Mohamed Farook Fazrin

[Email address]

Abstract

[Draw your reader in with an engaging abstract. It is typically a short summary of the document.   
When you’re ready to add your content, just click here and start typing.]

Online Student Accommodation System

[Document subtitle]

# ACKNOWLEDGEMENTS

# Executive Summary

The students of XYZ University are always seeking for properties for accommodation closer to their campus area. To support the student needs the university is planning to host an online Student accommodation system (SAS) to provide details of property-owners, their properties and of students seeking accommodation.

Some of the required functionalities of the system would be.

* Students and property owners can register with the service
* The property-owners can manage property details and update the system if their property is available or not available for rent.
* Students can search for properties and send requests to view the property to the relevant property owners
* Property owners can manage student’s view requests
* Admin can generate following reports and manage user accounts
  + List of users report
  + List of properties report
  + List of students report
  + List of owners report

Table of Contents

[ACKNOWLEDGEMENTS 1](#_Toc17302833)

[Executive Summary 2](#_Toc17302834)

[Table of Figures 4](#_Toc17302835)

[Task A 5](#_Toc17302836)

[1.1 Purpose 5](#_Toc17302837)

[1.2 List of Stakeholders 7](#_Toc17302838)

[1.3 Project Scope 7](#_Toc17302839)

[1.4 Requirement Clarity Index 8](#_Toc17302840)

[1.4.1 Functional Requirements 8](#_Toc17302841)

[1.4.2 Non-Functional Requirements 9](#_Toc17302842)

[Task B 10](#_Toc17302843)

[Level 1 Use case diagram 10](#_Toc17302844)

[Level 2 Use case diagram for student 11](#_Toc17302845)

[Level 2 Use case diagram for owner 12](#_Toc17302846)

[Level 2 Use case diagram for admin 13](#_Toc17302847)

[Class Diagram 14](#_Toc17302848)

[Class diagram explanation 14](#_Toc17302849)

[Sequence Diagram for accommodation booking system 15](#_Toc17302850)

[Task C 16](#_Toc17302851)

[Singleton Design Pattern 16](#_Toc17302852)

[Implementation 18](#_Toc17302853)

[Singleton design pattern : cons 18](#_Toc17302854)

[Singleton design pattern : pros 18](#_Toc17302855)

[Factory Method Pattern 18](#_Toc17302856)

[Advantage of Factory Design Pattern 18](#_Toc17302857)

[Usage of Factory Design Pattern 19](#_Toc17302858)

[Abstract Factory Pattern 19](#_Toc17302859)

[Advantages of Abstract Factory Pattern 19](#_Toc17302860)

[Disadvantages of Abstract Factory Pattern 20](#_Toc17302861)

[Task E 21](#_Toc17302862)

[Test Plan 21](#_Toc17302863)

[1.1. Purpose 21](#_Toc17302864)

[1.2. Project Overview 21](#_Toc17302865)

[1.3. Test Objectives 21](#_Toc17302866)

[1.4. Test Assumptions 22](#_Toc17302867)

[Test Principles 22](#_Toc17302868)

[Data Approach 23](#_Toc17302869)

[Functional Test 23](#_Toc17302870)

[Test Steps 24](#_Toc17302871)

[Test Cases 26](#_Toc17302872)

[Task E 27](#_Toc17302873)

[Admin Privileges 27](#_Toc17302874)

[Property Owners Privileges 29](#_Toc17302875)

[Task G 36](#_Toc17302876)

[Appendix 0](#_Toc17302877)

[References 0](#_Toc17302878)

# Table of Figures

[Figure 1 level 1 use case diagram 9](#_Toc17302784)

[Figure 2 use case diagram for student 10](#_Toc17302785)

[Figure 3 Level 2 Use case diagram for owner 11](#_Toc17302786)

[Figure 4 Level 2 Use case diagram for admin 12](#_Toc17302787)

[Figure 5 class diagram for student accommodation booking system 13](#_Toc17302788)

[Figure 6 overview of the singleton design pattern 15](#_Toc17302789)

[Figure 7 users-list report 26](#_Toc17302790)

[Figure 8 student-list report 27](#_Toc17302791)

[Figure 9 properties-list report 27](#_Toc17302792)

[Figure 10 owners-list report 27](file:///C:\Users\mfazrin\eclipse-workspace\acmd\Accommodation%20management%20system.docx#_Toc17302793)

[Figure 11 sign-up form 28](#_Toc17302794)

[Figure 12 update owners profile page 28](#_Toc17302795)

[Figure 13 profile-page 29](#_Toc17302796)

[Figure 14 my-properties page 29](#_Toc17302797)

[Figure 15 add-property page 30](#_Toc17302798)

[Figure 16 property-added to my property list 30](#_Toc17302799)

[Figure 17 payment confirmation pop up 31](#_Toc17302800)

[Figure 18 after payment confirmed 31](#_Toc17302801)

[Figure 19 view request to view properties 31](#_Toc17302802)

[Figure 20 change password function 32](#_Toc17302803)

[Figure 21 view all properties 32](#_Toc17302804)

[Figure 22 search only available properties with 'single' as search term 33](#_Toc17302805)

[Figure 23 send view request page 33](#_Toc17302806)

[Figure 24 my view requests page 34](#_Toc17302807)

[Figure 25 list of user accounts page 34](#_Toc17302808)

[Figure 26 Gantt chart for XYZ accommodation management project 0](#_Toc17302809)

# Task A

## Purpose

The students of XYZ University are always seeking for properties for accommodation closer to their campus area. To support the student needs the university is planning to host an online Student accommodation system (SAS) to provide details of property-owners, their properties and of students seeking accommodation. As of today, this whole operation is manual.

XYZ University has decided to automate above operations to achieve following objectives

1. Better and faster service to students

2. Better services to property owners

3. Increase the communication between students and property owners

The purpose of this document is to identify the high-end requirements for developing a web application system, to coordinate and monitor students and property owners for XYZ university. This document will explain operations of the student accommodation booking system which need to be monitored and optimized. The proposed system should automate the manual process

Some of the required functionalities are

* The Student ID, First Name, Last name, address email and telephone number of property-owners are kept in SAS. Each property is allocated a unique identifier as well as details such as the address, type of property [ex: Flat, terraced house, detached house...etc.], the maximum number of tenants it is suitable for and the amount of the rent. A fee is charged for each property that is added to the system.
* Students can register with the service providing their Student ID, First Name, Last name, address email and telephone number. The registered students will be provided with a list of available properties. Further the student can request for viewing the properly and system will notify this to the property-owners.
* The property-owners can update the system if their property is available or not available for rent.
* Also, the system must have the facility to provide reports such as available properties, registered students and current occupation details etc.

## List of Stakeholders

This proposed system will be expecting to be used by the following users

1. Students
2. Property Owners
3. Admin

## Project Scope

Once this system is implemented, whole manual processes related to accommodation booking process could be tracked in the system. Therefore, the project scope will have to be covered with following functional areas;

1. Login page for property owners and students

2. Manage property details by property owners

3. Search property details by students

4. Request to view the property through system

5. Accept/Reject the view request for property by respective property owners

6. Make a payment through system

7. View property details and availability of the property

8. View the student’s details

## Requirement Clarity Index

### 1.4.1 Functional Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Requirements | Description | Risk level | RCI |
| 1 | Use Accounts | User authentication with different types of privileges and access levels | Medium | 100% |
| 2 | Authentication | Proper Login system with username and password for each user (admin, student, owners ) | High | 100% |
| 3 | Student registration | Register student details | Medium | 100% |
| 4 | Edit student details | Edit the existing student details | Medium | 100% |
| 5 | Delete student details | Delete the existing student account | Medium | 100% |
| 6 | Owner registration | Register owner details | Medium | 100% |
| 7 | Edit owner details | Edit the existing owner details | Medium | 100% |
| 8 | Delete owner details | Delete the existing owner account | Medium | 100% |
| 9 | Create property details | Create new property details | Medium | 100% |
| 10 | Edit property details | Edit the existing property details | Medium | 100% |
| 11 | Delete property details | Delete the existing property details | Medium | 100% |
| 12 | Change the status of a property | Change property status as available or occupied | Medium | 100% |
| 13 | Search rooms by various criteria | Search room by specific parameters such as price, maximum number of tenants can stay, | Medium | 100% |
| 14 | View Request | Student can request to owners to view the property on specific date | Medium | 100% |
| 15 | Accept/Reject Request | Owners can accept or reject by request to view by student | Medium | 100% |
| 16 | View available properties | Report of all available properties | Medium | 100% |
| 17 | View all students | View all registered students | Medium | 100% |

### 1.4.2 Non-Functional Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Requirements | Description | Risk level | RCI |
| 1 | Web server | A web server to host the web application | High | 100% |
| 2 | Concurrency management | Server should be able to handle multiple concurrent requests by students and owners | High | 100% |

# Task B

## Level 1 Use case diagram

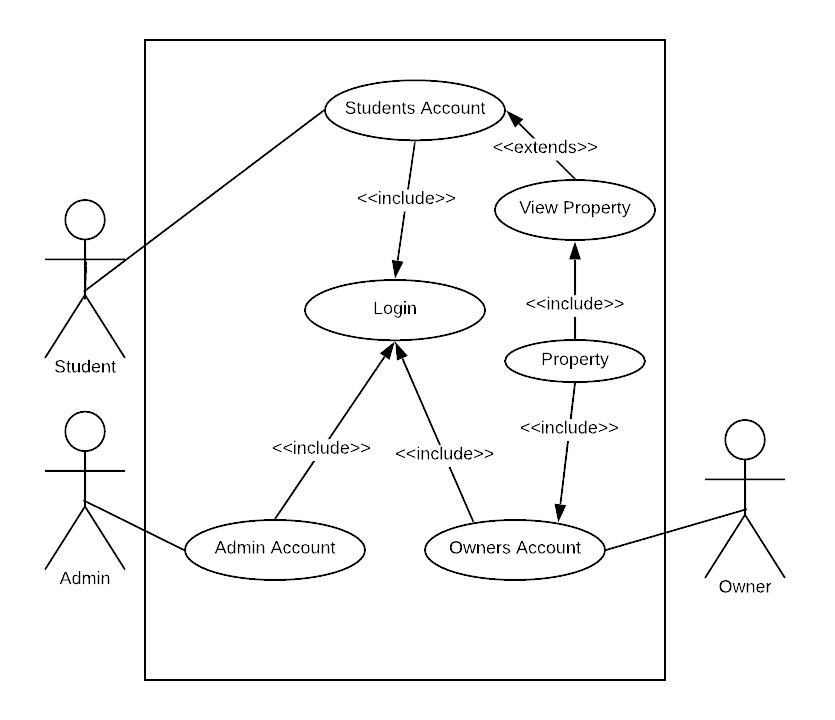


Figure level 1 use case diagram

### Level 2 Use case diagram for student

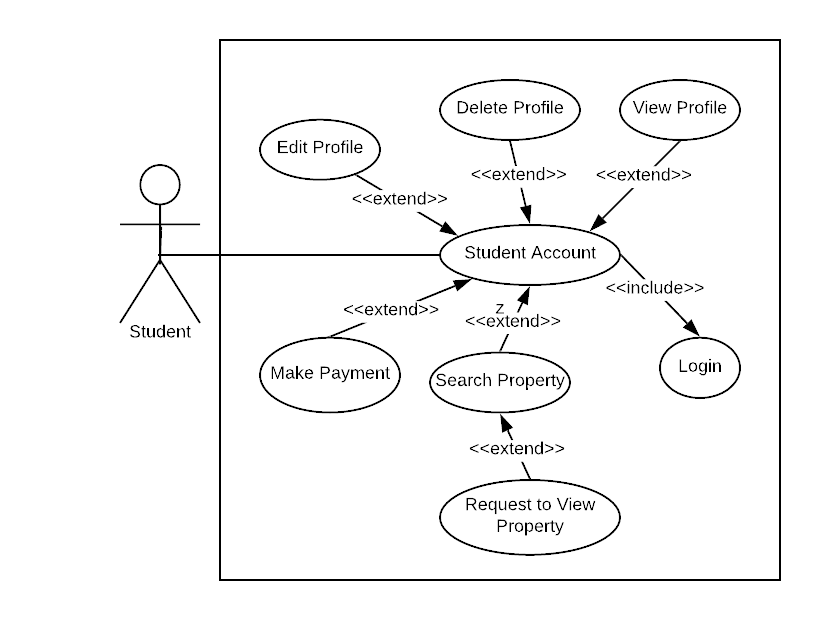


Figure use case diagram for student

### Level 2 Use case diagram for owner

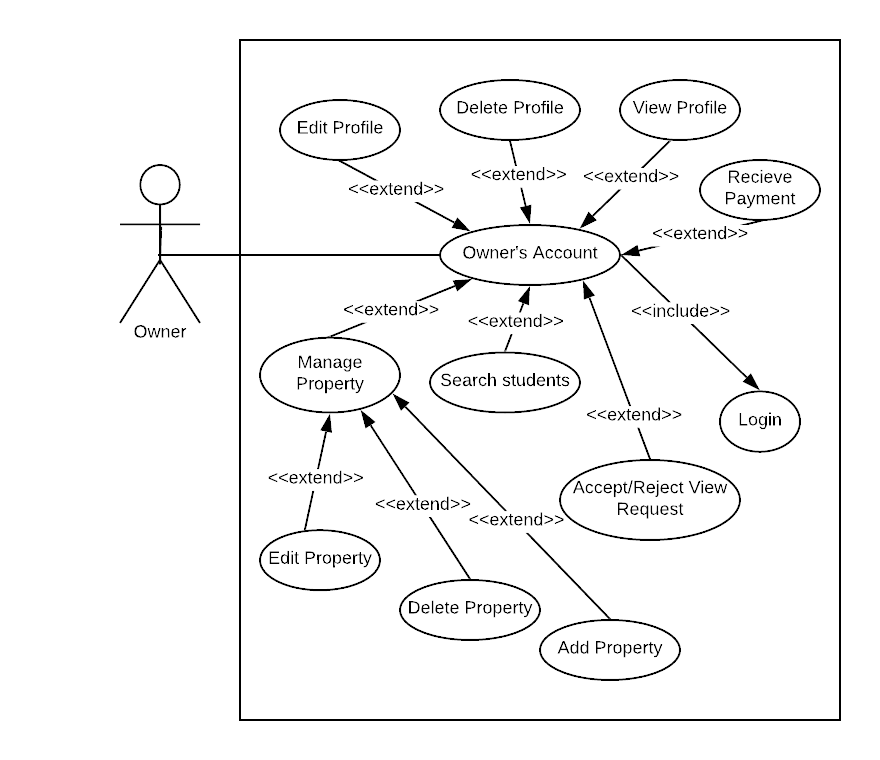


Figure Level 2 Use case diagram for owner

### Level 2 Use case diagram for admin

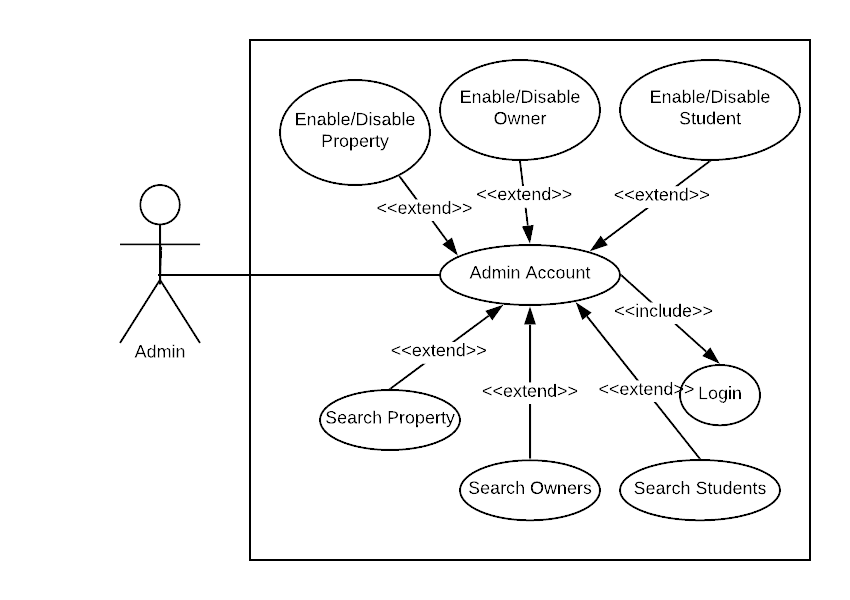


Figure Level 2 Use case diagram for admin

## Class Diagram

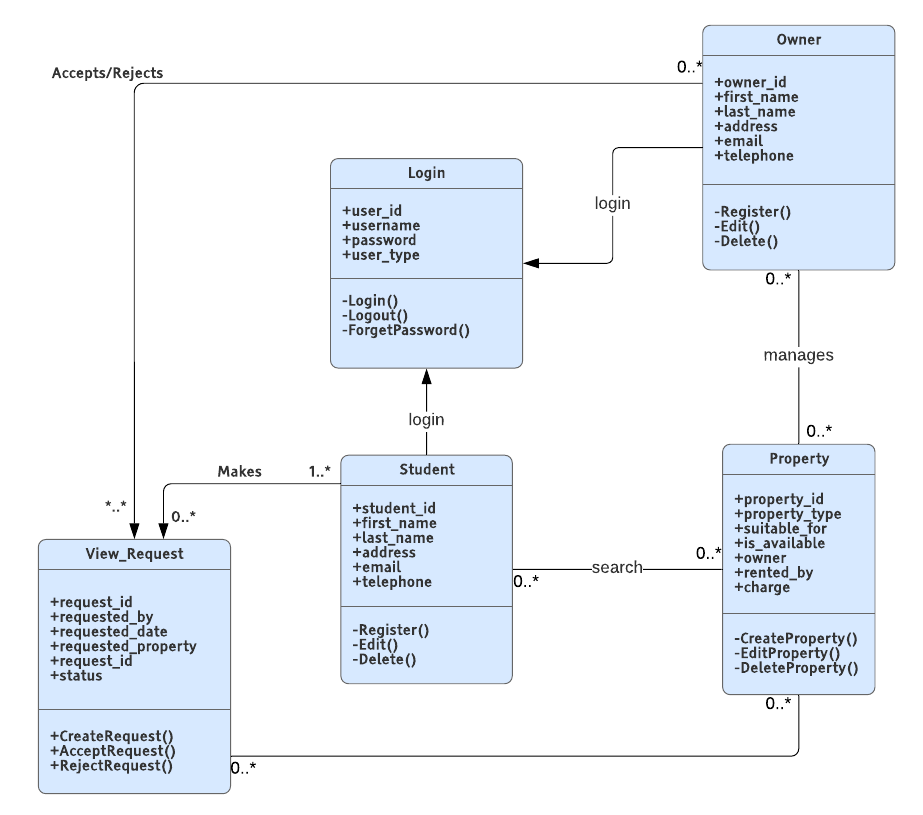


Figure class diagram for student accommodation booking system

### Class diagram explanation

|  |  |
| --- | --- |
| Class Names | Description |
| Login | Login class will handle authentication, authorization and forget password functions for student, owner and admin |
| Owner | Owner class will handle create, edit and delete owner details functions |
| Property | Property class will handle create, edit and delete property details this will including manage the availability of the property |
| Student | Student class will handle create, edit and delete student details |
| View\_Request | View request will handle create, accept and rejecting the view request of the property |

Table Class diagram explanation table

## Sequence Diagram for accommodation booking system

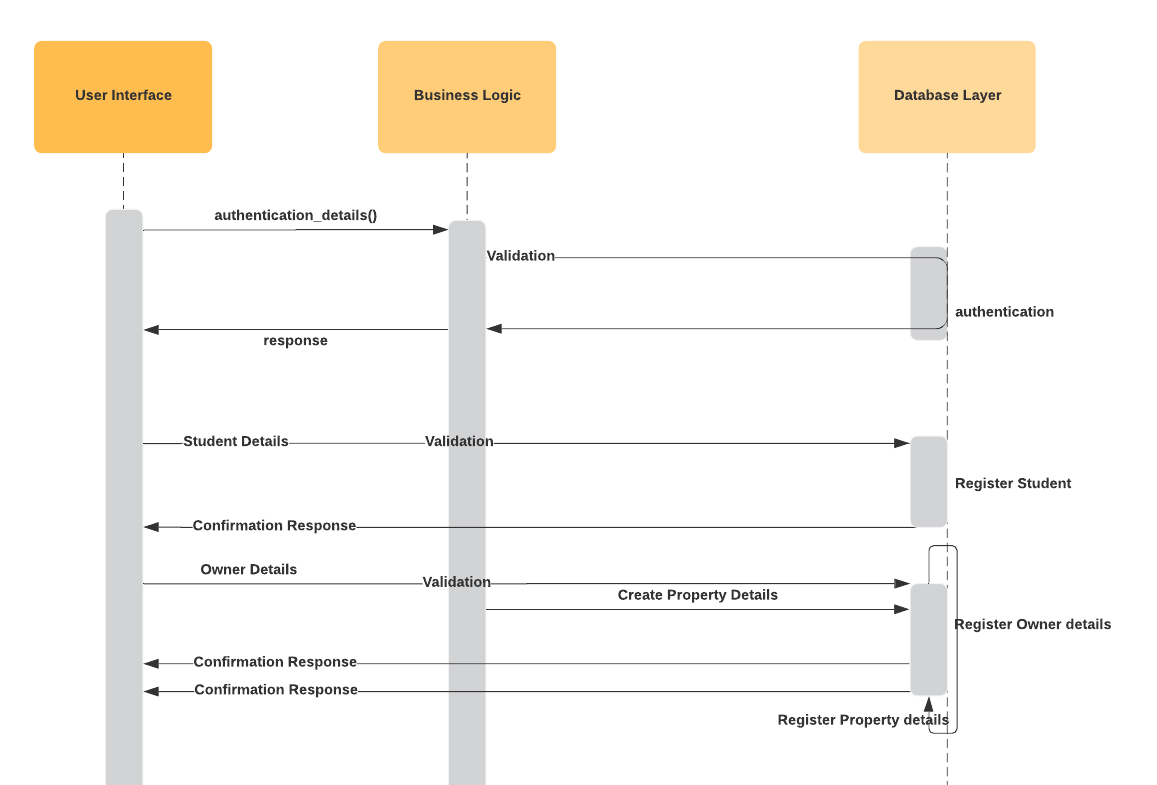


Table Sequence diagram for accommodation booking system

# Task C

Design patterns are conceptual tools for solving complex software problems. These patterns are simple and elegant solutions that have evolved over time and may have become generally accepted as the best way to address certain design (RudraSambyal, 2018),

## Singleton Design Pattern

The Singleton pattern encapsulates a shared resource within a single unique class instance. This instance arbitrates access to the resource and storage-related state information. A class method provides the reference to this instance, so there is no need to pass the reference around. Any object that has access to the Singleton’s class header can use the Singleton (RudraSambyal, 2018).

Following image will explain the overview of the singleton design pattern

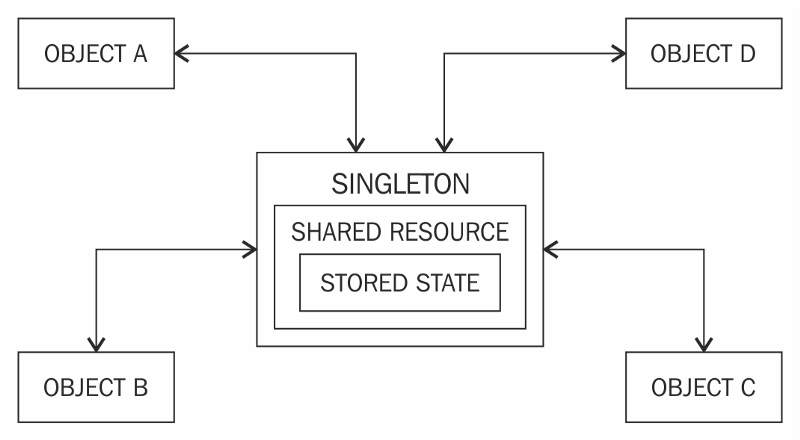


Figure overview of the singleton design pattern

This design pattern defines the structure of a class that can have only one instance. A Singleton encapsulates a unique resource and makes it readily available throughout the application. The resource might be hardware, a network service, a persistent store, or anything else that can be modelled as a unique object or service (RudraSambyal, 2018).

### Implementation

Implementation of the Singleton pattern often typically creates a single object using the factory method, and this instance/object is called a shared instance in most cases. Since the access to the instance is passed on through a class method, the need to create an object is eliminated (RudraSambyal, 2018).

### Singleton design pattern : cons

* Singletons hinder unit testing: A Singleton might cause issues for writing testable code if the object and the methods associated with it are so tightly coupled that it becomes impossible to test without writing a fully-functional class dedicated to the Singleton (RudraSambyal, 2018).
* Singletons create hidden dependencies: As the Singleton is readily available throughout the code base, it can be overused. Moreover, since its reference is not completely transparent while passing to different methods, it becomes difficult to track (RudraSambyal, 2018).

### Singleton design pattern : pros

To avoid these complications, when considering the Singleton pattern, need to make certain that the class is a Singleton. Also, while thinking of designing the Singleton design pattern, keep testing in mind and use dependency injection whenever possible — that is, try to pass the Singleton as a parameter to the initializer whenever possible (RudraSambyal, 2018).

## Factory Method Pattern

A Factory Pattern or Factory Method Pattern says that just define an interface or abstract class for creating an object but let the subclasses decide which class to instantiate. In other words, subclasses are responsible to create the instance of the class (javatpoint, 2019).

### Advantage of Factory Design Pattern

Factory Method Pattern allows the sub-classes to choose the type of objects to create. It promotes the loose-coupling by eliminating the need to bind application-specific classes into the code. That means the code interacts solely with the resultant interface or abstract class, so that it will work with any classes that implement that interface or that extends that abstract class (javatpoint, 2019).

### Usage of Factory Design Pattern

* When a class doesn't know what sub-classes will be required to create
* When a class wants that its sub-classes specify the objects to be created.
* When the parent classes choose the creation of objects to its sub-classes.

(javatpoint, 2019)

## Abstract Factory Pattern

Abstract Factory pattern is almost like Factory Pattern is considered as another layer of abstraction over factory pattern. Abstract Factory patterns work around a super-factory which creates other factories (Kumar, 2015).

Abstract factory pattern implementation provides a framework that allows to create objects that follow a general pattern. So, at runtime, abstract factory is coupled with any desired concrete factory which can create objects of desired type (Kumar, 2015).

Abstract Factory provides interfaces for creating families of related or dependent objects without specifying their concrete classes. Client software creates a concrete implementation of the abstract factory and then uses the generic interfaces to create the concrete objects that are part of the family of objects. The client does not know or care which concrete objects it gets from each of these concrete factories since it uses only the generic interfaces of their products. So, with this idea of Abstract Factory pattern, we will now try to create a design that will facilitate the creation of related objects (Kumar, 2015).

### Advantages of Abstract Factory Pattern

This pattern is particularly useful when the client doesn’t know exactly what type to create.

**Isolation of concrete classes:** The Abstract Factory pattern helps you control the classes of objects that an application creates. Because a factory encapsulates the responsibility and the process of creating product objects, it isolates clients from implementation classes. Clients manipulate instances through their abstract interfaces. Product class names are isolated in the implementation of the concrete factory; they do not appear in client code.

**Exchanging Product Families easily:** The class of a concrete factory appears only once in an application, that is where it’s instantiated. This makes it easy to change the concrete factory an application uses. It can use various product configurations simply by changing the concrete factory. Because an abstract factory creates a complete family of products, the whole product family changes at once (Kumar, 2015).

**Promoting consistency among products:** When product objects in a family are designed to work together, it’s important that an application use objects from only one family at a time. Abstract Factory makes this easy to enforce (Kumar, 2015).

### Disadvantages of Abstract Factory Pattern

Difficult to support new kind of products: Extending abstract factories to produce new kinds of Products isn’t easy. That’s because the Abstract Factory interface fixes the set of products that can be created. Supporting new kinds of products requires extending the factory interface, which involves changing the Abstract Factory class and all its subclasses (Kumar, 2015).

### Derived Conclusion

### Singleton pattern will be used to develop this specific system since data resource will have a single instance and utilized by many objects as per required, more over data connection pool will be utilized to interact with database which is designed in by default singleton pattern. There are some other UI elements also will be in singleton pattern such as navigation header menu will have a single instance which will be utilized in all the pages.

# Task E

## Test Plan

### 1.1. Purpose

This test plan describes the testing approach and overall framework that will drive the testing of the XYZ accommodation management system Version 1.0 ,

The document introduces:

Test Strategy: rules the test will be based on, including the givens of the project (e.g.: start / end dates, objectives, assumptions); description of the process to set up a valid test (e.g.: entry / exit criteria, creation of test cases, specific tasks to perform, scheduling, data strategy).

Execution Strategy: describes how the test will be performed and process to identify and report defects, and to fix and implement fixes.

Test Management: process to handle the logistics of the test and all the events that come up during execution (e.g.: communications, escalation procedures, risk and mitigation, team roster)

### 1.2. Project Overview

XYZ AMS is a useful web-based tool providing students of XYZ university with the ability to view and interact with relevant accommodation information such as search and contact the property owners to process the accommodation facilities requirements,

internet enabled PC without having to involve the HR department.

The functionality of this application enhances the availability of property owners through the entire system, making information available anywhere, anytime. all information is subject to university’s defined security policy, where student or property owners can only view the information authorized to view by both parties.

### 1.3. Test Objectives

The objective of the test is to verify that the functionality of XYZ accommodation management system works according to the specifications.

The test will execute and verify the test scripts, identify, fix and retest all high and medium severity defects per the entrance criteria, prioritize lower severity defects for future fixing.

The final product of the test is twofold:

* A production-ready software
* A set of stable test scripts that can be reused for functional and UAT(user acceptance test) test execution.

### 1.4. Test Assumptions

**Key Assumptions**

* Production like data required and be available in the system prior to start of Functional Testing
* In each testing phase, Cycle 3 will be initiated if the defect rate is high in Cycle 2.

**General**

* Exploratory Testing would be carried out once the build is ready for testing
* Performance testing is not considered for this estimation
* All the defects would come along with a snapshot JPEG format
* The project will provide test planning, test design and test execution support
* There is no environment downtime during test due to outages or defect fixes

### Test Principles

* Testing will be focused on meeting the business objectives, cost efficiency, and quality.
* There will be common, consistent procedures for all testing activities.
* Testing processes will be well defined, yet flexible, with the ability to change as needed.
* Testing activities will build upon previous stages to avoid redundancy or duplication of effort.
* Testing environment and data will emulate a production environment as much as possible.
* Testing will be a repeatable, quantifiable, and measurable activity.
* Testing will be divided into distinct phases, each with clearly defined objectives and goals.
* There will be entrance and exit criteria.

### Data Approach

In functional testing, XYZ accommodation management system will contain pre-loaded test data and which is used for testing activities.

### Functional Test

PURPOSE: Functional testing will be performed to check the functions of application. The

functional testing is carried out by feeding the input and validates the output from the

application.

**Test acceptance criteria**

1. Approved functional specification document, use case documents must be available

prior to start of test design phase.

2. Test cases approved and signed-off prior to start of test execution

3. Development completed, unit tested with pass status and results shared to testing team to avoid duplicate defects

4. Test environment with application installed, configured and ready to use state

## Test Steps

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Description | Steps | Test Data |
| 1 | Login student | 1. Launch application 2. Enter username and password 3. Click login button | Username : Vino  Password : 1234 |
| 2 | Update student profile | 1. Make changes in Profile details 2. Click save button | Enter new mail id : Vino@gmail.com |
| 3 | Search property | 1. Click properties tab 2. Enter a text and click search button | “Single Room” |
| 4 | View request | 1. Click on View Request link for specific property 2. Input the date of view the property 3. Click send request button | 31-08-2019 |
| 5 | My requests | 1. Click on my requests tab | ---- |
| 6 | Delete request | 1. Click on delete request link beside relevant request 2. Click ok for confirmation notification | --- |
| 7 | Login property owner | 1. Launch application 2. Enter username and password   click login button | Username: Vino  Password : abcd |
| 8 | Update owner’s profile | 1. Make changes in profile details   click save button | Enter new mail id : Vinoproperties  @gmail.com |
| 9 | View my properties | 1. Click my properties tab | ---- |
| 10 | View requests | 1. Click view requests tab | ---- |
| 11 | Accept view requests | 1. Click on accept link beside relevant view request | ---- |
| 12 | Reject view requests | 1. Click on reject link beside relevant view request | ---- |
| 13 | Log out | 1. Click log out tab | --- |
| 14 | Register new student | 1. Click Sign-up button from login page 2. Enter a username 3. Enter same password and confirm password fields 4. Click register button | Username: suja  Password: 1234 |

## Test Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Test ID | Expected result | Actual Result | Comments |
| 1 | Should be logged in successfully | Logged in successfully | Passed |
| 2 | Student profile should be updated successfully | Student profile updated successfully | Passed |
| 3 | Results should be displayed with relevant input text | Results displayed as per given input search text | Passed |
| 4 | View request should be added to my requests list | View request added to my requests list | Passed |
| 5 | All view requests should be displayed under my requests tab | All view requests displayed under my requests tab | Passed |
| 6 | Request should be deleted under my requests tab | Request deleted under my requests tab | Passed |
| 7 | Should be logged in successfully | Logged in successfully | Passed |
| 8 | Owners profile should be updated | Owners profile updated | Passed |
| 9 | All properties should be view under my properties of belongs to logged in owner | All properties view under my properties of belongs to logged in owner | Passed |
| 10 | All requests should be view under view requests of belongs to logged in owner | All requests view under view requests of belongs to logged in owner | Passed |

# Task E

XYZ Accommodation Management System has three types of users, with different privileges

1. Admin
2. Property Owners
3. Students

## Admin Privileges

Admin can enable or disable any users, and admins are able generate following reports in excel format

* Users reports

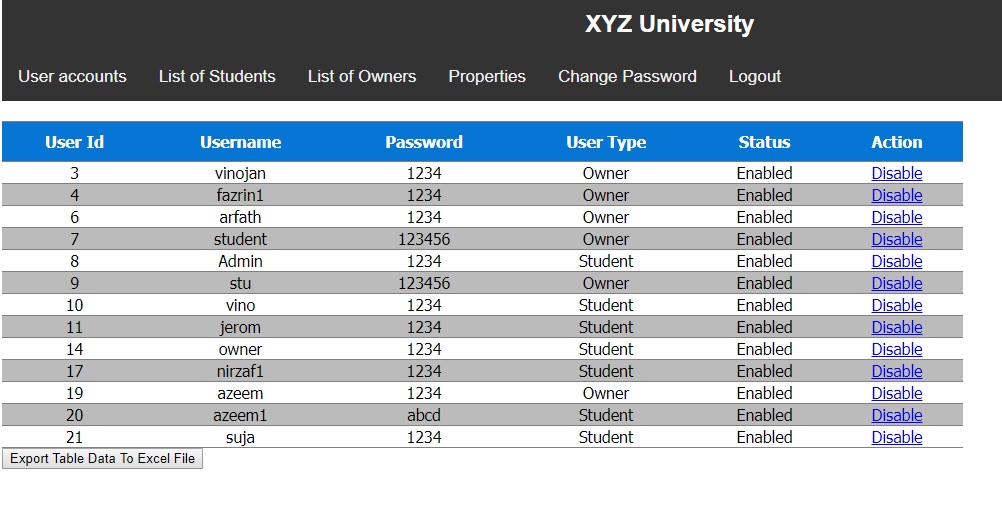


Figure users-list report

* Student list report

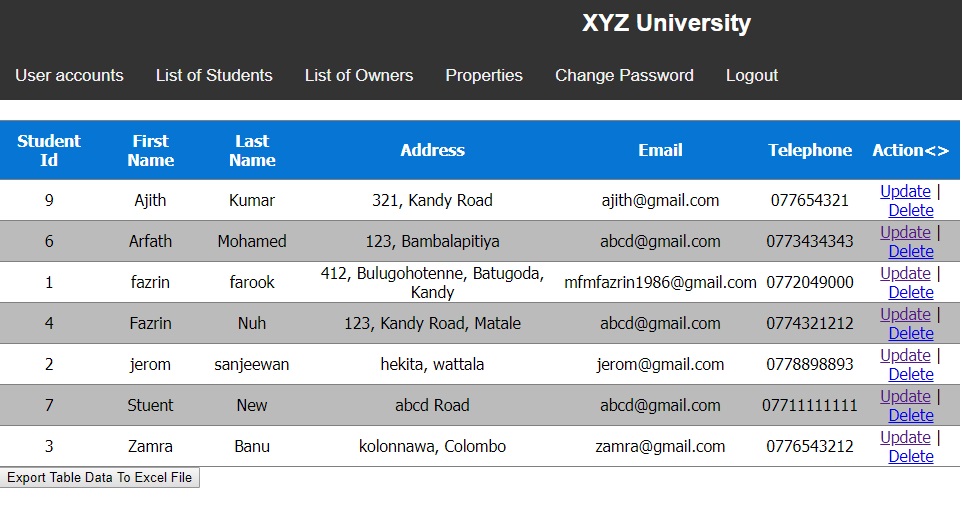


Figure student-list report

* Properties list report

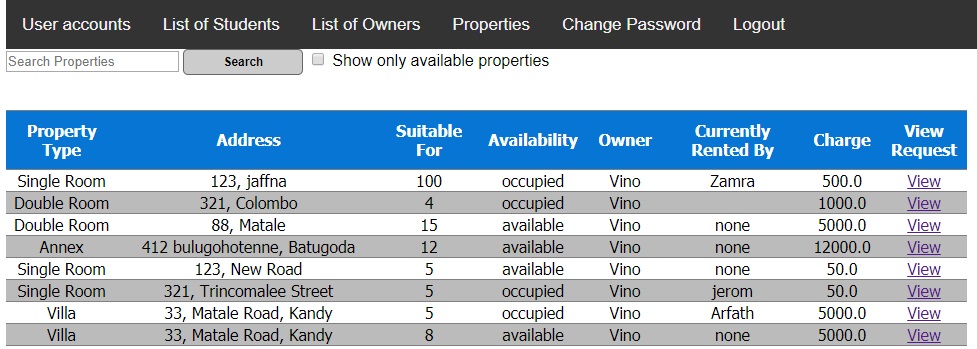


Figure properties-list report

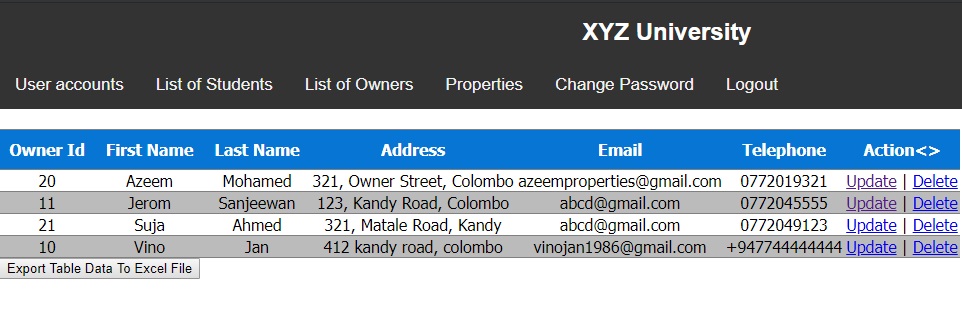
* Owners list report

Figure owners-list report

Admins has ultimate privilege to enable the users once they are registered with a system, moreover they can disable any users on anytime without any prior notification

## Property Owners Privileges

Property owners are able to register themselves and add properties, once they registered in the system their account should be approved by admin before they login, once it’s approved they are able to login with their username and password credentials, once they add the property it will not be listed for student’s view until they make the registration payment, once web application launched it will route to the login page,

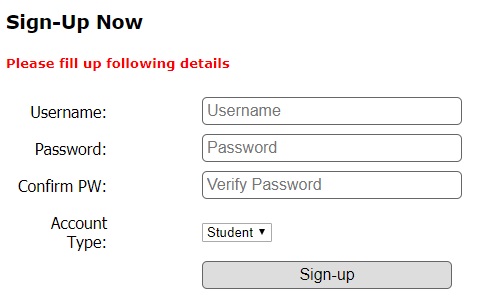


Figure sign-up form

While sign up in the account type drop down list user should select the account type as owner, once login it will route to the update profile page with default values, it should be updated with valid user details and click the save button

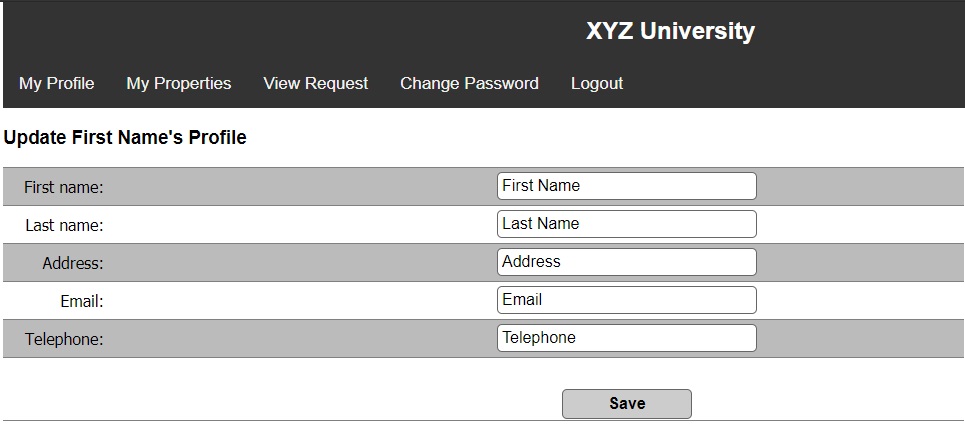


Figure update owners profile page

Once save button is clicked it will route to the profile page, these steps will be same for students user account as well

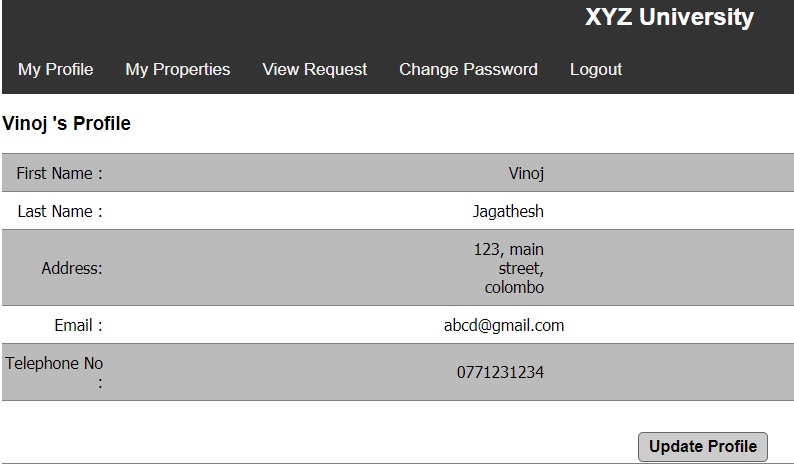


Figure profile-page

Under my properties there will be “Add Property” button to add properties, it will route to the add property page, relevant screenshots are attached below

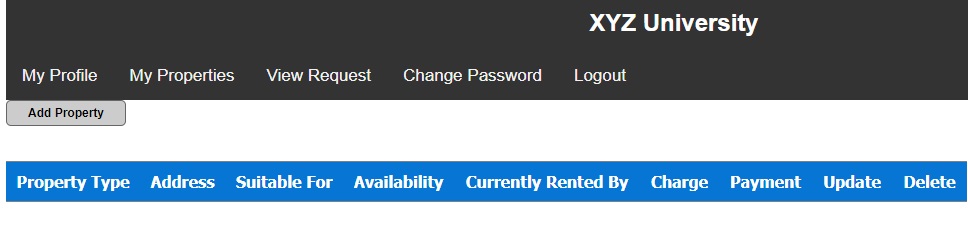


Figure my-properties page

**Add property page will view with following input fields**

* Property types (Need to select the property type from the drop-down list)
* Address (Need to enter property address)
* Suitable for (Need to enter the specific property suitable for how many persons)
* Charge (Need to enter monthly charge for this specific property)

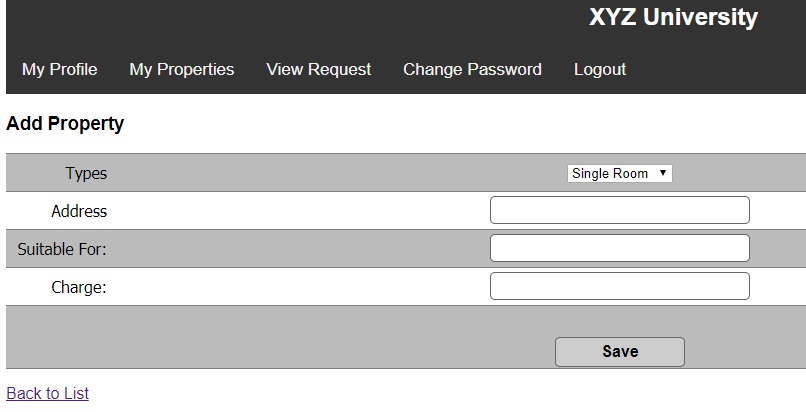


Figure add-property page

Once click the save button after add property it will rout to my properties page and property will be added to my properties tab

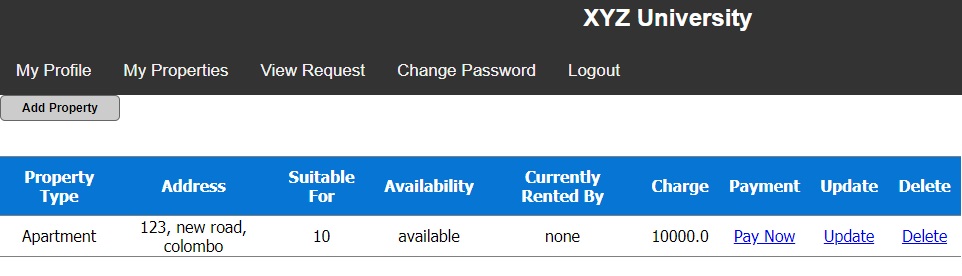


Figure property-added to my property list

Once property added to my property list need to make a payment through the link given in the same raw as “Pay Now” under “Payment” Column, once that link is clicked it will pop up the confirmation alert, after confirming the notification the link will be turned into “Paid”.

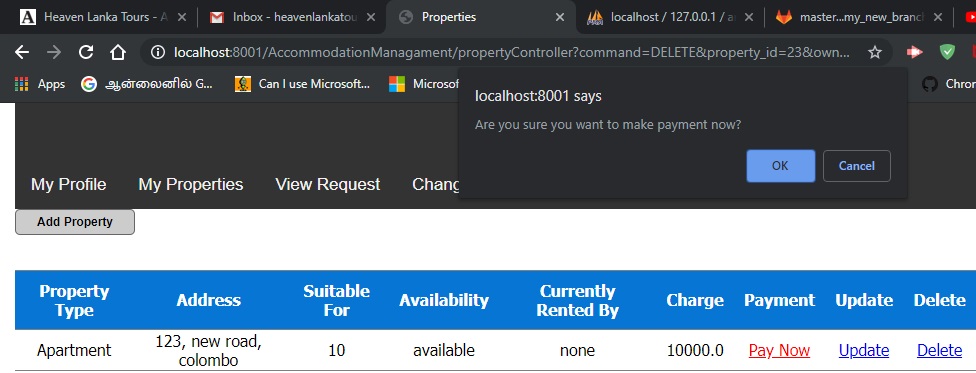


Figure payment confirmation pop up

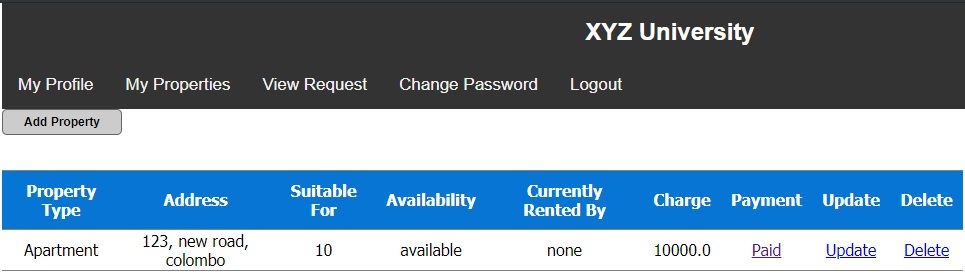


Figure after payment confirmed

View request tab will show all the property’s view request which were requested by students to view the properties which are posted by the current logged in owner,

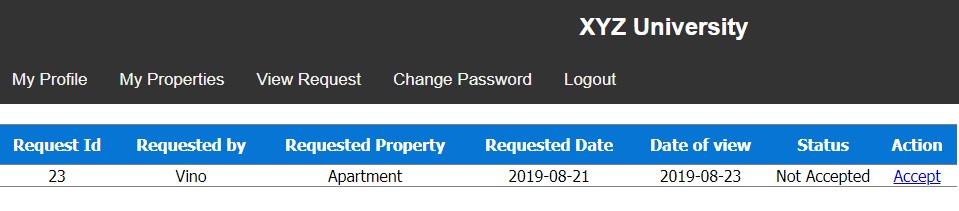


Figure view request to view properties

Under this tab property owner has a privilege to accept or reject the view request by clicking the link under action column.

**Change password**

Change password tab will allow users to change the password by entering new password in both password and confirm password fields, then click the “Change-Password” button, these steps are same for all three types of accounts(Property owners, Students, Admin).

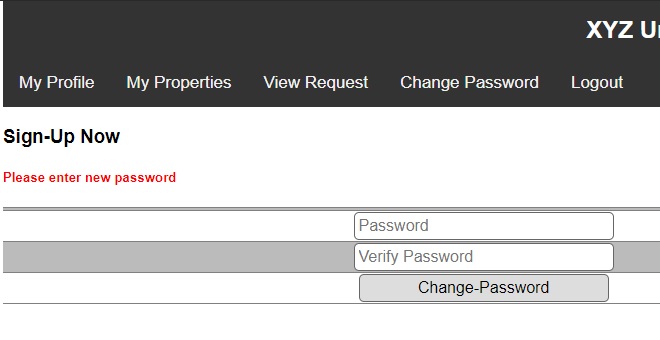
****

Figure change password function

Log out tab allows users to log out from the system, this function will clear all the cookies recorded about your login credentials, this step is same for all three types of accounts(Property owners, Students, Admin).

**View all listed properties**

After logged into the student account, under properties tab all the available properties will be listed,

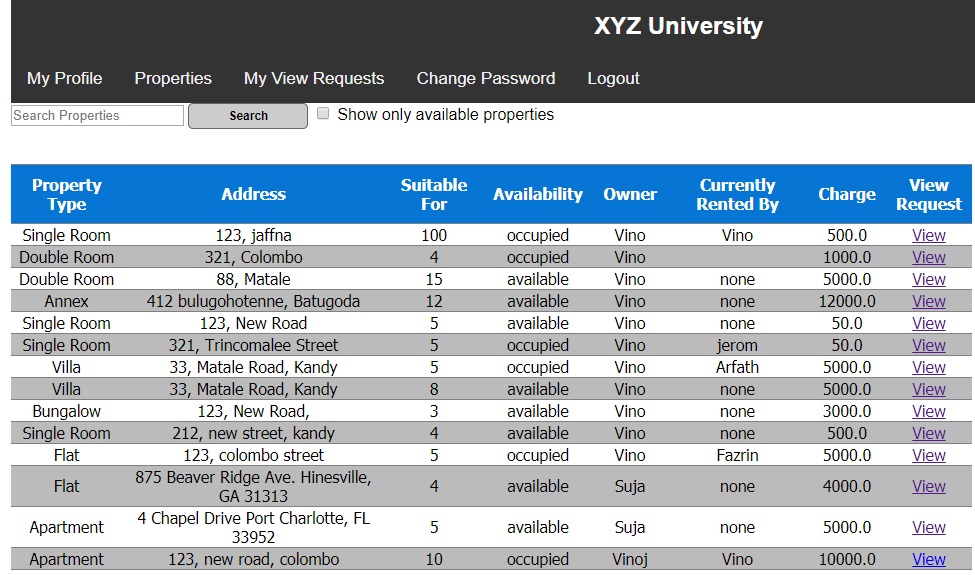


Figure view all properties

**Search properties**

to search properties, input the search term into the textbox and click the search button, to make sure to view only available properties “Show only available properties” check box should be checked

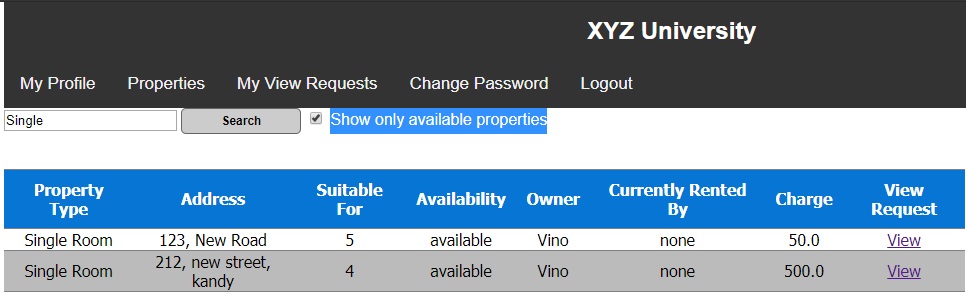


Figure search only available properties with 'single' as search term

**Send view request**

Click the “View” link under **“View Request”** column to send the view request to the selected property owner, student has to provide the expected date of viewing the property and click on the send request button

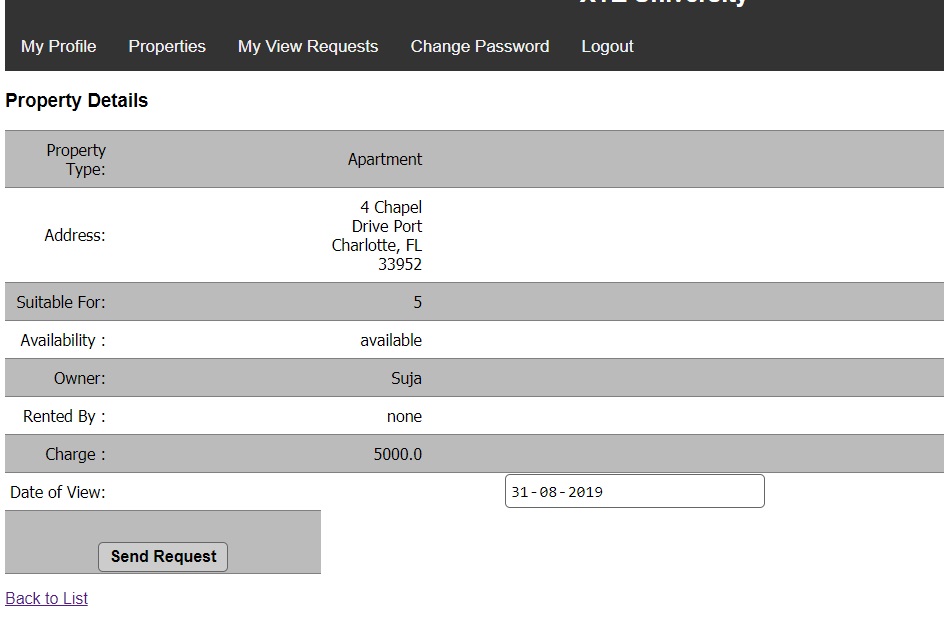


Figure send view request page

**My View Requests**

Logged in student can view all the “view requests” under **“My View Requests”** tab and able to check the existing view requests and request status whether accepted or not

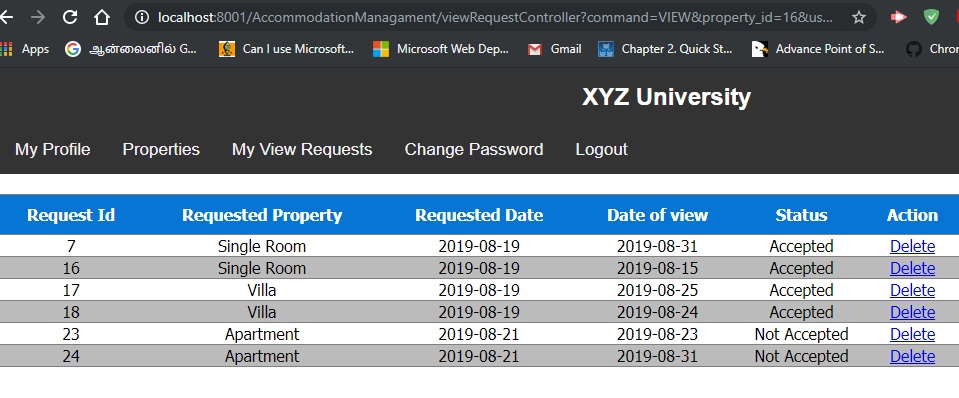


Figure my view requests page

**Enable or Disable Users by Admin**

Admin can enable or disable the user status under “Action” column,

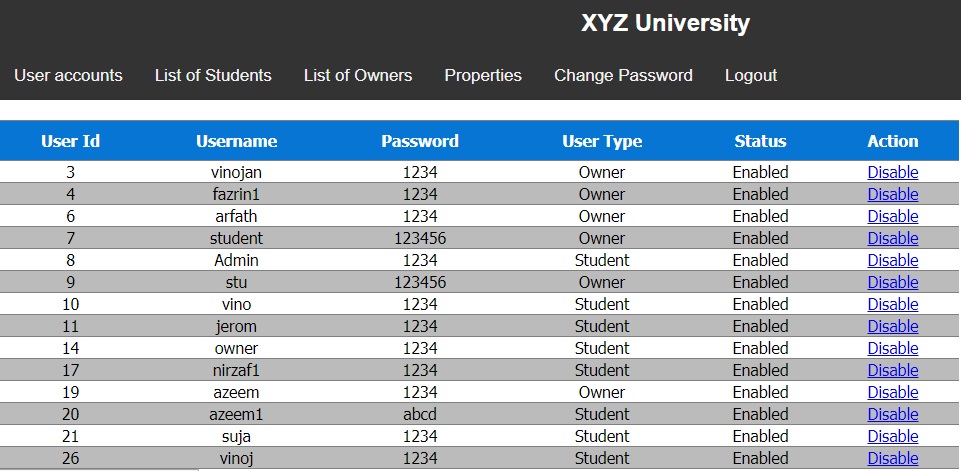


Figure list of user accounts page

# Task G

XYZ accommodation management system is developed in java web technology which includes JSP(Java Server Pages), Java EE is an open source development environment which is used to develop many enterprise solutions, due to its scalability and performance it is popular among the large scale solution providers, more over most of the libraries in Java EE is open source and free to use which are more than enough for current requirement of the particular project mentioned above.

Eclipse IDE is used to develop this project since it has many premium features to speed up the development and it is free to use; MySQL database server is used to store the data since it is very highly integrated with Java platform since both the technologies belongs to the same organization. More over it is free database tool to use even in enterprise solutions, Apache Tomcat server is used to run the web application and test the application, it is a light weight server to run web apps which is most suitable free tool for small projects to run on it.

MVC architecture approach is utilized as an architecture of this system which is useful to reduce the coupling between components and increase the cohesion, HTML, CSS and JavaScript languages used to develop the user interface of the application, JSP scriptlets are utilized in user interface to produce dynamic pages as per required,

Learning curve was bit high in this project due it’s requirement of full stack development technologies, currently this project has following pros and cons

Pros :

* All the required functions are available, and applications runs very smooth without any performance issues
* Since it is a web application it will be able to access through any operating system
* It is secured by proper validation mechanism with login credentials
* Back end business logics are developed in MVC architecture to scale according to requirement changes

Cons:

* User interface is not that much attractive enough
* Reports are not formatted in table format
* No additional innovative features are implemented
* Automatic backup for data is not configured

Future recommendations

* Payment feature for students to pay monthly rentals
* Adding images of properties while adding property details
* Realtime chatting and messaging features between students and owners through web application
* Communication feature between students to rent a shared property after discussion between students
* Commenting facility for properties and rating system from tenants as a feedback
* Voice calling feature through web application between students and property owners
* Search properties depend on location radius and distance based between the university and property
* Share property link to other registered students in the system
* Report to the admin about spamming property owners or students
* Automatic payment request notification to the students on specific time

# Appendix

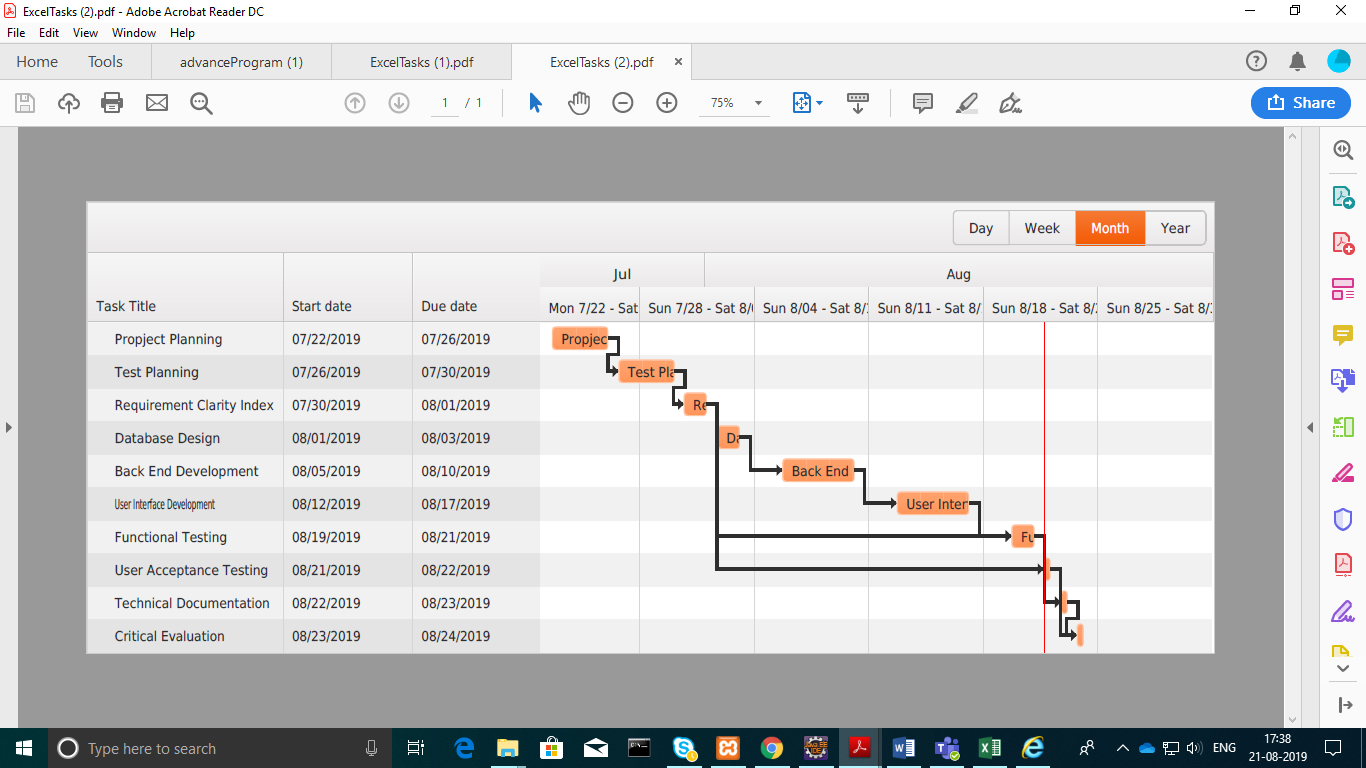


Figure Gantt chart for XYZ accommodation management project

# References

javatpoint. (2019, 08 02). *Factory Method Pattern*. Retrieved from https://www.javatpoint.com: https://www.javatpoint.com/factory-method-design-pattern

Kumar, S. (2015, 03 14). *Abstract Factory Pattern*. Retrieved from https://www.geeksforgeeks.org: https://www.geeksforgeeks.org/abstract-factory-pattern/

RudraSambyal. (2018, March 02). *Let’s examine the pros and cons of the Singleton design pattern*. Retrieved from www.freecodecamp.org: https://www.freecodecamp.org/news/singleton-design-pattern-pros-and-cons-e10f98e23d63/