

**CEN 4010 - Software Engineering**

**Team 2**

**int elligence;**

**Clinton Benton, David Hughes, Elizabeth Moreno,**

**Michael O’Donnell, Tommy Owens, Aaron Ulmer**

Table of Contents

[**1 - Overview** 3](#_Toc476945036)

[**2 - Design Rationale** 4](#_Toc476945037)

[**3A - High Level Design** 4](#_Toc476945038)

[**3B - Architectural Style** 5](#_Toc476945039)

[**3C – High Level System Diagram** 6](#_Toc476945040)

[**3D – Decomposition** 9](#_Toc476945041)

[**4 - Detailed Design** 10](#_Toc476945042)

[**4A - Sign-up and Login** 10](#_Toc476945043)

[**4B - Documentation** 11](#_Toc476945044)

[**4C – Properties** 14](#_Toc476945045)

[**4D - Billing** 16](#_Toc476945046)

[**5A – Risk** 17](#_Toc476945047)

[**5B – Assumptions** 17](#_Toc476945048)

# **1 - Overview**

The goal of the project is to build an internet based program to handle property rentals and transactions between landlords and tenants, so that they could better handle the system without being so hands on or in person. This is done to simplify and quickly pass the transactions between the two. Through this, searching for properties, leasing, management, and payments can all be condensed to one web-based app where either party can handle their own responsibilities of the process.

The program needs to be able to handle basic tenant and landlord interactions. This includes searching for, applying to, leasing a property from a landlord, and payment management and communication; along with the landlord to be able to upload their properties and relevant documents, lease them and manage their fees and other matters. The program will contain a process for potential tenants to search and view available property within the system and message the landlord of that property for potential application proceedings. The applications process is sorted out by the landlord with the reviewing and signing of the legal documentation by the tenant. The landlord either keeps the documents on file within this system or external.

The addition functionality of the system needs to include the basics of a rating system for both the tenant and landlord. The purpose of the rating system is to create a validation system for landlords on tenants to avoid individuals who have a history of breaking contracts and for preventing slumlords.

The implied task of the system includes a communication system consisting of a basic instant messaging protocols, including notifications through the system and email. The customer is strongly requesting that it also be SMS compatible, aside from being through the system and notifications.

The program will contain a payment notification system and an annotation method for the tenant and landlord to validate that the payment has been made in the appropriate amount for that month. The validation system is due to tenants paying part in cash and part in the system or to check if proper amounts. The payment system will be capable of being updated to handle the current system alongside additional features to the payment system at a later date. This process needs to be intuitive so that any user can gain a quick visual status of their property’s upcoming fees.

**Project Team**

|  |  |
| --- | --- |
| Clinton Benton | Software Engineer/Designer. High-Level Design and Assumptions/Risks |
| David Hughes | Database Management/Software Engineer. High-Level Design |
| Elizabeth Moreno | Lead Designer/Software Engineer. Design Rationale |
| Michael O’Donnell | Team Leader/Software Engineer. Overview, Table of Contents, Edits/Formatting |
| Tommy Owens | Software Engineer/Designer. Detailed Design |
| Aaron Ulmer | Software Engineer/Designer. Detailed Design |

# **2 - Design Rationale**

* Internet access is required for any user to interact with the system since it isn’t available in the form of a mobile application with offline access. Landlords should also keep in mind that not all potential tenants will always have access to an Internet connection and that the system expects the entire lease process to occur online. In the case that either the landlord or tenant cannot access the Internet for a prolonged period of time, a contingency plan should be set up so that both parties can manage the lease agreement offline.
* The accounting and billing aspect is outside the scope of this project and it is up to the administrator to either implement it, connect it to some other service such as PayPal, or accept payments by some other method. In the case it is handled through the system, the site or database administrator should ensure the database is secure enough to handle this kind of sensitive information.
* Basic security such as authentication and authorization will be implemented, however most services will be performed on the server-side to minimize the risk on the client side.
* It is up to the administrator to ensure the data stored in the database is backed up in the case of data loss. Cloud services are a popular option that provide this in addition to other benefits such as scalability, speed and migration tools.

# **3A - High Level Design**

The high-level design of the web application starts with both the Landlord and the Tenant registering, however the landlord will take place as the administrator. After both are registered they can then exchange documents and even sign digitally so there is no double entry required and can even save trees by not using paper. The other huge advantage is that all documents will be stored online so they can be retrieved later or anytime needed. Another feature is a scoring system set for both the landlord and the tenant to be able rate each other.

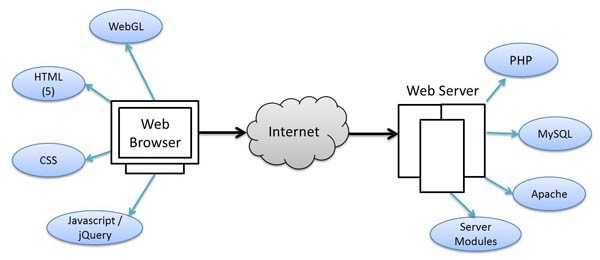
On the landlord’s end, after they sign into their account they can then add all the properties to the main page for them to view. After a property has been had been input by the landlord they can then notify a potential tenant and send them a unique I.D. to the property. When the landlord confirms the I.D. to the property and amounts are collected, the lease is then created and signed and sent on to be signed by the applicable party. Upon completion of the signature the document is stored in both the landlord and the tenant profiles for future viewing.

Another feature that creates extra ease for the landlord is when the tenant does not pay the rent on the due date the landlord can then generate a 3-day notice which is used in the State of Florida for the eviction process. Since a 15-day notice is almost the same exact document we also will add that in for the slight reason that a landlord needs to get rid of a tenant for no reason at all.

When the tenant signs into the system they will be notified of anything that will need attention. The page will display their score and the related property that they are registered with. When clicking on the property that they rent or properties, it will show them how much they owe on the property including late fees, etc. They can also pay for the property that they are renting.

All together this should save time for both the tenant and the landlord to create a non-confusing and private place to handle transactions. It will create a headache free environment where you can remember where you stored documents and always can send and receive payments right over your phone or any internet capable device.

# **3B - Architectural Style**



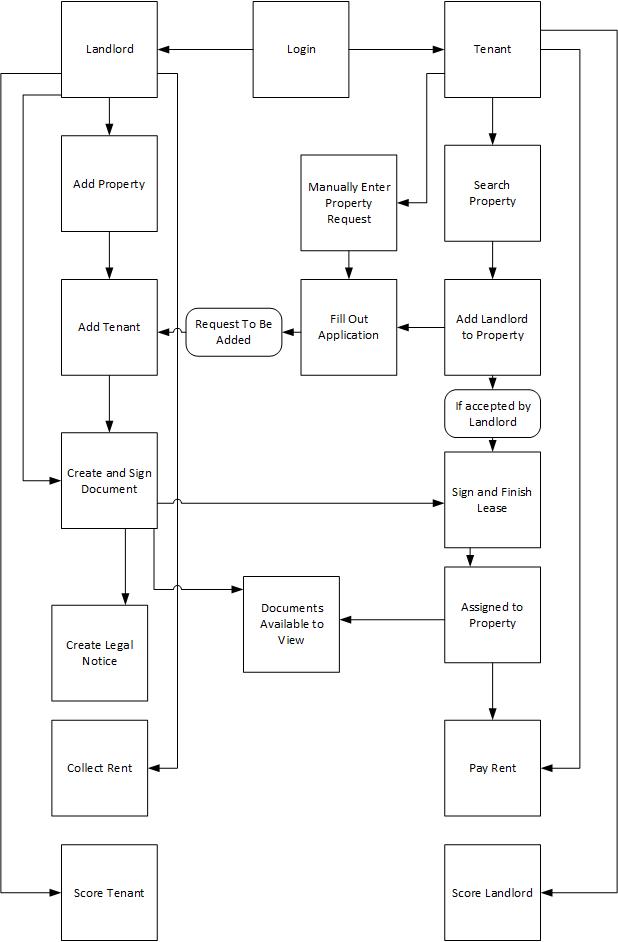
The web application makes use of client-server architectural style using a hosting service to handle these requirements and user requests through our database. In this sense, individual users connect to the application through the web, with the application being on a web server, and the application web server connecting to a database for the back-end. The site will make use of many different languages to makes it easier for all members of group. The server will be on a Linux and will make use of HTML, JavaScript, CSS, AJAX, and PHP for talking with the MySQL database and anything else that will help it accomplish the goals. The main GUI elements will make use of the HTML and CSS and the reactive components will use any extra languages. PHP will be used to communicate with the MySQL database which will be local to the server.

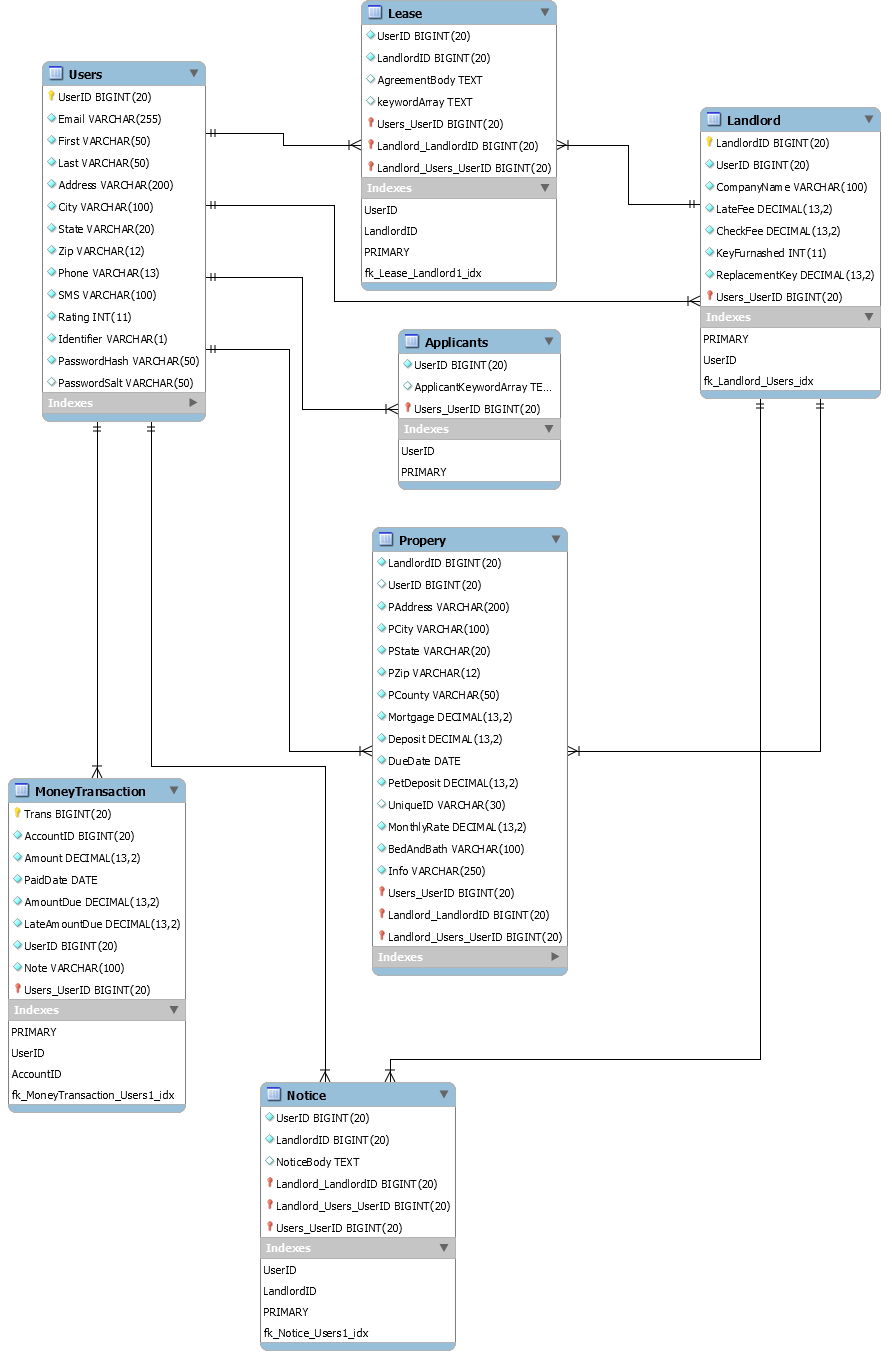
# **3C – High Level System Diagram**

The graphical representation below illustrates the high level representation of the operation of the program form the functional architecture of the program. For an expanded definition of the function shown below. Please refer to the use case documentation during the previous submission.

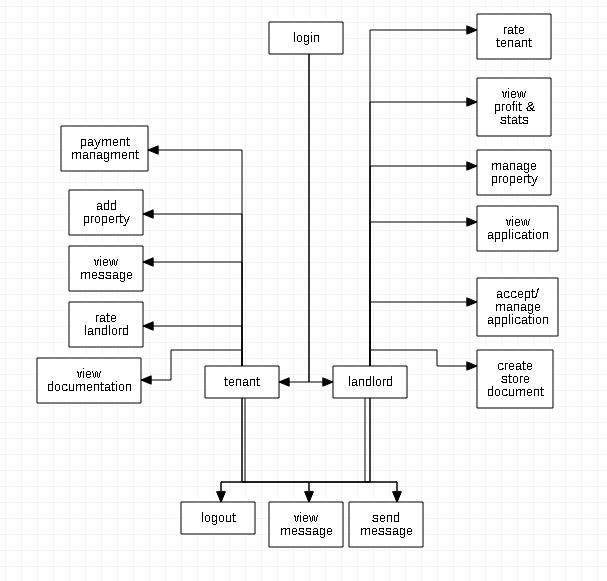
Below that diagram is a copy of the current E/R diagram of the database that has been established on MySQL via the program MySQL Workbench. It is broken down based on the data that is input. Both the landlord and the tenant start as a user and is distinguished by the identifier which for now will be “L” for landlord and “T” for tenant. This will make it so if later on if needed we can add on “M” for maintenance or “O” for office staff.

Once a landlord is established in the system they can then fill out all the things that would help autofill the lease. The Lease table is where all the information on the lease is set into the system so it can be retrieved later. Applicant table is where the tenants that have applied for the property are stored. Property table is where the landlord stores all the information on the property to be viewed by both parties. The Money-Transaction table is where all the payments information between both parties are kept. Last, Notice table is where all the 3-Day and 15-Day documentation will be kept for both parties to view.





# **3D – Decomposition**

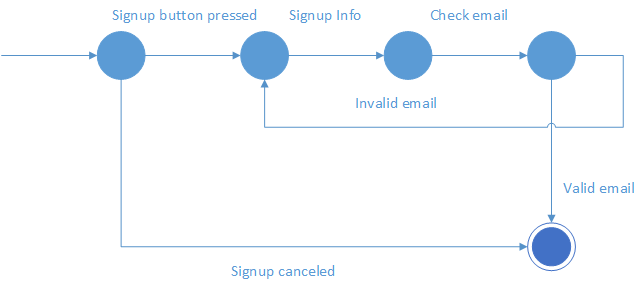


The decomposition strategy that is will be used is feature oriented and outside-in oriented. The choice to use these decomposition strategy is based on the model that is being enacted to the web format. The process of leasing a property is a pre-established routine with legal nuances within the practice so we ensure that the process is of an intuitive design with a sound implementation, which can be seen through this diagram.

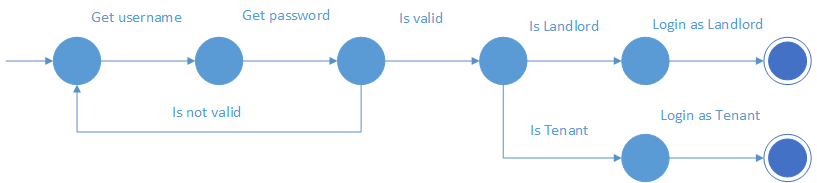
# **4 - Detailed Design**

The design model for this project is a web based implementation of a client server architecture. The server-side implementation uses a MySQL database to store data, an Apache Web server to store webpage documents and PHP documents for drafting virtual pdf documents on the fly. The client-side implementation uses a modern browser to display web pages using HTML5, CSS3, JavaScript, and WebGL.

# **4A - Sign-up and Login**



The above state machine represents the signup process. After the signup option is selected, items such as name and email address will be asked for. If the email has already been used by someone or is otherwise invalid, the software returns to the signup information gathering area. If the email is valid signup is complete and the user is directed to the login screen. The same thing happens should the user decide to cancel signup.

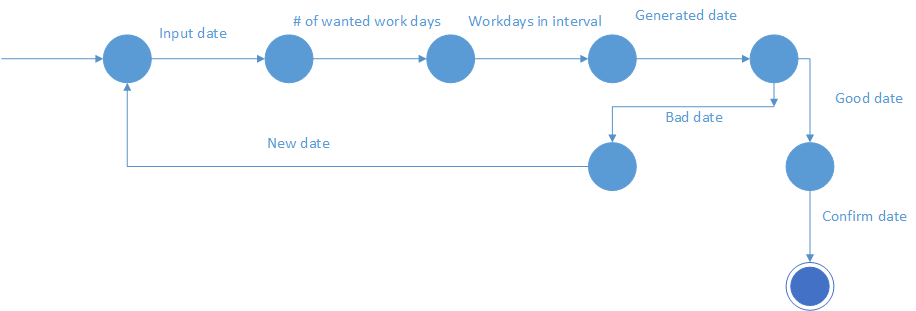


The above is a state diagram demonstrating how the login process works. If a username or password is invalid, the user gets returned to the login screen. Should the user be a valid user, the database checks if the user is either a Landlord or a Tenant. The system then logs the user in as the appropriate type with the ability to view what each user type is supposed to.

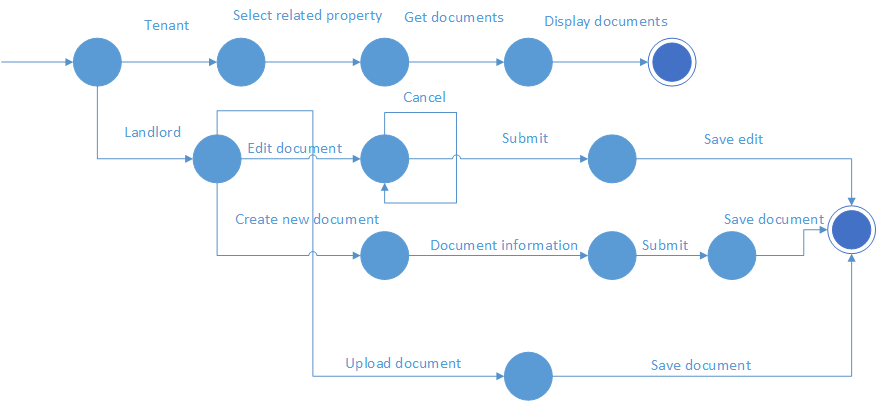
# **4B - Documentation**



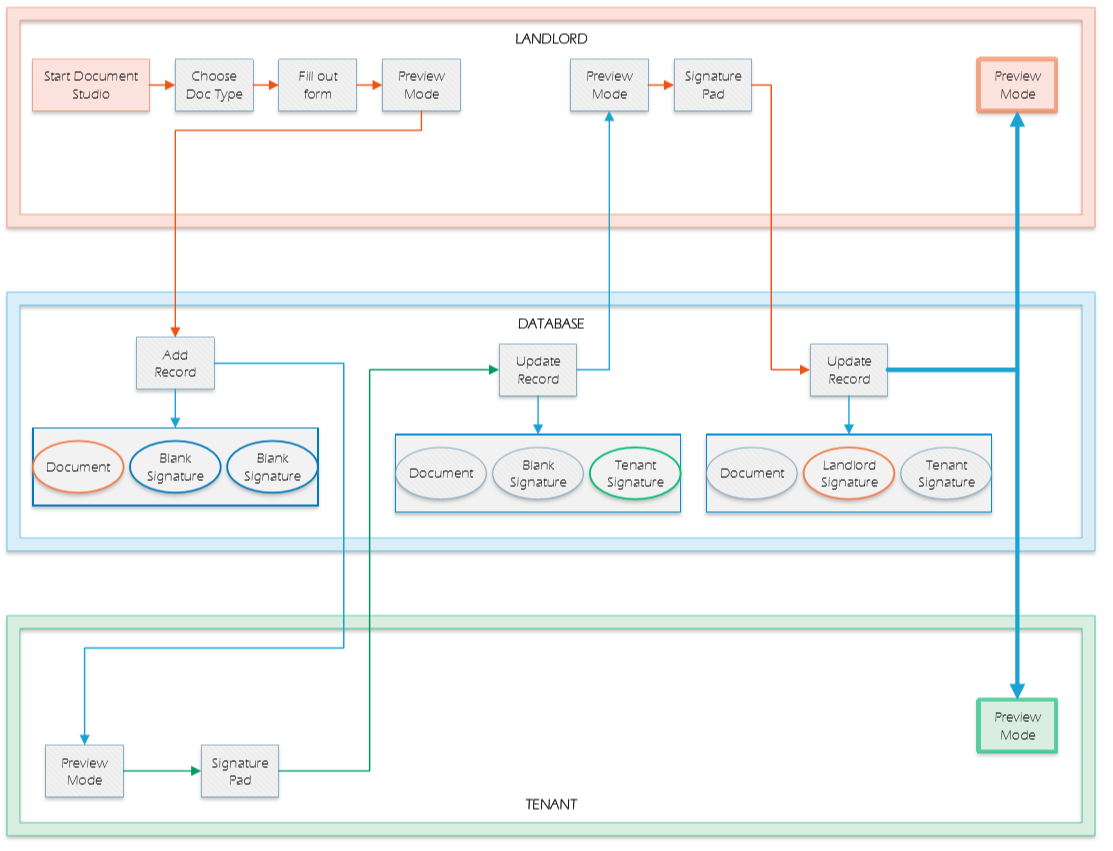
The above is a sample of how the signing of documents will appear. After signing, the user’s signature will be stored as an encrypted string to save space within the database. This signature is used with whatever document it is assigned to be used with.



The above state machine shows how a rent due date warning system is implemented. This event only accounts for weekdays with the software avoiding weekends and holidays. This section takes in two dates and a wanted number of workdays to generate an estimated target date. The section then generates the number of workdays in between the two dates and suggests a date in this interval which will fulfill the requirements. If the user likes the generated date, the user can use this as the end date on the warning document. Otherwise, the user can submit a new interval to try and get a date.



The above state machine shows the document viewing and document creation/alteration sections of the software. The software allows tenants to only view documents linked to properties that they are registered under. The landlords can edit, create, or upload documents to be eventually attached to a property. The edit option allows the landlord to open and edit a currently saved document. The create option allow the user to make new documents by using tools provided by the software. There is also an option to just upload documents if the landlord has one already created.

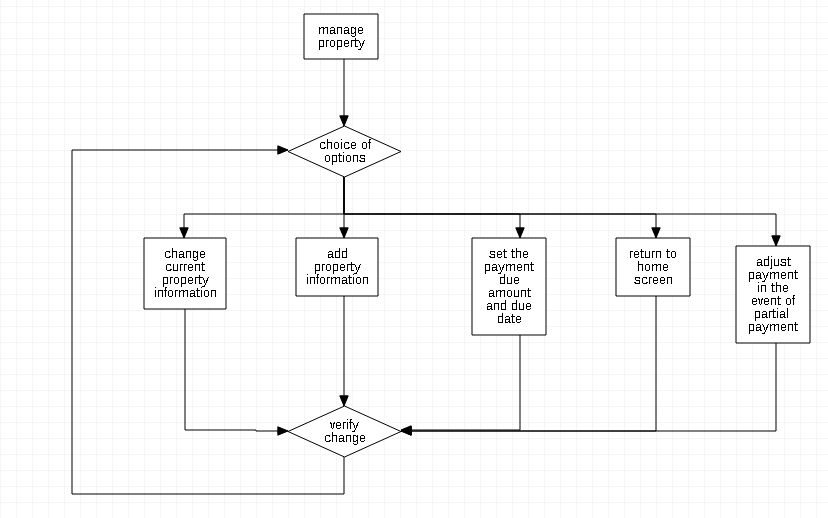


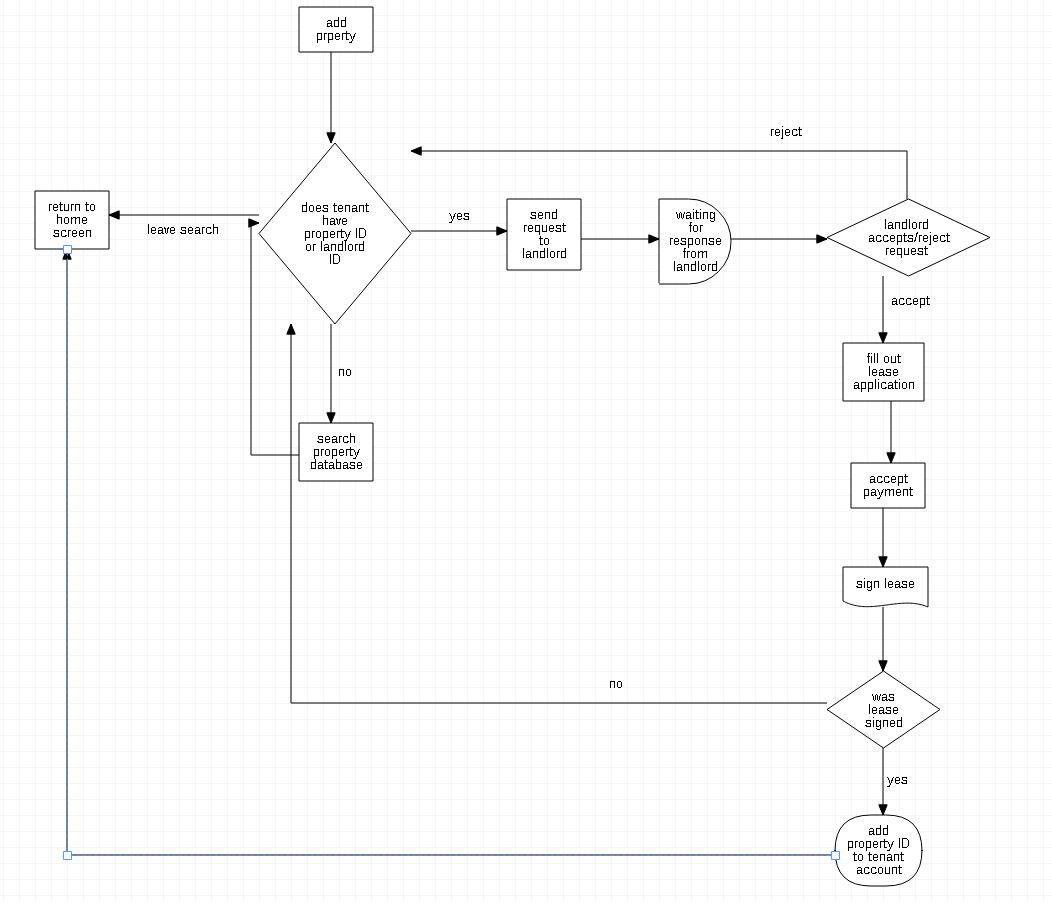
When a landlord wants to create a new document for a property, they are prompted to fill out a form based web page representing the fields necessary for either a Rental Application document or a Lease Agreement document. After filling in the form fields respectively, the landlord will be presented an editable document in the form of a text input box on a webpage. The landlord can then change the contents of the document before sending it to the tenant for review and signature. Once the landlord submits the document for the tenant to review, a record is populated in the database which holds the document text data as a string, and two separate fields filled with a placeholder text string which represents a blank signature. After the server adds the record to the database, the document is made viewable as a PDF to the tenant for signing. When the tenant decides to add their signature, the web page brings up a prompt for a signature pad in an overlaid window. Once the tenant has confirmed their signature, a string of the image data is encrypted and stored in the database. The document is then made viewable to the landlord in the form of a PDF with the tenant signature. Once the landlord signs the document in the same fashion as the tenant did, the database will update the signature field for the landlord and make the final signed document viewable to both tenant and landlord.

The documents will be viewed using an external class of PHP known as FPDF. This will pull the document string data from the database, as well as both encrypted image data strings from the signature fields. It then pull the data together and renders a PDF on the fly for the viewer.

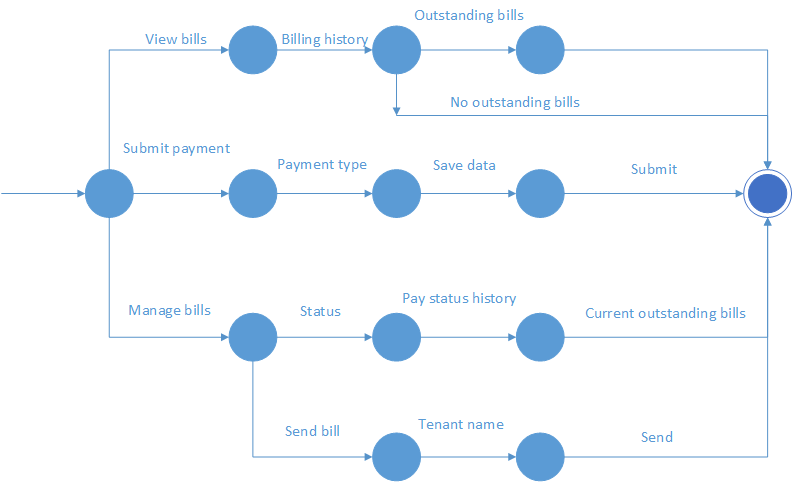
# **4C – Properties**

The diagrams below are flowcharts for the add property function of the tenant, and the manage property function of the landlord.





# **4D - Billing**



The above state machine shows the billing processes of view, manage, and submit. The tenant is the only user type that can access submit payment. This is used to have the tenant submit how they want to pay their bill. Only the landlord can access the manage bills option. This allows them to see the billing status of any tenant that is registered to their properties. They can also send a bill to a tenant along with setting things such as due date. Both user types can see the view bill history option. This allows the user to see billing history and their current billing status.

# **5A – Risk**

* At this point in the project, the main obstacle to the completing the project is the amount of available time each member of the team. All members of the group have either part-time/full-time jobs with two courses or are full time students.
* Additional two members of the group are in the process of taking COP-4813. This is causing a learning curve for some of the members for developing assigned projects. (the agile model is being used to mark the progress of the project)
* That Google's graph API will be function throughout the use of the application.

# **5B – Assumptions**

# 

* We assume each client device will be accessing our application using a supported web browser. We plan on supporting the latest stable versions of Chrome, Safari, Opera, Firefox and Edge. We are not planning to directly support Internet Explorer, due to it being a dead browser come this past January. As of this past January [Chrome](https://www.w3schools.com/browsers/browsers_chrome.asp) holds 73.7 %, [Firefox](https://www.w3schools.com/browsers/browsers_firefox.asp) holds 15.4 %, [IE/Edge](https://www.w3schools.com/browsers/browsers_explorer.asp) holds 4.9 %, [Safari](https://www.w3schools.com/browsers/browsers_safari.asp) holds 3.6 %, [Opera](https://www.w3schools.com/browsers/browsers_opera.asp) 1.0 %. By using this assumption we are supporting over 99 % of the internet browser used in the United States.
* We assume that the user will have internet access to complete this interaction on this platform.
* We assume the company hosting MySQL will not let the database go down.
* We assume that this product will be free of use due to the nature of the of the API’s used to support the product.