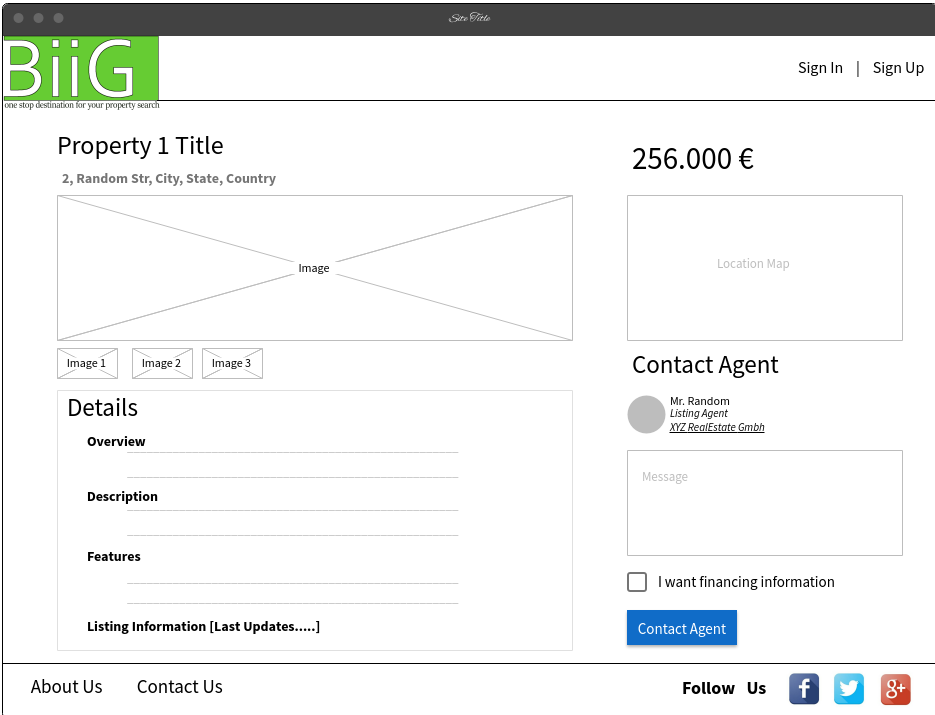
**4.2 Property Finder and Property Details**

In the property finder page the user can search for properties using the filters. The user can also sort the properties according to price and the time posted. Clicking on any property in the Property Finder opens the Property Details page.

4.2.1 Property Finder

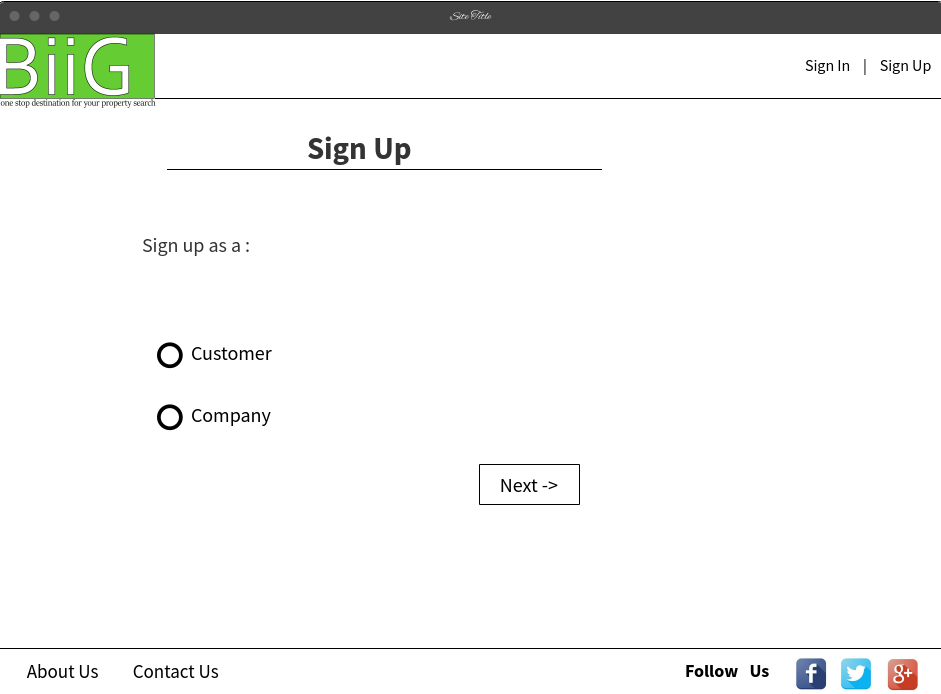
****

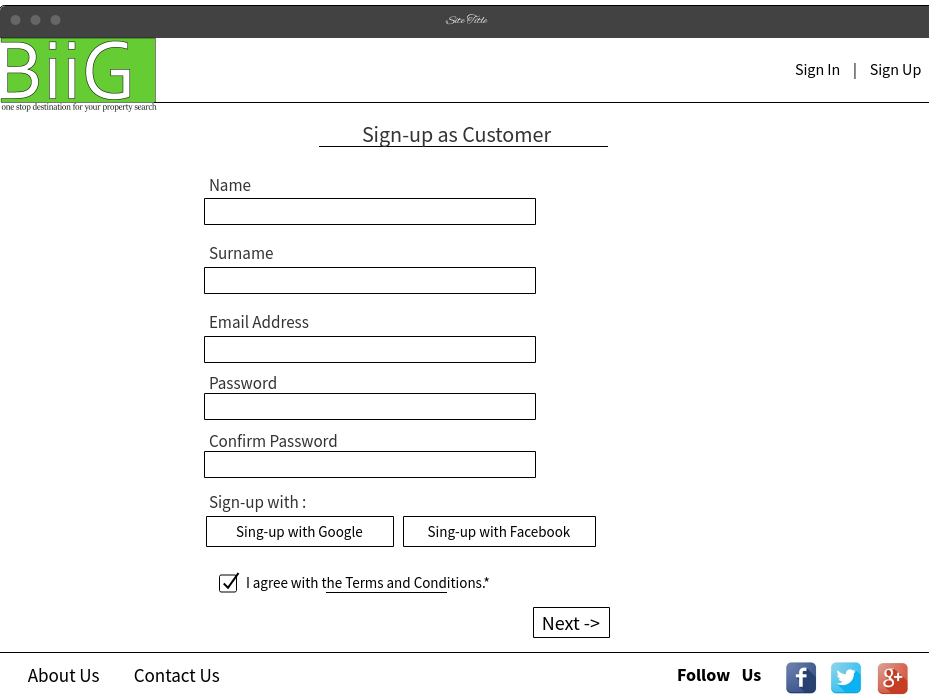
4.2.2 Property Details Page



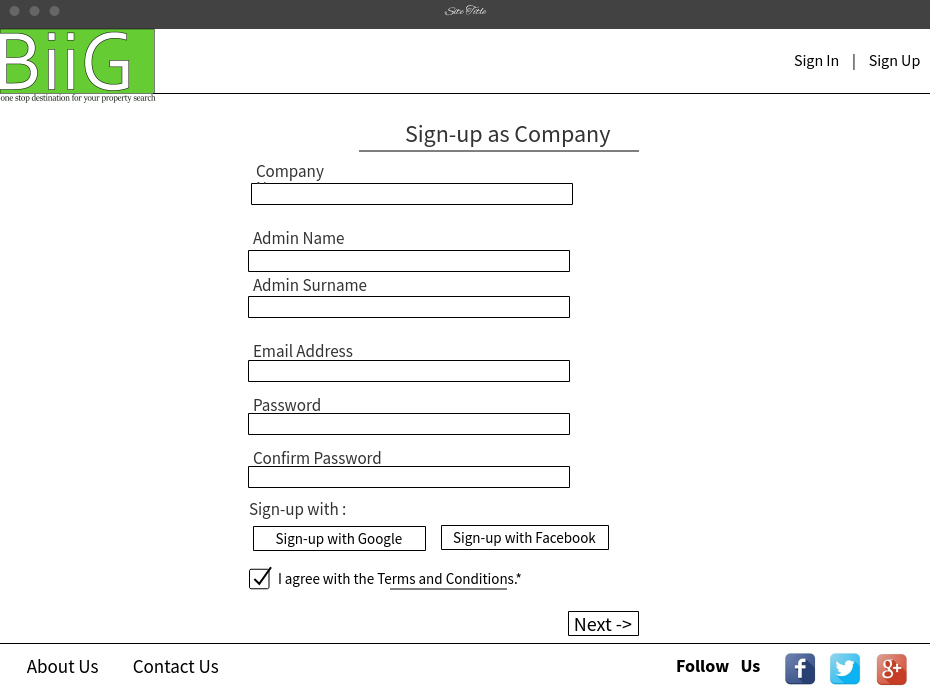
**4.3 Sign Up or Sign In**

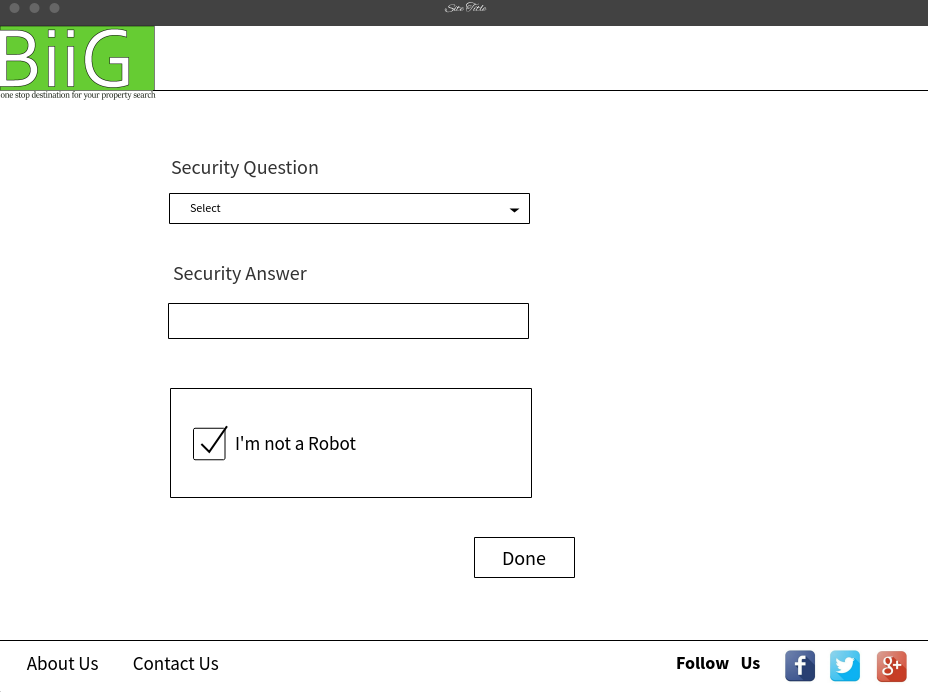
User can navigate to these pages by clicking on links **“Sign In” or “Sign Up”** from the header of any page. These pages provide users an option to create an account or sign in to their respective account.

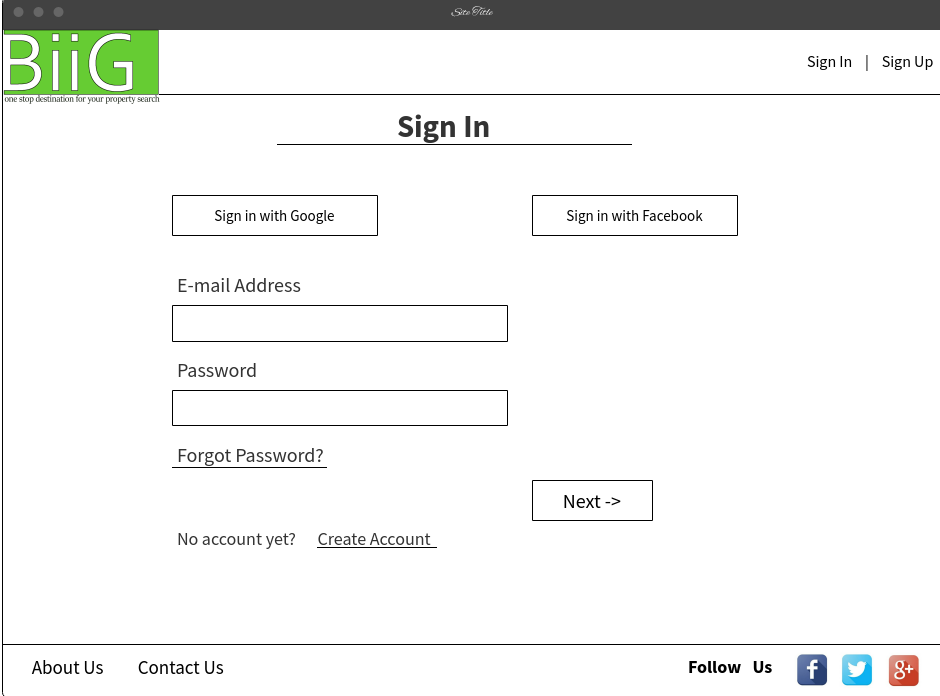
4.3.1 Sign up Option

4.3.2 Sign Up Customer

4.3.3. Sign Up Company

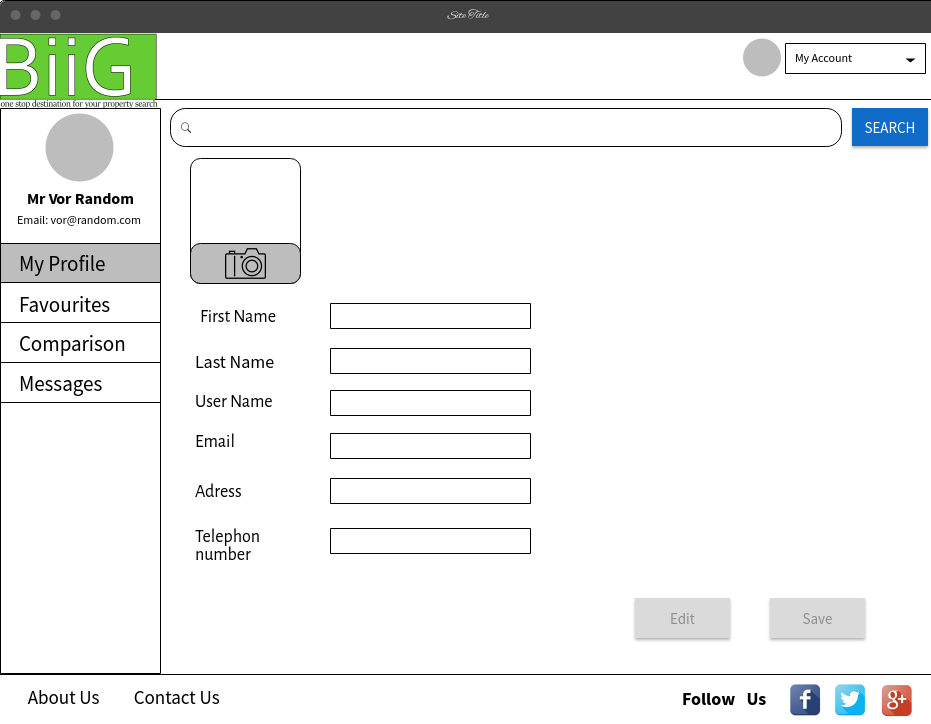


4.3.4 Security Question

4.3.5 Sign In:

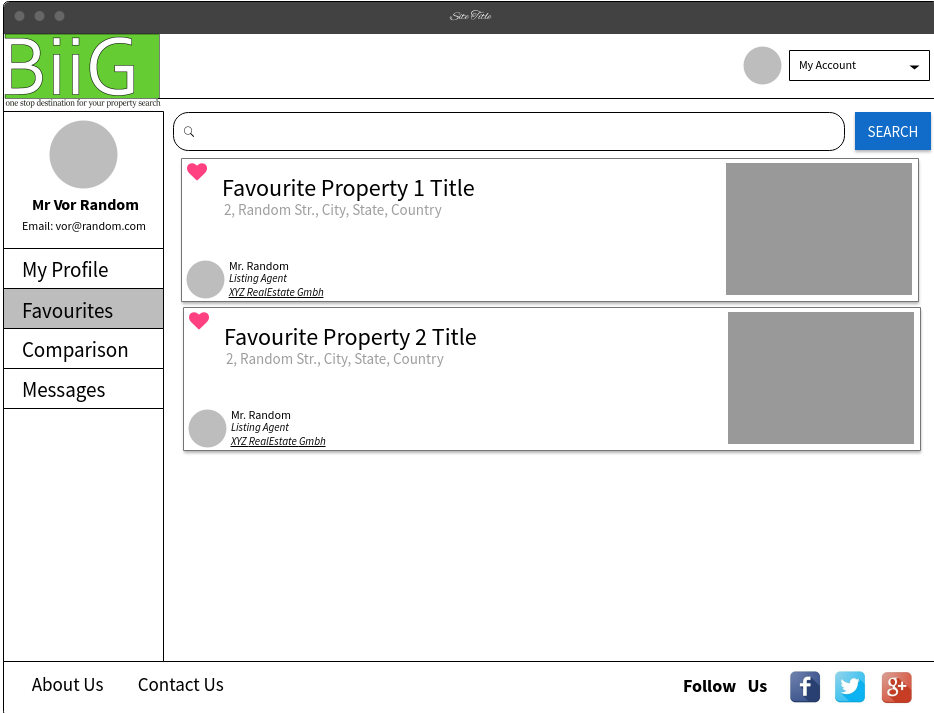
**4.4. Dashboard Customer**

4.4.1 Dashboard Customer Profile

The customer can view his profile details and can also update or edit the profile

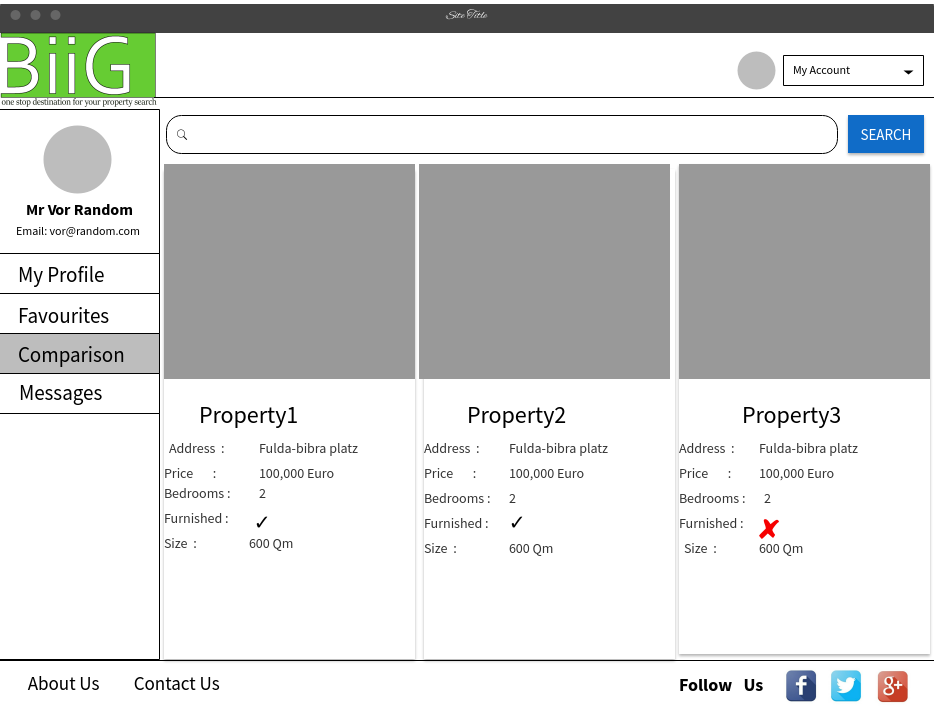
4.4.2 Dashboard Customer Favourites

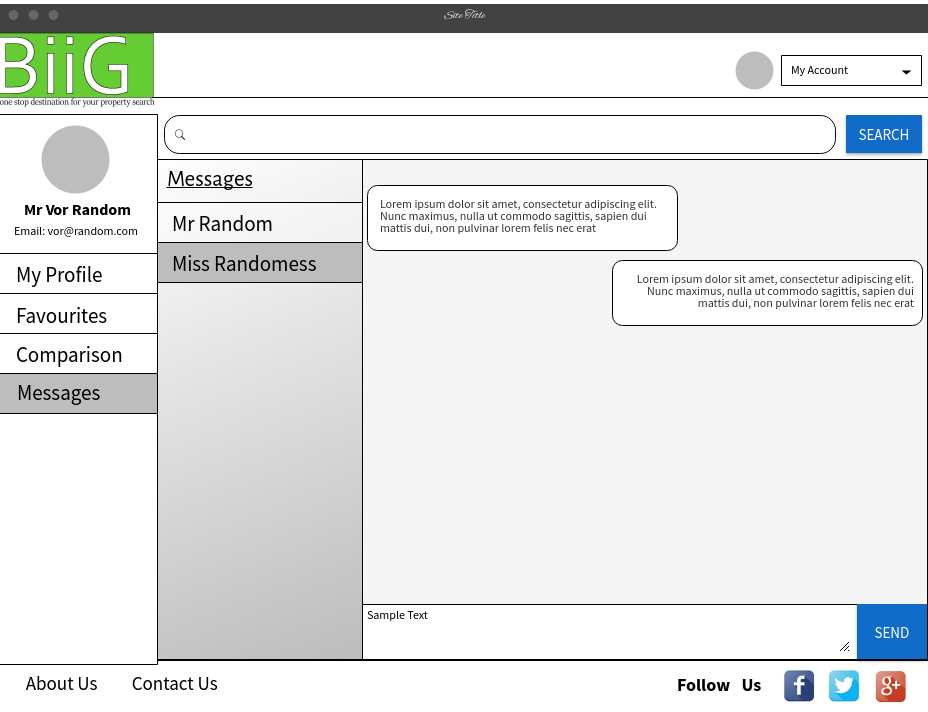
This page contains the list of favorite properties of the user. The user can delete a property from the list by deselecting the heart icon.



4.4.3 Dashboard Customer Comparison

On this page the customer can compare between different properties which are present in his favourites list. The comparison table will contain the pictures of the properties, address, price, number of rooms, area of the property and the furnished status.



4.4.4 Dashboard Customer Messages

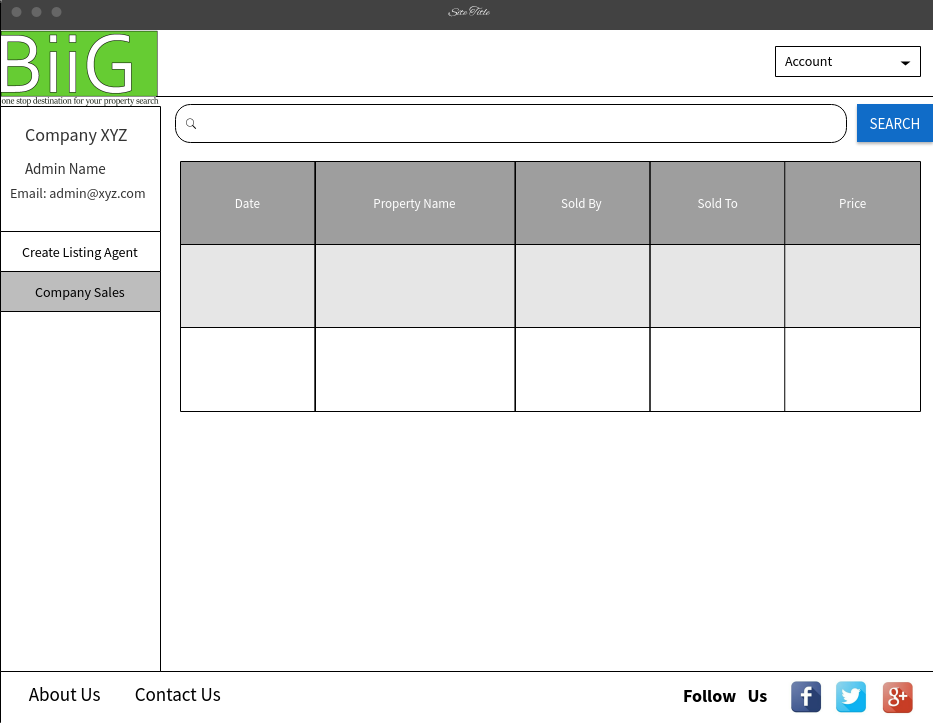
**4.5 Dashboard Company Admin**

4.5.1 Dashboard Create Listing Agents

Option for the Company Admin to create profile for Listing Agents

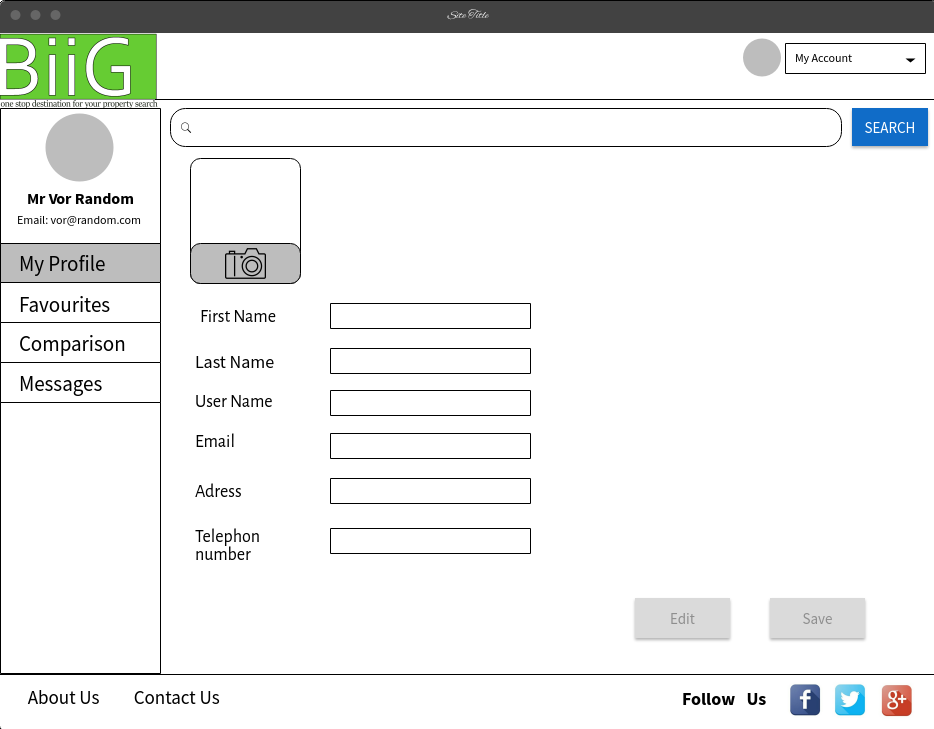


4.5.1 Dashboard Company Sales

Option for Company Admin to track sales

**4.6. Dashboard Listing Agent**

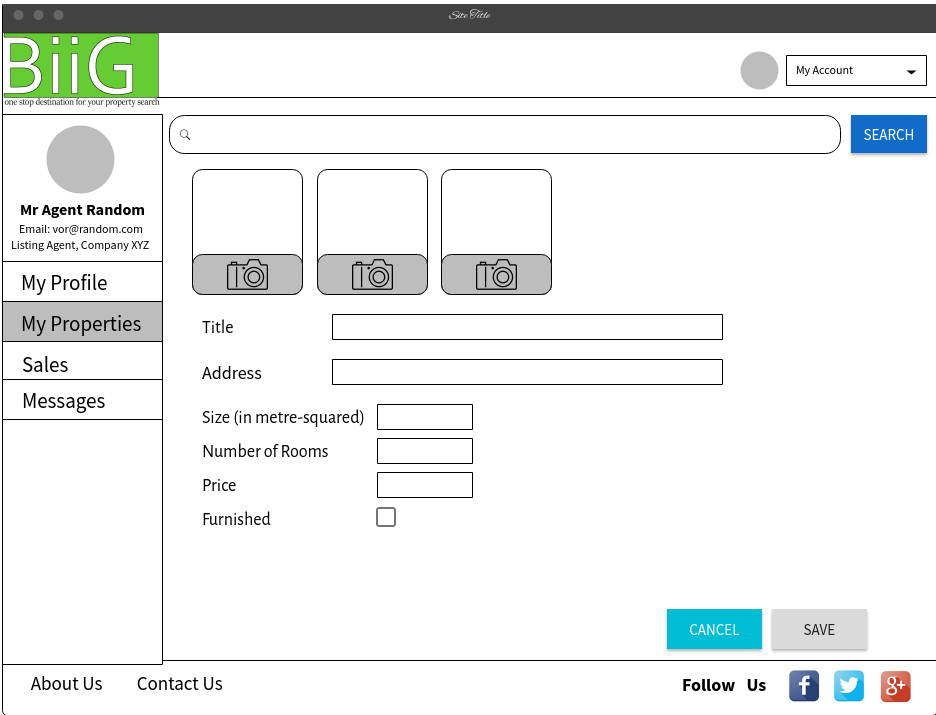
4.6.1 Dashboard Listing Agent Profile



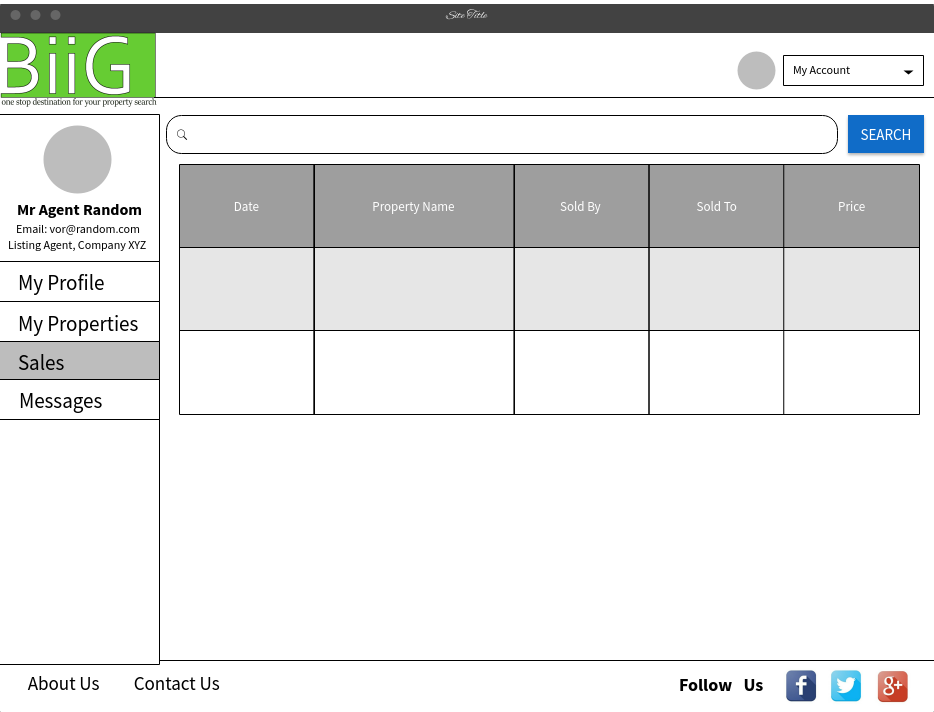
4.6.2 Dashboard Listing Agent Properties

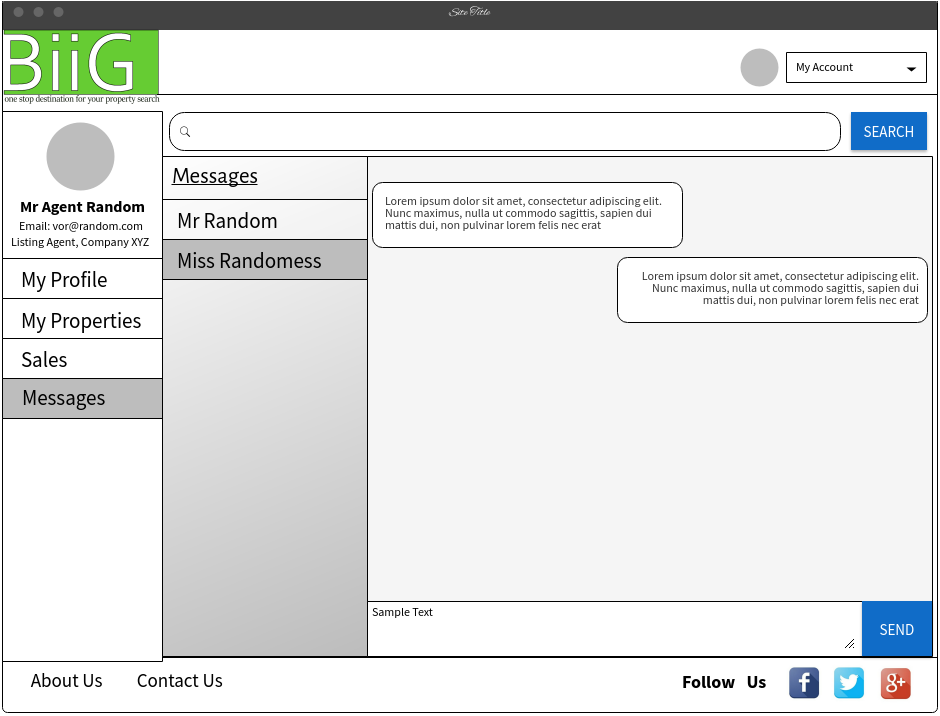


4.6.3 Dashboard Listing Agent Property Create

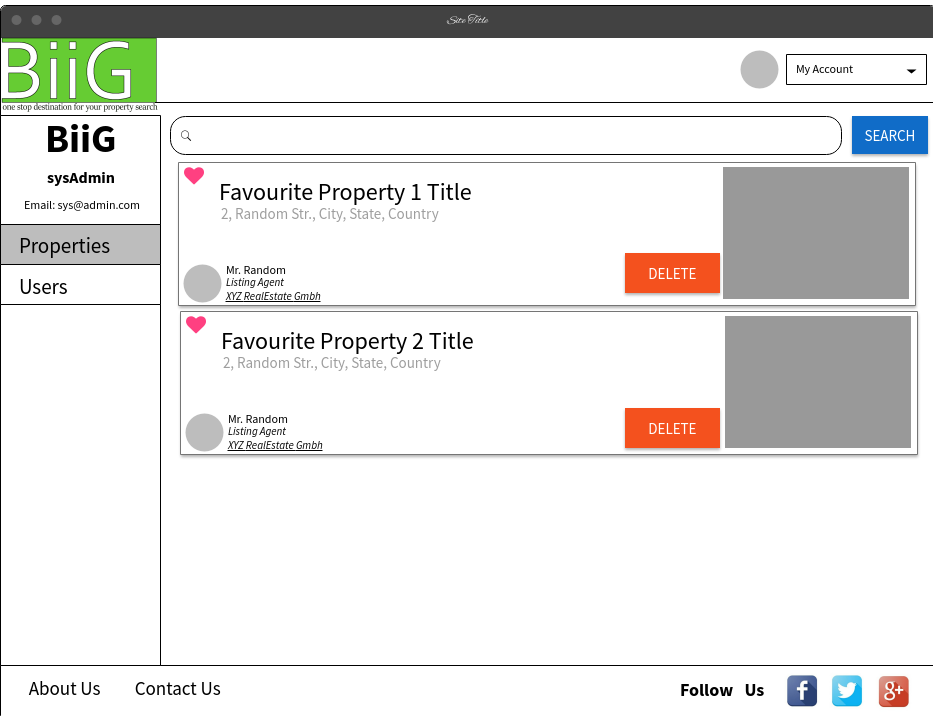


4.6.4 Dashboard Listing Agent Sales

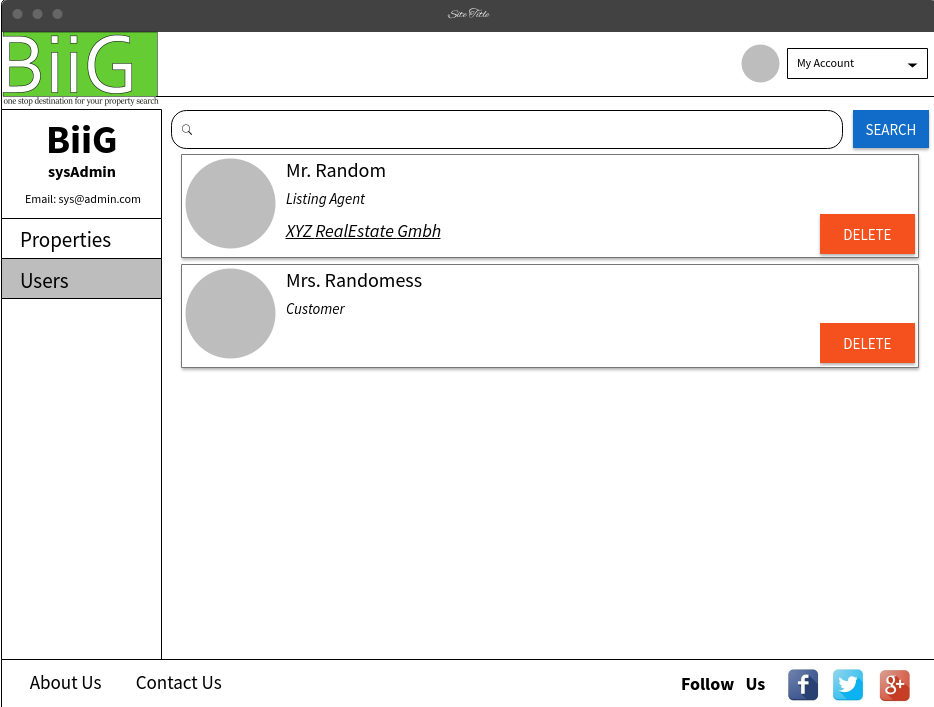


4.6.5 Dashboard Listing Agents Messages

**4.7 Dashboard Site Admin**

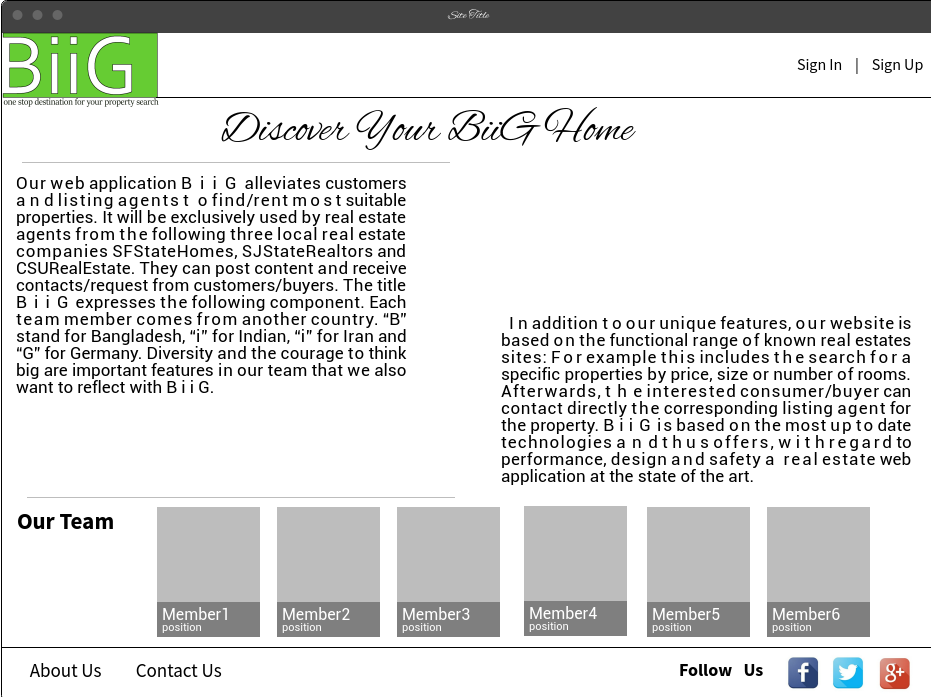
4.7.1 Dashboard Site Admin Remove Property

4.7.2 Dashboard Site Admin Remove User

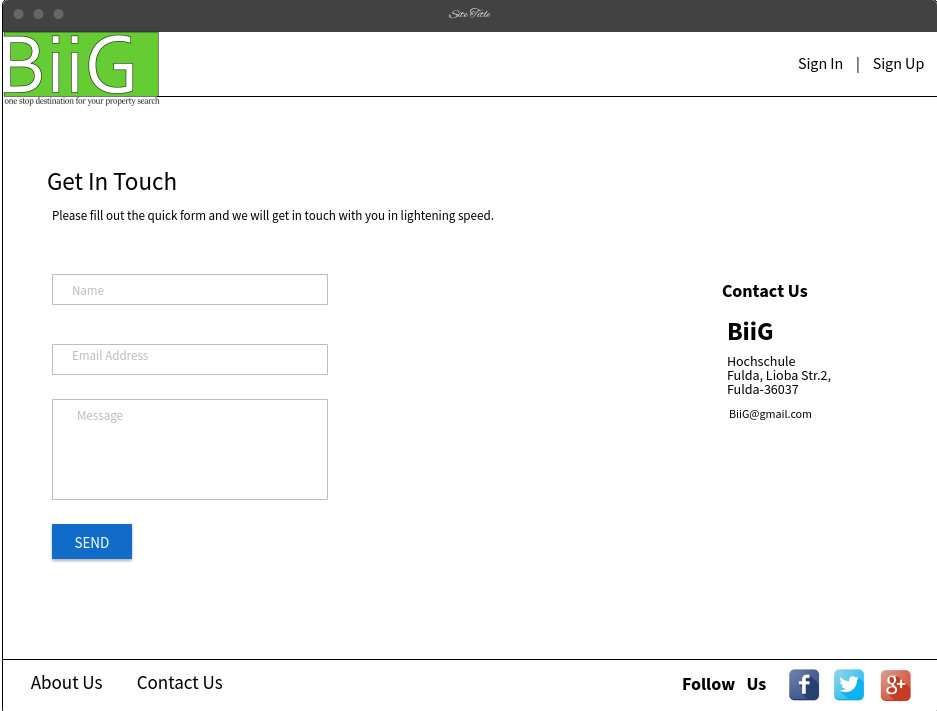


**4.8 About us and Contact Us**

4.8.1 About Us

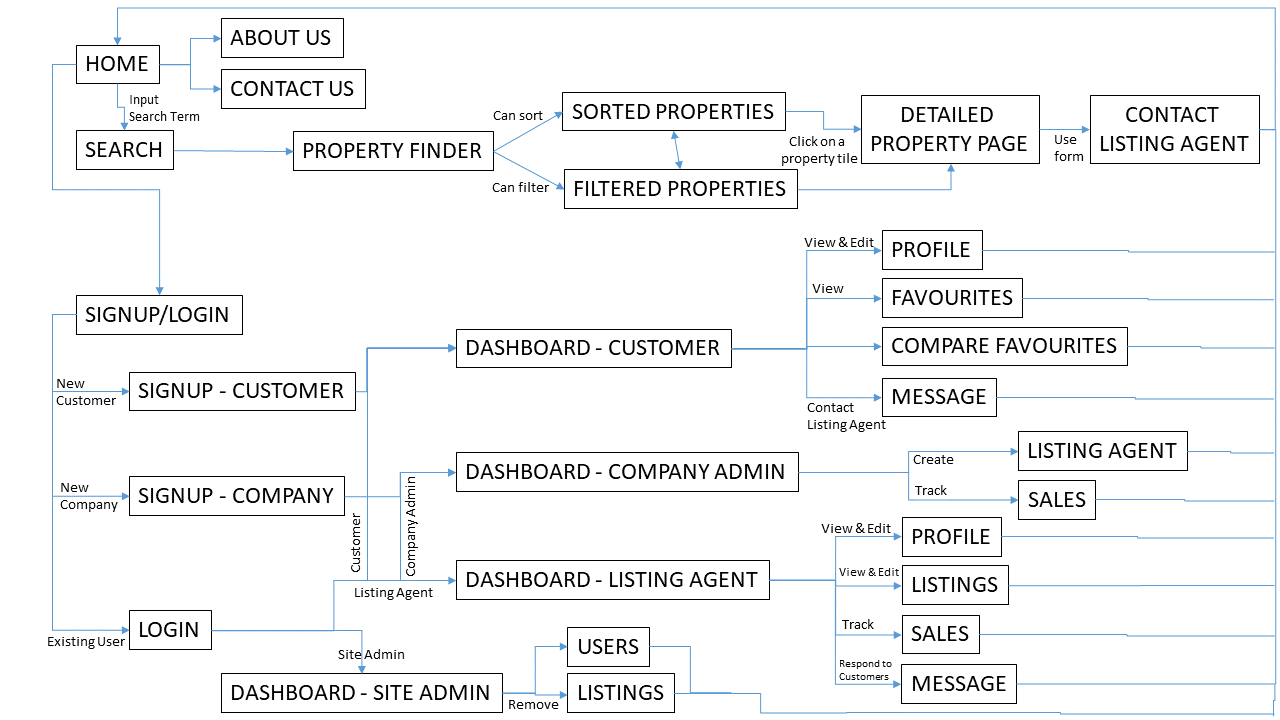


4.8.2 Contact Us



**4.9 Storyboard Overview**

This storyboard contains all the possible paths that a user can traverse in our application.



# 5. High Level Architecture and Database Organization

**5.1 High Level Architecture**

We will use the MVC architecture for our application.

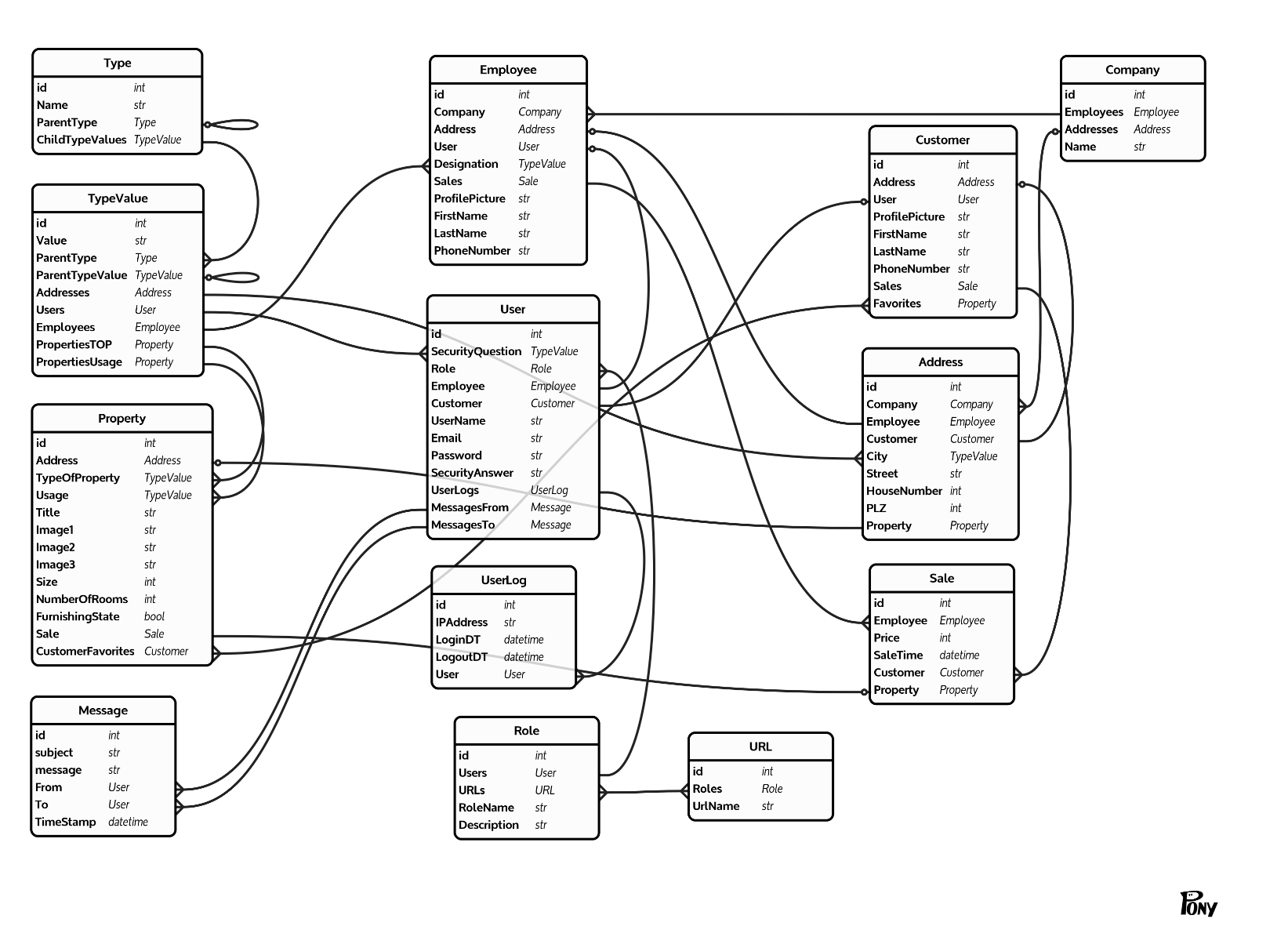
The **Model** would include a .js file to connect to our MySQL db and provide data whenever required. We will use ‘mysql’ node package for this.

The **View** would contain all our html files, static data (like images, stylesheets etc).

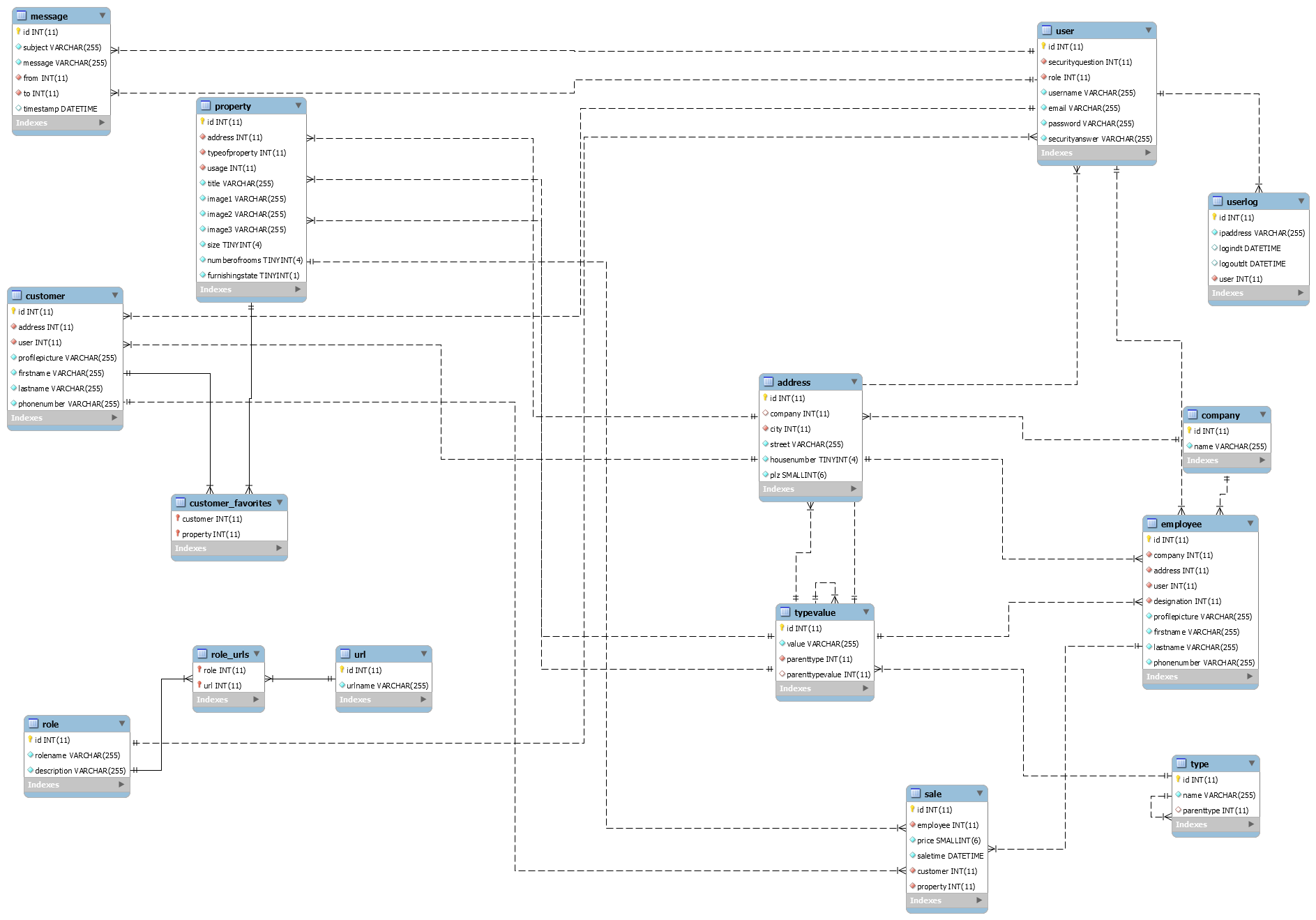
The **Controller** would contain the main app.js file which is the entry point to our application and also the file to route user requests. In addition, it will also contain all the business logic (calculate sales, search algorithms etc).

**Database Organization:**

Our database has 15 tables. The relations and attributes for each of them are defined below:

**Entity Relationship Model (PonyORM)**

**Entity Relationship Model (MySQL Workbench)**



**ENTITIES**

1. **Address**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI | NULL | auto\_increment |
| company | int(11) | YES | MUL | NULL |  |
| city | int(11) | NO | MUL | NULL |  |
| street | varchar(255) | NO |  | NULL |  |
| housenumber | tinyint(4) | NO |  | NULL |  |
| plz | smallint(6) | NO |  | NULL |  |

1. **company**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| name | varchar(255) | NO |  |  |  |

1. **customer**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| address | int(11) | NO | MUL |  |  |
| user | int(11) | NO | MUL |  |  |
| profilepicture | varchar(255) | NO |  |  |  |
| firstname | varchar(255) | NO |  |  |  |
| lastname | varchar(255) | NO |  |  |  |
| phonenumber | varchar(255) | NO |  |  |  |

1. **customer\_favorites**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| customer | int(11) | NO | PRI |  |  |
| property | int(11) | NO | PRI |  |  |

1. **employee**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| company | int(11) | NO | MUL |  |  |
| address | int(11) | NO | MUL |  |  |
| user | int(11) | NO | MUL |  |  |
| designation | int(11) | NO | MUL |  |  |
| profilepicture | varchar(255) | NO |  |  |  |
| firstname | varchar(255) | NO |  |  |  |
| lastname | varchar(255) | NO |  |  |  |
| phonenumber | varchar(255) | NO |  |  |  |

1. **message**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| subject | varchar(255) | NO |  |  |  |
| message | varchar(255) | NO |  |  |  |
| from | int(11) | NO | MUL |  |  |
| to | int(11) | NO | MUL |  |  |
| timestamp | datetime | YES |  |  |  |

1. **property**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| address | int(11) | NO | MUL |  |  |
| typeofproperty | int(11) | NO | MUL |  |  |
| usage | int(11) | NO | MUL |  |  |
| title | varchar(255) | NO |  |  |  |
| image1 | varchar(255) | NO |  |  |  |
| image2 | varchar(255) | NO |  |  |  |
| image3 | varchar(255) | NO |  |  |  |
| size | tinyint(4) | NO |  |  |  |
| numberofrooms | tinyint(4) | NO |  |  |  |
| furnishingstate | tinyint(1) | NO |  |  |  |

1. **role**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| rolename | varchar(255) | NO |  |  |  |
| description | varchar(255) | NO |  |  |  |

1. **role\_urls**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| role | int(11) | NO | PRI |  |  |
| url | int(11) | NO | PRI |  |  |

1. **sale**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| employee | int(11) | NO | MUL |  |  |
| price | smallint(6) | NO |  |  |  |
| saletime | datetime | NO |  |  |  |
| customer | int(11) | NO | MUL |  |  |
| property | int(11) | NO | MUL |  |  |

1. **type**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| name | varchar(255) | NO |  |  |  |
| parenttype | int(11) | YES | MUL |  |  |

1. **typevalue**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| value | varchar(255) | NO |  |  |  |
| parenttype | int(11) | NO | MUL |  |  |
| parenttypevalue | int(11) | YES | MUL |  |  |

1. **url**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| urlname | varchar(255) | NO |  |  |  |

1. **user**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| securityquestion | int(11) | NO | MUL |  |  |
| role | int(11) | NO | MUL |  |  |
| username | varchar(255) | NO |  |  |  |
| email | varchar(255) | NO |  |  |  |
| password | varchar(255) | NO |  |  |  |
| securityanswer | varchar(255) | NO |  |  |  |

1. **userlog**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(11) | NO | PRI |  | auto\_increment |
| ipaddress | varchar(255) | NO |  |  |  |
| logindt | datetime | YES |  |  |  |
| logoutdt | datetime | YES |  |  |  |
| user | int(11) | NO | MUL |  |  |

**REFERENCES**

|  |  |  |  |
| --- | --- | --- | --- |
| **TABLE\_NAME** | **COLUMN\_NAME** | **REFERENCED\_TABLE\_NAME** | **REFERENCED\_COLUMN\_NAME** |
| address | city | typevalue | id |
| address | company | company | id |
| customer | address | address | id |
| customer | user | user | id |
| customer\_favorites | customer | customer | id |
| customer\_favorites | property | property | id |
| employee | address | address | id |
| employee | company | company | id |
| employee | designation | typevalue | id |
| employee | user | user | id |
| message | from | user | id |
| message | to | user | id |
| property | address | address | id |
| property | typeofproperty | typevalue | id |
| property | usage | typevalue | id |
| role\_urls | role | role | id |
| role\_urls | url | url | id |
| sale | customer | customer | id |
| sale | employee | employee | id |
| sale | property | property | id |
| type | parenttype | type | id |
| typevalue | parenttype | type | id |
| typevalue | parenttypevalue | typevalue | id |
| user | role | role | id |
| user | securityquestion | typevalue | id |
| userlog | user | user | id |

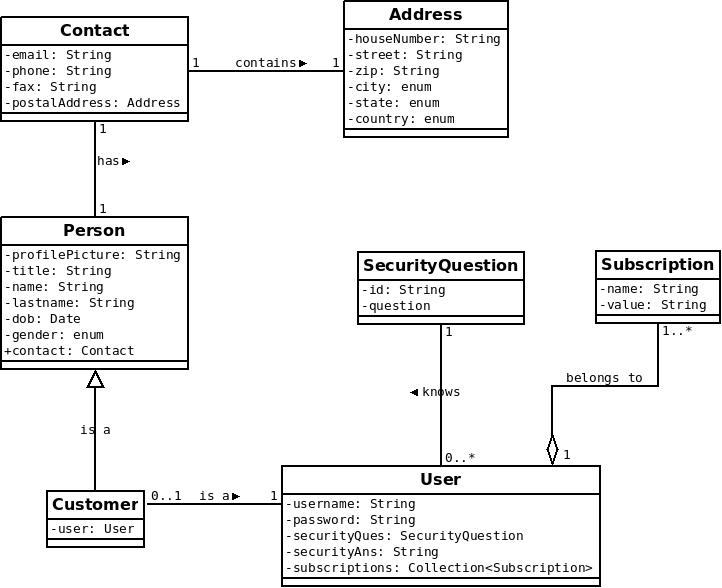
**Media Storage**Images and video/audio will be kept in file systems and the link to them will be stored in the database table wherever necessary.

**REST APIs**

|  |  |  |  |
| --- | --- | --- | --- |
| Resource | REST URI | HTTP Method | Description |
| Property | /biig/rest/properties | GET | Get all properties |
| Property | /biig/rest/properties?filter=*field*:*operator*:*value*&sort=field:*direction* | GET | Get a filtered and sorted properties using keys |
| Property | /biig/rest/properties/{id} | GET | Get a specific property using id |
| Property | /biig/rest/properties/ | POST | Add a property |
| Property | /biig/rest/properties/{id} | PUT | Update a property |
| Property | /biig/rest/properties/{id}/status | PUT | Update a property status (e.g: published, unpublished) |
| Property | /biig/rest/properties/{id}/state | PUT | Update a property state (e.g: ready for sale/listing, under discussion, sold, ready for rent, rented etc) |
| Property | /biig/rest/properties/{id} | DELETE | Delete a property |
| Sale | /biig/rest/properties/{id}/sales | GET | Get all sales of a specific property |
| Sale | /biig/rest/properties/{id}/sales/{sale\_id} | GET | Get a specific property's sale using id |
| Sale | /biig/rest/properties/{id}/sales/ | POST | Add a specific property's sale |
| Sale | /biig/rest/properties/{id}/sales/{sale\_id} | PUT | Update a specific property's sale |
| Sale | /biig/rest/properties/{id}/sales/{sale\_id}/status | PUT | Update a specific property's sale status (start, in progress, done) |
| Sale | /biig/rest/properties/{id}/sales/{sale\_id} | DELETE | Delete a specific property's sale |
| Rent | /biig/rest/properties/{id}/rents | GET | Get all rents of a specific property |
| Rent | /biig/rest/properties/{id}/rents/{rent\_id} | GET | Get a specific property's rent using id |
| Rent | /biig/rest/properties/{id}/rents/ | POST | Add a specific property's rent |
| Rent | /biig/rest/properties/{id}/rents/{rent\_id} | PUT | Update a specific property's rent |
| Rent | /biig/rest/properties/{id}/rents/{rent\_id}/status | PUT | Update a specific property's rent status (start, in progress, done) |
| Rent | /biig/rest/properties/{id}/rents/{rent\_id} | DELETE | Delete a specific property's rent |
| Lease | /biig/rest/properties/{id}/leases | GET | Get all leases of a specific property |
| Lease | /biig/rest/properties/{id}/leases/{lease\_id} | GET | Get a specific property's lease using id |
| Lease | /biig/rest/properties/{id}/leases/ | POST | Add a specific property's lease |
| Lease | /biig/rest/properties/{id}/leases/{lease\_id} | PUT | Update a specific property's lease |
| Lease | /biig/rest/properties/{id}/leases/{lease\_id}/status | PUT | Update a specific property's lease status (start, in progress, done) |
| Lease | /biig/rest/properties/{id}/leases/{lease\_id} | DELETE | Delete a specific property's lease |
| Mortgage | /biig/rest/properties/{id}/mortgages | GET | Get all mortgages of a specific property |
| Mortgage | /biig/rest/properties/{id}/mortgages/{mortgage\_id} | GET | Get a specific property's mortgage using id |
| Mortgage | /biig/rest/properties/{id}/mortgages/ | POST | Add a specific property's mortgage |
| Mortgage | /biig/rest/properties/{id}/mortgages/{mortgage\_id} | PUT | Update a specific property's mortgage |
| Mortgage | /biig/rest/properties/{id}/mortgages/{mortgage\_id}/status | PUT | Update a specific property's mortgage status (start, in progress, done) |
| Mortgage | /biig/rest/properties/{id}/mortgages/{mortgage\_id} | DELETE | Delete a specific property's mortgage |
| User | /biig/rest/users/{id} | GET | Get a specific user using id |
| User | /biig/rest/users/{id}/dashboard | GET | Get a specific user's dashboard using id |
| User | /biig/rest/users/{id}/messages | GET | Get a specific user's message list |
| User | /biig/rest/users/{id}/messages/{msg\_id} | GET | Get a specific user's message using id |
| User | /biig/rest/users/{id}/logs | GET | Get specific user's logs |
| User | /biig/rest/users/ | POST | Add a user |
| User | /biig/rest/users/login | POST | Login a user |
| User | /biig/rest/users/{id}/messages/send | POST | Send a specific user's message to another user |
| User | /biig/rest/users/{id} | PUT | Update a user |
| User | /biig/rest/users/{id}/logout | PUT | Logout a user |
| User | /biig/rest/users/{id}/reset-password | PUT | Reset a user's password |
| User | /biig/rest/users/{id}/change-username | PUT | Change a user's username |
| User | /biig/rest/users/{id}/subscribe | PUT | Subscribe a user in subscribtion list |
| User | /biig/rest/users/{id}/unsubscribe | PUT | Unsubscribe a user from subscribtion list |
| User | /biig/rest/users/{id} | DELETE | Delete a user |
| Customer | /biig/rest/customers | GET | Get all customers |
| Customer | /biig/rest/customers/{id} | GET | Get a specific customer using id |
| Customer | /biig/rest/customers/ | POST | Add a customer |
| Customer | /biig/rest/customers/{id} | PUT | Update a customer |
| Customer | /biig/rest/customers/{id} | DELETE | Delete a customer |
| Company | /biig/rest/companies | GET | Get all companies |
| Company | /biig/rest/companies/{id} | GET | Get a specific company using id |
| Company | /biig/rest/companies/ | POST | Add a company |
| Company | /biig/rest/companies/{id} | PUT | Update a company |
| Company | /biig/rest/companies/{id} | DELETE | Delete a company |
| Employee | /biig/rest/employees | GET | Get all employees |
| Employee | /biig/rest/employees?filter=*field*:*operator*:*value*&sort=field:*direction* | GET | Get a filtered and sorted employees using keys |
| Employee | /biig/rest/employees/{id} | GET | Get a specific employee using id |
| Employee | /biig/rest/employees/ | POST | Add a employee |
| Employee | /biig/rest/employees/{id} | PUT | Update a employee |
| Employee | /biig/rest/employees/{id} | DELETE | Delete a employee |
| Role | /biig/rest/roles | GET | Get all roles |
| Role | /biig/rest/roles/{id} | GET | Get a specific role using id |
| Role | /biig/rest/roles/ | POST | Add a role |
| Role | /biig/rest/roles/{id} | PUT | Update a role |
| Role | /biig/rest/roles/{id} | DELETE | Delete a role |
| Url | /biig/rest/urls | GET | Get all urls |
| Url | /biig/rest/urls/{id} | GET | Get a specific url using id |
| Url | /biig/rest/urls/ | POST | Add a url |
| Url | /biig/rest/urls/{id} | PUT | Update a url |
| Url | /biig/rest/urls/{id} | DELETE | Delete a url |
| Type | /biig/rest/types | GET | Get all types |
| Type | /biig/rest/types/{id} | GET | Get a specific type using id |
| Type | /biig/rest/types/ | POST | Add a type |
| Type | /biig/rest/types/{id} | PUT | Update a type |
| Type | /biig/rest/types/{id} | DELETE | Delete a type |
| Type Value | /biig/rest/types/{id}/values | GET | Get values of a specific type |
| Type Value | /biig/rest/types/{id}/values/{value\_id} | GET | Get a specific value using id of a specific type |
| Type Value | /biig/rest/types/{id}/values/ | POST | Add a value of a specific type |
| Type Value | /biig/rest/types/{id}/values/{value\_id} | PUT | Update a value of a specific type |
| Type Value | /biig/rest/types/{id}/values/{value\_id} | DELETE | Delete a value of a specific type |

# 6. High Level UML Diagrams

5.1 UML Class Diagram

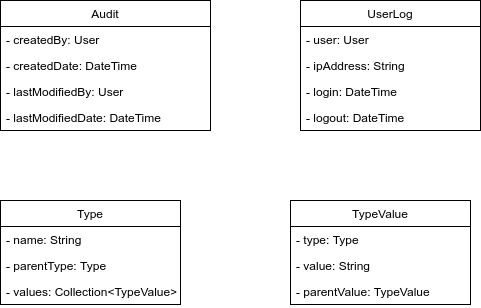


5.2 Class Diagrams

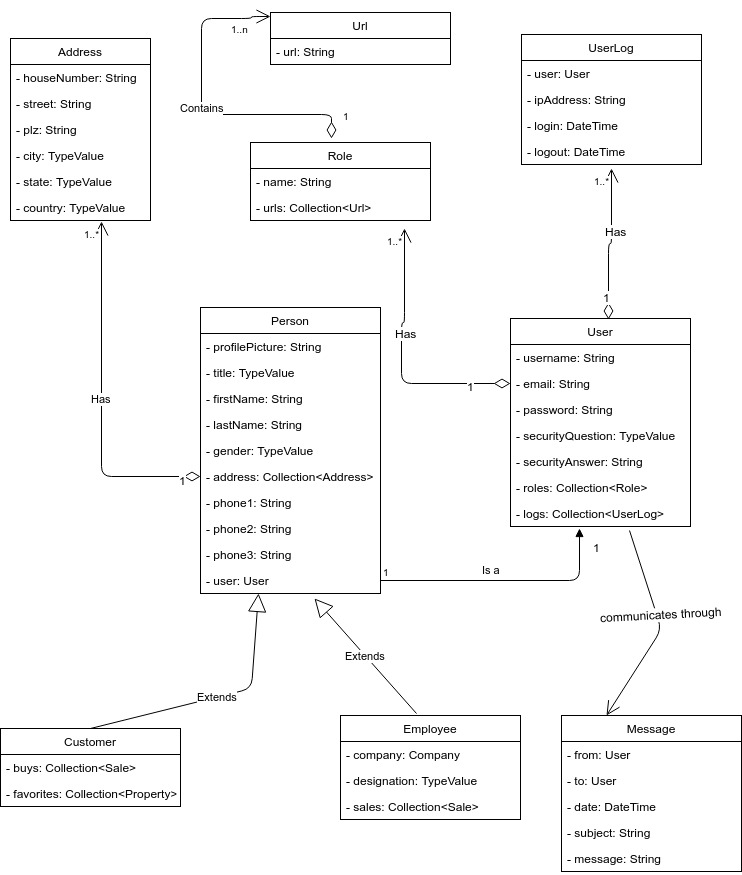
5.2.0 Class Diagram overview



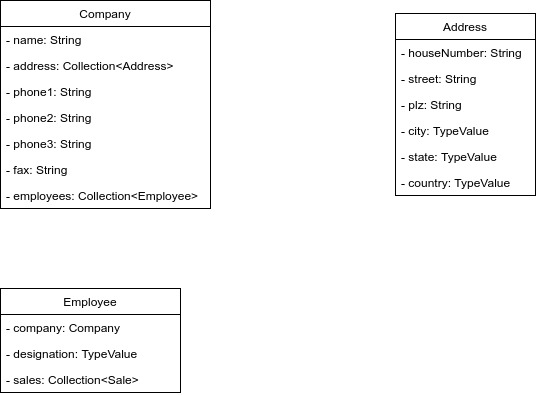
5.2.1. Class Diagram 1



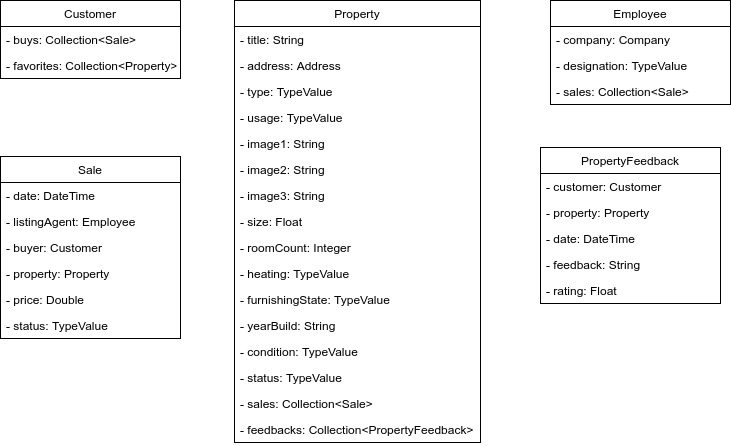
5.2.1 Class Diagram 2



5.2.3 Class Diagram 3



5.2.4 Class Diagram 4



# 7. Key Risks

**7.1 Skill Risk**

7.1.1 Members are not skilled enough to complete their project task before the weekly deadline.

7.1.2 Members have not enough experience with version control systems like Git.

**7.2 Schedule Risks**

7.2.1 Team members get sick or are not capable to work on their task anymore

7.2.2 Deadline of the Project in in two weeks. It can be very challenging to work at the

Same time on front/back end, integration and deployment.

7.3.3 Project is in an advance state, too many features were planned at the beginning.

Project is in risk of not getting complete in time.

**7.3 Technical Risks**

7.3.1Loss of work progress due to hardware and software errors

**7.4 Teamwork Risks**

7.4.1Member skip their tasks, don’t cooperate with other team members

7.4.2 Using only remote communication can lead to misunderstandings and can cost time

7.4.3 Member does not tell that he has a problem or tell it too late so that it will comprise the project

**7.5 Legal/Content Risks**

7.5.1 Using images and videos not created be us, can lead to a copyright strike

**7.6 Plan to resolve risks**

The whole team has to be committed to the project, which means every member should focus on finishing his given task before the deadline. This task will be shared throughout the team member with their specific skill set.

Every team member should discuss problems/changings/recommendations for tasks he/she sees to be fit as soon as possible with the help of our provided communication task management tools.

During our project progression, we will follow our best practices, using a consistent time management to meet the project expectations.

We will work as a unit and try to help each other out in every given topic.