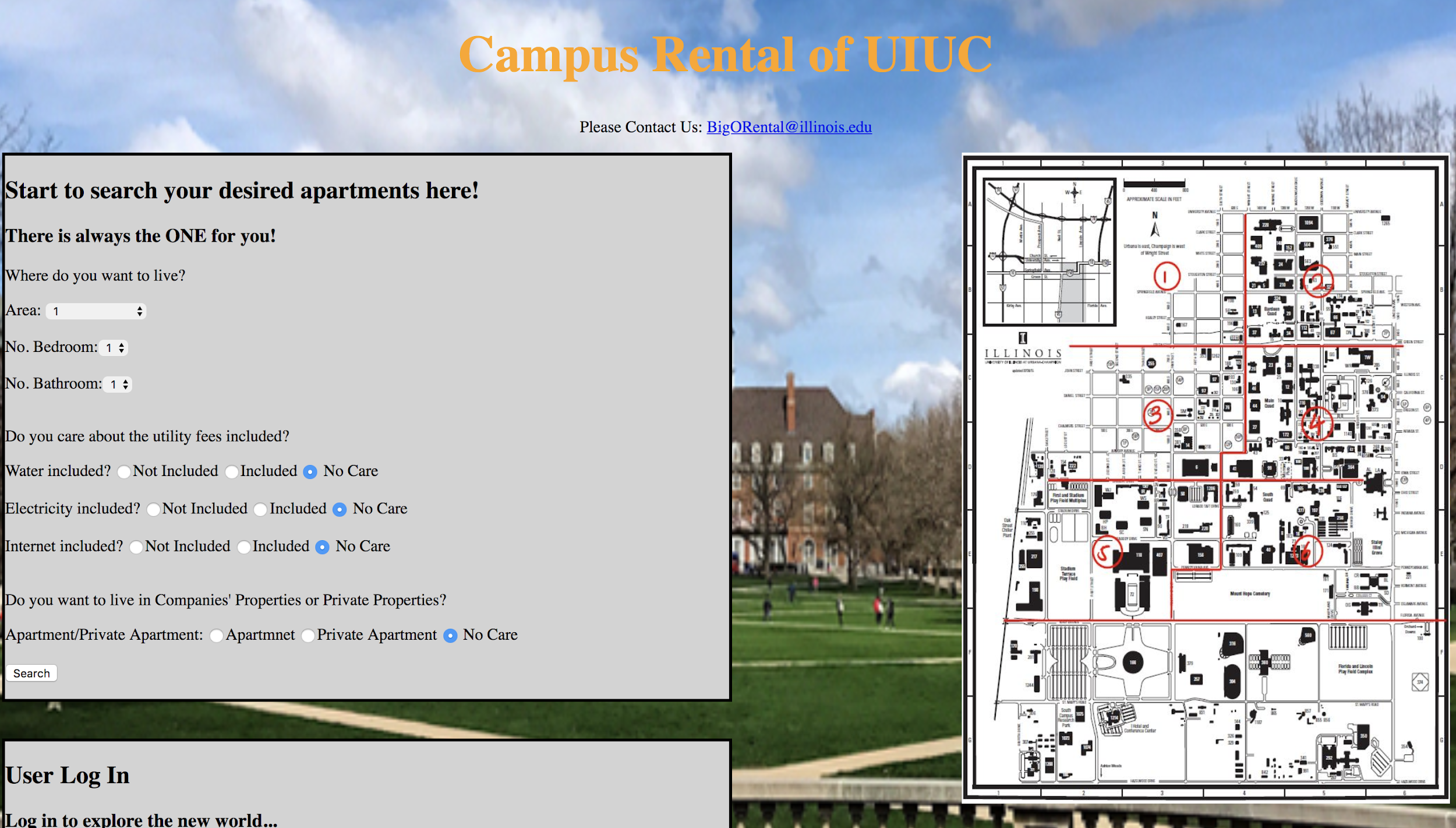
BigO\_Rental

Created by: BigO\_Range



**Anqi Shen, Yiran Li, Zheng Liu**

04.23.2018

CS 411 Project

# 

# BRIEF DESCRIPTION OF PROJECT

Our project is to implement an apartment search website which can help students find the ideal place to live for their college life. Our website allowed the search of apartment according to the number of bedroom, number of bath, and the utilities (such as water, electricity, and internet) to find the lists of apartment/housing fitting the requirements of our users. Besides, our website can make the recommendation lists of apartment for our users to help them find their home when studying in University of Illinois Urbana Champaign.

# USEFULNESS OF PROJECT

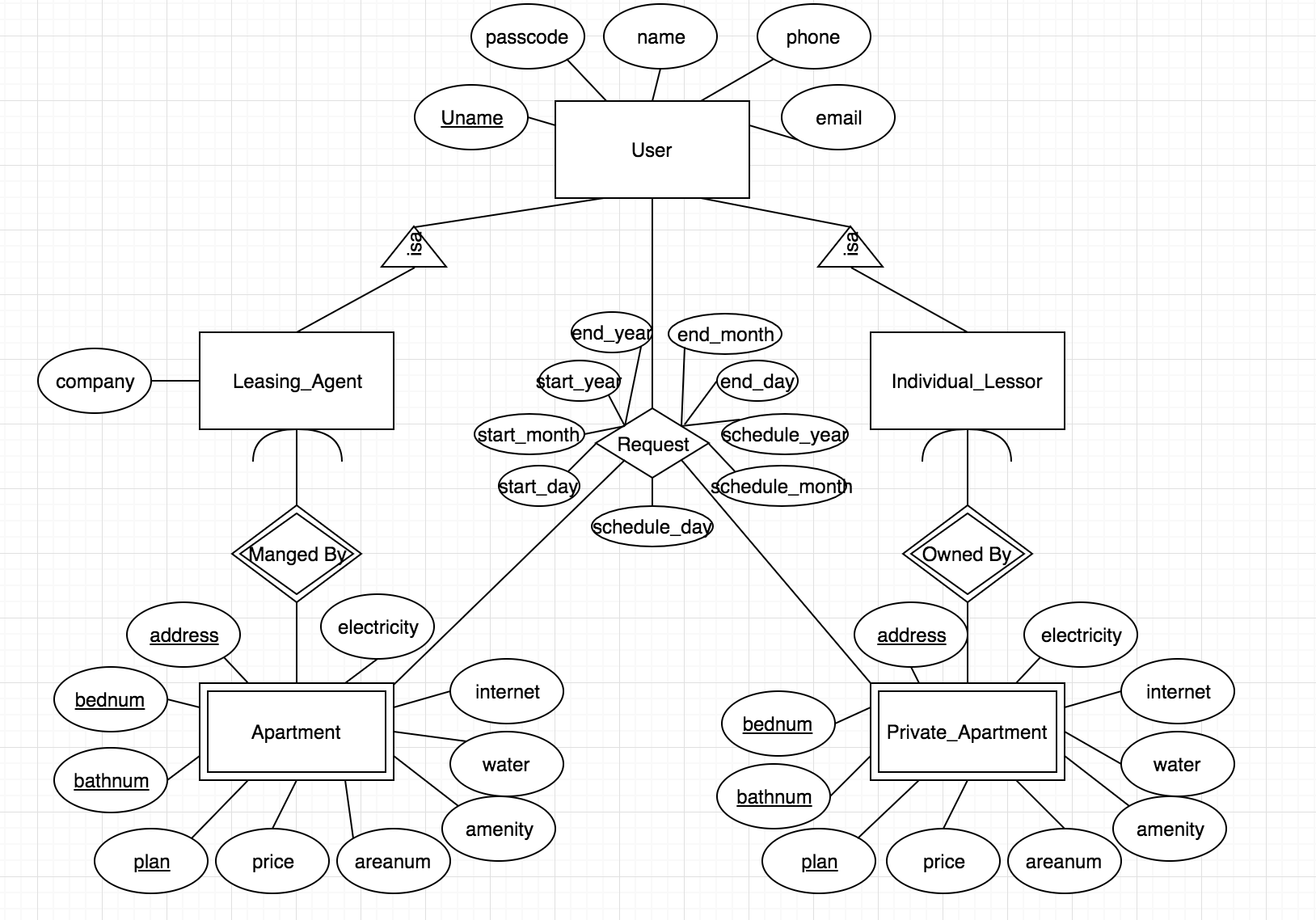
While many students or faculty staffs are not local residences of Champaign or Urbana, the require of housing is extremely high in university area. However, the prices of dorms or university founded student apartment are so expensive, so the rental apartments are the first choice for most students and faculty. The increasing new apartment building in university area is the evidence for that. Nevertheless, the housings/apartments in university area are held by different property companies and it is too bothering to visit the the property websites one by one. Therefore, we build a website which collects most apartments information belonged to different property companies in university area for the convenience of our users.

# DATA USED IN THE DATABASE

Our database are made by five data entity sets and one relation set. The data sets includes the user information and the apartment information. The apartment information are mainly divided into two data entity sets: Apartment and Private Apartment. The apartment are the rental housing/apartment owned or operated by rental property companies, while private apartments are owned by private individuals. The user data are also divided into two part: normal users and leasing agents. Normal user can search the housing information on the website and put their unused rental apartment on the website to find the lessee. Leasing agent are the account users of the property companies and they can update their housing information if they want. The relationship data set called Request is used for normal user if they want to contact the apartment owner to schedule an appointment or require more information (our website support email in website function).

# ER DIAGRAM & SCHEMA

ER Diagram:



Relational Schema:

1. User (Uname, passcode, name, phone, email);
2. Leasing\_Agent (Uname, company);
3. Individual\_Lessor (Uname);
4. Apartment (Uname, address, bednum, bathnum, plan, price, areanum, amenity, water, internet, electricity);
5. Private\_Apartment (Uname, address, bednum, bathnum, plan, price, areanum, amenity, water, internet, electricity);
6. Request (Uname\_Lessor, Uname\_Lessee, address, bednum, bathnum, plan, start\_year, start\_month, start\_day, end\_year, end\_month, end\_day, schedule\_year, schedule\_month, schedule\_day);

# DATA SOURCE

The source of our data based are found on the different property company websites which own the buildings in university area. Besides, we also get some data from some rental website to get individuals’ private housing information. We collected the data online and filling them into the microsoft excel form built based on the attributes of our data entity sets. After we collected the enough data we need, we load them into the php database.

# FUNCTIONALITY

1. Basic Function:

Insert:

* Can insert new data entity to the user data entity set tables (including Normal user, Individual lessor, and Leasing agent), when the new customer sign up a new user account.
* Can insert new data entity to the Apartment data entity set table, if the property company want to add a new apartment/housing information.
* Can Insert new data entity to the Private\_Apartment data entity set table, if individual lessor want to add a new housing/apartment for rental on the website.
* Can insert new relation entity to the Request relation table, if the user want to schedule a tour with the owner of the housing/apartment.

Update:

* Can update data entity in the Apartment data entity set table, if the property company want to modify apartment/housing information.
* Can update data entity in the Private\_Apartment data entity set table, if individual lessor want to modify new housing/apartment for rental on the website.

Delete:

* Can delete data entity from the Apartment data entity set table, if the property company want to delete new apartment/housing information.
* Can delete data entity from the Private\_Apartment data entity set table, if individual lessor want to delete housing/apartment for rental on the website.
* Can delete relation entity from the Request relation table, if the user (only normal user who send the schedule request) want to delete scheduled tour with the owner of the housing/apartment.

Search:

* The normal user search allow the users (including Normal user, Individual lessor, and Leasing agent) to search the apartment/housing information according to the input requirements. The Individual Lessor and Leasing Agent can search all the apartments and housing information they owned.

Schedule Request Search (Join used here):

* When the user want to see their scheduled request, our program would process the SQL query with the join of the three tables (User, request, and Apartment/ Private\_apartment) and show the required information to the user. (See details in the SQL code below).

1. Advance Function:

* Recommendation:

Our website can build a recommendation list based on the schedule request of the user to help our user to find out the ideal living place in their college life.

* Google Map API:

When our users search the apartment, the information list of the apartments can be both seen as the table on the web page and as the markers on the google map api with the housing information if the mouse is placed on the marker.

* Online Scheduler Emailing:

Our website allows the customers to fill the form of schedule request or information request of their target apartment, and our website would automatically help customers send their requests to the Leasing Agent or Individual Lessor.

# ONE BASIC FUNCTION DESCRIPTION

**Search:**

* Request Search (Join):

When the user want to see the schedule requests he or she have sent to the Leasing Agent or Individual Lessor. The SQL query we implemented would finish search with the join of three tables to get the all the information the user needed to see his/her scheduled requests (including address, area number, Lessor, badroom number, bathroom number, floor plan, contract start date, contract end date, schedule meeting date, and email). In this part we have to do the join of three tables, because Request is a relation table and the other information are saved in the other data entity set tables (Apartment/Private Apartment and User). (The code can be seen below)

* Housing Information Search:

The housing information search can be divided in two parts: before login and after login. The housing information search before the user login would only show the corresponding Apartment/Housing information based on the customer inputs. The after login housing information search would not only show the information but also the functionality of schedule (additional schedule button) which allow our user to schedule their interested apartments. The SQL query are processed mainly upon the Apartment table and Private Apartment table(The code can be seen below).

* User login Search:

The user login search is the search of user information based on input user name and user password. Only the recorded user (information stored in the User data entity table) can login to the website. The registered users can enjoy the additional service on the website. The SQL query can mainly focused on the User data entity table (The code can be seen below).

# SQL CODE

1. **Insert:** We have two functionalities that involved in using INSERT query: new user sign-up and new housing information adding.
   1. The SQL code we used for user sign-up is: "INSERT INTO User (Uname, passcode, name, phone) VALUES ('$Uname', '$passcode', '$name', $phone)". Additionally, since we have three different types of user: lessee only, leasing agent and individual lessor, depending on the type chosen by the user, we may need to insert the user information into the table of Leasing\_Agent or the table of Individual\_Lessor as well due to the reason that we are using ER model to store subclasses. The queries for inserting to one of these two tables are: "INSERT INTO Leasing\_Agent (Uname, company) VALUES ('$Uname', '$company')" and "INSERT INTO Individual\_Lessor (Uname) VALUES ('$Uname')".
   2. The SQL code we used for adding new housing information into our database, which is only allowed by leasing agents or individual lessors. Since we have two different types of apartments: leased by companies or leased by individuals. To adding new apartments managed by companies, named as Apartment, the query we use is "INSERT INTO Apartment (Uname, address, bednum, bathnum, plan, price, water, electricity, internet, amenity, areanum, company) VALUES ('$Uname', '$address', $bednum, $bathnum, '$plan', $price, '$water', '$electricity', '$internet', '$amenity', $areanum, '$company')". And the query we use for inserting new apartment information into Private\_Apartment is "INSERT INTO Private\_Apartment (Uname, address, bednum, bathnum, plan, price, water, electricity, internet, amenity, areanum, company) VALUES ('$Uname', '$address', $bednum, $bathnum, '$plan', $price, '$water', '$electricity', '$internet', '$amenity', $areanum, 'private')"

In both two parts of inserting, the values are provided by users, which is gotten by the forms in the user interface.

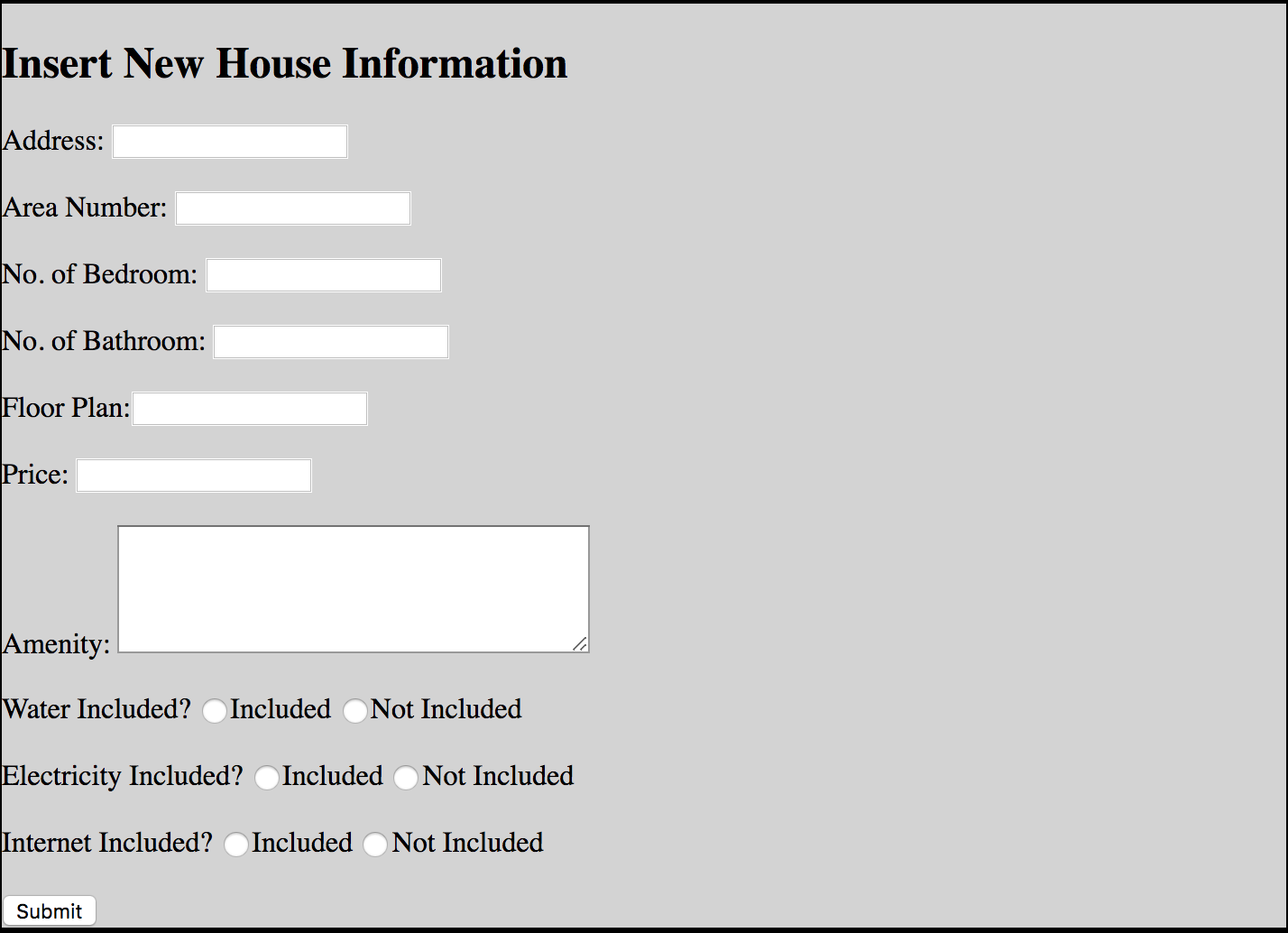
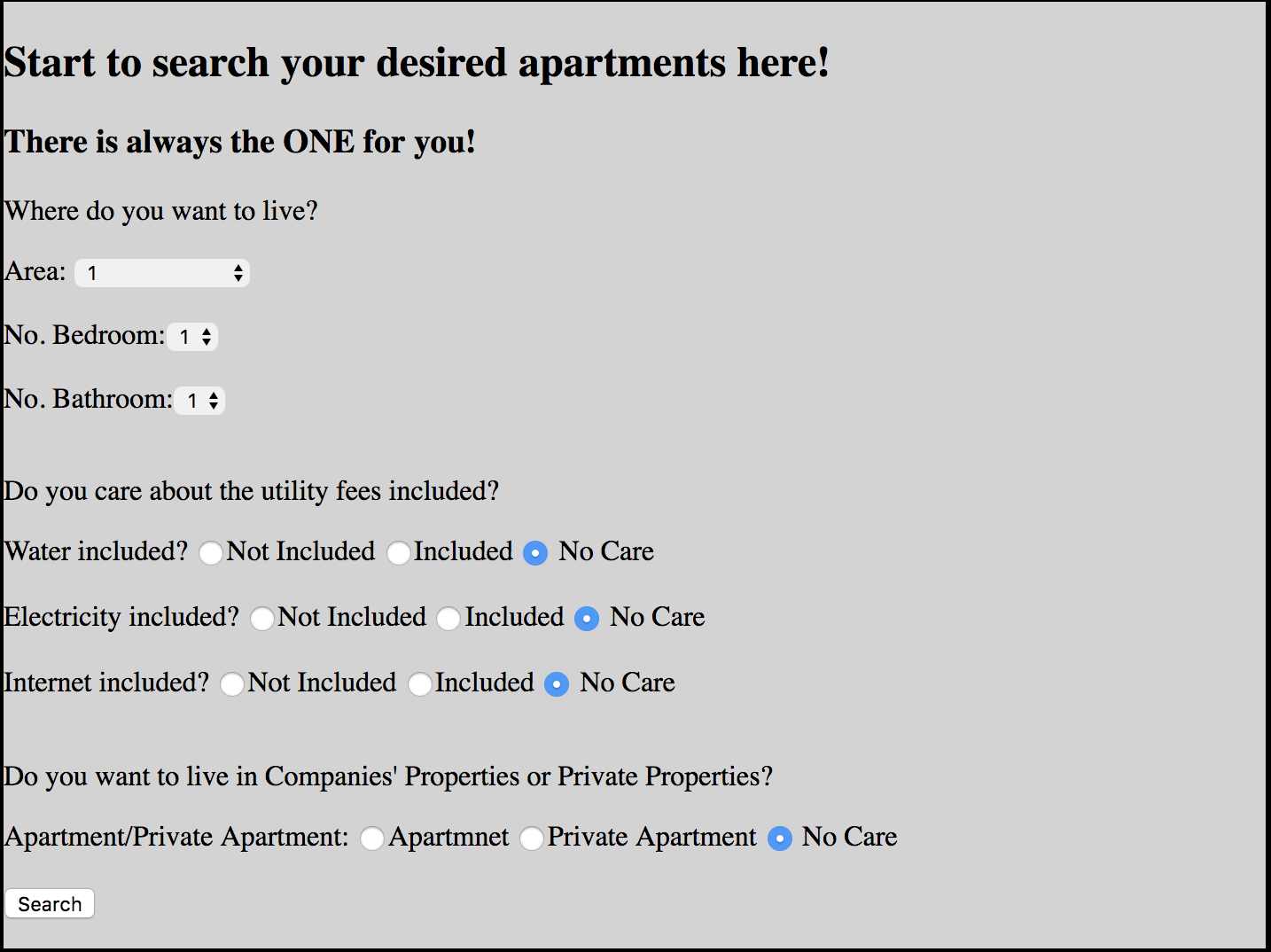
1. **Update**: We use UPDATE query for leasing agents and individual lessors to update their housing information. In the user interface, we have an update link for each records of the current user’s houses, which are provided by the search functions we did (it will be described in the later part). When the user clicks “update” of one record, the user interface will show up the current information of this record in the form and the user can choose which part he or she want to update except the primary key attributes. If the current user is a leasing agent, the query we used is the following: “UPDATE Apartment SET new\_updated\_attribute=$new\_value WHERE address='$address' AND Uname='$Uname' AND bednum='$bednum' AND bathnum='$bathnum' AND plan='$plan'”. If the current user is an individual lessor, the query we used is the following: “UPDATE Private\_Apartment SET new\_updated\_attribute=$new\_value WHERE address='$address' AND Uname='$Uname' AND bednum='$bednum' AND bathnum='$bathnum' AND plan='$plan'”. For both Apartment and Private\_Apartment, the primary key is the combination of attributes Uname, address, bednum, bathnum and plan.
2. **Delete**: We use DELETE query for two parts: deleting housing information records (only allowed by users as leasing agents or individual lessors) and deleting already-scheduled tours (only allowed by users as lessees).
   1. For deleting housing information records stored in either Apartment table or Private\_Apartment table, similarly to updating, each record showing in “My Houses” will have a “delete” link. When “delete” is clicked, this record will be removed from the database. The query we use is: "DELETE FROM Apartment WHERE address='$address' AND Uname='$Uname' AND bednum=$bednum AND bathnum=$bathnum AND plan='$plan'" when the record is a company property. If the record is a private apartment, then the query we use is: "DELETE FROM Private\_Apartment WHERE address='$address' AND Uname='$Uname' AND bednum=$bednum AND bathnum=$bathnum AND plan='$plan'". As defined, the primary key for both Apartment and Private\_Apartment is the combination of Uname, address, bednum, bathnum and plan. All these attributes data are gotten by the current line’s data id in the user interface.
   2. For deleting scheduled requests, similarly, there is a “delete” link of each record showing in the “My Schedule” table. If the user who is a lessee click “Delete”, which means he or she does not want to see the apartment anymore, the record of this request will be deleted from our database. The query we use is the following: "DELETE FROM Request WHERE address='$address' AND bednum=$bednum AND bathnum=$bathnum AND plan='$plan' AND Uname\_Lessor='$Uname\_Lessor' AND Uname\_Lessee='$Uname'". The primary key is gotten from the corresponding data-id in the table and Request is the table that stored the scheduled tour requests.
3. **Search**: We use multiple searches for several different functionalities: user login, search housing information, show “My Apartments” information, show “My Message”, show “My Schedule”.
   1. User login: using the user name entered by the user on the “Login” part of the website as the primary key Uname, we search User table and get the passcode of this Uname. Then we compare the passcode we found with the passcode entered by the user. If they are the same, then login successes; otherwise, ask the user to try again if the passcodes are not the same or the Uname has not been found in our database. The query we use is: “SELECT \* FROM User WHERE passcode='$passcode' AND Uname='$Uname'”.
   2. Search housing information: Based on the constraints entered by the user using the searching interface on the website, we search our Apartment table and/or Private\_Apartment Table. The query we use is: “SELECT address, price, bednum, bathnum, plan, water, electricity, internet,amenity, areanum, company, lat, lng FROM Apartment WHERE areanum = $area AND internet = '$internet' AND bednum = $bed AND bathnum = $bath” for searching Apartment if the user only want to see the company properties; or, “SELECT address, price, bednum, bathnum, plan, water, electricity, internet,amenity, areanum, company, lat, lng FROM Private\_Apartment WHERE areanum = $area AND internet = '$internet' AND bednum = $bed AND bathnum = $bath” if the user only want to see apartments leased by individual lessors. We use both queries if the user does not care the type of apartment. Then we print out all searching results on the website with “Schedule” and “Delete” link for each record.
   3. Show “My Messages”: This functionality is for leasing agents or individual lessors to check who has scheduled a meeting or a tour on his or her properties and show each schedule’s details including housing information, lessee information, contract start and end time and schedule time. The query we use is: "SELECT Request.address, Uname\_Lessee, Request.bednum, Request.bathnum, Request.plan, start\_year, start\_month, start\_day, end\_year, end\_month, end\_day, schedule\_year, schedule\_month, schedule\_day, Apartment.amenity, Apartment.areanum, User.phone FROM Request, Apartment, User WHERE Uname\_Lessor ='$Uname' AND Request.address=Apartment.address AND Request.bednum=Apartment.bednum AND Request.bathnum=Apartment.bathnum AND Request.plan=Apartment.plan AND User.Uname=Request.Uname\_Lessee" (for company properties) and "SELECT Request.address, Uname\_Lessee, Request.bednum, Request.bathnum, Request.plan, start\_year, start\_month, start\_day, end\_year, end\_month, end\_day, schedule\_year, schedule\_month, schedule\_day, Private\_Apartment.amenity, Private\_Apartment.areanum, User.phone FROM Request, Private\_Apartment, User WHERE Uname\_Lessor ='$Uname' AND Request.address=Private\_Apartment.address AND Request.bednum=Private\_Apartment.bednum AND Request.bathnum=Private\_Apartment.bathnum AND Request.plan=Private\_Apartment.plan AND User.Uname=Request.Uname\_Lessee" (for private apartments). In this query, we join three tables Request, Apartment/Private\_Apartment and User to get the search results. Then we print out the search results on the website.
   4. Show “My Schedule”: This functionality is for normal user (lessee) to see his or her already scheduled tours. We first check the user’s scheduled tours with company properties and then check the user’s scheduled tours with private apartments. The queries we use are: "SELECT Request.address, Uname\_Lessor, Request.bednum, Request.bathnum, Request.plan, start\_year, start\_month, start\_day, end\_year, end\_month, end\_day, schedule\_year, schedule\_month, schedule\_day, Apartment.amenity, Apartment.areanum, User.email FROM Request, Apartment, User WHERE Uname\_Lessee ='$Uname' AND Request.address=Apartment.address AND Request.bednum=Apartment.bednum AND Request.bathnum=Apartment.bathnum AND Request.plan=Apartment.plan AND Request.type='A' AND User.Uname=Uname\_Lessor" (for company properties) and SELECT Request.address, Uname\_Lessor, Request.bednum, Request.bathnum, Request.plan, start\_year, start\_month, start\_day, end\_year, end\_month, end\_day, schedule\_year, schedule\_month, schedule\_day, Private\_Apartment.amenity, Private\_Apartment.areanum, User.email FROM Request, Private\_Apartment, User WHERE Uname\_Lessee ='$Uname' AND Request.address=Private\_Apartment.address AND Request.bednum=Private\_Apartment.bednum AND Request.bathnum=Private\_Apartment.bathnum AND Request.plan=Private\_Apartment.plan AND Request.type='B' AND User.Uname=Uname\_Lessor" (for private apartment). Both queries join three tables in order to find the schedule details including housing information, lessor information and schedule time. Then we print out our search results on the website with “Email” and “Delete” link for each record.

# DATAFLOW

First, on our website, we have forms to collect data we need for our database. These forms are either directly showing up on the main page or they can be popped out when corresponding buttons.

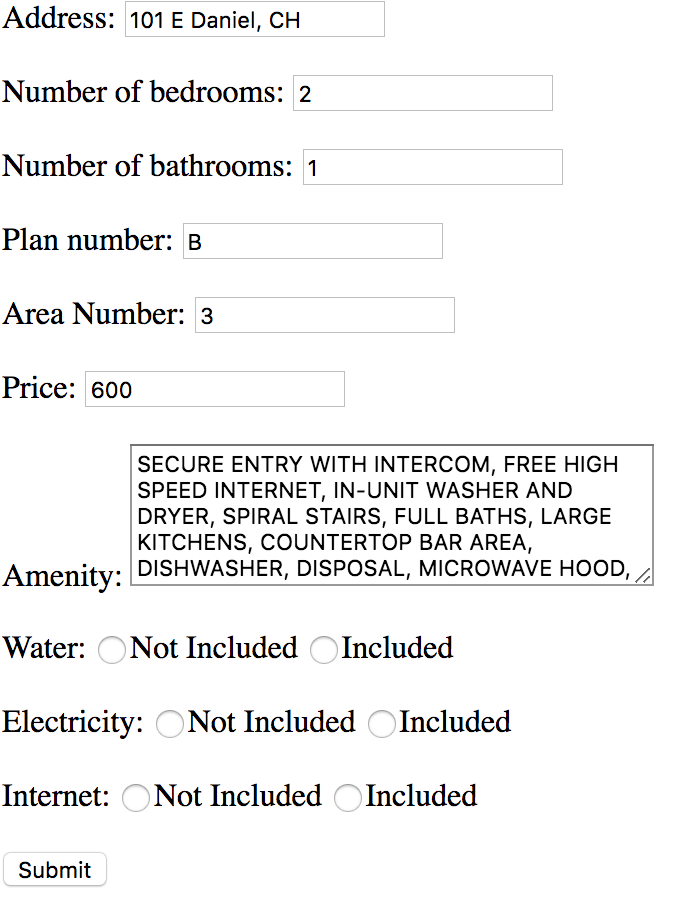
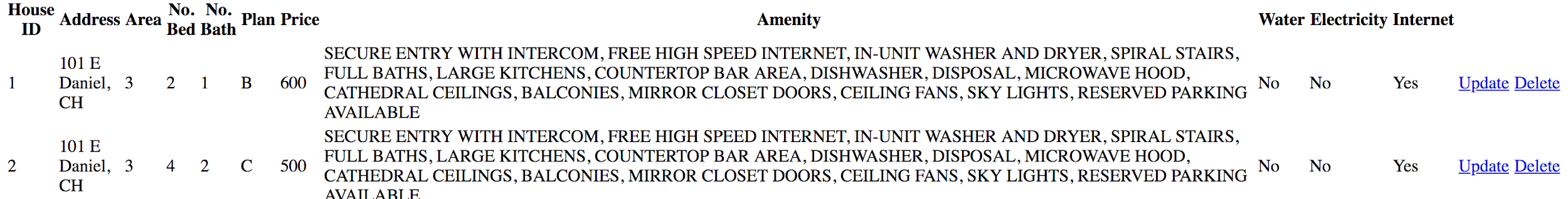
Examples:

Search and inserting new housing information (leasing agents or individual lessors) are directly show up forms to collect data on the main page:



Updating housing information forms need to be popped out only when corresponding “Update” has been clicked:





# 

# 

# 

# 

# 

# 

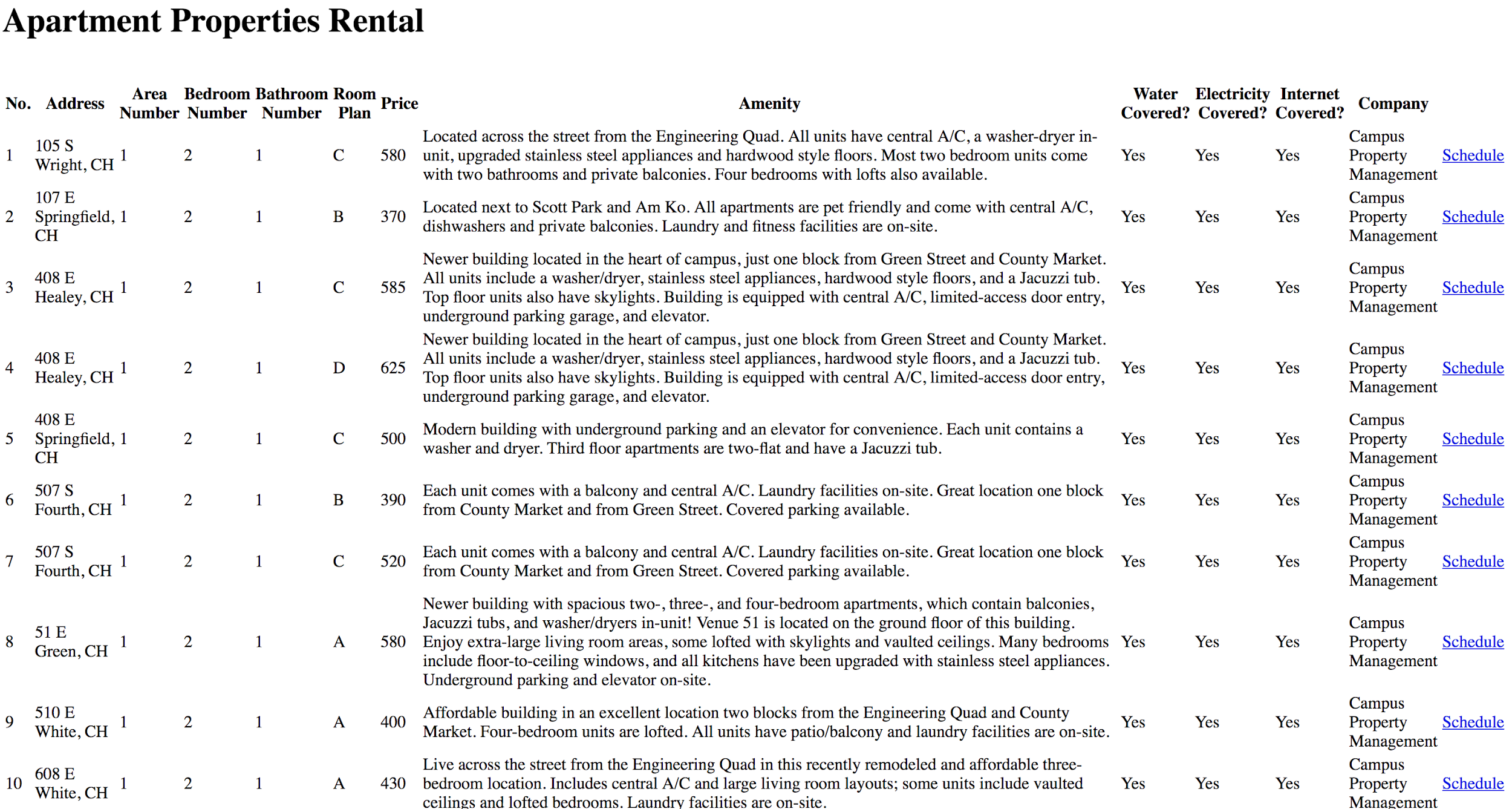
# 

# 

After the user click “Submit” on the website, the data entered in the form, which is constructed in the html file, will be get by the corresponding php file we link to the click button action by method “\_GET”. Then in our php files, we use the data we get from the html forms as the values for our SQL queries.

After getting the results from our SQL queries, we print out our results using “echo” in our php files and the content to echo is in the format of html table, defined by <tr></tr>. This format makes the results easier to read than just simply print out each result.

Example: If we search the houses that are in area No. 1 with 2 bedroom and 1 bathroom and we do want water, electricity and internet all included, but we do not care whether it is a company property or a private apartment, then the search results will be showed in the following way:



# ADVANCED FUNCTIONS DESCRIPTION

* Recommendation:

Our website can build a recommendation list based on the schedule request of the user to help our user to find out the ideal living place in their college life. This is a very useful function on our website, while the user of our website might not have patience to search every potential living place. The algorithm of this function involves to find the mode of the room layout (number of bedroom and number of bathroom), because we need to know what kind of the apartment layout the user are most interested. Besides, the program we implemented allow to find the range of the housing prices that our customer have requested to visit; therefore, the recommendation can have the list of housing with the price that fit the budget of our customer, while prices is always one of most significant factor in housing rental. The recommendation function would build a list of apartment/housing information based on our housing information database and the estimation of housing style due to the user request history to help our user find apartment/house fitted their expectation.

* Google Map API (citations in code):

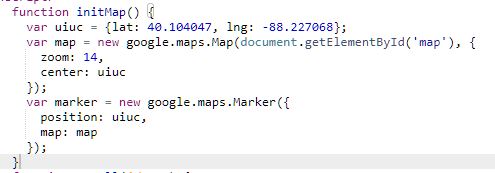
When our users search the apartment, the information list of the apartments can be both seen as the table on the web page and as the markers on the google map api with the housing information if the mouse is placed on the marker. While no one in our group have any experience of Javascript, html, php, or API operations, we did spend a lot of time to figure out how to add the Google Map API to our website with the functionality as our expectation. According to the google api manual, we first acquire a API key to the Google Map which allows us to display/connect the Google map on our website. Later, we defined the basic settings of google map (size, room, etc) and located it at the bottom our website. The most important and difficult task in this function is to output the housing information and display the markers of our housings on the Google map. We use the dom xml as the tool to arrange the data we get from the SQL query as an array/tree. Then, we have to find out the way to use the Google Map Api functions to input the addresses of our apartments and output it as the markers on the map with click-on-shown information (including highest price in this apartment building, the apartment index for the convenience of our user to find more details above the map, and the address of the building).

* Online Scheduler Emailing:

Our website allows the customers to fill the form of schedule request or information request of their target apartment, and our website would automatically help customers send their requests to the Leasing Agent or Individual Lessor. If the signed user of our website click the email button in the My Schedule page, the web page will be redirected to the email form web page. In that page, the customers are required to filling their email address and email content, and they can see the contact email address of the Leasing Agent or Individual Lessor; the email contained the requires of the customers will be automatically sent to the Leasing Agent and the Individual Lessor after a simple click on send button. This function allows the person to person contact for our customers as the replacement of the online chat, while we cannot find the way to implement the online chat functionality on the website in our proposal.

# CONCLUSION

Technical Challenges: We used Google Map API in our project to show the address and price of apartments in the search table with JavaScript. This is totally a new knowledge that we have never learned and used. First of all, you need to get the key from Google API service and it’s free. Then you need to init a map in your html to show it.



Then, in our database, we stored the latitude and longitude as attributes for each apartment. Then you need to convert these values to a XML file for writing data to Google API.



Next, we created a function called “addMarkers” to put each marker onto the map and set the label of each marker to show the address and highest price of that apartment. For this part, you need to read the marker data in from the XML file that you created. For each marker, you need to get the address value of that maker and price for that address. The latitude and longitude values must be converted to float number firstly. And use the google.maps.LatLng to set the point. After this step, you should be able to see a map with some markers on it. But those markers can not show your information. To set the information, please imagine that each marker is separated into different division. And in each division, you can create elements and append them. For example, we created two elements, ‘strong’ and ‘text’, and we set the string that we want to show to them. Then the API can read each element to show on each marker.



Difference from the proposal:

We did not implement online chatting function, because we did not figure out how to send packet between two people. And none of us took related courses for this, so we decided to implement the mail function. And the email function is also useful because it is still an efficient way to contact the lessor for any question.

And secondly, we did not implement the prediction for apartments’ price. This is because we can not get the history prices of each apartment, those companies did not post them online. So we have to abandon this function.

Thirdly, we added a scheduler function which can let normal lessee to schedule a checking time for their desired apartments. We think this is useful and currently, those main apartment companies’ do not apply such service.

Lastly, we changed the ER diagram a little bit. We changed some attributes of some entities to store our data more efficiently.

# LABOR DIVISION

There are three members in our groups: Yiran Li, Anqi Shen and Zheng Liu. Mostly, all of our three members are cooperated on each part. The labor division generally looks like following:

* Yiran Li: Collecting housing data from already-existed leasing companies’ website (MHM, JSM, CPM, ACC, ADV); handling searching housing information; part of API implementation; html designing; writing report.
* Anqi Shen: User sign-up and login; inserting, updating and deleting housing information; handling requests and sending email in normal users; handling showing up messages in leasing agents and individual lessors; part of API implementation; writing report.
* Zheng Liu: Collecting housing data from already-existed leasing companies’ website (MHM, JSM, CPM, ACC, ADV); Recommendation algorithm implementation; html design; modified updating and deleting housing information and schedules; video recording; writing report.