**Design Documentation**

**System Architecture**

**Architectural patterns used**

We are using the Client-Server architectural pattern as using multiple servers allows a high capacity of users and if demand for the system increases, it will make it easier to add servers to cope for the extra processing. Servers can also be easily changed without changing the entire system (Sommerville, 2011, p. 161). Several servers are used to display the website, house listings and data, save and retrieve backups and also retrieve user data. This separates each server from each other making management of each a lot easier as well as making each individual server much more efficient than just one machine with multiple servers handling all the processing. This is great as we identified that our system requires high performance so that it can search a database and return results to the user quickly. Each server is a single point of failure and is prone to hacking or downtime, so to prevent loss of data we have included a backup server in the design as well.

The type of application architecture used is that of a layered information system. We made this decision based on the recommendations that were made in Software Engineering (Sommerville, 2011, pp.157-159). The system we are making provides a massive database of house listings that will need to be accessible from the web server (when the user queries the database via the website).Each layer provides certain functionality and is accessible by the above and below layers. When a query is sent, it will go through each layer until the correct information is brought forth from the correct databases and storage systems. Then the data received will be sent to the web server and thus to the user’s browser.

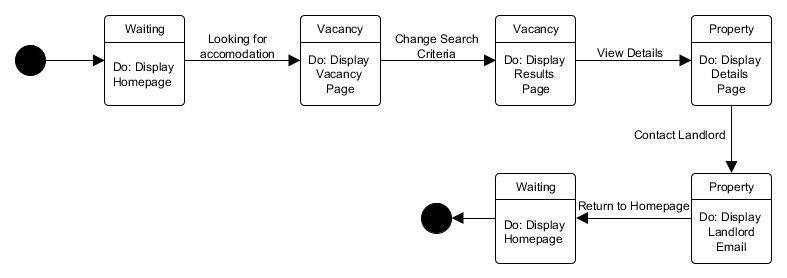
The web server is in charge of communicating with the user and implementing the user interface for the user's browser as well as sending/receiving and presenting data from the other servers. The database servers (and picture server) are separate due the fact that we want security on just our user database and to reduce the risk of losing data if a server did go down. This is good as our need of availability of the system based on our non-functional requirements was strong. Each database server (and picture server) moves information to and from each of their respective databases and the web server.

|  |  |
| --- | --- |
| **Client-Server layout (Figure 1)** | **Layered Information System Architecture (Figure 2)** |

**Representation**

**State Diagram (Figure 3)**

This diagram represents one example of a user interacting with the system, and shows the transitions between different system states that this interaction results in.

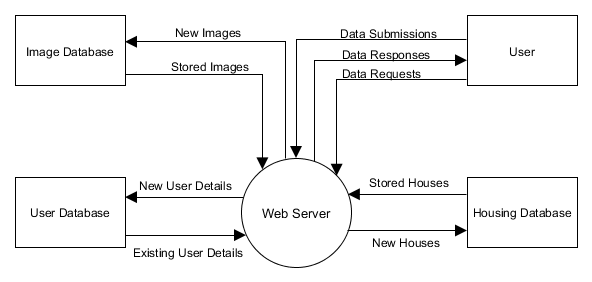


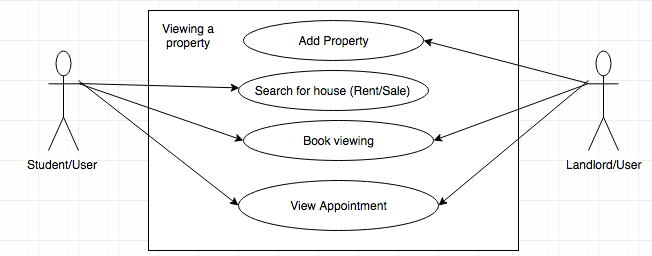
**Class Diagram (Figure 4)**

Class diagram showing relations between the main parts of our system



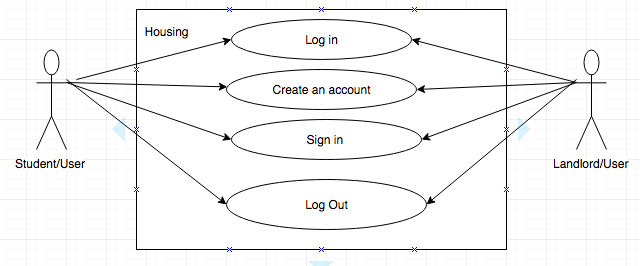
**Context Diagram of the System (Figure 5)**

Context diagram showing the connections between the central web server and other components of the system.

**Use Case Models** 

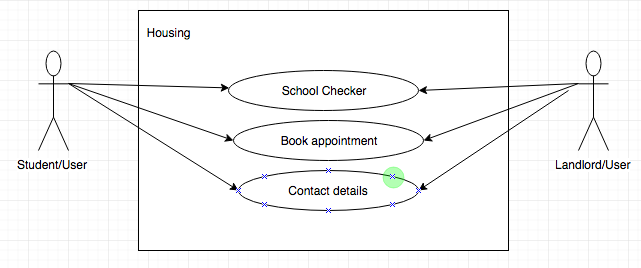
**Figure 5.1**

In this diagram it shows how the user/Landlord adds a property which then the user/student is able to search for a particular proper which is either up for rent or sale. The users can book a viewing the other user/landlord can viewing the bookings and also they are able to view to the whatever appointment thats are made from viewings.



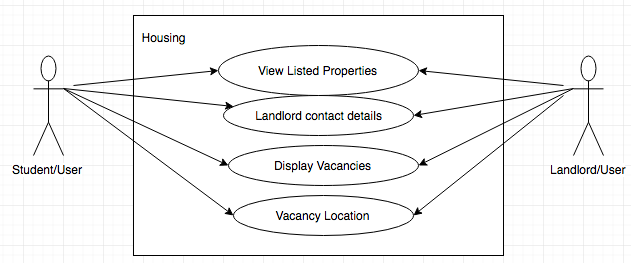
**Figure 5.2**

The diagram shows how the the student/user will have to log in or either create an account to be able to access the website fully. The same principle applies for the landlord/user they will also have to have an account before they will be able to post properties online.



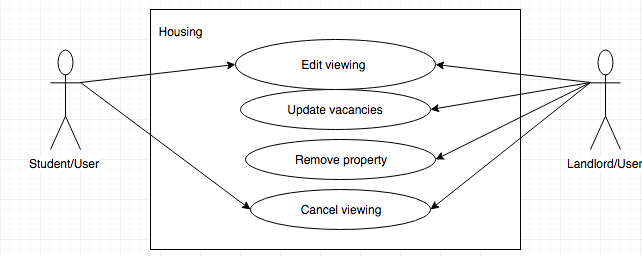
**Figure 5.3**

This diagram shows how the users are able to view how far the property they looking for is from the university. They can also contact the agency or contact the landlord/user directly to book an appointment if needed.



**Figure 5.4**

In this diagram the student/user is able to view the listed properties on the website. Then contact the landlord/user if needed. Both users will are able to view the location of listed vacancy that have be displayed on the website.

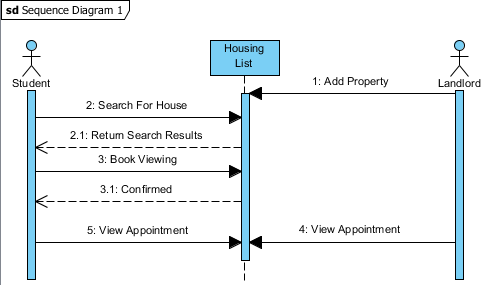


**Figure 5.5**

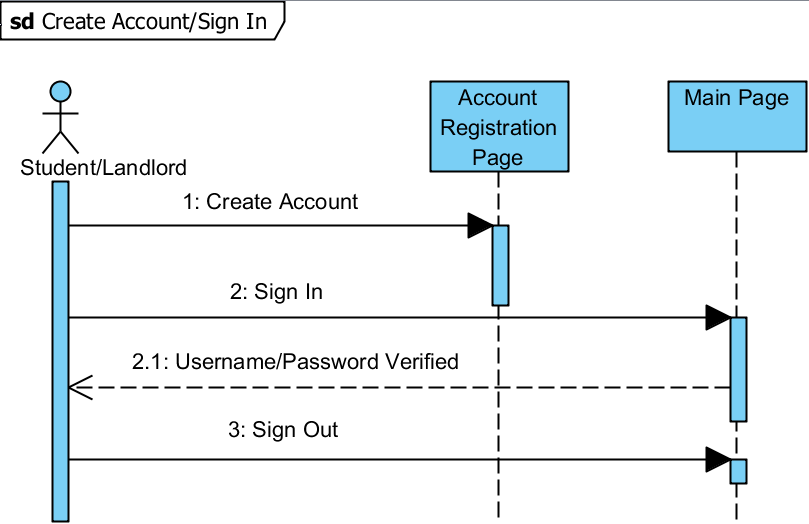
In this particular use case diagram the Student/User can edit the viewings that they've booked on a particular property, the same principle applies for the Landlord/user the can also edit a scheduled viewing. The landlord/user can update a vacancy on the site to fit a certain criteria the can remove the the listed property and both users are able to cancel on a scheduled viewing for a particular property.

**Sequence Diagrams**

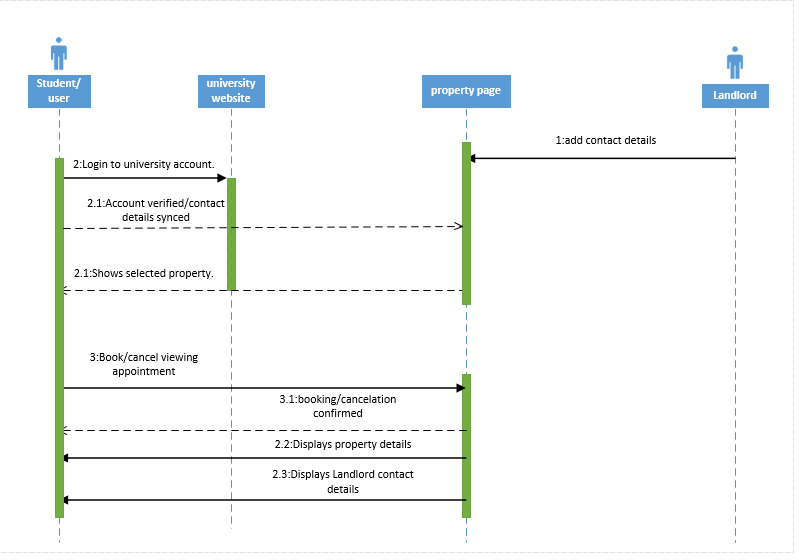
**Figure 6.1 - Based on use case Figure 5.1**



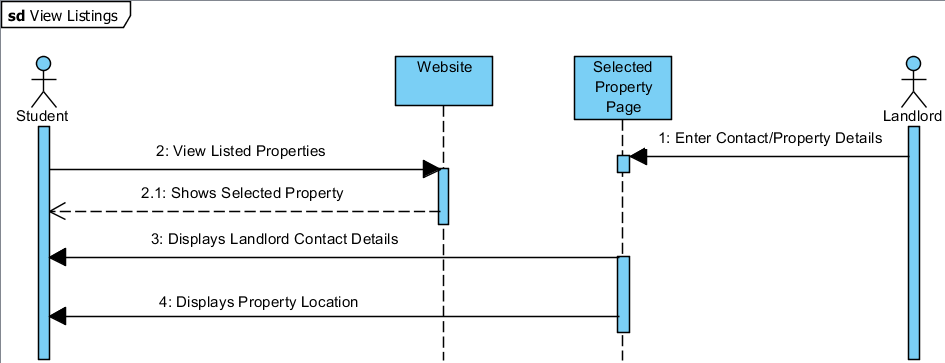
**Figure 6.2 - Based on use case Figure 5.2**



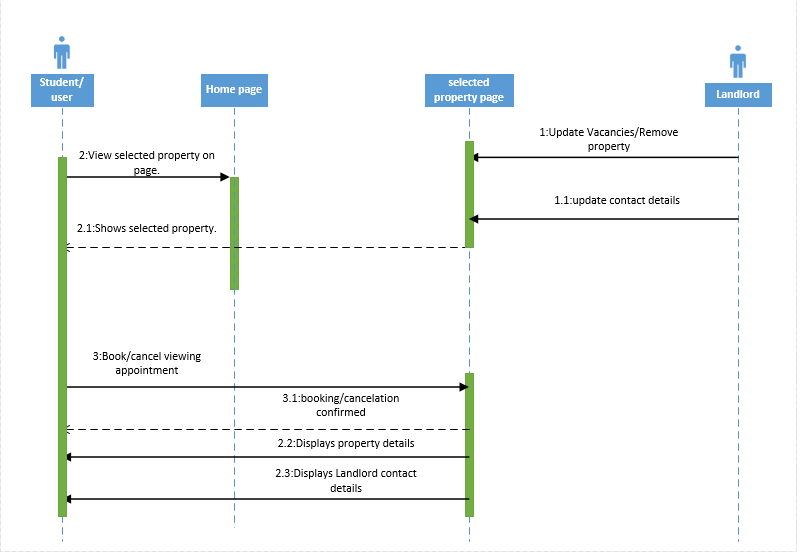
**Figure 6.3 - Based on use case Figure 5.3**



**Figure 6.4 - Based on use case Figure 5.4**



**Figure 6.5 - Based on use case Figure 5.5**



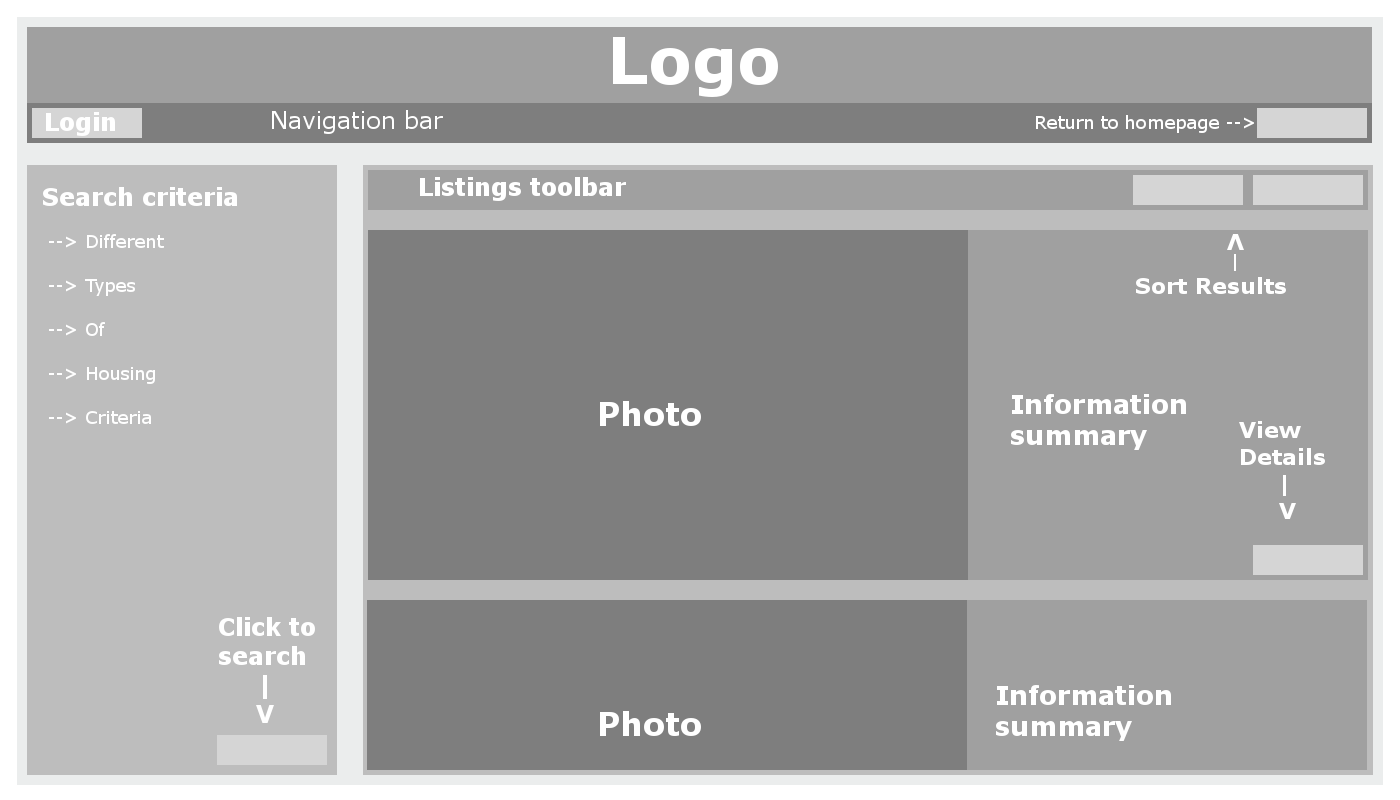
**UI Mock Ups**

The following images are just drafts and prototypes, the design may still change upon completion. This may depend on certain conclusions of usability that we have yet to make.

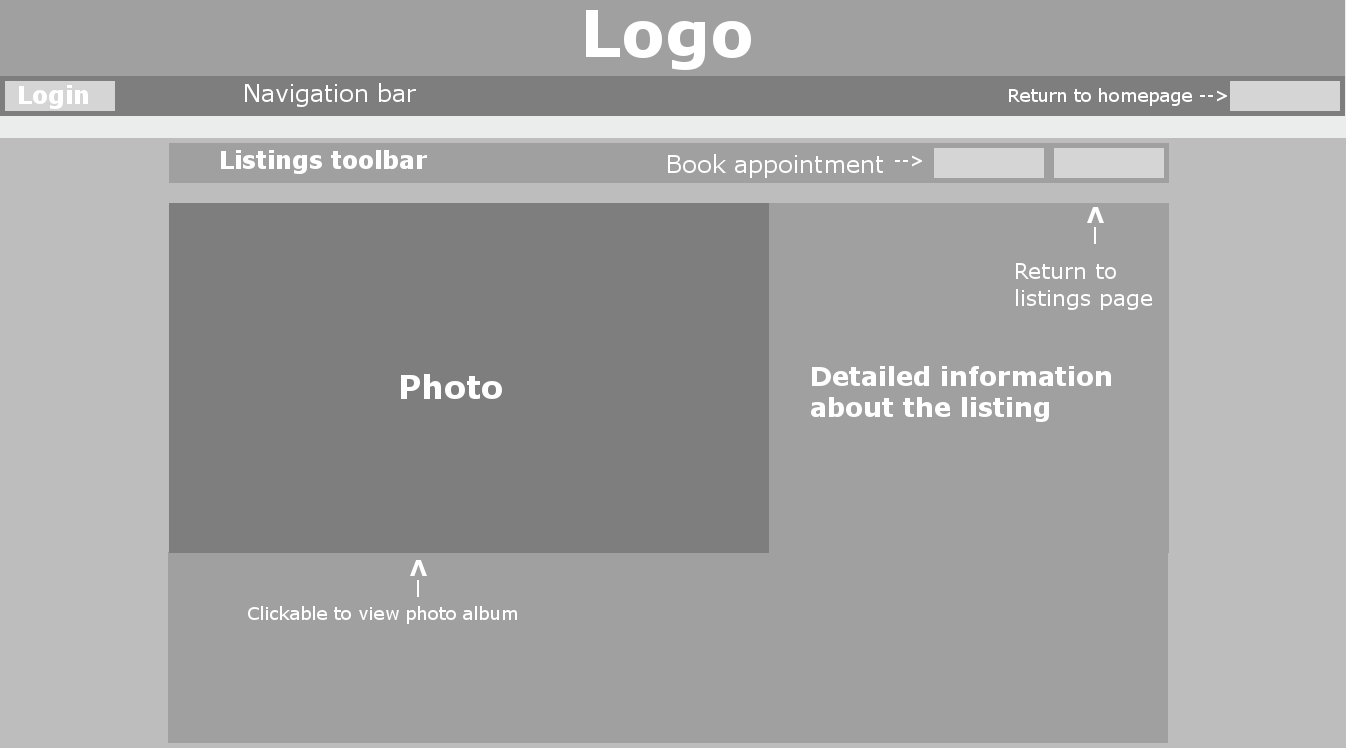
The home page is pretty straight forward, you are greeted by two big buttons. If you are a student that wishes to look for housing, you will click the “I am looking for accommodation button”. If you are a landlord that wants to advertise their properties to students, then you will click “I want to offer accomodation”.



If a student wishes to search for accommodation, they will then be taken to the vacancy page. On this page the student will be able to use the menu on the left to define a number of search criteria. These criteria are related to pricing, location, amenities etc.. when the student has selected the criteria and pressed the search button, the webpage will output a list of results that corresponds to the given criteria. Each result will consist of an image (that may be part of a larger album of images of the listing) and a summary with the most important details about the property. in the lower right corner of each listing there is a button which can take you to a page with more detailed information about the listing.



In the detailed listing view, the user will find all the information we have on a property, such as energy ratings, distance from campus, contact details and more. By clicking on the photo on the listing, the user can view an album of photos about the property in greater detail (providing the landlord included any photos). The buttons in the upper right corner of the listing view will either take the user back to the listings catalogue or give you the option to book an appointment with the landlord. Students have the option to create an account where they will be able to bookmark interesting vacancies for later reference. They can login (or create a new account) through the login button in the upper left corner.



Landlords will be required to make an account in order to be able to advertise their vacant properties. The menu below gives the landlord an overview of the properties which they have currently added to the site. These properties can be sorted in similar ways like the listings view, they can also be sorted on whether they are currently vacant or not, since properties that are currently being rented out should not be visible to students. Properties can be added , removed or have their details changed by the buttons on the right side of the property list. The sidebar contains various account management settings.



**Appendix**

References

Sommerville, I. (2011). *Software engineering*. Boston: Pearson.

**Declaration**

|  |  |  |
| --- | --- | --- |
| Name | Has worked on | Signature |
| Matthew Anthony James Hawkins | System Architecture, architectural patterns, and the class diagram. |  |
| Edward Bonsu | Use case models | E.A |
| Hong Hon Ping | Sequence Diagram | 16106611_1903739913190204_202435250_o.jpg |
| Guillaume Van de Sype | UI mock up designs | signature.jpg |
| Joshua Bowen | State diagram, context diagram |  |
| mouloud diram | sequence diagrams | myt.jpg |