SW Engineering CSC648/848 Section 01 Fall2016

Online Apartment Rental Site Tailored to SFSU Students

GatorRent

Team 05

Local

Soumithri Chilakamarri (Team Lead) schilaka@mail.sfsu.edu

Matthew Wishoff (C.T.O.)

Kevin Fang

Guanming Pan

Jeffrey Ilar

Emil Santos

"Milestone 2"

10-26-2016

History Table

S.No.	Version	Revision Date	Revision Description
1.	1.0.	10-26-2016	First Draft

1. USE CASES

1.1. Student:

Tony is a 20 year old **student** at San Francisco State looking to find an **apartment**.With **GatorRent** he is able to narrow his search to apartments with two bedrooms. Tony also filters apartments by price and pets allowed aswell. In just one hour of using GatorRent Tony finds two suitable aprtments and sends a **message** to both posters.

Sandra uses **GatorRent** to search for leaseholders specifically subletting a single **room.** By using the map feature of **GatorRent** Sandra is able to get a nice idea of the relative distance between the listings and school. Within a day Sandra is able to find several apartments in Park Merced subletting rooms.

1.2. Leaseholder:

Dorothy is a 57 year old **leaseholder** to a **house** near San Francisco State and she would like to advertise it for rent. She does not like long registration processes on websites. Dorothy wants to post **pictures** of the apartment and also set the price she is renting for. She also wants renters to know the house includes its own washing machine and dryer. Lastly she wants to tell renters she is willing to negotiate the price as a **special detail**. **GatorRent** is able to meet all of her needs. After going through the quick registration process; she uploads her pictures, sets the desired price, and indicates the house has a washer and dryer. Before finalizing the posts she writes that the **price** is negotiable. Within three weeks she meets five potential rentees and is able to sublet the house the next month.

1.3. Roommate Finder:

Henry is a **student** at San Francisco State and is looking to replace his roommate. With **GatorRent** he is able to indicate he is looking for a non-smoker who does not drink. Henry also lists specific details of the apartment such as size, number of roommates, the total rental price

that is paid by the roommates each month. Within a week, Henry receives messages from 14 interested **students**.

1.4. Administrator:

Miranda is one of the hired **administrators** of GatorRent. Her job is to review **flagged** postings and deletes them if they violate the site's guidelines. She is also in charge of support and as such she can change user passwords and respond to any user messages. On rare occasions she also deletes user accounts responsible for repeatedly posting offensive content.

2. DATA DEFINITION

2.1. User : Any person who uses GatorRent is a User.

2.1.1. Student : A User who is studying in SFSU is a Student. A Student uses GatorRent for finding the desired room. They may also use GatorRent to find a roommate.

2.1.2. Leaseholder : A User who would like to advertise a space for renting in GatorRent is a Leaseholder.

2.1.3. Administrator : A User who has control over all GatorRent data.

2.1.4. Guest : A User who has not signed into an account. Guests are able to view Listings but can not create or respond to them.

2.2. Account : An Account is the user information in GatorRent. An Account is created when a user signs up in GatorRent. Only users with an account may create or respond to listings. Accounts are created when Users Sign Up.

2.2.1. First Name

2.2.2. Last Name

2.2.3. Email

2.2.4. Password

2..4: Address: The location of the Listing. For Roommate Finder, the address is the location of where the poster lives.

2.4.1. Address

2.4.2. District

2.4.3. City

2.4.4. State

2.4.4. Country

2.4.4. Zipcode

2.5. Listing: Homes open for Students to rent. There are three types of housing available;

single room, apartment, and an entire house.
2.5.1 Number of Bedrooms **2.5.2 Number of Bathrooms** **2.5.3 Floor Size** **2.5.4 Rent** . **2.5.5 Lease Time** **2.5.6 Number of Tenants** **2.5.7 Relative Distance** **2.5.8 Date** **2.5.9 Post Date** **2.5.10 Relative Distance** **2.5.11 Leaseholder ID** **2.5.12 Picture ID** **2.5.13 Adress ID**
2.6. Roommate: A person who will occupy the same housing as another.
2.4.1 Graduation Year **2.4.2 Major** **2.4.3 Minor**
2.7 Picture: A Picture is an image of the housing for rent.
2.4.1 Picture ID **2.4.2 Image File**
2.8 Special Detail: A detail about a Listing written by the Listing creator.
2.9 Flag: A listing that is marked for Administrative review.

3. LIST OF FUNCTIONAL SPECIFICATIONS

Section	Functional Specification	Priority
3.1.0	The **Student** shall be able to search the website for rental listings	1
3.1.1	The **Student** shall be able to filter listings by marking box settings that fit their needs	1
3.1.1	The **Student** shall be able to create an account	1
3.1.2	The **Student** shall be able to login to an account	1
3.1.3	The **Student** shall be able to contact a Leaseholder or other Students	1
3.1.4	The **Student** shall be able to view pictures	1
3.1.5	The **Leaseholder** shall be able to create an account	1
3.1.6	The **Leaseholder** shall be able to update their posting	1
3.1.7	The **Leaseholder** shall be able to login to their account	1
3.1.8	The **Leaseholder** shall be able to list posting(rentals)	1
3.1.9	The **Leaseholder** shall be able to post pictures of their rental	1
3.1.10	The **Leaseholder** shall be able to specifically lease the room to SFSU students	1
3.1.11	The **Administrator**s shall be able to log into an administrative account through MySQL Workbench	1
3.1.12	The **Administrator**s shall be able to delete posts	1
3.1.13	The **Administrator**s shall be able to view an admin panel	1
3.1.14	The **Administrator**s shall be able to remove/ban users	1

3.2.0	The **Student** shall be able to find the approximate location of renting apartments using Google Maps while preserving privacy of landlords	2
3.2.1	The **Student** shall be able to create postings looking for roommates with Roommate Finder	2
3.2.2	All **Users** shall be able to retrieve thier password if it is forgotten	2
3.2.3	The **Leaseholder** shall be able to mention special details	2
3.3.0	The **Student** shall be able to report false claims	3
3.3.1	The **Student** shall be able to favorite certain rentals	3
3.3.2	The **Leaseholder** shall be able to post videos of their rental	3
3.3.3	The **Administrator**s shall be able to change passwords	3
3.3.4	The **Administrator**s shall be able to communicate with Students/Leaseholders	3

4. LIST OF NON FUNCTIONAL SPECIFICATIONS

4.1 Security

4.1.1. Application shall be hosted and deployed on Amazon Web Services as specified in the class

4.1.2. Data shall be stored in the MySQL database on the class server in the team's account

4.1.3. Application shall be served from the team's account

4.1.4. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.

4.1.5. Messaging between users shall be done only by class approved methods to avoid issues of security with e-mail services.

4.2. Performance

4.2.1. No more than 50 concurrent users shall be accessing the application at any time

4.2.2.Google analytics shall be added for major site functions.

4.2.3. Site security: basic best practices shall be applied (as covered in the class)

4.3. Optimization

4.3.1. Application shall be optimized for standard desktop/laptop browsers, and shall render correctly on the two latest versions of all major browsers: Mozilla, Safari, Chrome. It shall degrade nicely for different sized windows using class approved programming technology and frameworks so it can be adequately rendered on mobile devices

4.3.2. The language used shall be English.

4.3.3. Application shall be very easy to use and intuitive. No prior training shall be required to use the website.

4.3.4. Pay functionality (how to pay for goods and services) shall be simulated with proper UI, no backend.

4.4 Best Practices

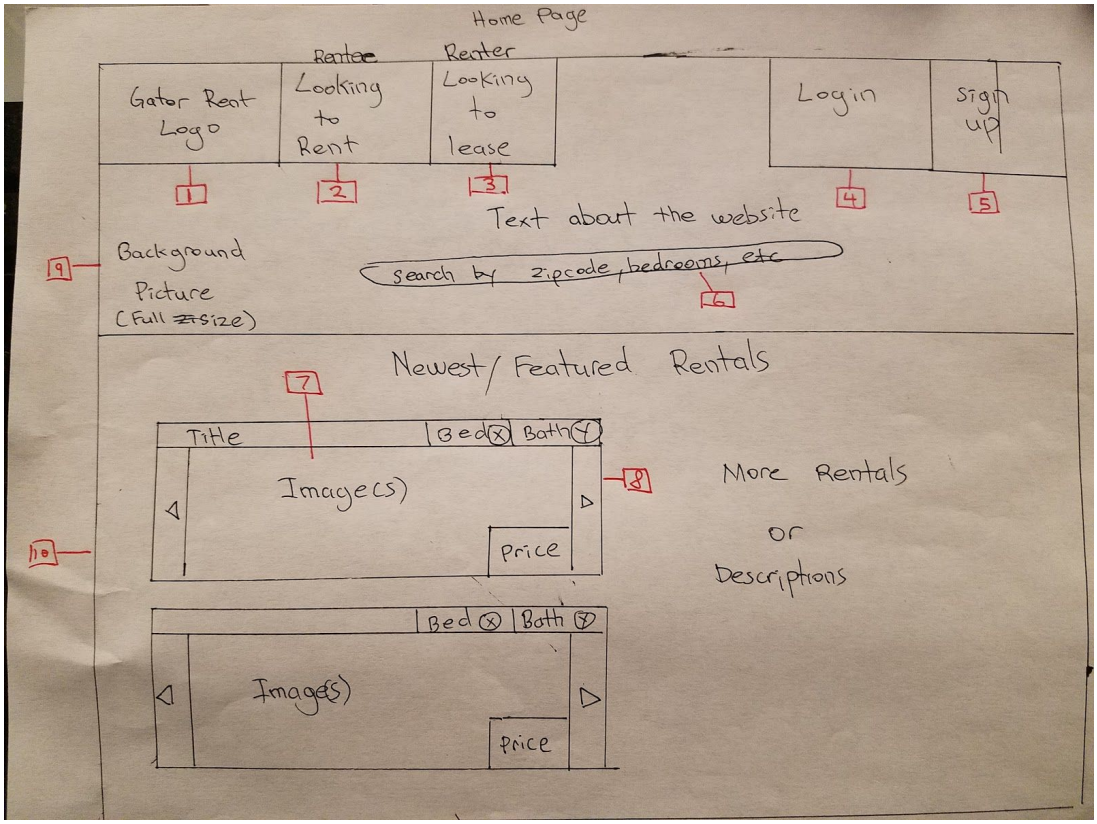
4.4.1. Application shall be developed using class provided LAMP stack

4.4.2. Application shall be developed using pre-approved set of SW development and collaborative tools provided in the class.

4.4.3. The website shall prominently display the following text on all pages "SFSU Software Engineering Project, Fall 2016. For Demonstration Only". (Important so as to not confuse this with a real application)

5. UI Mockups and Storyboards

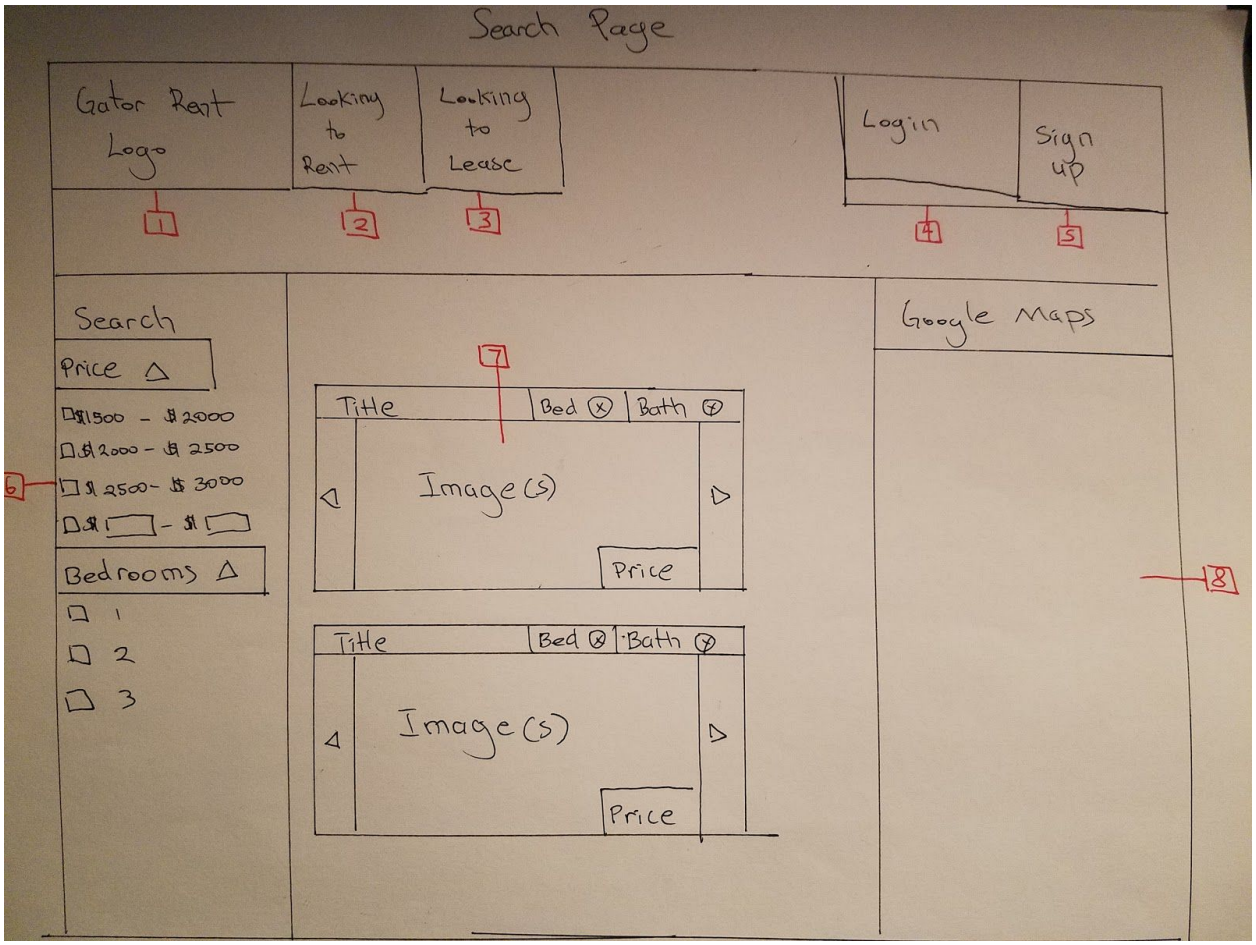
Home Page



Note: Tabs are part of a Navigation Bar

1. Home button titled Gator Rent (with a logo)
2. Links to a page where you can search for rentals
3. Links to a page where you can create a listing
4. Opens a login window
5. Opens a sign up window
6. Search bar
7. Clickable image of a posting
8. Navigate through various photos
9. Initial landing page
10. List of newest rentals

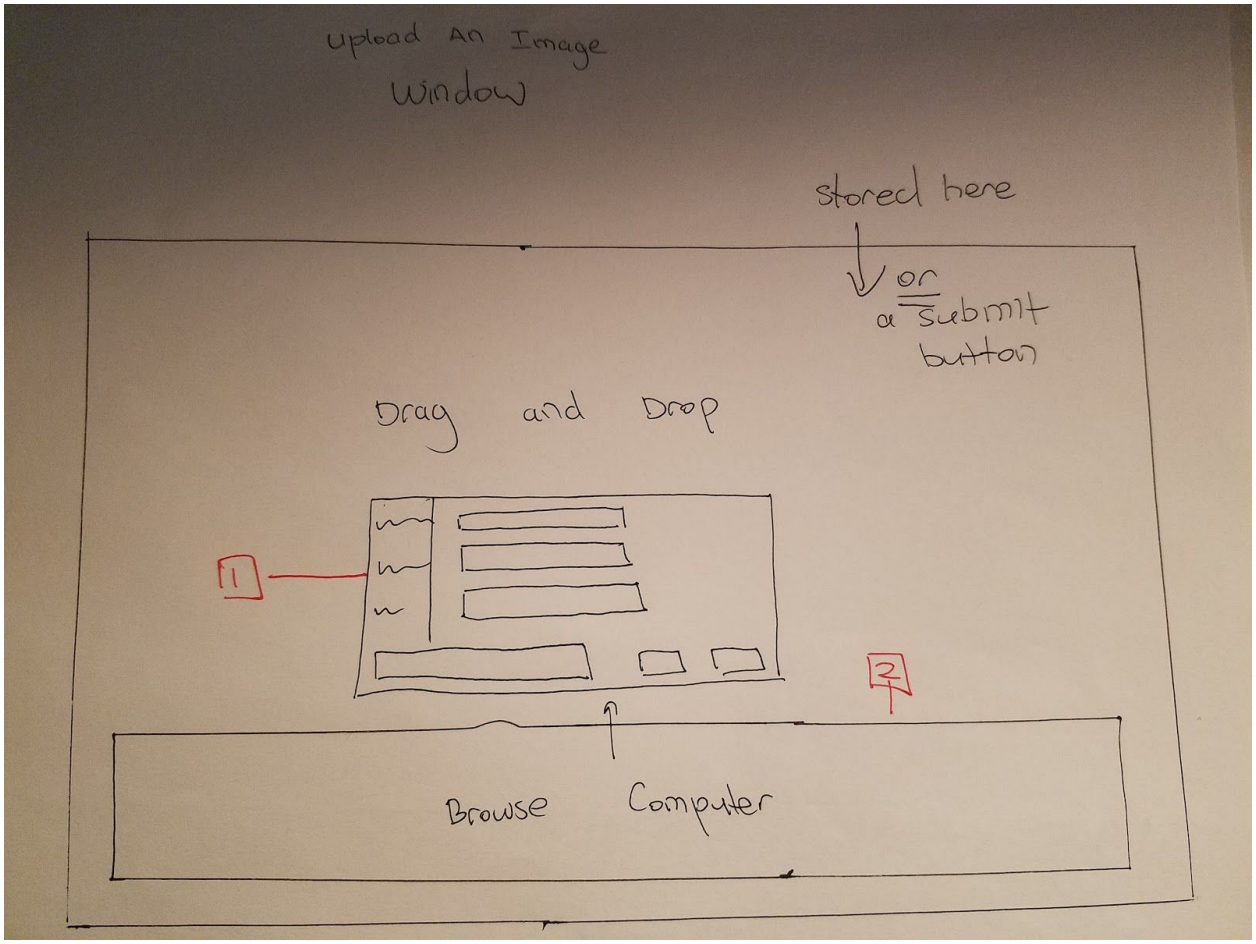
Search Results Page



Note: Tabs are part of a Navigation Bar

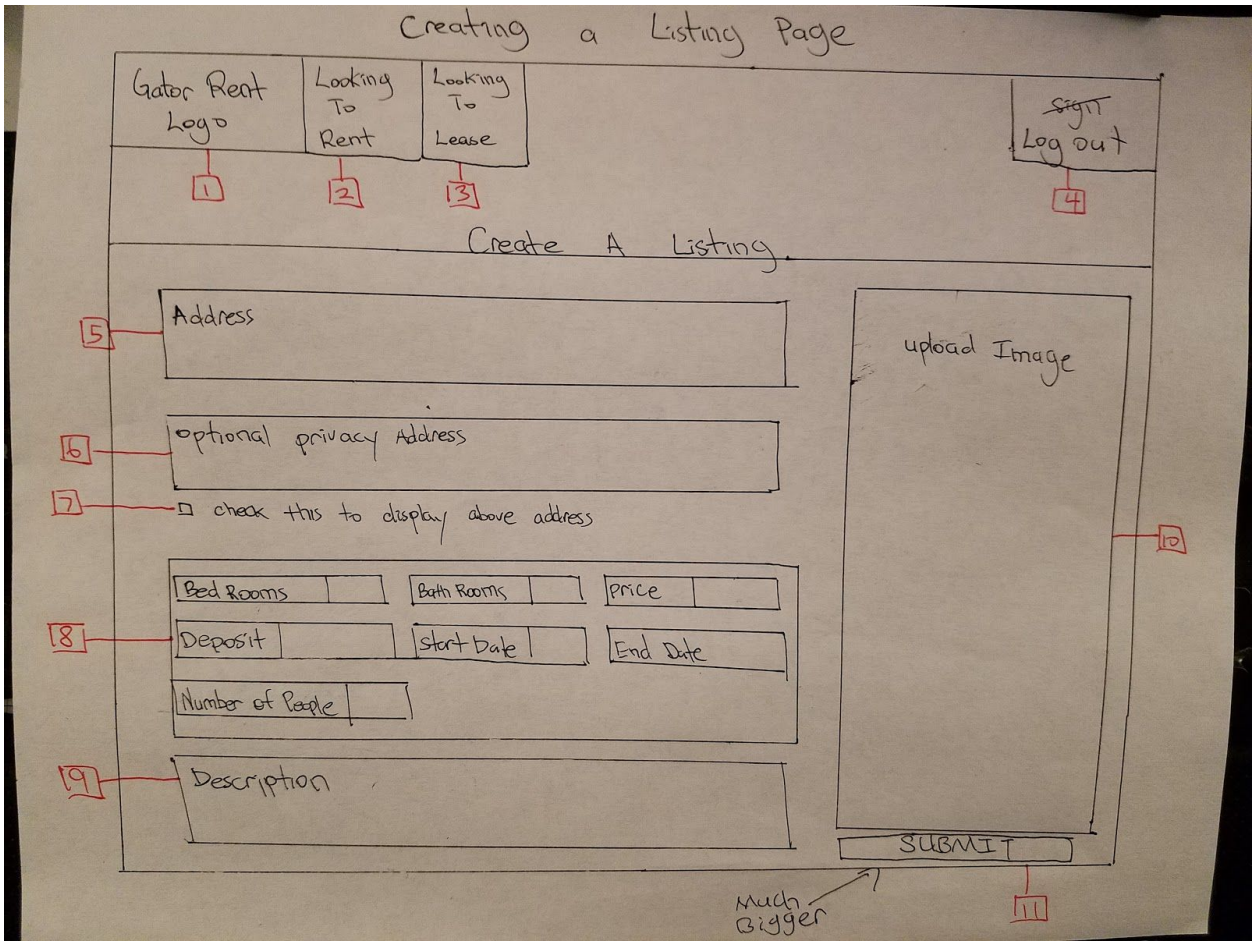
1. Home button titled Gator Rent (with a logo)
2. Links to a page where you can search for rentals
3. Links to a page where you can create a listing
4. Opens a login window
5. Opens a sign up window
6. Filters that allow the user to adjust their search
7. Sample information of a rental
8. Location of where the rental is on google maps.

Upload Page



1. Upload image by searching through computer folder system.
2. Upload image by dragging it into the box.

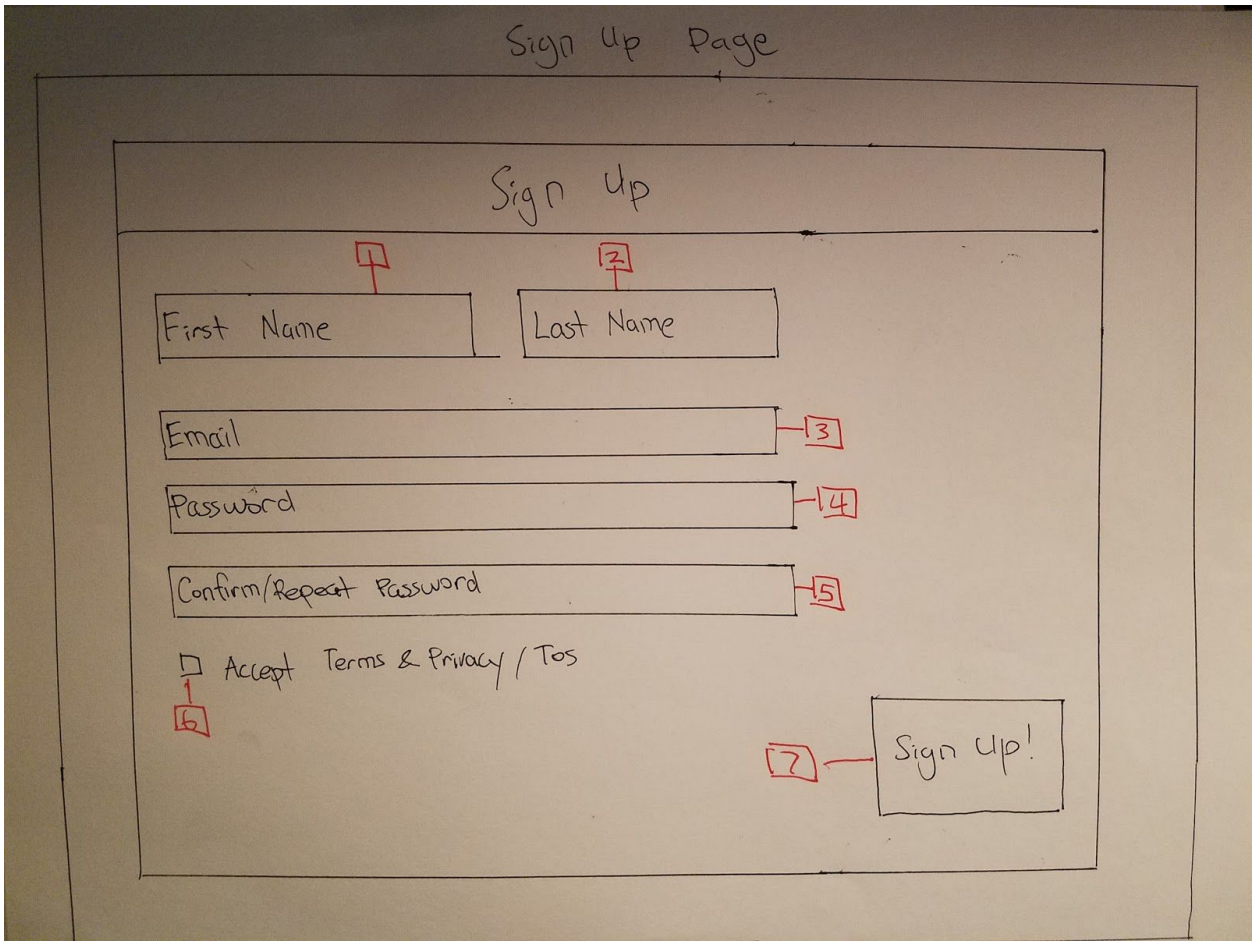
Create a Listing



Note: Tabs are part of a Navigation Bar

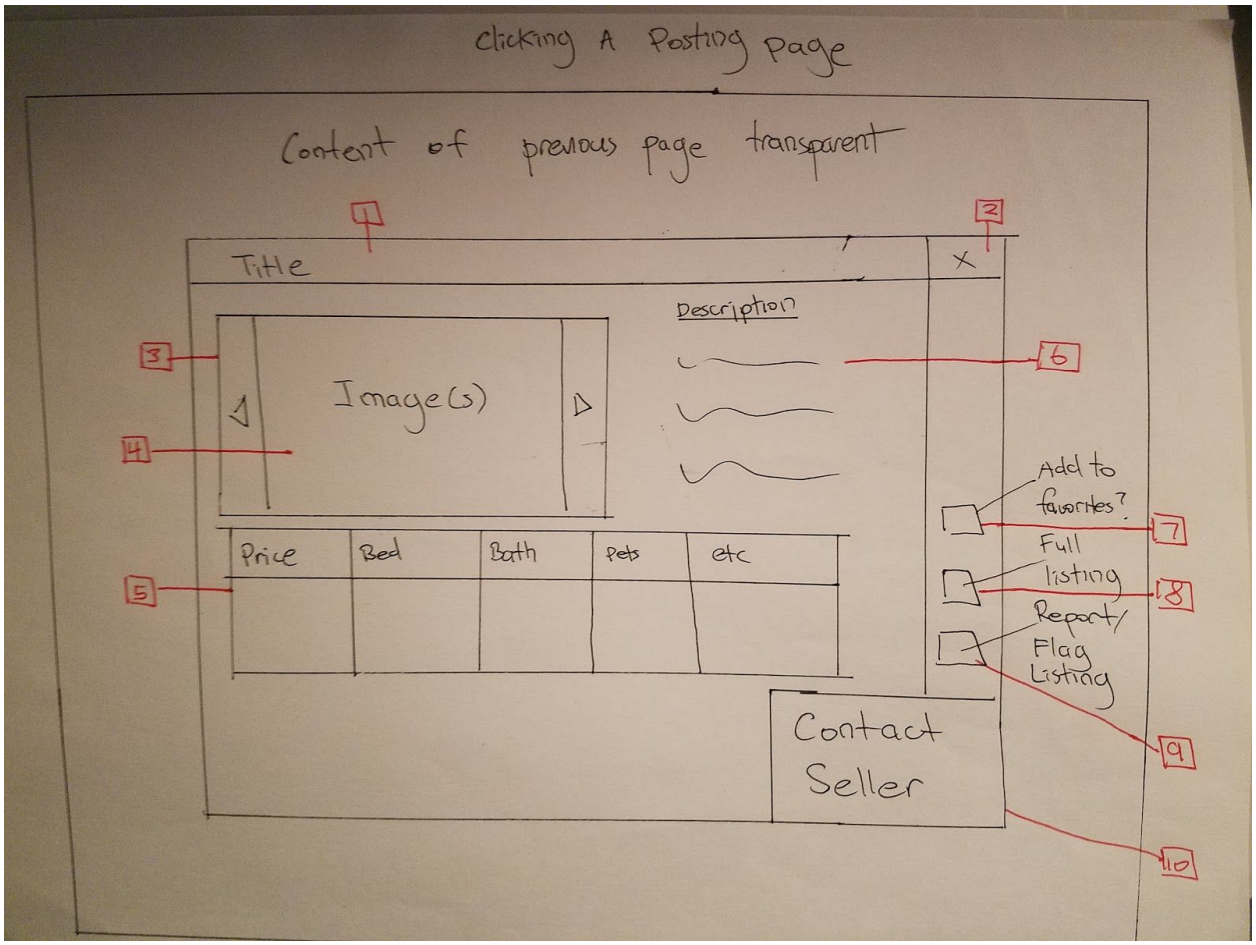
1. Home button titled Gator Rent (with a logo).
2. Links to a page where you can search for rentals.
3. Links to a page where you can create a listing.
4. Logs user out of their account.
5. Prompts user for address.
6. Prompts user for an alternative address they'd like to use..
7. Button that allows user to use their alternative address information.
8. Prompts user for basic house information.
9. Prompts user for description of the house.
10. Upload an image or images of the house.
11. Button to submit the listing.

Sign up Page



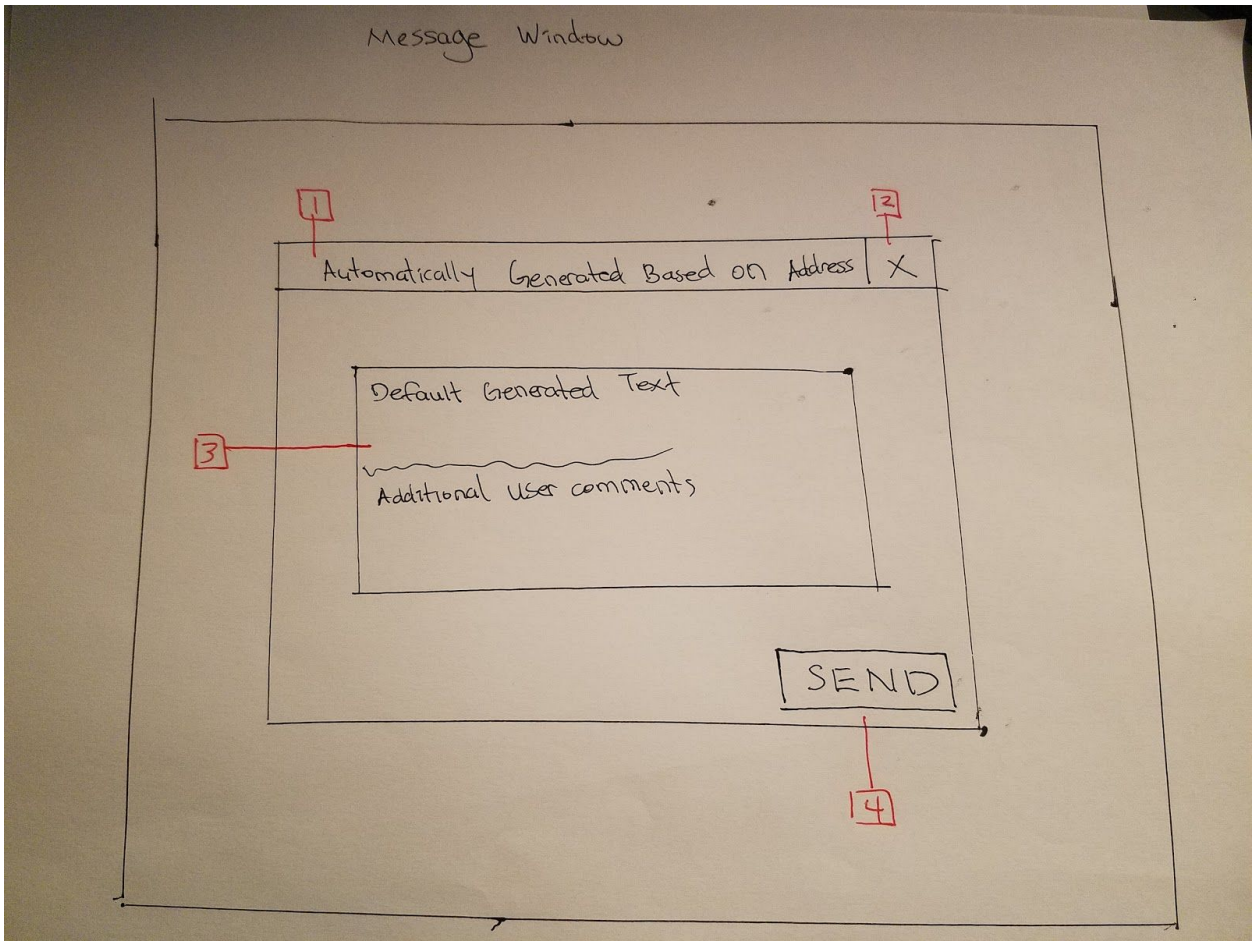
1. Prompts user for first name.
2. Prompts user for last name.
3. Prompts user for email.
4. Prompts user for password
5. Prompts user to confirm password
6. Forces users to accept TOS and Privacy agreements
7. Sign up button/Complete.

Clicking a Posting window



1. Title of posting.
2. Exit window and return to your original page.
3. Arrows to move slider.
4. Clickable images.
5. Specific details about the rental.
6. Description of the rental.
7. Button to add listing to favorites/remembered
8. Button to see the full listing
9. Button to flag/report listing
10. Contact the seller

Message window



1. Automatically generated title based on address
2. Exit the window
3. Automatically generated initial text, user may enter more text.
4. Button to send message.

6. HIGH-LEVEL SYSTEM ARCHITECTURE, DATABASE ORGANIZATION

The System architecture for **GatorRent** uses two servers - ***sfsuswe.com*** and ***sweng.education***. GatorRent is developed and hosted on ***sfsuswe.com*** sever, which consists of LAMP stack, MySQL database, LINUX shell accounts and MINI php frameworks. The other server, ***sweng.education*** has GitLab accounts used as a cross-communication platform for code deployment and QA testing.

The front-end development uses LAMP stack tool and Bootstrap framework. So, what is LAMP stack? **LAMP stack** tool is a popular open source web platform commonly used to run dynamic web sites and servers. It is very flexible choice for developing web applications which need high performance and reliability. It is Linux, Apache, MySQL, PHP, Python, Perl compatible platform.

Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML and CSS based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only.

Every website we find on internet needs to be developed has a foundation of skeleton structure. Likewise, the skeleton structure for our website is MINI. **MINI** is an extremely simple and easy to understand skeleton PHP application. Why MINI? Well, the straight-forward answer would be MINI is easy to install, runs nearly everywhere and doesn't make things more complicated than necessary.

In this project, the back-end development uses PHP as the scripting language. What is PHP? Why do we use it? Firstly **PHP**, *Hypertext Preprocessor*, is a server-side scripting language basically designed for web development. PHP code is interpreted by a web server via a PHP

processor module, which generates the resulting web page. For the later question, the answer is that PHP is free and easy to code. PHP, though GNU General Public License (GPL) incompatible, can be embedded into HTML code and can be used in combination with various web template systems, web content management systems and web frameworks..

The data for the website is stored in ***sfsuswe.com*** server. **MySQL**, an Open Source Relational SQL database management system, is hosted on this server. It is one of the best RDBMS being used for developing web-based software applications in terms of usage and practice.

HTML, **CSS** and **jQuery** are the scripting languages used at the front-end design of GatorRent. **HTML** (HyperText Markup Language) is the basic building block of most web pages. **CSS** (Cascading Style Sheet) is combined with HTML to allow the developers to style each element individually, or group elements together for a uniform look. **jQuery** is another tool which will be used to design GatorRent. jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications.

The compatible browsers for GatorRent are:

- Mozilla Version: 44.0.2 and 45.0
- Chrome Version: 49.0.2623 and 48.0.2564

For the other server, ***sweng.education***, has **GitLab** which is an application to code, test, and deploy code together. It provides Git repository management with fine grained access controls, code reviews, issue tracking, activity feeds, wikis, and continuous integration. GitLab consists of Repository, a file storage manager, and Master which is a branch that contains the entire code of the website.

During the last 2 weeks of semester, the website will be hosted on **Amazon Cloud**, a cloud storage application managed by Amazon. The service offers secure cloud storage, file backup, file sharing, and Photo printing. Using an Amazon account, the files and folders can be

transferred and managed from multiple devices including web browsers, desktop applications, mobiles, and tablets. The Amazon server specifications is listed below:
Hardware 64 bit Intel(R) Xeon(R) CPU E52651 v2 @ 1.80GHz, 4GB RAM.

APIs

Google Maps API : The Google Maps API allow for the embedding of Google Maps onto web pages of outside developers, using a simple JavaScript interface or a Flash interface.

Google Analytics API : The Google Analytics Embed API is a JavaScript library that allows you to easily create and embed a dashboard on a third-party website in a matter of minutes. It gives you a set of pluggable components that can work together to build complex tools, making it both simple and powerful at the same time.

Data Organization:

Image Format/Size: Format GIF, JPG and PNG will be the accepted image formats for our website. The resolution of the image file is 1600 x 900 ppi and the size is 317.42 KB.

Thumbnail Format: Thumbnails will be saved as images of the same format as the original in a maximum of 144 pixels in the longest direction.

Metadata:

Image information regarding the filename, the size, the resolution, the upload date, the thumbnail path, the filename of the thumbnail will be stored in the file system. The images are stored in file systems not in blobs. An advantage of file system approach is the efficiency and performance of data retrieval from the database.

Related Data:

The data of the users will be stored in database tables. Whenever a user signs up, new user data record is generated and stored. The User_id of a user is unique like the email address and password. For the user who are students, we store additional information such as grad_year, minor and minor.

Search Architecture:

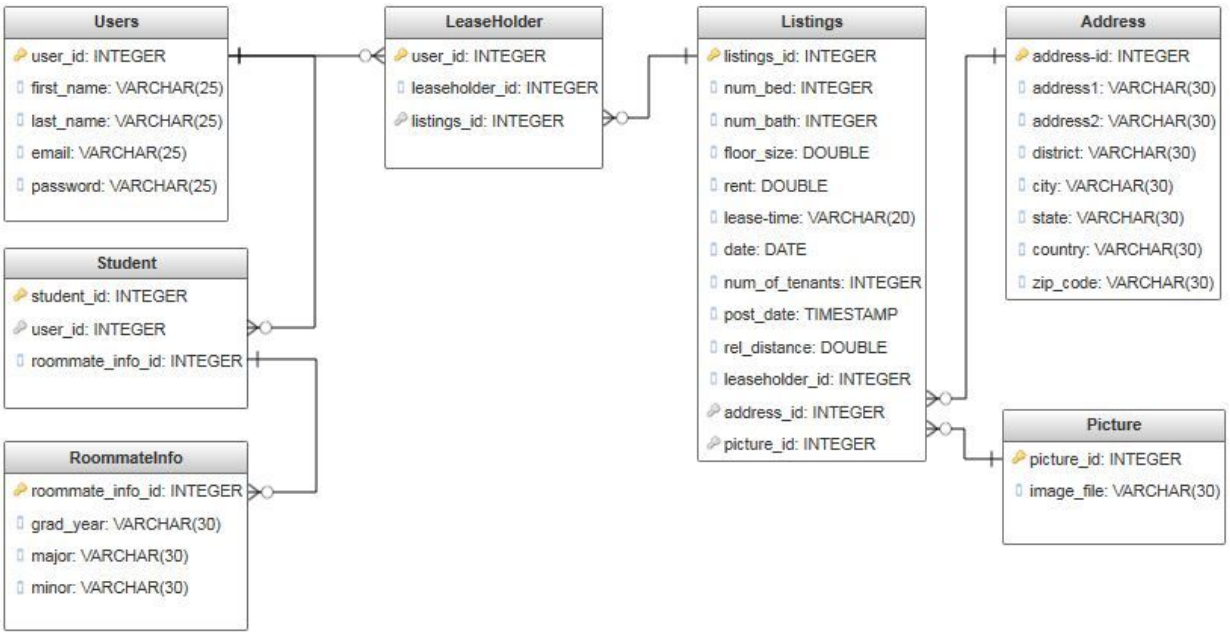
GatorRent uses %LIKE search architecture. Using the LIKE keyword, we pass in the percentage sign (%), which is a wildcard character that returns zero or more characters and our name variable from the search field. As a result, the LIKE keyword (in conjunction with our wildcard character) will find any name that matches in our database table.

For example,

$sql="SELECT user_id, first_name, last_name FROM Users WHERE first_name LIKE '%" . $first_name. "%' OR last_name LIKE '%" . $last_name ."%'";

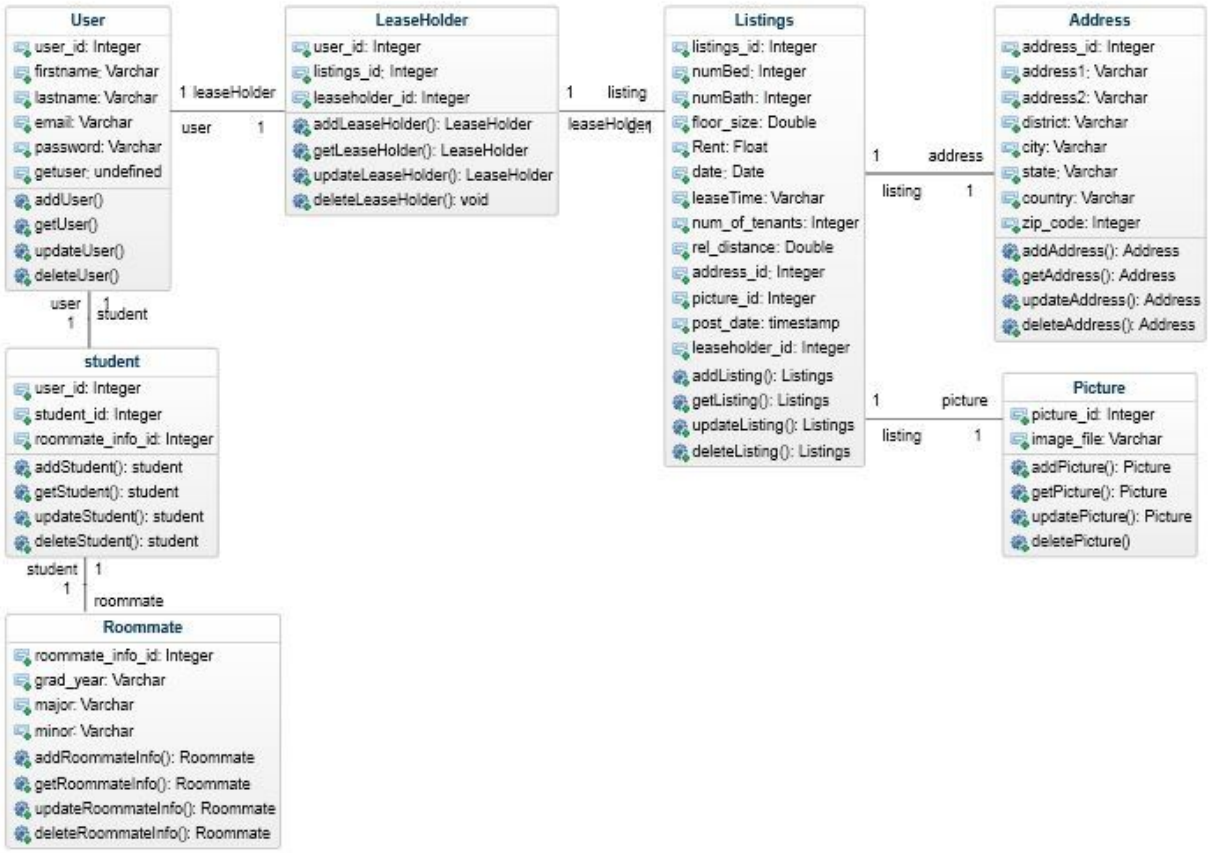
The above query retrieves the list of all the user data from database when either first name or last name of the user. Similarly, GatorRent does a %LIKE search when a user enters the area (apartments to be found in that area) in the search bar. Say that the user enters ‘park’ in the search bar, GatorRent displays all the listings present in ‘park ’ area.

Database Organization

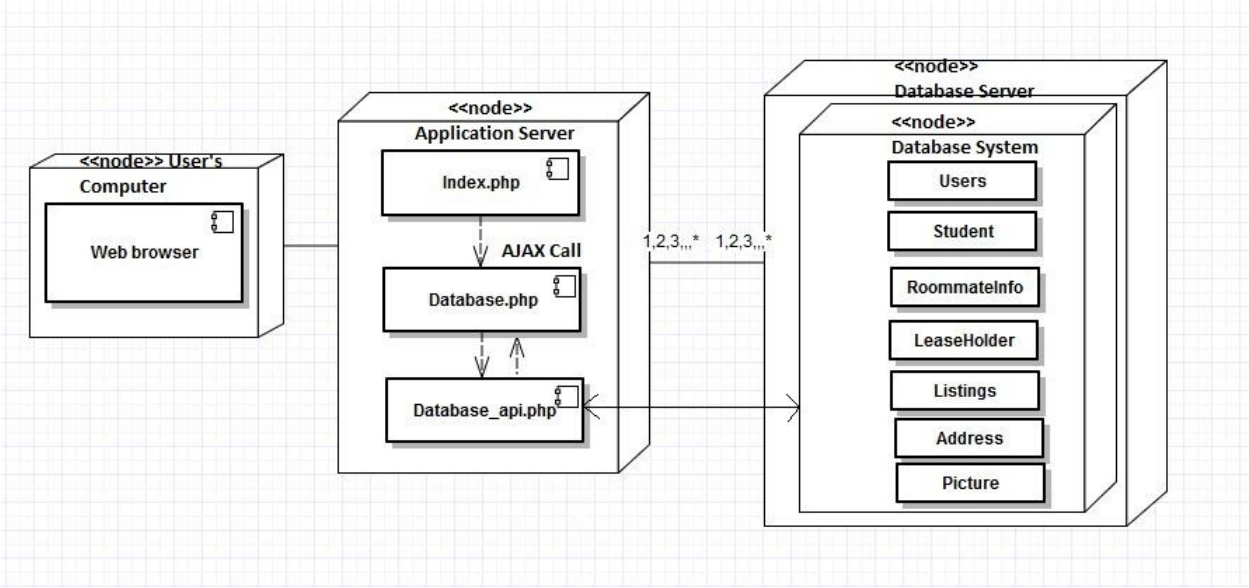


7. UML Class Diagram

7.1. High Level UML Class Diagram:



7.2 UML Component and Deployment Diagram



8. High Level APIs

User

addUser() : adds a new user account to the database.

getUser() : fetches the details of user from the database

updateUser() : edits the attributes of user to the database

deleteUser() : deletes the user account from the database

Student

addStudent() : adds a new student account to the database

getStudent() : retrieves the student account from the database

updateStudent() : updates/edits the student account in the database

deleteStudent() : deletes the student account from the database

Roommate

addRoommateInfo() : adds a new roommate information (like grad year) to the database

getRoommateInfo() : retrieves the roommate information account from the database

updateRoommateInfo() : updates/edits the roommate information account in the database

deleteRoommateInfo() : deletes the roommate information account from the database

LeaseHolder

addLeaseHolder() : adds a new LeaseHolder account to the database

getLeaseHolderr() : retrieves the LeaseHolder account from the database

updateLeaseHolder() : updates/edits the LeaseHolder account in the database

deleteLeaseHolder() : deletes the LeaseHolder account from the database

Listings

addListing() : adds new Listings (like Rent) to the database

getListing() : retrieves the Listings from the database

updateListing() : updates/edits the Listings in the database

deleteListing() : deletes the Listings from the database

Address

addAddress() : adds a new address information (like city) to the database

getAddress() : retrieves the address information from the database

updateAddress() : updates/edits the address information in the database

deleteAddress() : deletes the address information from the database

Picture

addPicture() : adds the new picture information (like name) to the database

getPicture() : retrieves the picture information from the database

updatePicture() : updates/edits the picture information in the database

deletePicture() : deletes the picture information from the database

9. Key Risks

Skill Risks:

PHP

PHP is new to majority of our team since no one worked on it before. Learning PHP is one of the vital tasks for both, the frontend and the backend, since many website developers today use the language to increase their site's performance. In order to bridge this gap, our team worked together to complete PHP course in codeacademy.com. In addition, we also used the online resources such as w3schools.com, stackoverflow.com to clarify few questions we could not understand.

MySQL Workbench 6.3

All of our team members have worked with MySQL using the Command shell or the UNIX shell but none of them have worked MySQl using the Workbench. Designing the database schema is a major task and few of our members took this task of using Workbench to design databases while guiding the inexperienced members.

GitLab

Only few of our team members have prior experience using the GitLab software. Even though the git commands are similar to GitHub, a lot of care must be taken while pushing the code to the repository. Extreme care should be taken while pushing and committing to a branch since any small code conflict in master will cause the whole website to crash. We have organized a small session to learn GitLab and its functionality so that our team members can communicate within ourselves while pushing the code into development branch to ensure that the website does not crash.

Schedule Risks:

The time and resources are the two major factors involved in creating the website. Taking time into consideration, our team members have work and other classes and is a potential risk in completing the work. So, we decided to meet every Thursday outside of class for an hour. This meeting is a scrum where we would discuss on the progress of each member and the future tasks we have to complete. If any member needs assistance, all the team members sit together and help to solve the issue. Like this, we are dedicated in finishing all the milestones set forth in the project.

Technical Risks:

Cross-platform compatibility

A technical issue is the Cross-platform compatibility. It's hard to get websites to look and function perfectly on the browsers taking considerations of the orientation and the screen size. When we try to optimize for a browser, it may de-optimize for another. In order to ensure that GatorRent works on the specified browser, our team would make the website to be standard-compliant.

Teamwork Risks:

An important teamwork risk would be the mutual agreement between the front-end and the back-end members. Although the front-end and back-end work independently, there might be situations where we should work together. Some of the decisions made by the front-end team may not agree with the back-end team. We resolve these issues by taking the factors such as risks and time involved in both the approaches. Later the best possible approach is decided by the entire team to deliver the output as estimated.

Legal/Content Risks:

GatorRent allows the Leaseholders to post their pictures of rental apartments. However, there may be spam users who upload the images which that violate our terms and conditions. To resolve this issue, we provide our users to flag their posts which they find inappropriate. The Admin will check the flagged posts and removes these spam posts and block users that violate our terms and conditions.

10. TEAM

1. Soumithri Chilakamarri : Team Lead and front-end developer for GatorRent
2. Matthew Wishoff : Tech Lead and Back-end developer for GatorRent
3. Kevin Fang : Front-end developer of GatorRent
4. Guanming Pan : Front-end developer of GatorRent
5. Jeffrey Ilar : Back-end developer of GatorRent
6. Emil Santos : Back-end developer of GatorRent