

Online Coworking Spaces Management & Booking System

J.N. Selvaraja

2020



Online Coworking Spaces Management & Booking System

**A dissertation submitted for the Degree of Master of
Information Technology**

J.N. Selvaraja

University of Colombo School of Computing

2020



Abstract

Coworking spaces are an emerging trend around the world serving startups and companies that require a temporary or a flexible office space for a reasonable price. With the need for coworking spaces developing in Sri Lanka, this study is aimed to provide an online one-stop platform to book and rent out coworking spaces with ease. The study identified several problems with the prevailing booking system of the coworking space providers. Several companies which offer coworking spaces use a manual system to take bookings and few of the companies that have an online platform, do not have a well-organized system to manage their day to day activities. Furthermore, customers who want to book coworking spaces are faced with the difficulty of visiting each individual company websites to make their bookings as there is no one single platform for users to browse through all the possible spaces at once.

With the objective to overcome the above issues and to increase the productivity of the existing systems, a detailed study was conducted on the manual system, and on the individual online systems used by some of the companies. Based on the detailed study, an online coworking space management and booking system was designed to address the identified issues. The system was developed using the CodeIgniter framework using PHP, MySQL and Bootstrap. Furthermore, this system adheres to the Xtreme Programming methodology, which was well suited for this project.

Based on the post-development evaluations, the system received satisfactory client responses and clients were willing to go live with the website. Thus, the primary objective of the study was achieved successfully. The system facilitates the coworking space owners to advertise the type of spaces they offer along with a booking management system for their systems while customers can browse all the different types of spaces and book it for the day and time they want via single platform.

However, the system requires further development in areas of performance and usability of the website and integrating an online payment feature to accommodate users to make payments with ease. These changes are expected to be done in sprints and can be released to live in the future, making the designed system on par with International Standards.

Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name : Joel Nilenth Selvaraja

Registration Number: 2017/MIT/072

Index Number: 17550722

Signature:

Date:

This is to certify that this thesis is based on the work of Mr. Nilenth Joel Selvaraja under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name: Dr. H.A. Caldera

Signature:

Date:

Acknowledgement

I would like to express my deepest appreciation to all who supported and motivated myself in completing this thesis.

First, I would like to thank my supervisor Dr. H.A. Caldera, who helped with stimulating suggestions, guidance and encouragement throughout the project period especially in conducting the research and writing of the thesis.

In addition, I would like to thank all the members of the Academic and non-academic staff at UCSC for the support and assistance offered to myself in numerous ways.

Last but not least I express my whole hearted gratitude to my family, my friends and for the Management and staff at Pragmatic Labs, for willingly giving their utmost support, advice and motivation complete my project successfully.

Table of Contents

Abstract	ii
Declaration	iii
Acknowledgement.....	iv
List of Figures	viii
List of Tables.....	ix
List of Abbreviations.....	x
Chapter 1: Introduction	1
1.1 Introduction	1
1.2 Motivation	2
1.3 Objectives.....	4
1.4 Scope	4
1.4.1 Coworking Space Admin scope	5
1.4.2 Customer Scope.....	6
1.4.3 Out of Scope.....	6
2 Chapter 2: Background.....	7
2.1 Problem Analysis.....	7
2.1.1 Problems faced by Coworking Space Providers.....	7
2.1.2 Problems faced by Coworking Space Customers.....	8
2.2 Solutions identified from our analysis.....	9
2.3 Existing System.....	11
2.3.1 Workflow of the existing system.....	11
2.3.2 Pros of Current System.....	13
2.3.3 Cons of Current System.....	13
2.4 Similar Systems	13
2.5 Requirement Analysis	16
2.5.1 Functional Requirements.....	16
2.5.2 Non Functional Requirements	22
2.5.3 Software Requirements	23
2.5.4 Hardware Requirements	24
2.6 User Classes and Characteristics	24
2.7 Technical Constraints – Design and Implementation.....	25
2.8 Assumptions and Dependencies	25

3	Chapter 3: Methodology	26
3.1	Choice of Development Methodology	26
3.1.1	Waterfall Model.....	26
3.1.2	Scrum Methodology	28
3.1.3	Extreme Programming (XP).....	29
3.1.4	Selection of Appropriate Methodology	31
3.2	Selection of Modelling Language	32
3.2.1	Activity Diagram.....	33
3.3	User Interface Design.....	35
3.4	Implementation.....	36
3.4.1	Implementation Languages and Tools.....	36
3.4.2	Implementation Strategies	38
3.4.3	Installation Procedure	39
4	Chapter 4: Evaluation.....	40
4.1	Test Plan.....	40
4.1.1	Black Box Testing.....	40
4.1.2	White Box Testing.....	40
4.1.3	Unit Testing.....	41
4.1.4	Integration Testing.....	41
4.1.5	System Testing	41
4.1.6	User Acceptance Testing.....	41
4.1.7	Testing Strategies	41
4.1.8	Exit Criteria	43
4.2	Test Results	43
4.3	Evaluation.....	45
4.4	Evaluation Results.....	47
4.4.1	Evaluation Result for Appearance.....	47
4.4.2	Evaluation Result for Functionality.....	48
4.4.3	Evaluation Result for Usability	49
4.4.4	Evaluation Result for Performance.....	50
4.5	Overall Evaluation.....	51
5	Chapter 5: Conclusion	53
5.1	Problems encountered and lessons learnt	54
5.2	Future Enhancements	55
6	References	56

Appendix A: User Manual.....	59
A.1 Home page.....	59
A.2 Customer Sign up page.....	59
A.3 Customer Login Page	60
A.4 Tables Page.....	60
A.5 Table Details Page.....	61
A.6 Customer Booking Form	61
A.7 My bookings section.....	62
A.8 Admin panel page.....	62
A.9 Admin Sign up and Create Space page.....	63
A.10 Admin dashboard page	63
A.11 Tables Page.....	64
A.12 Add New table Page	64
A.13 Bookings Page	65
A.14 Admin Settings Page	65
Appendix B: Important Code Segments.....	66
B.1 Code segment for database configuration settings.....	66
B.2 Code segment for admin registration and create spaces	67
B.3 Code segment for create table process.....	68
B.4 Code segment for update booking status	69
Appendix C: Use case diagram	70
Appendix D: Activity Diagram	71
D.1 Activity diagram of Coworking Space Admin	71
D.2 Activity diagram of Customer	72
Appendix E: Test case Document	73
Appendix F: Gantt chart.....	78

List of Figures

Figure 2.1. Workflow of the existing system	11
Figure 2.2. Hub9 WhatsApp booking interface.....	14
Figure 2.3. Loft1024 booking screen	15
Figure 2.4. Use case diagram Level 1	18
Figure 2.5. Use case diagram Level 2 - coworking space management.....	19
Figure 2.6. Use case diagram Level 2 - Booking Spaces	20
Figure 2.7. Use case diagram Level 2 - Manage Bookings	21
Figure 3.1. Waterfall Model	27
Figure 3.2. Scrum Methodology.....	28
Figure 3.3. Extreme Programming Methodology.....	29
Figure 3.4. Activity Diagram – coworking Space Admin	33
Figure 3.5. Activity Diagram – Customer	34
Figure 4.1. Graphical Representation of Evaluation Result for appearance.....	47
Figure 4.2. Graphical Representation of Evaluation Result for Functionality	48
Figure 4.3. Graphical Representation of Evaluation Result for Usability	49
Figure 4.4. Graphical Representation of Evaluation Result for Performance	50

List of Tables

Table 2.1. Use Case Description - coworking space Management	19
Table 2.2. Use Case Description - Booking Spaces	20
Table 2.3. Use Case Description - Manage Bookings	21
Table 4.1. Test Report	43
Table 4.2. Likert scale options and weights	45
Table 4.3. Software Evaluation Form.....	46
Table 4.4. Evaluation Result for Appearance.....	47
Table 4.5. Evaluation Result for Functionality.....	48
Table 4.6. Evaluation Result for Usability	49
Table 4.7. Evaluation Result for Performance	50

List of Abbreviations

CSRF	Cross-site request forgery
GB	Gigabyte
GHz	Gigahertz
HTTP	Hypertext Transfer Protocol
IT	Information Technology
LAMP	Linux, Apache, MySQL, PHP
MVC	Model View Controller
PHP	Hypertext Preprocessor
RAM	Random Access Memory
SEO	Search engine optimization
SSD	Solid-state drive
UI	User interface
UML	Unified Modeling Language
UX	User experience
XSS	Cross-site scripting
XP	Extreme programming

Chapter 1: Introduction

1.1 Introduction

Coworking spaces were introduced more than a decade ago but its existence came to the knowledge of many only a few years ago. Now coworking spaces can be found in many cities and is a growing business model around the world. Coworking spaces are type of shared offices, spaces or desks that are offered with the complete infrastructure of an entire office which are rented out by Space Admins at an hourly or daily rate, or longer. Knowledge workers, company employees who want to avoid the commute and freelancers who are tired of working at the kitchen table, are fueling the demand for coworking spaces. It is an environment where people from different professions are working in the same space, and, thus, can help foster collaboration.

In Sri Lanka, coworking spaces are a growing trend and many coworking spaces are emerging all around the country especially in Colombo. Many startups in Sri Lanka prefer to use coworking spaces to operate their business rather than private offices mainly to avoid the fixed cost incurred in an owned space and to avoid the inflexibility to scale up or down when needed. Not only Sri Lankans but there are also many foreign companies, which setup offices in Sri Lanka for business ventures, which opt to work in coworking spaces.

Pragmatic Labs is an independent software testing space admin for IT and IT enabled organizations around the globe. They are currently providing testing services including Software Performance Testing, Automation testing and Manual Testing for clients from Australia, Canada and Sri Lanka. Currently they have around 30 employees and are based in Colombo, Sri Lanka. In 2010, Pragmatic Labs opened as a startup with four employees inside a cubicle rented out from a private educational organization. During the first two years, they continued their office in that cubicle and as their number of clients and employees increased, they moved to larger spaces and are now occupied in a three floor building in the heart of Colombo.

This three-floor office space resulted in excess and idle space and the company was opting for ways to use the idle space efficiently. Though the initial strategy of the company was to rent out the space for a long term, the company subsequently realized that during certain parts of the year they required the excess space to conduct their own workshops and meetings and therefore could not rent it for longer periods. Hence, the company was on the lookout for opportunities which could make use of that space during short terms. Subsequently, the company was made aware about the concept of coworking spaces and started implementing it in their office space.

The coworking space concept was an efficient way of using excess space for Pragmatic labs and some of their local clients were interested in that concept and implemented it in their own offices. Pragmatic Labs and their clients used a manual paper-based system for the bookings where customers can call and book a space they required and the bookings were recorded manually in a register. Over time, this system became very complicated, as they had to handle large number of calls from customers where they had to explain to each person about the type of spaces they rent, available time slots and faced booking complications. Their clients who were implementing the coworking service also faced similar issues. This resulted in the management of Pragmatic Labs requiring a solution for this issue and requested to offer a solution by conducting a research on their customers, clients who are providing coworking spaces and other coworking space admins.

1.2 Motivation

One of the main issues related to coworking spaces in Sri Lanka is finding a coworking space congruent to their needs. Many coworking spaces have their own individual website to advertise and to accept bookings, therefore when a customer wants to a book a space, the customer is compelled to visit each provider's website and check the spaces, facilities and time slots available. There are many coworking spaces especially new ones, which do not have a website to advertise details for potential customers. Due to this many customers remain loyal to few popular coworking places with an online presence which are always booked to full capacity, while other coworking spaces which do not have an online presence, are continued to be idling without bookings due to lack of awareness. Owing to the same reason, customers

have to travel a long distances to find a coworking space, which defeats one of the main purposes of coworking spaces even though there could be coworking spaces nearby which the customer is not aware about.

Another issue related to coworking spaces is that Space Admins find it hard to promote their spaces and customers who use coworking spaces find it hard to get a reliable source of information about these spaces. People who use coworking spaces are a niche segment and it is not easy for a newly opened coworking space company to target such customers precisely. Due to this as mentioned earlier, many coworking space companies are idled even though there is a strong demand by a specific segment. For many companies especially startups running their businesses in a coworking space is more profitable than a private place but they fear moving to them due to lack of awareness and proper information about the spaces. Many coworking space customers mainly depend on word of mouth to find the place that suits them and there is no other neutral source to know secure, reliable and flexible coworking spaces. There are many companies in Colombo like Pragmatic Labs with excess office space, which do not use the excess space optimally. These companies avoid renting their excess spaces long term due to costs related to tenancy agreements, and other management issues. The purpose of our system is to help such companies to adapt to the concept of coworking spaces by helping them to rent out their excess spaces on a daily or hourly basis. Our system will help to manage the marketing and the business management aspect, which will be profitable to the company as well as help customers who seek coworking spaces.

Coworking spaces are a concept that should be encouraged at this day and age where startups are promoted to improve the country's economy especially in countries such as Sri Lanka where entrepreneurial ventures are supported. One of the main reasons many startups fail is due to the expenses incurred to own an office space. In cities like Colombo, there's very limited office spaces for rent and the ones available are very expensive making it difficult to bear for startups. Coworking spaces are a great solution to avoid these costs and the purpose of our system is to help startups and freelancers to find such spaces with ease and convenience.

1.3 Objectives

The main objective of this project is to create an online platform for all Coworking Space Admins where they can promote and rent out their spaces and connect them with their prospective clients via single platform. In achieving this objective a website is built, which fulfills the following goals.

- A system where coworking space providers can promote their company and display the provided facilities.
- A system where Space Admins can display the structures of the offices, spaces or desks they provide along with their hourly rates.
- A system where coworking space providers can manage their bookings which will be helpful for new space providers who don't have their own booking system or private website.
- A system where space providers can receive feedback and reviews about their services and how they can improve their spaces.
- A system where space providers can generate reports and statistics regarding the bookings they received.
- A system where customers can check the details of different space providers and all the facilities provided by them.
- A system where customers can search and filter out all coworking spaces according to the facilities provided, service rates and other essentials.
- A system which is mobile friendly where users can book spaces they need on the go.
- A system where users can rate and review the spaces they have worked in and provide feedback to the Space Admins.

1.4 Scope

The scope of this project is divided into two based on the two types of users; the Coworking Space Admin and the Customer

1.4.1 Coworking Space Admin scope

1. Space Creation Module

- Space Admin should be able to create his company in the site and enter details of his company such as location, Amenities provided by company, Photos of the location, opening and closing times etc.
- Space Admin should be able to create individual spaces in his company in the site such as an individual table or a table for a group or a private room and the number of people who can use it and charge for each space
- Space Admin should be able to display all the free amenities provided in each space such as printers, coffee machine, table lamps etc.
- Space Admin should be able to display the charges for additional facilities such as printouts, projectors etc.

2. Space Reservation Module

- Space Admin should be able to allocate daily or hourly slots for each space and should be able to check booking for each slot for each space.
- Space Admin should be able to block slots for the orders received privately.

3. Space dashboard Module

Space Admin should be able to track bookings, time used, and see who is working where and when using the activity dashboards.

4. Space reporting Module

- Space Admin should be able to read reviews and ratings his space has received.
- Space Admin should be able to generate a monthly report of all the bookings with different type of data to make data-driven decisions.

1.4.2 Customer Scope

1. Public Space Details Module

- Customer should be able to see the Space Admin Company's details page with their location details, facilities provided, photos, opening and closing times etc.
- Customer should be able to see the various spaces available in the particular company and the facilities provided in each of them and cost for each space and if there are any payable facilities along with the charges for each of them.

2. Public Space Search Module

- Customer should be able to search a location and all space companies in and around that, location should be displayed to the customer.
- Customer should be able to filter types of spaces required by location, type of customer (Individual or Group), amenities available etc.
- Customer should be able to see space suggestions according to categories such as Most liked, Most Reviewed, Beach View etc.

3. Public Booking Module

- Customer should be able to book an available time slot in a space either per hour or per day and as soon as a booking has been made, the space should not be available for further bookings.
- Customer should be able to cancel a booking or a facility he booked.

4. Public Review and Rating Module

- Customer should be able to like and rate the spaces he has worked in and provide a review about it
- Customer should be able to generate a monthly report of the bookings made along with other details such as costs.

1.4.3 Out of Scope

This system scope will not contain the online payment facilities for the customers.

Chapter 2: Background

In this chapter, we initially present the problems faced by both coworking space providers and coworking space customers. Subsequently, we analyse the existing and the proposed system in depth identifying the functional, non- functional, software and hardware requirements needed to develop the proposed system.

2.1 Problem Analysis

In order to understand the problems in coworking space arena, we conducted interviews and surveys with coworking space providers and coworking space customers. Furthermore, we were curious to understand as to why some entities such as startups and freelancers, continue to use traditional options for space management rather than coworking option. Therefore, we held interviews and surveys with such entities as well to understand their perspective on coworking spaces and their reasons for not using them.

Following are some of the issues discovered through our research.

2.1.1 Problems faced by Coworking Space Providers

Difficulty in managing a Coworking Space

There are many companies, which operate with excess space in their offices and are ready to change it to a coworking space similar to our client but they give up the idea due to the complexity of the process especially due to the need to perform it manually. Our client started their business by taking bookings through calls and e-mails but over time, the process became cumbersome. This created the need for a proper management system to manage the bookings and payments. Even when considering management systems, many existing coworking space companies have invested significantly on building these systems. However, most of such systems are very basic and do not provide valuable information the company need to gather to improve their business.

Difficulty in Promoting a Coworking Space

The customers who use coworking spaces belong to a niche segment and it is not easy for a newly opened coworking space company to target these customers. Coworking spaces are mostly used by entrepreneurs, IT employees, startups, freelancers etc. and reaching them through traditional marketing techniques is a difficult task.

One of the newly opened companies we interviewed said that they engaged in many marketing campaigns through newspaper ads but no significant improvements were visible in their business. Many newly opened coworking spaces do similar mistakes and due to this, many coworking space companies are idled even though there is a heavy demand for such spaces.

Lack of knowledge regarding the industry

Many Sri Lankan companies who want to enter the coworking space industry do not have proper know-how on how to start it and what procedure to be followed. Many companies we interviewed communicated to us that they had to rely on foreign consultants to understand the resources and technology needed. Furthermore, they explained how they had to go around many government departments in order to get the correct licenses and certificates, as there was not a proper guide for them to follow.

2.1.2 Problems faced by Coworking Space Customers

Difficulty in finding Coworking Spaces

One of the main issues related to coworking spaces in Sri Lanka is the difficulty in finding a good and near-by coworking space. Many coworking spaces have their own private website to advertise and book their spaces. Therefore when a user wants to book a space he has to individually go to each private website and check the spaces, facilities and time slots available. Similarly, there are many coworking spaces especially new ones, which do not have a website. Therefore, customers have to call and verify whether spaces are available.

Due to this, many customers prefer the few popular coworking space companies that have a website. Hence, these popular spaces are always booked while coworking spaces without an online presence are idled without bookings creating an imbalance in the industry.

Lack of knowledge regarding the benefits of coworking spaces

For many startups and other small businesses, running their business in coworking spaces is more profitable than a private property. However, many such Sri Lankan businesses are not aware that such a facility exists and therefore overspend on long-term rent and building new offices.

Another issue that small businesses and startups face is the fear moving to them due to lack of awareness and proper information about the coworking spaces. Many coworking space customers mainly depend on word of mouth to find such spaces that suits them and there is no other neutral source to provide information on how secure and reliable these places are and how flexible and profitable they are for them.

2.2 Solutions identified from our analysis

After our analysis, we put forward our observations and conclusions to the client. Discussions were held on the possible solutions we had developed and accommodated client's suggestions and customizations as follows.

- First, we discussed about how coworking space companies were unable to promote their business and in order to solve it our solution will be a system, which will allow coworking space admins to promote their company and display the facilities they provide.
- Next problem we looked into was how companies, which did not have a website, have to answer customer calls and explain the facilities they provide and their rates. Therefore, the solution we wanted to incorporate into our system was to allow Space Admins to display the structures of the offices, spaces or desks they provide along with their hourly rates.

- Next problem is the issue where most coworking spaces did not have a system to manage their bookings and had to resort to manual systems. To solve this problem our system will have a feature for coworking space admins to manage their bookings, which will be helpful for new space admins who do not have their own booking system or private website.
- Next we looked into on how there was no system for customers to find various coworking spaces around the country and how they have to visit individual sites to know about various spaces. In order to resolve this in our system customers can check the details of different space admins and all the facilities provided by them in one single platform. Furthermore, customers also will be able to search and filter out all coworking spaces according to the facilities provided, service rates and other essentials according to their preference.
- Next problem we looked into was how customers did not have a way to review and recommend various spaces they used and how other customers didn't have a source to know feedback about various spaces. In order to resolve this we will allow users to rate and review the spaces they have worked in and provide feedback to the space admins. This will also allow coworking space companies to receive feedback and reviews about their services and how they can improve their spaces.
- When coworking services manage their bookings manually or use a simple system they will find it hard to obtain various valuable data about their business, which if received can be used to develop their business. In order to solve this we will introduce a feature where Space Admins can generate monthly reports regarding their bookings.
- Most private websites owned by coworking spaces are not user friendly and many of them have to be accessed through desktop to make the bookings. In order to solve this our system will be designed in a mobile friendly way where users can book spaces they need on the go.

2.3 Existing System

The existing system of Pragmatic Labs is a manual system, which provides its customers with services such as providing information about the spaces, information about the free slots available, information about amenities offered, and booking of spaces etc.

With demand for coworking spaces improving, Pragmatic labs faced difficulties in handling frequent queries, which resulted in waste of time, cost and effort for both the company and its potential customers. This led to the need of improving the current manual system to be more efficient and user friendly. Before development of the requested system, a detailed study on the existing manual system of Pragmatic Labs was conducted to identify the workflow and frequent queries made by the customers.

2.3.1 Workflow of the existing system

The workflow of the existing system is shown in Figure 2.1.

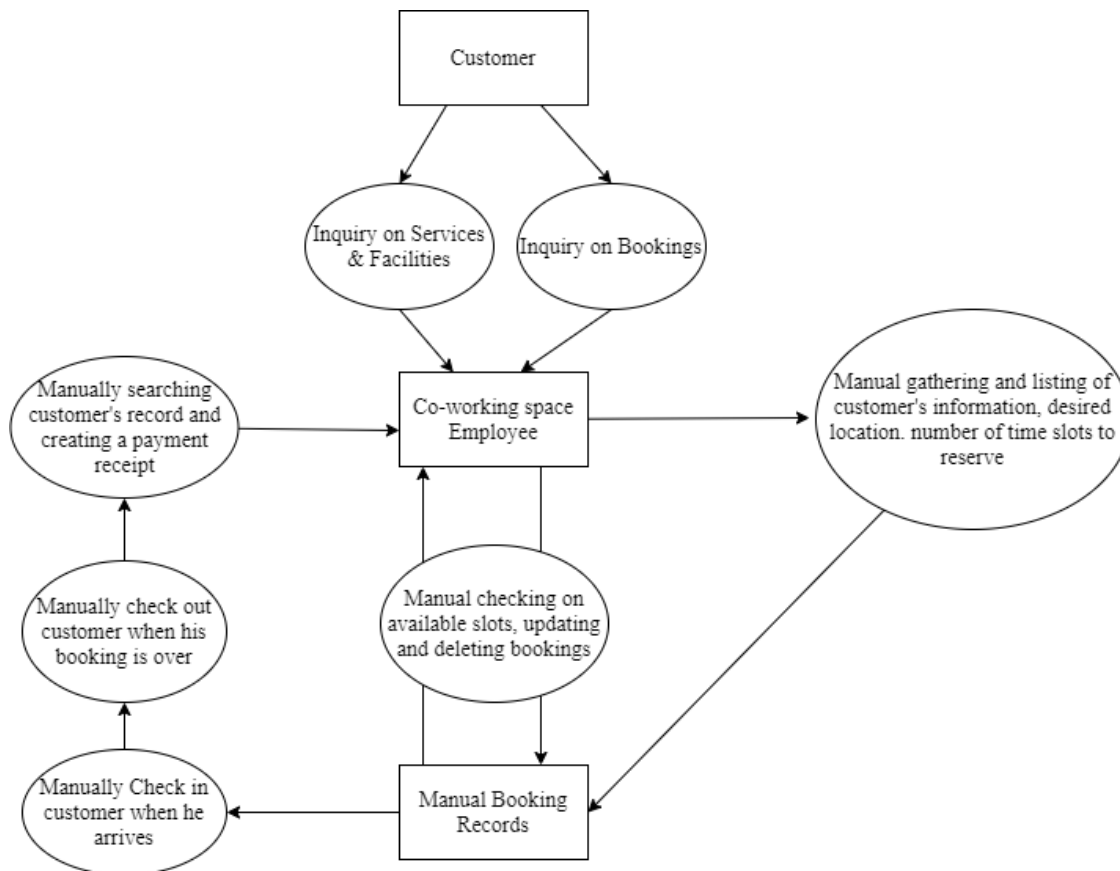


Figure 2.1. Workflow of the existing system

According to the above workflow diagram and the study conducted about the system, the existing system can be elaborated as follows.

- The customer makes a call to the coworking space company in order to know about the details of the coworking space such as the locations, amenities provided, booking methods, rules & conditions etc.
- The company offering the coworking space has to have a designated employee to answer such calls from customers and respond to the inquiries.
- The customer has to call and ask the coworking space whether a particular time slot is free to book or ask which time slots are freely available to book.
- The designated employee has to go through the logbook to see whether a particular time slot is available and if not he has to tell the customer one by one the time slots that are available.
- The customer has to provide all his details and the booking slots he wants to book when he makes a booking.
- The designated employee has to manually enter the booking details in his logbook and enter the details of the customer.
- When the customer comes to the space, the designated employee has to manually enter the check in and check out time of the customer in the logbook.
- When the customer checks out, the designated employee has to manually verify the check in and check out time, calculate the charges and generate a payment receipt.
- If the coworking space owner asks for a report about the bookings and payments, the designated employee has to go through his logbook and manually make a report.

After identifying the workflow, focus was directed towards the pros and cons in using the manual system. Accordingly, following were identified.

2.3.2 Pros of Current System

- In comparison to a fully-fledged website, which will involve higher costs in terms of designing and implementation, the current manual system will be less costly in the short term. However, it must be noted that this benefit is not significant enough to outweigh the following cons in the long term.

2.3.3 Cons of Current System

- Since the booking details are stored in manual logbooks, retrieval of information to respond to queries is slow and inefficient
- The waiting time is high as customers have to call the office and hold calls until designated employee answers the calls to make an inquiry and verify availability to make bookings
- Furthermore, waiting time for new customers is relatively higher as there are more queries on the spaces, amenities, rules of the location in comparison to a repetitive customer
- The employee has to repeat the same details about their services and facilities to each and every customer
- The employee has to manually record the customer details with the bookings, which could lead to errors and miscommunications
- Difficulty in data modification especially when a customer cancels or postpones a booking
- Instances of revenue loss to the company as some customers do not communicate cancellations ahead since no payment was made at the time of booking
- There is a probability of data loss due to records being maintained manually
- Having a designated employee for inquiries and bookings can become expensive in the long term
- Time consuming and expensive to produce reports

2.4 Similar Systems

Proposed system is an online web based management system for coworking spaces that will facilitate users to search for coworking spaces available around the country, book spaces, and view additional facilities. Furthermore, it will facilitate coworking space owners to create coworking spaces, display the facilities available and manage bookings without directly

booking in person or via calls. Upon a preliminary research conducted on the internet following systems were identified as similar to the proposed system of this study.

Hub9 is a Sri Lankan coworking space company that has its own website to market their facilities and services [1]. Hub9 was a pioneer in Sri Lanka when it comes to coworking spaces. Therefore through their website we were able to understand what type of spaces are rented and what type of services are provided in coworking spaces in Sri Lanka.

However, they do not have a proper online booking system and depend on booking through WhatsApp. When Book now option is clicked it leads you to the WhatsApp app where user has to place a booking enquiry message and wait till the space owner reply and confirm the order. Some of the features we were able to learn through this website were the time slots preferred by Sri Lankan customers and the type of spaces provided by Sri Lankan coworking vendors. Figure 2.2 shows the WhatsApp interface through which bookings are made for Hub9.

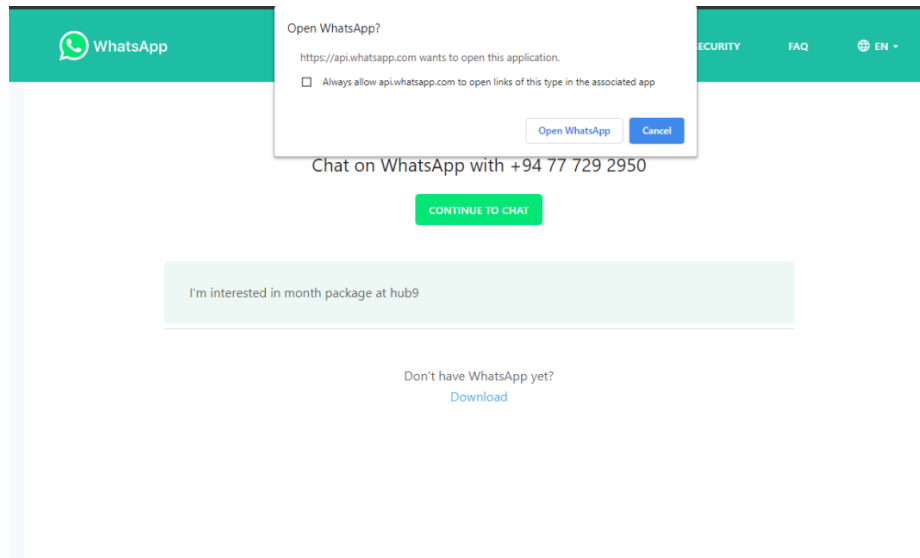


Figure 2.2. Hub9 WhatsApp booking interface

Loft1024 is another Sri Lankan coworking space company that has its own website to market their facilities and services [2]. Through their website, we were able to understand type of add-ons expected by Sri Lankan customers such as Wi-Fi, water bottles, printers etc. and the type of facilities provided by Sri Lankan coworking vendors.

However, they too do not have a proper online booking system. In the current system, customers have to make the booking and have to call the space company to confirm their booking. The current booking screen of LOFT1024 is shown in figure 2.3.

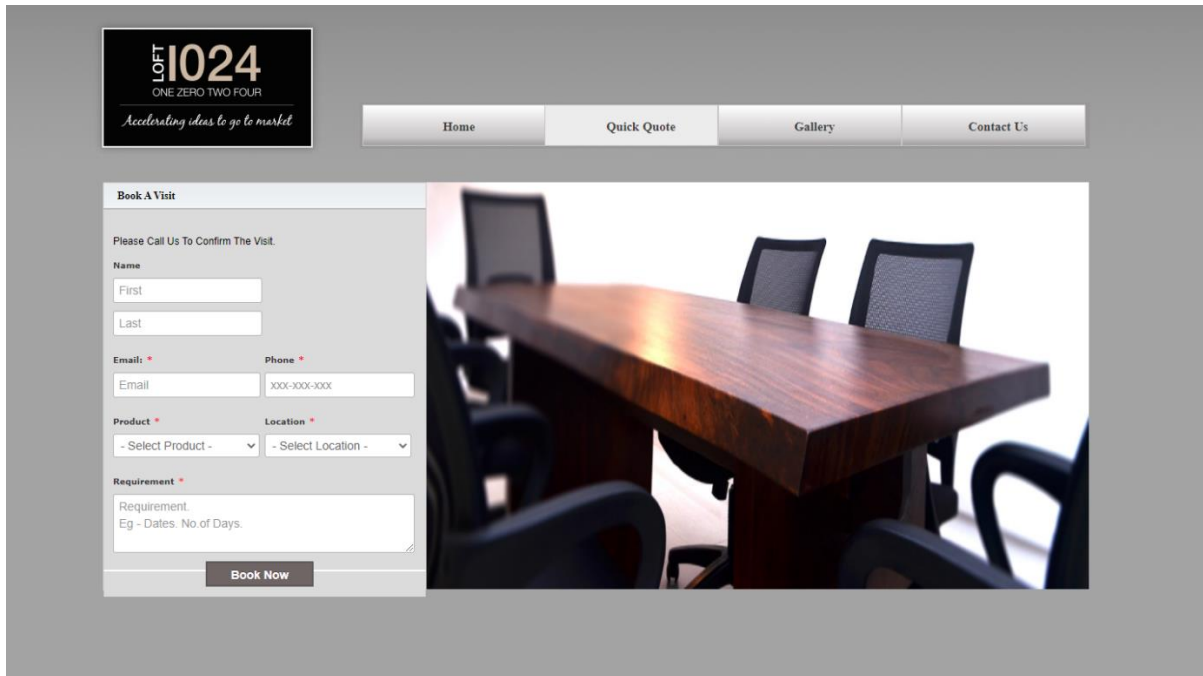


Figure 2.3. Loft1024 booking screen

Sharedesk.net is an international website to book coworking spaces [3]. The features found in this website such as how the booking system works, how the process of registration of spaces work around the world were studied to build our website. We analyzed this site's process flow and studied how it can be adapted to the Sri Lankan coworking space context that we studied in Hub9.

Millionspaces.com is a Sri Lankan website that is used to book concert halls, meeting rooms, sports stadiums and other spaces around Sri Lanka [4]. They are a well-recognized name when it comes to booking venues in Sri Lanka and their business model was studied to understand what type of facilities are provided to attract vendors and customers. Their site does not have a dedicated booking style that is suitable for coworking spaces. Therefore, we adapted their business model and built a website exclusively for coworking spaces.

Booking.com is a travel meta-search engine for lodging reservations and one of the world's leading digital travel companies [5]. They have a very good search and filter tool for searching

different locations and filtering out locations according to facilities and we were able to study it in order to implement it to our Space search.

2.5 Requirement Analysis

A thorough requirement analysis was conducted before the system was built and following are some of the steps taken for the analysis.

- A fact gathering was conducted to find the requirement of the system's end users.
- A feasibility study was conducted determining whether the project is economically, socially, technologically and organizationally feasible.
- A research was conducted to determine how the end users would operate the system and how the system would be used to fulfill the current requested services.

2.5.1 Functional Requirements

Following are the list of Functional requirements in the system, which we have been identified during the requirement analysis.

- Registration and Login - User should be able to register as a customer or a space owner using the signup page where he can select his role. The username and password will allow them to login and according to the role they signed up, they can access different features.
- Space Company Creation – Space owner should be able to create his company in the site and enter details of his company.
- Individual Space Creation - Space owner should be able to create individual spaces in his company in the site such as an individual table or a table for a group or a private room and the number of people who can use it and charges for each space.
- Space Facilities creation and display - Space owner should be able to display the charges for additional facilities such as printouts, projectors etc.
- Allocate time slots – Space owner should be able to allocate daily or hourly slots for each space and should be able to check booking for each slots for each space.

- Block time slots - Space owner should be able to block slots for the orders received privately.
- Track bookings - Space owners should be able to track bookings, time used, and access on who's working where and when.
- Generate Reports – Space owners should be able to generate reports of the orders they received.
- View Space Details - User should be able to see the space company's details page and see the various spaces available in the particular company and the facilities provided in each of them and cost for each space.
- Search Spaces - User should be able to search a location and all space companies in and around that location.
- Filter Spaces - User should be able to filter types of spaces required by using the filters available.
- Book Spaces - User should be able to book an available time slot in a space.
- Cancel bookings - User should be able to cancel a booking or a facility booked.
- Rate and Review Spaces - User should be able to rate the spaces worked in and provide a review.

The main functionalities from the above-mentioned functionalities and their internal and external influences are represented using Use case diagrams below.

The Level-1 Use case Diagram of the System is shown in figure 2.4.

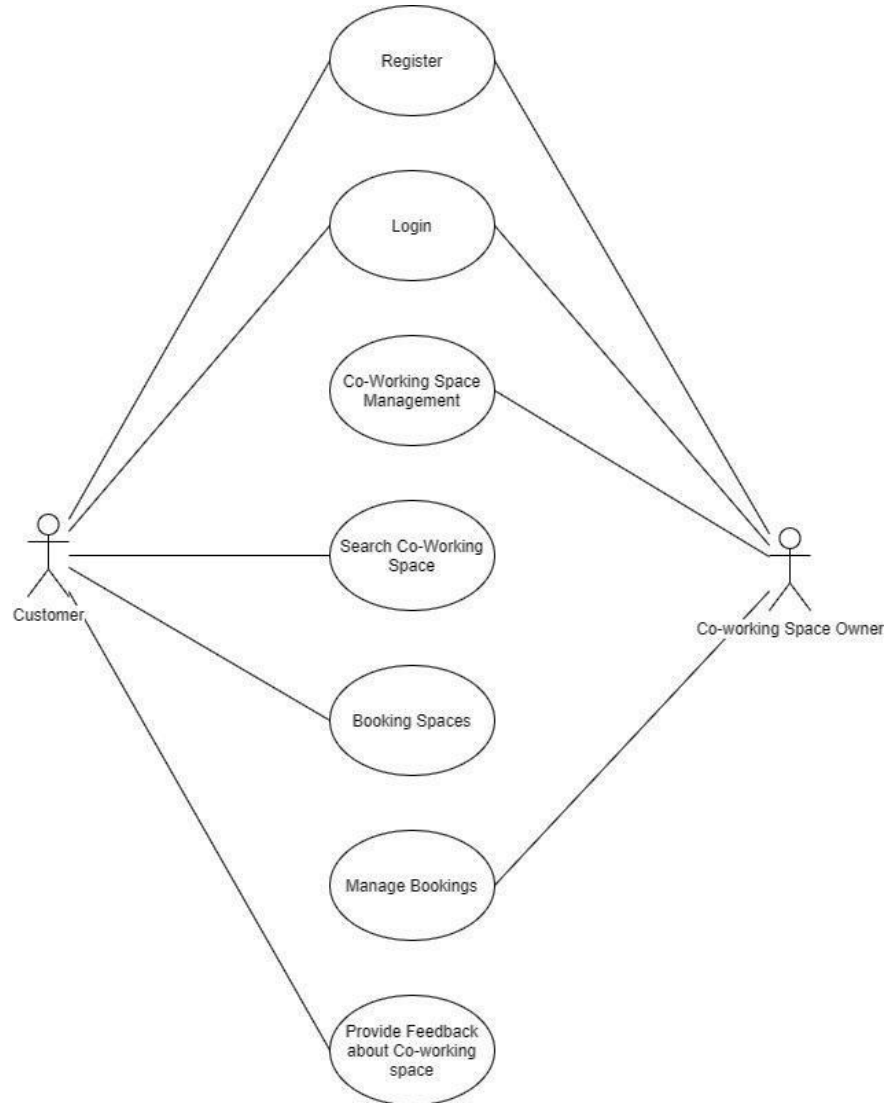


Figure 2.4. Use case diagram Level 1

Use case diagram for coworking space management is shown in figure 2.5.

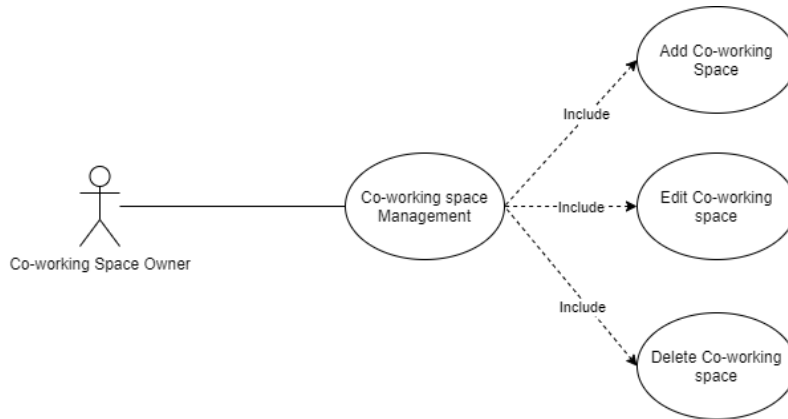


Figure 2.5. Use case diagram Level 2 - coworking space management

The use case description for coworking space management is shown in table 2.1.

Table 2.1. Use Case Description - coworking space Management

Use case ID	1
Use case Name	coworking space management
Description	Managing coworking space by adding, updating and deleting coworking spaces and it's details
Actors	coworking space owner
Pre-Condition	Space owner should be login as Space owner
Post-Condition	coworking space is created coworking space is updated coworking space is deleted
Related Use cases	Includes: Create coworking space Update coworking space Delete coworking space
Flow of Events	User login as coworking space administrator Navigate to Admin Page Add a new coworking space Update coworking space details Publish coworking space details
Alternative Flows	User doesn't publish the coworking space and navigates back to the admin page

Use case diagram for booking spaces is shown in figure 2.6.

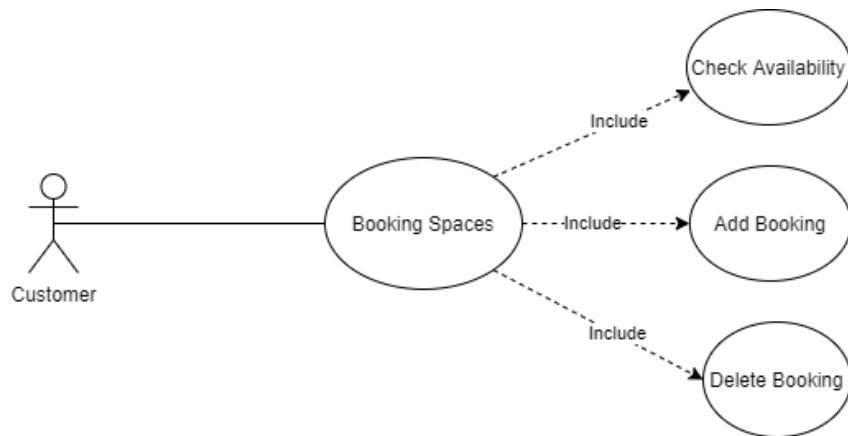


Figure 2.6. Use case diagram Level 2 - Booking Spaces

The use case description for booking spaces is shown in table 2.2.

Table 2.2. Use Case Description - Booking Spaces

Use case ID	2
Use case Name	Booking spaces
Description	Add a booking to a space
Actors	Customer
Pre-Condition	Customer should login as customer Booking slot should be available
Post-Condition	Booking is created in the space for the particular time slot
Related Use cases	Includes: Add Booking Delete Booking
Flow of Events	User login as Customer Navigate to Space of his choice Check whether time slot and space is available for the time he needs Update coworking space details Publish coworking space details
Alternative Flows	User doesn't click the confirm booking button and navigates back to the coworking space User cancels the booking as soon as the booking is made

Use case diagram for manage bookings is shown in figure 2.7.

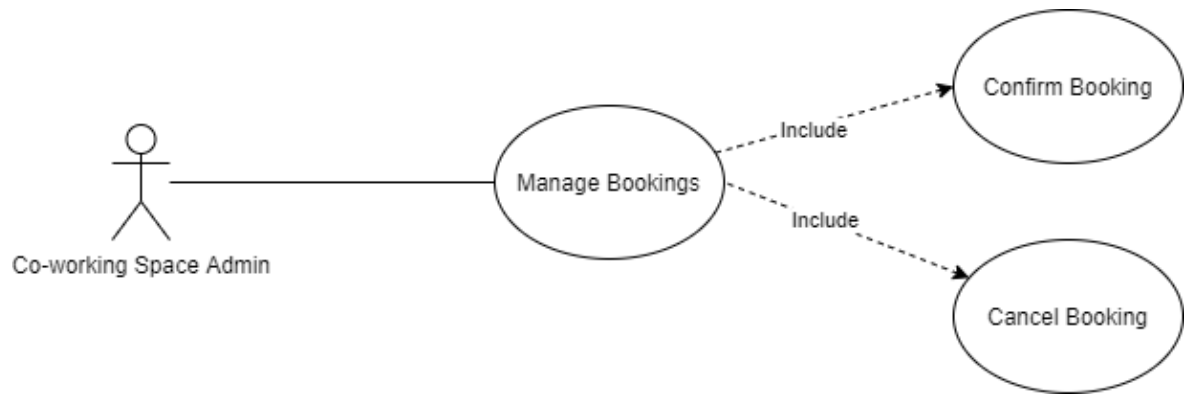


Figure 2.7. Use case diagram Level 2 - Manage Bookings

The use case description for manage bookings is shown in table 2.3.

Table 2.3. Use Case Description - Manage Bookings

Use case ID	3
Use case Name	Manage Bookings
Description	Admin should be able to manage the bookings made by the customer by approving and cancelling the booking
Actors	coworking space Admin
Pre-Condition	User should login as Admin Bookings that are pending should be available
Post-Condition	Booking is approved by the admin Booking is cancelled by the admin
Related Use cases	Includes: Confirm Booking Cancel Booking
Flow of Events	User login as coworking space admin Navigate to pending booking page Approve the pending bookings Cancel the pending bookings
Alternative Flows	User doesn't click the confirm booking or cancel booking button and navigates back to the admin page

2.5.2 Non Functional Requirements

Following are the non-functional requirements in the system:

- **Performance** – The system should respond quickly as soon as the user makes a request. In addition, the different screens should load quickly for the user. The front-page load time must be no more than 5 seconds for users that access the website using an LTE connection.
- **Availability** – The system should be available to the user on the internet 24x7. If there is a pre-planned down time or scheduled maintenance periods it needs to be informed early so that inconveniences will be minimized.
- **Scalability** - The system should be able to grow by serving more users, processing more data, and doing more transactions without negatively influencing on its performance. In order to do this, the system should be able to scale for a larger audience just by adding memory, servers, or disk space rather than making change in code in the future.
- **Security** – Users should be able to register to the system and should be able to login to the system using his/her username and password. Data validation should be present on all screens to prevent unauthorized access to user details. The database will be protected and encrypted by standard practices.
- **Maintainability** – The IT division will maintain system and they will be provided with documentation with instructions on how to use and maintain the system.
- **Usability** – User should be able to understand easily how to handle the system by searching the locations and make the booking. An assessment will be carried out to see the average time it takes to accomplish a booking, how many tasks a user can complete without any help, the number of transactions completed without errors, etc.

2.5.3 Software Requirements

2.5.3.1 Programming Language and Framework – Codeigniter PHP framework

When we did the research on what tools to use to build the site we came across PHP and JavaScript. We did a thorough research on both and decided to go with PHP mainly due to the support it provided to build e-commerce websites. JavaScript topped PHP in many aspects such as speed and extensibility but when considering factors like the learning curve and the support given to build e-commerce websites, PHP was much preferred option than JavaScript.

After choosing PHP, we did a thorough research on whether to go for core PHP or for a framework like CodeIgniter. During the research we found that, using a framework like CodeIgniter is more beneficial considering factors such as the user friendliness, security and the support to build large dynamic web applications. We even considered other PHP frameworks such as Laravel, which has some good features such as Built-in Database Query Builder, “Artisan” Command-Line Interface. But CodeIgniter’s lightweight and high speed along with the easier learning curve was more impressive.

2.5.3.2 Database – MYSQL

When we were looking for a Database, two options attracted us the most namely MySQL & PostgreSQL. PostgreSQL is an open source, feature-rich database that is perceived as a go-to solution for performing complicated, high-volume data operations [6]. PostgreSQL contains more features compared to other databases and is extensible due to its ability to define data types, index types, and functional languages.

While researching on both these databases, we understood that when building a system that needs to handle complex queries and massive databases, PostgreSQL is the preferred choice. However, if you are building a simpler database that is easy to set up and manage, fast and reliable then MySQL is the most preferred choice. Considering our system doesn’t have any complicated, high volume data we selected MySQL and the fact that it is part of the LAMP stack (an open-source suite of web applications that consists of Linux, Apache HTTP Server, MySQL, and PHP) made our choice even easier.

2.5.3.3 Other Development Software

The development is done using a system running Microsoft Windows 10 operating system. Visual Studio Code Version 1.41 is the code editor used for the development.

2.5.3.4 Client Software

Any of the web or Mobile browsers such as Microsoft Edge, Mozilla Firefox, Google Chrome, MAC Safari Browser, Apple Safari Browser, Android Chrome Browser

2.5.4 Hardware Requirements

- Development - Intel i5 or better processor, 7th generation or newer (Virtualization must be supported), 500 GB or larger SSD, Minimum 8 GB of RAM (12GB -16GB RAM recommended)
- Client - An Intel Pentium 4 processor or later that is SSE2 capable, 100MB of free hard drive space and 128MB of RAM.
- Server – Windows 2008 R2, Processor 2.0 GHz, 8 GB Ram & 2GB+ Free Disk Space

2.6 User Classes and Characteristics

There are two types of users that interact with the system, the customer and the coworking Space owner/ management. Each of these two types of users has different uses of the system so each of them has their own requirements.

The customers want to book coworking spaces and they can use the website to find a coworking space. This means that the user have to be able to search for coworking space, choose a coworking space from that search and then navigate to it. Furthermore, they should be able to select the timeslot and make the booking. Therefore, the customer needs basic knowledge of English language and computer literacy and ability to work in an internet environment.

The coworking Space owner/ Management can login as an admin to the website and access the admin portal where he will be able to add, edit and delete a coworking space, facilities and other information and view all the booking details. Therefore, the customer needs basic

knowledge of English language and computer literacy and ability to work in an internet environment.

2.7 Technical Constraints – Design and Implementation

- The information of all users, spaces and bookings must be stored in a database that is accessible by the website.
- The system should be available 24 hours a day.
- Users will access the system from any device that has a browser and an internet connection.
- Users must have their correct e-mails and passwords to login to their online accounts and do actions.
- The website should be responsive so that it will be visible clearly in all devices both desktop and mobile.
- Security of the system such as database security, database backup, database restore and effects of power failure is the responsibility of the IT team.

2.8 Assumptions and Dependencies

- The system is a web based application therefore there is a need for 24 hours internet connectivity.
- It is assumed that the user runs an operating system, which supports internet browsing.
- It is assumed that the user is familiar with an internet browser and familiar with handling the keyboard and mouse.
- The users should know the English language as the user interface will be provided in English language

Chapter 3: Methodology

The presence of a good design is an essential prerequisite for a successful system implementation. In this chapter, we have documented the design considerations that helped to plan, design and implement this system.

3.1 Choice of Development Methodology

Software development methodology is a series of processes used in managing a software development project. This usually addresses issues like choosing features for a version of the software, planning different tasks, tools, techniques, release dates, resources allocated, and planning of the tests. Practically different organizations implement their Software project management in different ways and it sometimes changes from project to project. A single methodology is not suitable for all types of circumstances [7]. In order to build our system, we studied the following methodologies to see which of them will be more suitable.

- Waterfall Model
- Scrum
- Extreme programming

3.1.1 Waterfall Model

The waterfall model is a linear, sequential approach to the software development life cycle (SDLC) where each phase depends on the deliverables of the previous one. This is ideal for projects that are well documented, contains requirements that are fixed and not ambiguous, adequate resources, a well-established timeline and surely-knew technology [8].

Figure 3.1 illustration is a representation of the different phases of the Waterfall Model.

Waterfall model

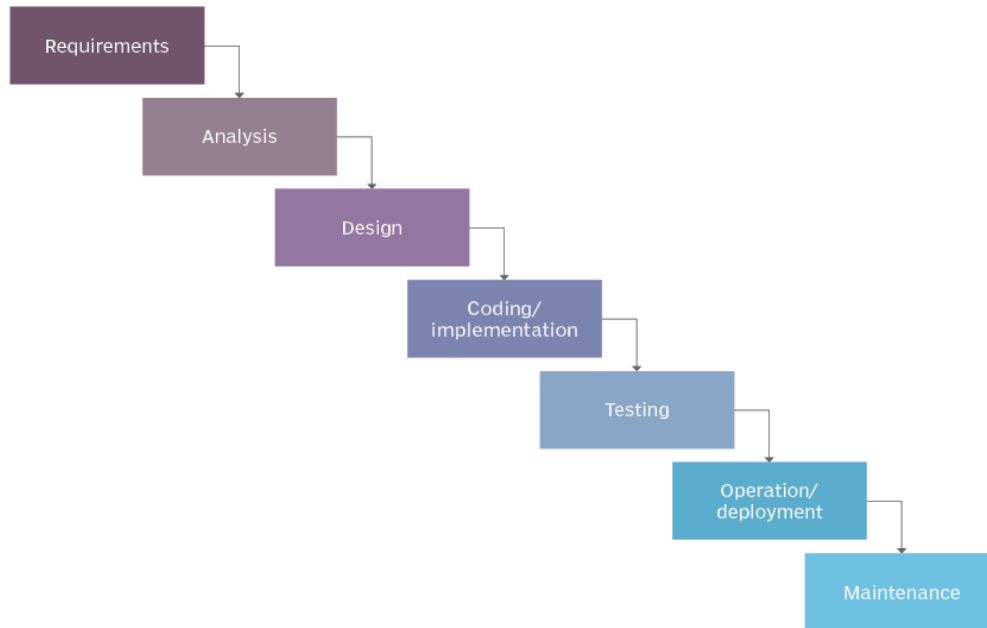


Figure 3.1. Waterfall Model

Why Waterfall methodology will not suit our project

- When the application is in the testing stage, the waterfall model doesn't permit a lot of reflection or revision. Therefore, it is extremely hard to return and change something that was not considered in the concept stage. There is a risk of client asking for changes during the project therefore waterfall will not suit our project.
- A working software is not produced until late during the life cycle. There are certain parts of this project where there is need to build, test, and have to consult the client for his opinion. In such a situation, waterfall model will not support the project.
- The waterfall model is not appropriate for projects where requirements are at a moderate to high risk of evolving. In our project, the client can change the requirement and request to introduce new improvements regularly. Therefore, this process model will not suit our project.

3.1.2 Scrum Methodology

Scrum is one of the most popular agile development methodology used in software development based on iterative and incremental processes. The primary objective of scrum is to ensure the customer's need in an environment of transparency in communication, collective accountability and continuous progress [9].

The product owner creates a product backlog after working with his team to identify and prioritize system features. The product backlog consists of a list of everything required to successfully deliver a working software, including features, bug fixes, non-functional requirements, etc. Once the product backlog is prioritized, cross-functional teams will deliver shippable increments of software in sprints. Once a sprint's product backlog is finalized, only team members can add functionality to it. After the completion of a sprint, the product backlog is analyzed and reordered, and the next set of features that are going to be delivered is selected for the next Sprint [10].

Figure 3.2 illustration is a representation of the different phases of the Scrum methodology.

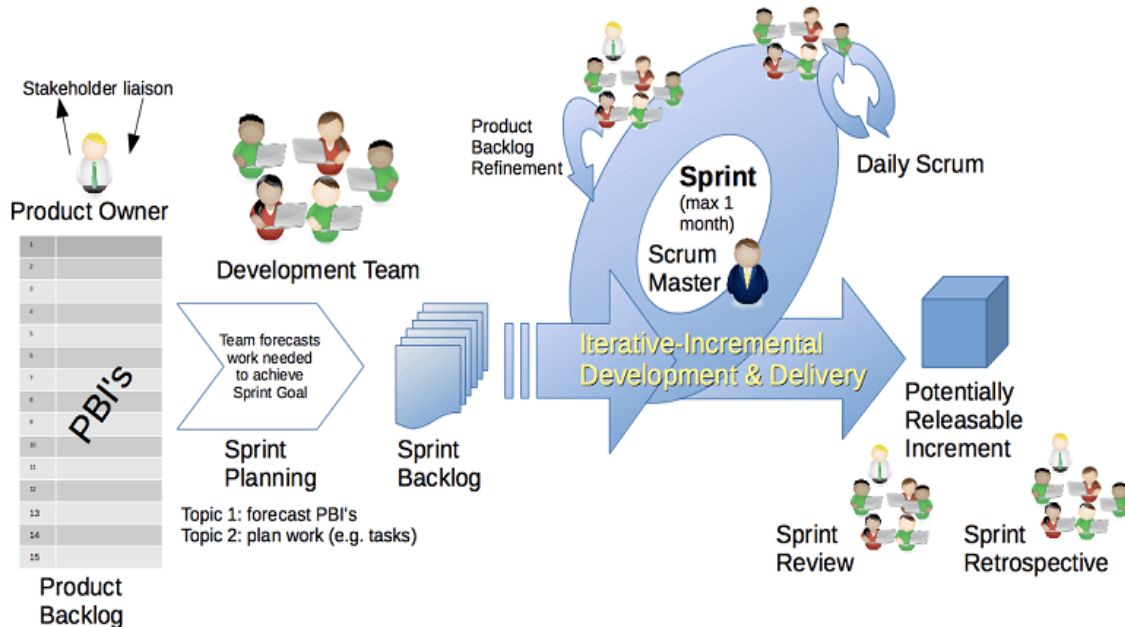


Figure 3.2. Scrum Methodology

Source: [11]

Why Scrum methodology will not suit the project

Scrum is a framework for a cross-functional team to work and collaborate under conditions of uncertainty and conflicting priorities. When working by yourself, there is no need to solve many of the problems that Scrum was invented to solve. Implementation of Scrum will result in too much time being spent in coordinating processes and therefore will out way larger benefits the framework offers. Therefore, most of it would be irrelevant or overkill.

3.1.3 Extreme Programming (XP)

Extreme Programming (XP) is a software development methodology whose primary intention is to improve software quality and responsiveness according to changing client requirements. High client involvement, rapid feedbacks, nonstop testing and constant planning are some of the principle features of extreme programming which assists to deliver working software usually every 1 to 3 weeks [10].

In XP, the customer is expected to drive the project by working closely with the developers especially in activities such as providing user stories and prioritizing them. The developers will then take those user stories according to priority make estimates, plan, develop, test and deliver them in iterations. Figure 3.3. illustration is a representation of the different phases of the Xtreme Programming methodology.

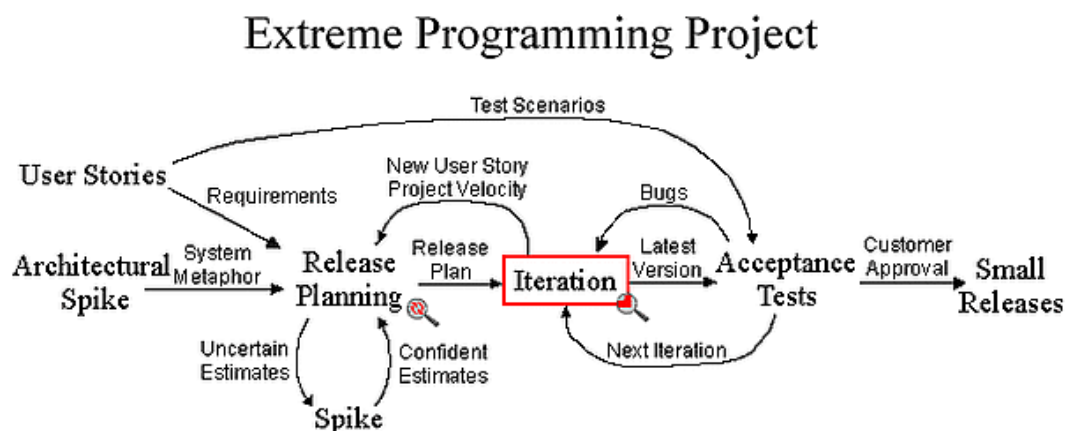


Figure 3.3. Extreme Programming Methodology

Source: [12]

There are circumstances where XP centers around having little teams, like for example, doing development in teams of two also known as pair programming or having another developer to do the code review [13]. There are situations where pair programming is impossible like in a situation where only one developer is involved in the project. In such situations, we can scale XP to come up with a methodology that utilizes its practices, yet in a structure where a single developer is involved just like in a traditional project.

A single developer using XP is not an unfamiliar idea. XP41, i.e. "XP for one" is the term used to describe it. In the following sections, we will look into how a single developer can use XP and why we chose it to build our system.

Extreme programming for one

XP was created in light of five core values: Communication, Simplicity, Feedback, Respect and Courage. From these five core values, twelve core practices were derived and below we will discuss how all of these 12 core practices can be applied for a project with a single developer [14].

- **The Planning Game:** At the point when you have a client, you can get together with him to choose what features of the system will be of the greatest value to the business.
- **Small Releases:** Releases can be made frequently and early when you are a lone programmer than with a team of developers
- **Metaphor:** Each project has a “system of names” and description which helps to guide the development process and communication between all parties. As a single developer you can use and refine whatever metaphor suits you best.
- **Simple Design:** When you are a single developer, it is anything but difficult to keep up a design which is simple. The best way to resolve this is to have a high-level design as a goal but develop an alternative less complex design that conveys all the functionality
- **Coding Standards:** When you are a single developer, how you decide to code is your coding standard

- **Testing:** When you are a single developer, test suites can be handily composed and used for what you have developed
- **40-Hour Week:** Hold on to a 40-hour week and stop for the day when you feel like, which means stop working when you believe you are not productive anymore or stressed
- **On-site Customer:** At the point when you have a customer or client then correspondence through email or telephone will deal with this issue as long as the customers are available to correspond.
- **Continuous Integration:** When there is no other person making any changes to your codebase and you are working alone, then there won't be any issues and integrations will not cause any conflicts.
- **Collective Code Ownership:** There will be no issues when you are the only developer and you will own all the source code.
- **Refactoring:** When you are a lone programmer you can and should work on refactoring as far as possible with the exception of in circumstances where you don't have consent to change code you don't own.

The only core practice that is hard to implement as a single developer is the following

- **Pair Programming:** As a lone programmer, advantages of pair programming are lost but you could still ask a cohort or a friend to give you reminders regularly. In these situations, informal walkthroughs and test first philosophy can become handy.

3.1.4 Selection of Appropriate Methodology

For this project, Extreme Programming for One development methodology was chosen after considering all the above-mentioned methodologies due to the following reasons.

- Due to the close contact with the client, extreme programming for one reduces the risks related to programming or related to project failure by ensuring that the client gets exactly what he wants.

- Extreme Programming for one is a customized methodology for a single developer and considering that this project will be implemented by a single developer all the guidelines are practically implementable.
- It can manage a series of common pitfalls, slips in schedules, delaying of projects, high defect rates and broken builds by applying its principles, which can come very handy when managing a project like ours.
- Constant feedback from the customers is encouraged through this methodology compared to others. Demonstrating the software early and often, listening carefully to customer feedback and making any changes needed helps the project to move in the right direction.
- Compared to other Methodologies continuous testing is encouraged more in this methodology, which helps to maintain a stable software.
- Compared to other methodologies extreme programming for one gives priority to simplicity, which helps to create extremely simple code that can be improved whenever it is needed.

3.2 Selection of Modelling Language

Unified Modelling Language (UML) was selected as the modelling language for the design of the solution. UML is an industrially well-recognized model language, which is used to design the behavior and structure of a system. Following are some of the reasons why we selected UML as our modelling language.

- UML can be utilized to model any type of application, running on any type and combination of hardware, operating system, programming language, and network, in UML [15]
- Reusable code is easily identified and coded with increased efficiency. Due to this, there will be lower amount of effort and lower development costs [16]
- It can specify the functionalities of the system in an implementation independent manner
- ‘Holes’ in the logic can be easily identified in the design drawings

The flow of our system using different types of UML diagrams will be represented below.

3.2.1 Activity Diagram

An Activity diagram is a flowchart to represent the flow of the system from one operation of the system to another [17]. In the following activity diagrams, we will illustrate the flow of the system for the coworking space admin and the customer.

Activity diagram for coworking space admin is shown in Figure 3.4.

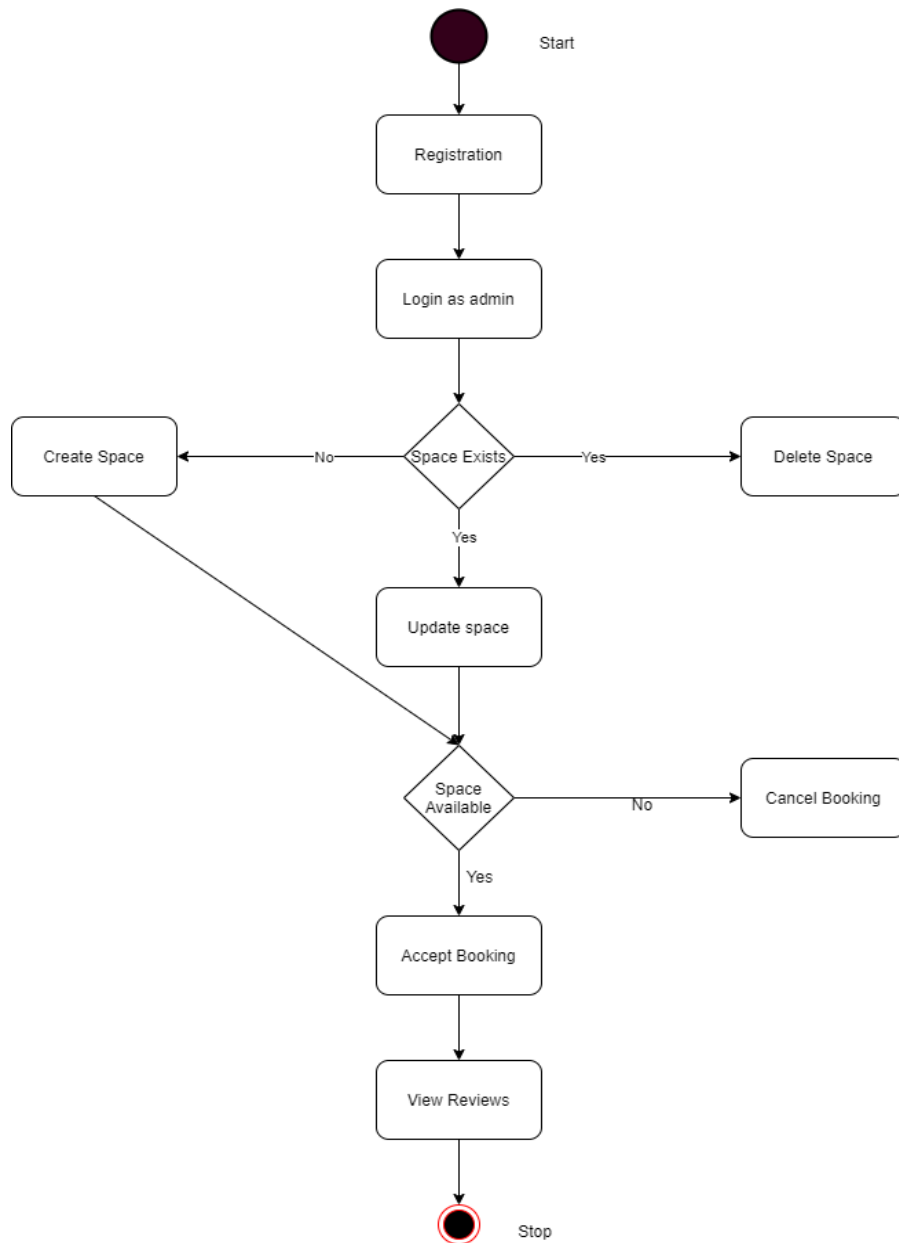


Figure 3.4. Activity Diagram – coworking Space Admin

Activity diagram for customer is shown in Figure 3.5.

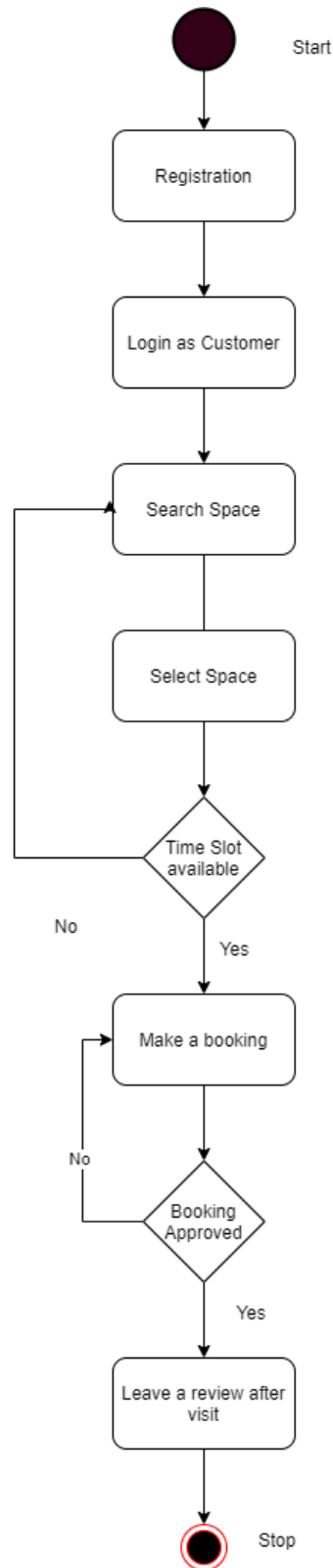


Figure 3.5. Activity Diagram – Customer

3.3 User Interface Design

The objective of a good UI should be to make client and customer's interactions straightforward, instinctive and efficient. A website like ours should have an appealing design to increase the user experience that can affect the performance of the sales [18]. Furthermore, a site like ours should be accessible anytime, anywhere therefore priority should also be given to access the website from any device whether it is web or mobile.

Considering all the above factors, we decided to design a mobile friendly website with a responsive design for the Customers so that they can browse and make bookings from anywhere using their web or mobile device browsers. In order to capture the attention and elevate the user experience of the customers we have also adapted the following UI design elements into our website:

- Easy to understand navigation – A simple menu structure can provide positive customer experience and increased conversion rates
- Attractive call to action buttons – Attractive call to action buttons such as signup and book now buttons can trigger customers to take instant action which provides a massive impact on conversion rates.
- Search bar that is well positioned – A good search bar that is in an ideal position can lead customers to get what they expect quickly and if the results they obtain is what they were looking for, then the sales will increase for our clients.
- Straightforward booking and checkout process – Many online customers prefer a booking and checkout process with minimum steps therefore we should reduce the amount of steps and amount of information gathered in the entire process which can lead customers to visit our site again.

We have implemented all the above-mentioned features in our customer website. Please refer “Appendix A: User Manual” for the UI designs and the user flow of our website.

3.4 Implementation

In this phase, we will discuss on how our system is developed which includes details of the tools and techniques used to implement this system such as the frameworks, programming languages, scripting languages, database systems etc. Furthermore, we will also discuss about the installation of the system, technical support and user training.

3.4.1 Implementation Languages and Tools

When selecting the implementation tools, it is essential to check whether the tools we use will match the purpose of our system. Especially for a system like ours, we should pick tools to build a site, which is intuitive, secure, fast and easy to use. After a detailed research these are the tools we went on to select to build our system.

3.4.1.1 Codeigniter Framework

Codeigniter is a robust open-source PHP driven framework built for developers who need a straightforward and exquisite toolkit to develop multi-featured web applications [19]. Codeigniter follows Model-View-Controller (MVC) design pattern, which separates the application logic from the presentation logic and business logic. This helps the webpages to have minimal scripting, as the presentation is discrete from the PHP scripting.

Figure 3.6. below depicts how the data flow in the MVC framework works in CodeIgniter.

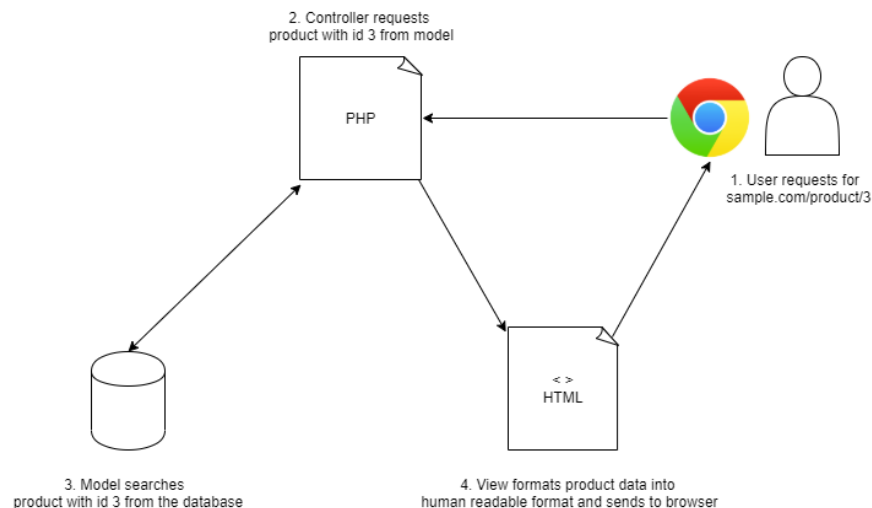


Figure 3.6. MVC in CodeIgniter

Source: [20]

CodeIgniter has the following features:

- Codeigniter is free to use as it is licensed under Massachusetts Institute of Technology
- Codeigniter core system requires only a small library therefore it is incredibly light weighted and different libraries can be included upon request based on your needs
- Codeigniter generates URLs that are SEO friendly and clean
- Codeigniter contains many inbuilt security features to secure the web applications such as CSRF protection, Password handling, XSS Filtering and Input data validation [21]
- It is simple and easy to use as it provides an easy interface, this also makes Codeigniter easy to expand and productive

3.4.1.2 MySQL Database & phpMyAdmin

MySQL is an open-source relational database management system, which is capable of working on many system platforms. It is capable of increasing performance and durability by partitioning the tables and replicating data [22]. In order to use MySQL knowing standard SQL commands is sufficient and there is no need to learn new syntax to work on it. The memory leakage is very low in this database therefore the efficiency of the memory is high. It also allows the developers high productivity by using Triggers, Stored procedures, and views [23].

In order to handle the administration of MySQL over the web we will be using the phpMyAdmin tool. PhpMyAdmin is capable of importing and exporting data in various formats such as CSV, SQL, XML and is capable of administering multiple servers. It is also capable to support almost all of the MySQL features.

3.4.1.3 Bootstrap

Bootstrap is an open-source front-end-framework consisting of HTML, CSS and JS tools for creating responsive. Bootstrap is easily customizable and it's mobile-first approach helps to build websites that is responsive for all types of device browsers. As it comes bundled with many components such as menus, alert boxes and JavaScript plugins it makes the job of developing the front-end of your system attractive and efficient. Bootstrap is a very supportive tool to build your front end when you are developing your website using CodeIgniter framework.

3.4.1.4 Development Tools

Along with the above-mentioned tools, a text editor to write the code in is needed. Any text editor, including the Windows Notepad, would in theory do, but one that is a bit more targeted towards programming is certainly an advantage. For this project, we will be using Visual Studio code due to its features such as intellisense, which helps for code completion, syntax highlighting and bracket matching which can become very effective in PHP development.

3.4.2 Implementation Strategies

In software engineering, it is essential to maintain a specific style on the programming and follow various "good practices". Maintaining good practices helps to reduce the number of errors in the code as well as help to identify the errors in the code easily. These practices will improve the readability of the code and helps to maintain the code easily. Following are some of the coding standards that were maintained by us in this project.

The naming of classes, methods, constants, variables is one significant part of maintaining a coding standard and considering how PHP is a case-sensitive programming language it is essential we follow this. Class files should always start with an uppercase letter and should match the name of the class itself. Any other file name other than class files such as configurations, views, generic scripts, etc. should be in all lowercase.

Class names ought to consistently start with a capitalized letter. If the class names consists of multiple words, they need to be separated with an underscore, and not camel cased. Class methods should be altogether lowercased and named to explain their function, ideally by including a verb. Here too multiple words should be separated with an underscore.

The naming of variables are similar to the naming of class methods where they contain only lowercase letters, separated by underscore separators. Constants contain only uppercase letters and underscore separators separate multiple words [24].

White spaces were inserted at certain places in the code to make it more open which will improve readability. In order to get understandable and maintainable code, comments were used which gives a total view of what a section of code does. This can become handy when a new developer joins the team and needs to understand what happens in each code section [25].

3.4.3 Installation Procedure

3.4.3.1 Server Installation

The system is developed using Codeigniter framework therefore it can be installed in a local server or a web server or even in a cloud server depending on the client's request. Many shared hosting services even have dedicated Codeigniter hosting plans which makes the server installation process easy. MySQL database can also be hosted in web servers and controlled through phpMyAdmin. Since our client had his own hosting server, we installed the system in that server itself.

3.4.3.2 Client Installation

Day by day, more and more devices come with varying screen resolutions, definitions and orientations. With the popularity of the android, iPhone, iPad and advanced smartphones, many new devices are able to switch from portrait to landscape at the user's whim. Our system is designed in a way to support all these designs and orientations.

The customers' site is mobile-responsive; therefore, customers can make their bookings from wherever they want on the go, using web browsers or using mobile browsers in their phones. All they have to do is take their device, go to the browser, enter the URL and access our system. Admin users can access the admin site using a web browser. Users can use any web browser to access these sites. Example: Microsoft Edge, Safari for Mac, Mozilla Firefox, Google Chrome etc.

Chapter 4: Evaluation

After the completion of the system, Testing and Evaluation were carried out with customers and coworking space providers. The aim of conducting testing and evaluation was to determine the quality and accuracy of the website, how users behave on different webpages and identify areas of further development. In this chapter, we have presented the testing plan, the test results and evaluation of the results.

4.1 Test Plan

Test Plan is a comprehensive document that describes the test strategy, objectives, schedule, estimation and deliverables and resources required for testing. It helps to understand the effort required to validate the quality of a software and functions as a blueprint for the testing process [26].

First, we need to plan & identify the different types of testing methods available to understand the strengths and weaknesses of the system. White box testing and Black box testing are two main testing methods used in software testing.

4.1.1 Black Box Testing

Black box testing is a testing technique in which functionality of the system is tested without peering at the internal code structure, design and implementation details. This type of testing depends completely on system requirements and specifications, and primary focus is given on inputs and output of the software system [27]. Black box testing was conducted in our system by both the developers and the end users.

4.1.2 White Box Testing

White Box testing is a testing technique in which software's internal structure and design is tested through the derivation of test data from the program logic. The code is visible to the tester in this type of testing. Primary focus of this testing is verifying the flow of inputs and outputs through the application, improving design and usability and strengthening security [28]. In white-box testing the tester chooses inputs to exercise paths through the code and

determine the appropriate outputs. As the testing is done with knowledge of the internal structure of the program only developers will be involved in this type of testing.

Furthermore, the following types of testing were conducted in the system.

4.1.3 Unit Testing

Unit testing is a software testing method where smallest testable parts of a software known as units or components are tested to determine whether each unit of the software code performs as expected. The developers do this testing after each component is developed.

4.1.4 Integration Testing

Individual units and components that were unit tested have to be integrated to guarantee they work well when integrated. Now and again individual components and code fragments may work fine however when they are integrated with different modules, it might give bugs. The motivation behind integration testing is to recognize such situations with integration.

4.1.5 System Testing

After successfully completing the unit testing and integration testing, the entire system has to be tested in an environment that is similar to the future production environment. System testing aims at identifying problems that emerge in real world working environment.

4.1.6 User Acceptance Testing

The system should be tested by the end users in a real environment and should verify whether the intended functionalities specified in specifications were been implemented. The client, end users and Industry experts, will conduct this testing. Acceptance from the clients and other users indicates the system has met its objectives.

4.1.7 Testing Strategies

Listed below are some of the testing strategies adopted for the testing process.

4.1.7.1 Prioritize the tests

Before beginning the testing process, we prioritized the type of tests that should be done according to the importance as we have a limited time. We decided that developers will conduct unit testing after each component is developed and when multiple components are integrated developers will conduct an integration test. After the development of the system is complete priority will be given to the system test that will be conducted by developers so that we will be able to fix any major bugs before conducting the user testing.

After the developers complete the system test, end users, which includes our clients from Pragmatic Labs will be allowed to test the system. The bugs identified by them will be recorded and also other observations such as user behavior in different functionalities will also be recorded. Based on the observations and time available the system will be given to industry experts to be tested and their thoughts and observations will be recorded. Furthermore, discussions will be conducted with the experts on how to improve the current system based on observations made through user testing.

4.1.7.2 Create Test cases

Test case is a set of actions executed to verify a particular feature or functionality of your system, which usually contains test steps, test data, precondition, and post condition developed to verify any requirement. Test cases was crucial for both the system testing done by developers as well as User acceptance testing in order to keep track what tests were done and instruct the users on the areas to be tested. Please refer “Appendix E: Test case document” for all the test cases used to test the system.

4.1.7.3 Create Test reports

Test report is a summary of test activities and final test results, which assesses how well the testing is performed and provides information about the tests we have conducted in order to show the customers whether the system is ready to go live. If the test report informs that there are many defects remaining in the product, the stakeholder can delay the release until all the defects are fixed.

4.1.8 Exit Criteria

An exit criterion decides the completion or termination of the testing task. These are the criteria/requirements, which must be met to complete a test phase. By having clear exit criteria, the evaluation of potential solutions becomes far less subjective.

Following are list of exit criteria decided for our project.

- The regression test suite is executed (all test cases).
- A complete test round should be conducted on top of the production data especially for ongoing projects.
- All critical priority (Blockers/ Highest/ High) issues have resolved and completed.
- 95% of the medium bugs have been resolved.
- The application should be tested with slow connectivity/ no connectivity.

4.2 Test Results

After the development process was completed, the test cases were executed in order to assess how many of the test cases passed and whether the system was stable. Table 4.1. shows the test report of the tests we conducted, which contains a summary of the areas we tested and test results we obtained after the test cases were executed.

Table 4.1. Test Report

Test Report			
Test Cycle	System Test		
Executed	Passed	100	
	Failed	0	
	Total Test Cases Executed		100
Pending			0
In Progress			0
Blocked			0

Test Planned (Executed + Pending + In Progress + Blocked)						130
Functions	Description	% TCs Executed	%TCs Passed	TCs Pending	Priority	Remarks
Signup as coworking space owner	Verify new coworking space owner is created	100	100	0	High	
Signup as user	Verify new customer user is created	100	100	0	High	
Login as coworking space owner	Verify user can login as coworking space owner	100	100	0	High	
Login as customer	Verify user can login as customer	100	100	0	High	
Create coworking Space	Verify coworking space owner can create a space	100	100	0	High	
Edit coworking Space	Verify coworking space owner can edit a space	100	100	0	High	
Search coworking Space	Verify customer can search for a space	100	100	0	High	
View coworking Spaces	Verify customer can view spaces	100	100	0	High	
Make a Booking	Verify customer can make a booking	100	100	0	High	
Delete a Booking	Verify customer can delete a booking	100	100	0	High	
Accept a Booking	Verify customer can accept a booking	100	100	0	High	
Decline a Booking	Verify customer can decline a booking	100	100	0	High	

With the Test case pass rate being 100%, we were confident enough to release it to our client for User Testing and obtain feedback from them.

4.3 Evaluation

The primary goal of the evaluation process is to decide whether it accomplishes the ideas presented in the initial overview, solves the problems that were identified and assesses the quality of the system. Furthermore, the evaluation process will also assess the software development methodology that was used throughout the development of the framework, the usefulness of the technologies and tools used and accuracy of the estimations and the usefulness of the client reviews.

Since we used Xtreme Programming (XP) methodology as the Development methodology, the evaluation was done after each component was delivered. After each component was delivered, a review meeting was held with the client and feedback was obtained which was used for corrective maintenance and change management. Furthermore, after the product was fully implemented feedback was obtained from the clients as well as real world users.

The feedback was obtained using a software evaluation form with a prepared list of criteria where a measurement against a Likert scale was used to quantify the feedback. The feedback process was done anonymously among the members of the clients and real world users in order to get genuine reviews. Table 4.2 lists the options and weights provided in the Likert scale.

Table 4.2. Likert scale options and weights

Likert Option	Value
Very Poor	1
Poor	2
Satisfactory	3
Good	4
Excellent	5

The evaluation form used to evaluate the system is displayed in Table 4.3.

Table 4.3. Software Evaluation Form

Online coworking Space Booking System						
Software Evaluation Form						
Parameters	Ratings					Remarks
	1	2	3	4	5	
1. Appearance						
User interfaces are attractive						
The color codes and combinations were suitable						
The font sizes & styles were appropriate & readable						
2. Usability						
Navigation was easy between the screens						
Menus and options were easy to understand						
Data validation was helpful						
The requested output results are successful						
3. Functionality						
Registration & Login Function						
Create, Update & Delete Spaces Function						
Search Spaces Function						
Display Space details function						
Booking Spaces Function						
Manage Booking Function						
Overall Flow of the system						
4. Performance						
Page load time for different pages						
Response time for all the requests made						

4.4 Evaluation Results

4.4.1 Evaluation Result for Appearance

The results obtained from the evaluation of appearance are displayed in Table 4.4 & Figure 4.1.

Table 4.4. Evaluation Result for Appearance

Likert Option	Result	Percentage %
Excellent	3	15
Good	9	45
Satisfactory	5	25
Poor	3	15
Very poor	0	0

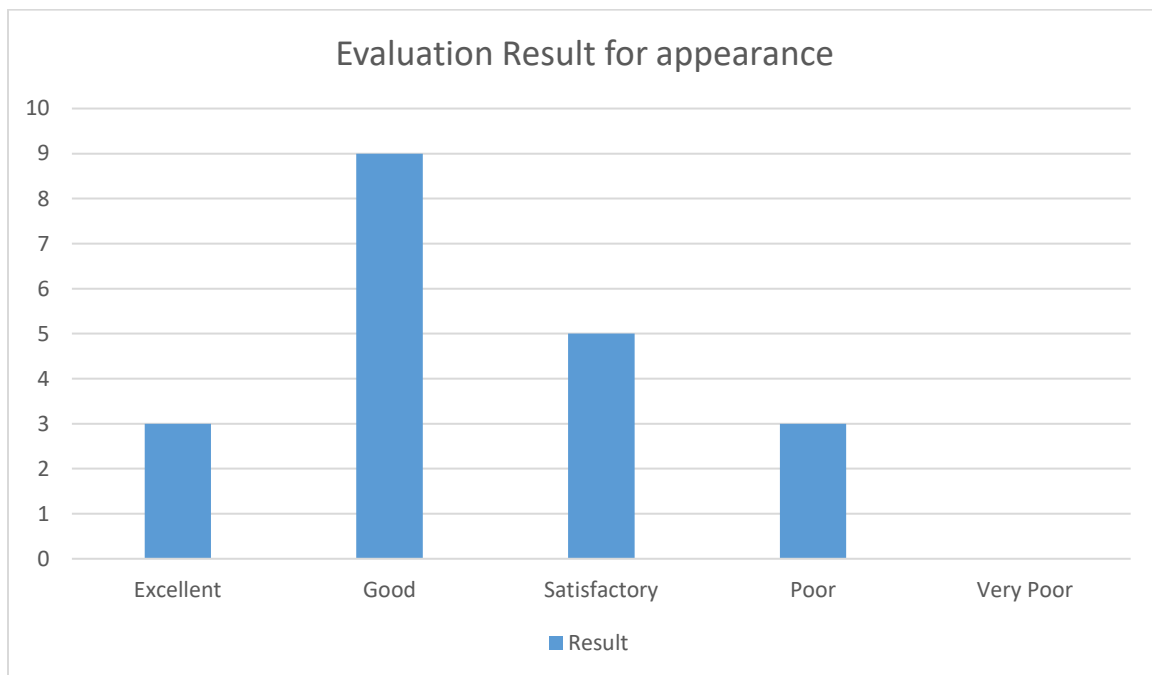


Figure 4.1. Graphical Representation of Evaluation Result for appearance

According to the results obtained, it is clear that the User Interfaces are attractive and users are happy with the appearance of the website. 45% of the users have rated the appearance as good and 15% of the users have rated the appearance as excellent.

4.4.2 Evaluation Result for Functionality

The results obtained from the evaluation of functionality are displayed in Table 4.5 & Figure 4.2.

Table 4.5. Evaluation Result for Functionality

Likert Option	Result	Percentage %
Excellent	5	25
Good	10	50
Satisfactory	4	20
Poor	1	5
Very poor	0	0

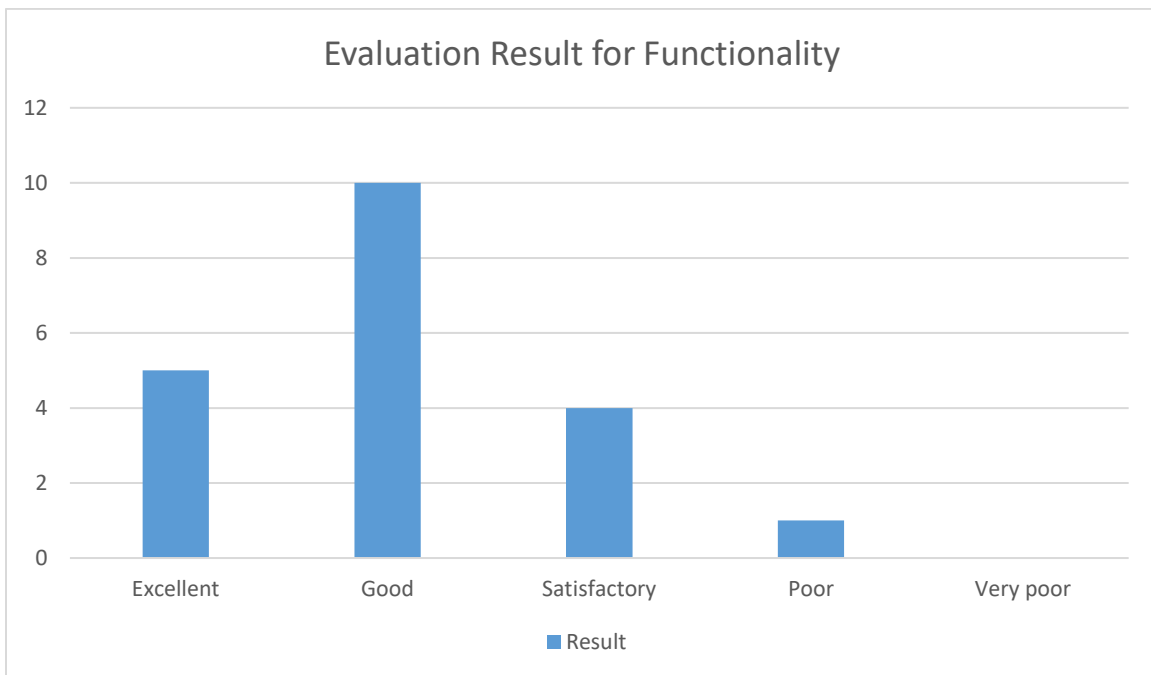


Figure 4.2. Graphical Representation of Evaluation Result for Functionality

According to the results obtained, it is clear that the functionalities of the system are effective and users are satisfied with the website and its functionalities. 50% of the users feel the functionality is good and 25% of the users feel the functionality is excellent.

4.4.3 Evaluation Result for Usability

The results obtained from the evaluation of usability are displayed in Table 4.6 & Figure 4.3.

Table 4.6. Evaluation Result for Usability

Likert Option	Result	Percentage %
Excellent	1	5
Good	6	30
Satisfactory	6	30
Poor	5	25
Very poor	2	10

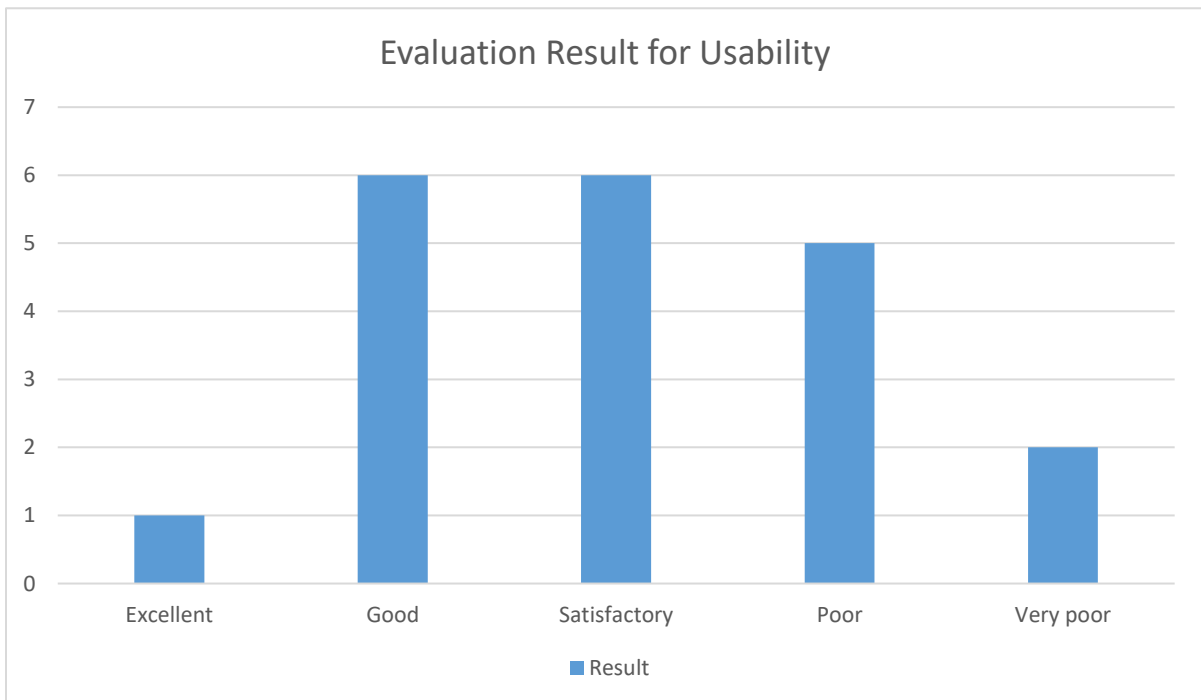


Figure 4.3. Graphical Representation of Evaluation Result for Usability

The feedback for the usability is decent with 30% of users finding it good and another 30% finding it satisfactory and 5% finding it excellent. But we will be working more to improve this aspect in the future to make it more user friendly for the user.

4.4.4 Evaluation Result for Performance

The results obtained from the evaluation of performance are displayed in Table 4.7 & Figure 4.4.

Table 4.7. Evaluation Result for Performance

Likert Option	Result	Percentage %
Excellent	1	5
Good	7	35
Satisfactory	4	20
Poor	6	30
Very poor	2	10

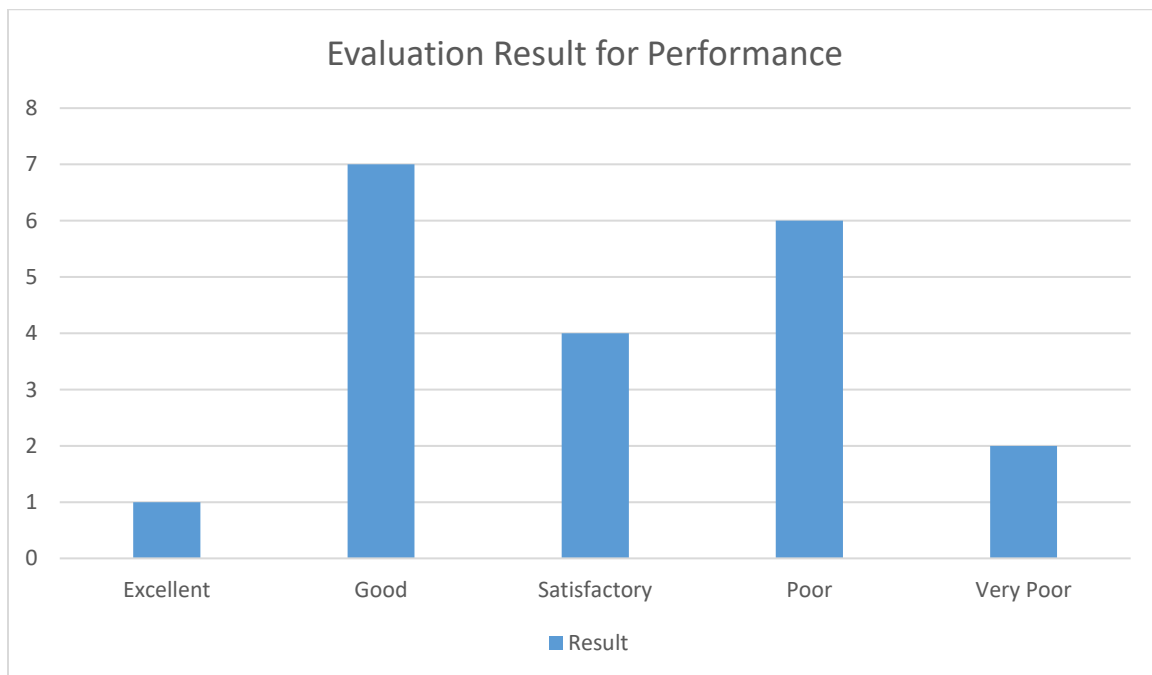


Figure 4.4. Graphical Representation of Evaluation Result for Performance

Performance is an area, which requires improvement as it generated mixed reviews from the users. Considering that 40% of the users rated the performance as poor or very poor, indicates

that there is need for improvement. However, it should be noted that considering that this is the first live release of the system, 60% of the users rating the performance as satisfactory and above indicates that the proposed system's performance is better than the existing system.

4.5 Overall Evaluation

The objective of this system was to build a website where coworking space owners especially ones who use a manual system can promote and sell their spaces while coworking space users can search for a space that will be convenient for them and book it online with ease. The objective has been achieved through the built system and anyone who wants to promote their coworking space or wants to book a coworking space can easily fulfill their needs through the website.

When comparing our site with some of the similar systems around the world it has most of the major functionalities found in them and it has adapted itself to match the requirements of the users of Sri Lanka. Even though when it comes to factors like performance, usability is not equal to the International standards, it is a good start for a product like this system in the Sri Lankan market and can be improved further in the future to match the International standards of similar websites.

When the study for this system was conducted, some of the Sri Lankan users of coworking spaces were interviewed and it was found out that some of the requirements expected among the Sri Lankan customers were different from the users abroad. One such factor was value for money. Most international users and international websites emphasize and advertise about the price of the space but when it comes to Sri Lankan users, significant focus is placed on amenities they receive with the coworking space and the charges for each of them. We understood this through interviews where some interviewees expressed how two different spaces charge the same amount for the spaces but the price for amenities like printouts are significantly different compared to the other. This is one of the many observations we made and hence we implemented the feature where the coworking space owners can display the prices of the amenities they provide for the users' ease. This option may sound simple but this was one of the most appreciated features by many of users. This also made us understand the importance of doing a detailed study with real life users during the planning phase.

We used the Xtreme Programming for one methodology as the development methodology of the study and using this methodology helped significantly to develop this system efficiently especially with frequent client feedback. There were instances where some of the features implemented were different from what the client expected but the client feedback in between releases helped to rectify the misunderstanding and helped build a system we all agreed upon. It was further decided that in future we would be using this methodology for any similar projects due to its efficiency.

When it comes to the tools used, Codeigniter was a good choice for the development and many features of it made the development process easier. There were some complaints regarding the performance not being equal to international standards but we think this can be improved in the future by incorporating latest technologies with Codeigniter and the current system is a good start for it.

We were overwhelmed by the response we got regarding the user interface of our website and how it attracted our users. However, the usability of the site can be improved further based on the ongoing feedback from users. Overall, we believe the system was a success and a few tweaks in the future can help us reach the International standard.

Chapter 5: Conclusion

Pragmatic Labs and other coworking space providers are now able to manage their coworking space bookings with ease using our system and have received many new customers after implementing our system. We have received several positive comments from both coworking space providers and coworking space users with regard to the ease of making bookings and convenience of exploring new coworking spaces, as highlighted in Chapter 4.

The evaluation of the test results generated the following conclusions about the finer details of the website.

- The appearance of the website can be concluded as attractive since 60% of the users rated the attractiveness as good or excellent (45% of the users have rated the appearance as good and 15% of the users have rated the appearance as Excellent).
- The functionalities of the system can be concluded as effective and efficient since 75% of the users rated the functionality as good or excellent (50% of the users have rated the functionality as good and 25% of the users have rated the functionality as Excellent).
- The usability aspect of the system can be concluded as satisfactory indicating room for further improvement. Only 30% of users rated the usability as good while another 30% rated the usability as satisfactory. Only 5% rated the usability as Excellent.
- Performance is an area which requires further improvement as 40% of the users rated performance as poor or very poor. However, considering it is the first release, we can conclude that performance is at an acceptable level for an initial release as majority of the users (60% of the users) have rated in the range of satisfactory to excellent.

Overall, based on the test results and the evaluation, it can be concluded that the system is in a very satisfactory level and clients are positive to go live with the website. This indicates that primary objective of the study has been successfully achieved.

5.1 Problems encountered and lessons learnt

Following were some of the problems encountered during this project.

- There were no standard set of data collected by different coworking space companies, and the types of data collected by each company differs.
- Getting hold of coworking space providers for the research was a tough task and some providers were not supportive with the idea of marketing their spaces in a platform which enabled easy comparison with other competitors.
- Many coworking space providers were not comfortable in providing data about their customers' behavior.
- Adapting to the MVC architecture of CodeIgniter was challenging. Hence, the time taken to study the framework was longer than expected. This caused considerable delay in completing the project within the agreed timeline.

Following were some of the lessons learnt during this project.

- Many of the problems were solved easily when we conducted interviews and meetings with the industry experts. One such meeting was fruitful for us, as it helped to create a standard set of data to be collected from the customer and avoid collection of unwanted data.
- Negotiations with the clients were required to build their trust with regard to handling of data and to convince them to be on board with the project by highlighting the long term benefits from the system especially in terms of awareness created, as most providers were concerned about the increased competition. .
- Build good rapport with the clients, especially through regular communication and by sharing the progress regularly with the client through workable prototypes.
- Online tutorials are a good way to learn new technologies and it is also important to follow the articles & speeches by experienced developers of that technology to understand the depth of the technologies and how to implement them.

The most important lesson we learned from this project was the significance of planning and implementing the project and adapting to changes in the timeline. Throughout the project, there were many deviations from the agreed timeline due to the customer feedback.

It is advisable to develop project timelines with extra time to adapt to these feedback and changes.

5.2 Future Enhancements

With regard to the future enhancements, significant priority will be placed to improve the performance and usability of the website according to the feedback received. To improve the performance in future, we are looking forward to use caching techniques such as Memcache, which is useful for reducing database load, along with few other planned performance improvement techniques. In order to improve the Usability, we will be consulting some UI & UX experts in the industry on how to improve the flow and the outlook of our website.

In addition, the online payment feature is not yet implemented in this website. Therefore, focus will be placed to introduce a payment platform to accommodate users to make the payments through Credit & Debit cards with ease. Emphasis will be placed to incorporate payments through mobile payment apps especially Sri Lankan payment gateways such as Frimi, Flash and International Digital wallet apps such as PayPal, Venmo etc. considering the target crowd.

Following are some other enhancements and features to be incorporated in the future.

- E-mail alerts to coworking space admin whenever a booking is received for their coworking space.
- SMS alerts to coworking space user whenever a booking is confirmed.
- Online payment for amenities such as printers, coffee etc.
- Integration of bookings made to online calendars such as Google calendar.
- Multiple account management where a single user can manage multiple coworking spaces.

We believe these changes can be done in sprints and can be released to live in the future, making our site on par to international coworking websites.

References

- [1] Hub9, January 2020. [Online]. Available: <http://www.hub9.space/>.
- [2] LOFT1024, "Loft One Zero Two Four," LOFT1024, 2019. [Online]. Available: <https://www.loft1024.com/>. [Accessed 14 November 2020].
- [3] Sharedesk, "Sharedesk.net," Sharedesk, [Online]. Available: <https://www.sharedesk.net/>. [Accessed 24 January 2020].
- [4] MillionSpaces, "Millionspaces," Auxenta, [Online]. Available: <https://www.millionspaces.com/#/home>. [Accessed 24 January 2020].
- [5] Booking.com, "Booking.com," Booking.com, [Online]. Available: [booking.com/content/about.en-gb.html](https://www.booking.com/content/about.en-gb.html). [Accessed 24 January 2020].
- [6] M. Smallcombe, "PostgreSQL vs. MySQL: Which One Is Better for Your Use Case?," xplenty, 15 June 2019. [Online]. Available: <https://www.xplenty.com/blog/postgresql-vs-mysql-which-one-is-better-for-your-use-case/>. [Accessed 07 May 2020].
- [7] D. Young, "Software Development Methodologies," *White Paper*, 2013.
- [8] Techtarget, "waterfall model," Techtarget, February 2019. [Online]. Available: <https://searchsoftwarequality.techtarget.com/definition/waterfall-model>. [Accessed 27 February 2020].
- [9] ScrumStudy, A Guide to the SCRUM BODY OF KNOWLEDGE, Arizona: SCRUMstudy, 2016.
- [10] Blueprint, "Agile Methodologies," Blueprint, 2020. [Online]. Available: <https://www.blueprintsys.com/agile-development-101/agile-methodologies>. [Accessed 27 February 2020].
- [11] Wikimedia Commons contributors, "Scrum Framework," Wikimedia Commons, the free media repository., 7 December 2019. [Online]. Available: https://commons.wikimedia.org/w/index.php?title=File:Scrum_Framework.png&oldid=379333554. [Accessed 12 June 2020].
- [12] M. G. F. M. N. W. T. M. a. D. B. Rittenbruch, "Extreme Participation - Moving Extreme Programming Towards Participatory Design," in *Proceedings of the Seventh Biennial Participatory Design Conference*, Queensland, 2002.
- [13] J. R., "What is eXtreme Programming?," 2000. [Online]. Available: https://ronjeffries.com/xprog/what_is_xp.htm. [Accessed 28 February 2020].
- [14] R. Agarwal and D. Umphress, "Extreme Programming for a Single Person Team," in *46th Annual Southeast Regional Conference*, Auburn, Alabama, 2008.

- [15] Object Management Group, "WHAT IS UML," Object Management Group, 2005. [Online]. Available: <https://www.uml.org/what-is-uml.htm>. [Accessed 19 June 2020].
- [16] SPEC India, "What are the benefits of UML?," SPEC India, 17 07 2008. [Online]. Available: <https://www.spec-india.com/blog/what-are-the-benefits-of-uml>. [Accessed 25 February 2020].
- [17] TutorialsPoint, "UML - Activity Diagrams," TutorialsPoint, 2020. [Online]. Available: https://www.tutorialspoint.com/uml/uml_activity_diagram.htm. [Accessed 8 May 2020].
- [18] K. Corrigan, "User Interface (UI)," Oberlo, 31 August 2019. [Online]. Available: <https://www.oberlo.com/ecommerce-wiki/user-interface-ui>. [Accessed 19 June 2020].
- [19] EllisLab, "Codeigniter," EllisLab, 2020. [Online]. Available: <https://codeigniter.com/>. [Accessed 18 June 2020].
- [20] guru99, "CodeIgniter MVC(Model View Controller) Framework with Example," guru99, 2020. [Online]. Available: <https://www.guru99.com/codeigniter-mvc-framework.html>. [Accessed 18 June 2020].
- [21] C. Nachre, "Strong Features of CodeIgniter: Powerful PHP Framework," <https://yourstory.com/>, 29 April 2019. [Online]. Available: <https://yourstory.com/mystory/strong-features-of-codeigniter-powerful-php-framew>. [Accessed 18 June 2020].
- [22] M. Rouse, "MySQL," Techtarget, July 2018. [Online]. Available: <https://searchoracle.techtarget.com/definition/MySQL>. [Accessed 18 June 2020].
- [23] javatpoint, "MySQL Features," javatpoint, 2020. [Online]. Available: <https://www.javatpoint.com/mysql-features>. [Accessed 18 June 2020].
- [24] British Columbia Institute of Technology, "PHP Style Guide," 2019. [Online]. Available: <https://codeigniter.com/userguide3/general/styleguide.html>. [Accessed 13 June 2020].
- [25] V. Brox, Date estimation in lineage-linked databases, Newcastle: University of Newcastle upon Tyne, 2000.
- [26] Guru99, "How to Create a Test Plan," Guru99, 2020. [Online]. Available: <https://www.guru99.com/what-everybody-ought-to-know-about-test-planing.html>. [Accessed 8 May 2020].
- [27] Guru99, "What is BLACK Box Testing? Techniques, Example & Types," Guru99, 2020. [Online]. Available: <https://www.guru99.com/black-box-testing.html>. [Accessed 28 February 2020].
- [28] Guru99, "What is WHITE Box Testing? Techniques, Example, Types & Tools," Guru99, 2020. [Online]. Available: <https://www.guru99.com/white-box-testing.html>. [Accessed 29 February 2020].
- [29] R. Deane, "Coworking spaces in and around Colombo.," Sunday Times, 17 February 2019. [Online]. Available: <http://www.sundaytimes.lk/190217/magazine/coworking-spaces-in-and-around-colombo-335825.htm>. [Accessed 23 January 2020].

- [30] G. Ibasco, "COWORKING SPACE CASE STUDY: THE TRANSITION FROM A "TRADITIONAL OFFICE" TO COWORKING," www.gorillaspace.co, 28 August 2019. [Online]. Available: <https://www.gorillaspace.co/blog/traditional-to-coworking-space-case-study/>. [Accessed 23 January 2020].
- [31] J. v. d. K. R. A.-M. & T. A. Minou Weijs-Perrée, "Analysing User preferences co-working space characteristics, Building Research and Information," *Building Research & Information*, vol. 5, no. 47, pp. 534-548, 2018.
- [32] Bookmyshow, "Bookmyshow.com," Bookmyshow, [Online]. Available: <https://lk.bookmyshow.com/>. [Accessed 24 January 2020].
- [33] Datamation, "8 Major Advantages of Using MySQL," Datamation, 16 Novemeber 2016. [Online]. Available: <https://www.datamation.com/storage/8-major-advantages-of-using-mysql.html>. [Accessed 05 May 2020].
- [34] Guru99, "Black Box Testing Vs. White Box Testing: Key Differences," Guru99, 2020. [Online]. Available: <https://www.guru99.com/back-box-vs-white-box-testing.html>. [Accessed 7 May 2020].

Appendix A: User Manual

A.1 Home page

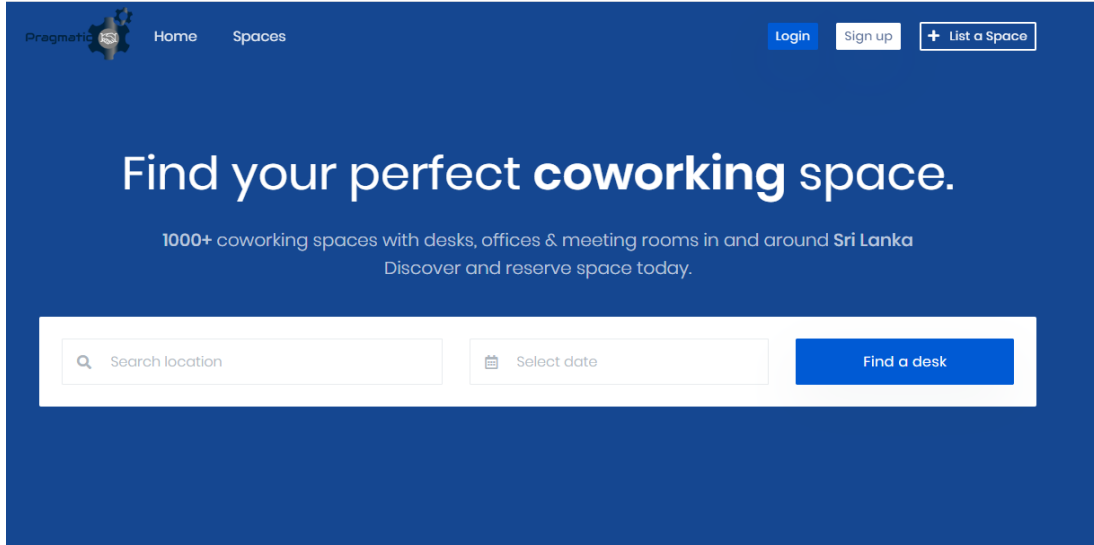


Figure A.0.1: Home page

A.2 Customer Sign up page

Customers who have not registered should click the “Sign Up” button.

The image shows the customer sign up page. The title is 'JOIN PRAGMATIC' with the subtitle 'Drop in to beautiful workspaces all around your city.' The form includes fields for 'Enter email', 'Enter first name', 'Enter last name', 'Enter address', 'Enter phone', 'Enter post code', 'Enter city', 'Enter country', 'Password', and 'Confirm password'. There is a checkbox for 'I agree to the terms and conditions' and a 'Create account' button. To the right of the form is an illustration of a building with a sign that says 'AD' and a thumbs-up icon.

Figure A.2: Customer sign up page

A.3 Customer Login Page

Customers who have signed up can click the “Login” button and navigate to the login page.

WELCOME BACK!
Please log back in to access your account.

Enter email

This field is required.

Password

Sign in

Not registered? [Create account](#) [Lost password?](#)

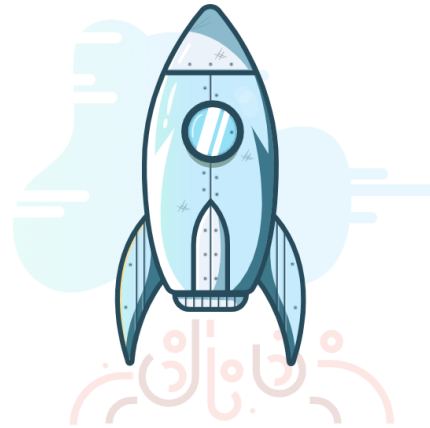



Figure A.3: Customer login page

A.4 Tables Page

Customers can select the available tables listed in spaces page.


Home · 435, 15a Longden Hill, Colombo 00500 · WeHive

Space in 435, 15a Longden Hill, Colombo 00500



Single Person tables for you to work individually. W750 x D800 x H750 sized tables for each person with rolling chair will be provided. Free usage of A/C, power cords and reading light will be provided

Table No	Charge
Table 1	700



Single Person tables for you to work individually. W750 x D800 x H750 sized tables for each person with rolling chair will be provided. Free usage of A/C and reading light will be provided

Table No	Charge
Table 2	700

Figure A.4: Tables page

A.5 Table Details Page

Customers can check the details and facilities provided in each table in the table details page.

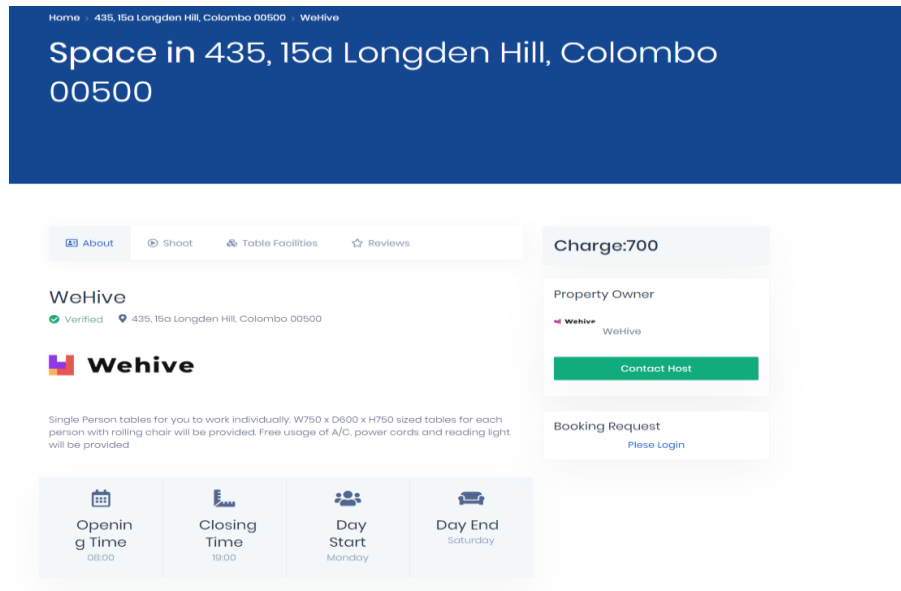


Figure A.5: Table details page

A.6 Customer Booking Form

Customers can book the table by logging in and filling the booking form in the table details page.

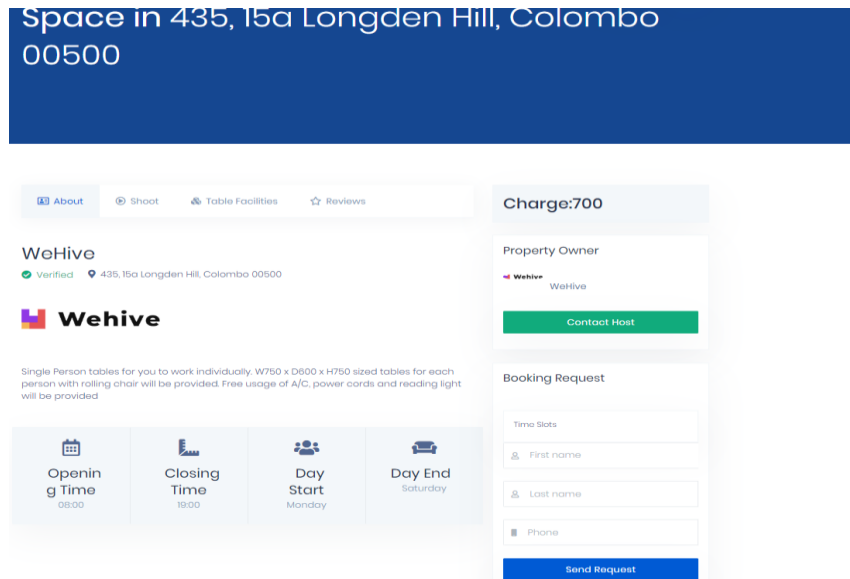


Figure A.6: Customer booking form

A.7 My bookings section

Customers can check the bookings they made in the my bookings section of the my account page.

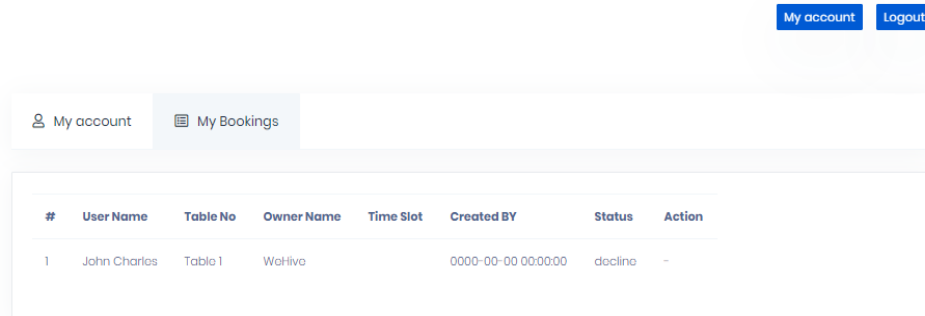


Figure A.7: My bookings section

A.8 Admin panel page

Admins who needs to register or manage their space should click the “List a Space” button to navigate to the Admin Panel.

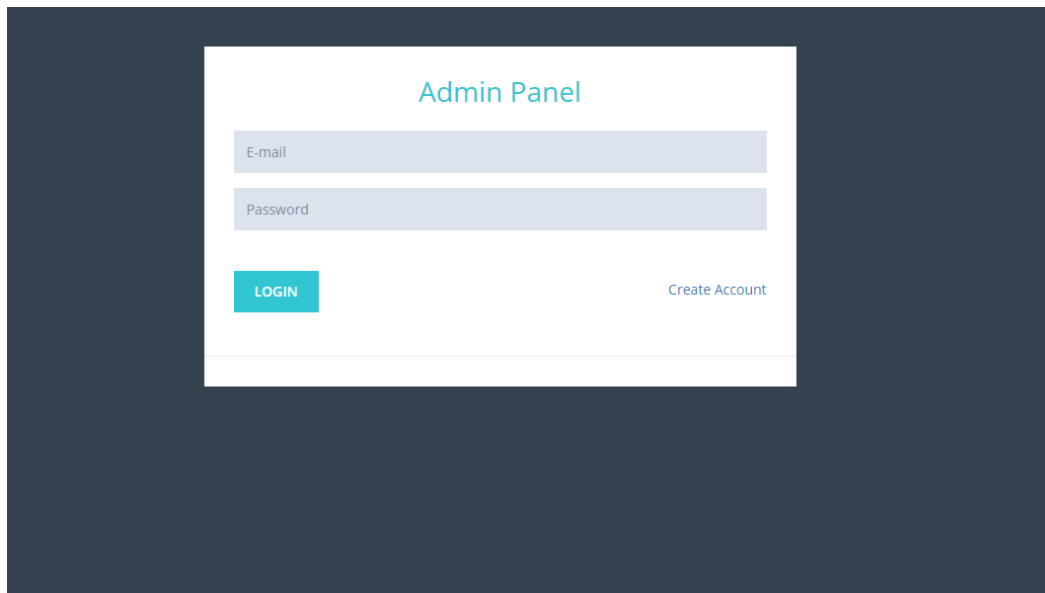
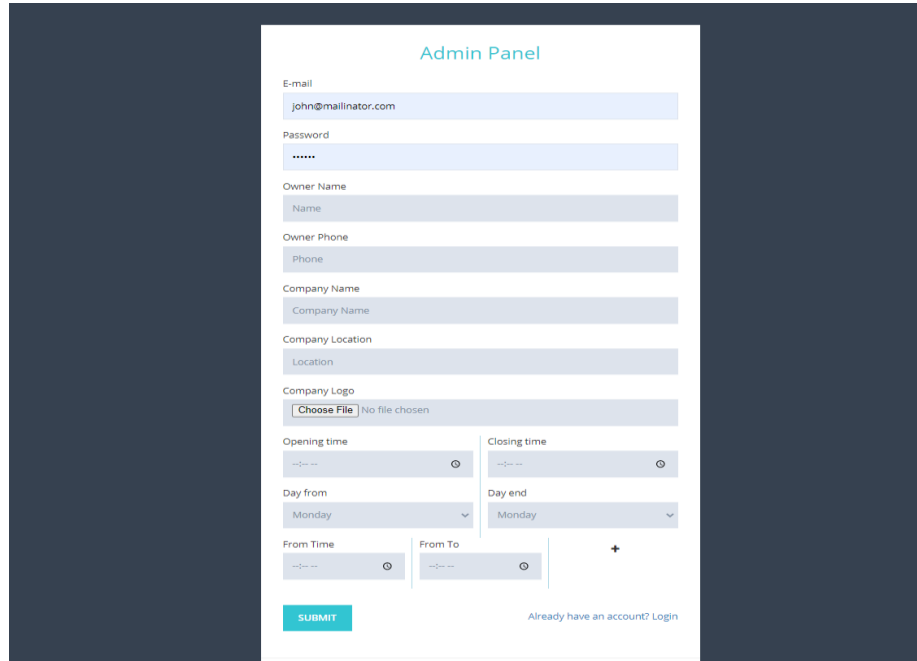


Figure A.8: Admin panel page

A.9 Admin Sign up and Create Space page

Admins who have not signed up or created their space click the “Create Account” button and navigate to admin sign up and Create Space page.



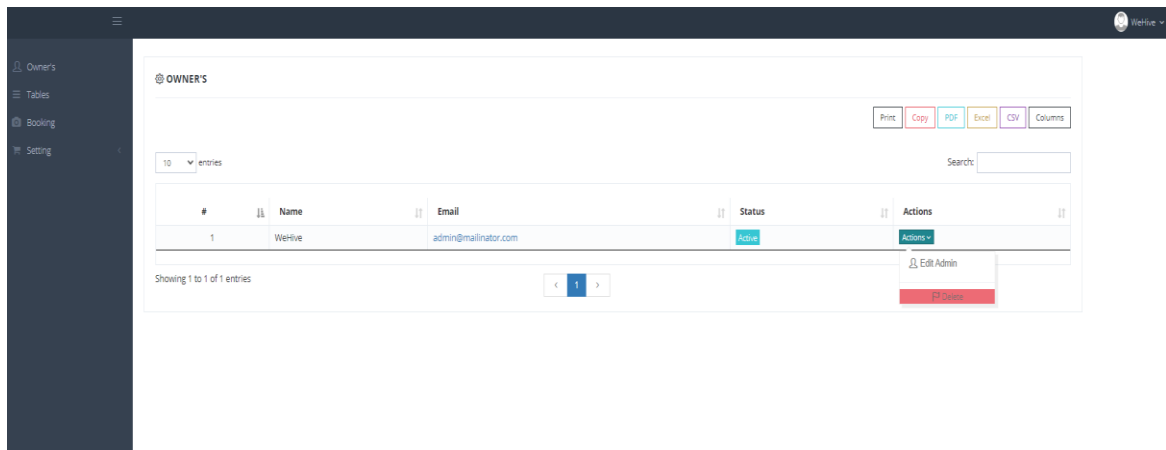
The image shows a web form titled "Admin Panel" for creating an account. It includes the following fields and options:

- E-mail: john@mailinator.com
- Password: masked with dots
- Owner Name: Name
- Owner Phone: Phone
- Company Name: Company Name
- Company Location: Location
- Company Logo: Choose File (No file chosen)
- Opening time: time picker
- Closing time: time picker
- Day from: Monday (dropdown)
- Day end: Monday (dropdown)
- From Time: time picker
- From To: time picker
- A blue "SUBMIT" button is at the bottom left.
- A link "Already have an account? Login" is at the bottom right.

Figure A.9: Admin sign up and create space page

A.10 Admin dashboard page

Admins who have signed up can login and they will be navigated to the Admin Dashboard. Admin can update his details or delete the space using the actions in this page.



The image shows an admin dashboard for "OWNER'S". It features a sidebar with navigation options: Owner's, Tables, Booking, and Setting. The main content area displays a table of owner entries with the following data:

#	Name	Email	Status	Actions
1	Weilive	admin@mailinator.com	Active	Actions > Edit Admin Delete

Additional UI elements include a search bar, a "Showing 1 to 1 of 1 entries" indicator, and a pagination control showing "1".

Figure A.10: Admin dashboard page

A.11 Tables Page

Admin can see the tables created in his space by navigating to the Tables menu in the dashboard page. He can update or delete the tables using the edit and delete option in the actions. He can generate reports of the tables using the reports option on top right.

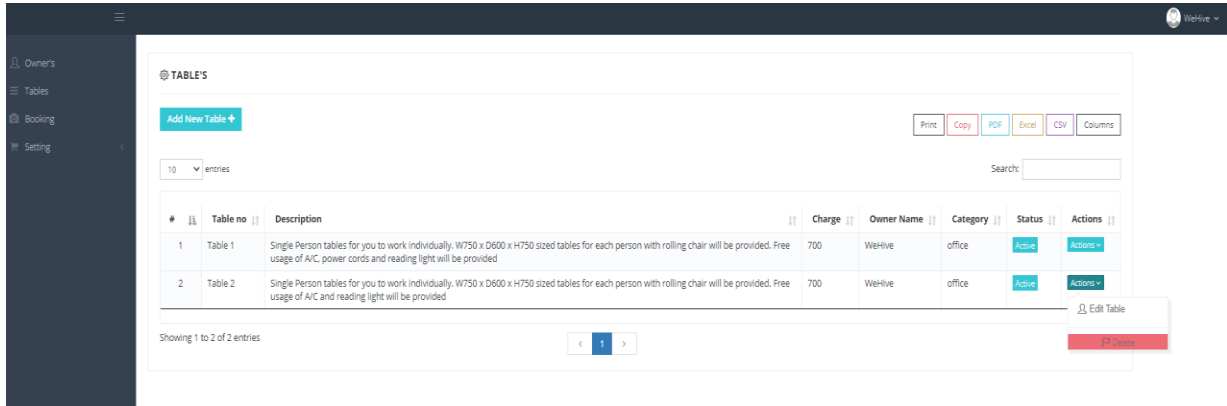


Figure A.11: Tables page

A.12 Add New table Page

Admins can create new tables by clicking the Add new table button, which will navigate the admin to add new table page where the admin can add details of the table such as description, images, charges, facilities and price.

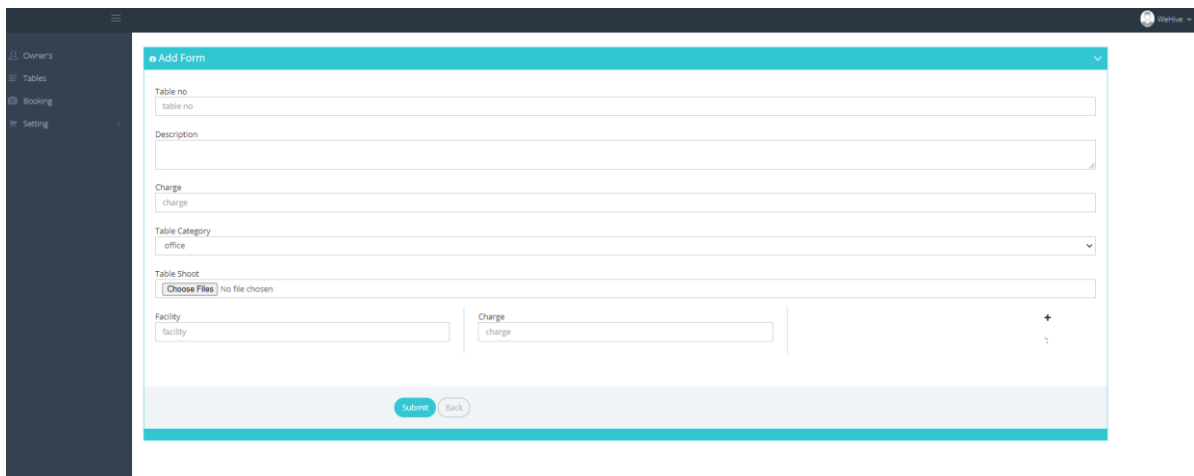


Figure A.12: Add new table page

A.13 Bookings Page

Admin can see the bookings he received to his space by navigating to the Bookings menu in the dashboard page. He can accept or reject the bookings using the edit option in the actions columns. He can generate reports of the bookings using the reports option on top right.

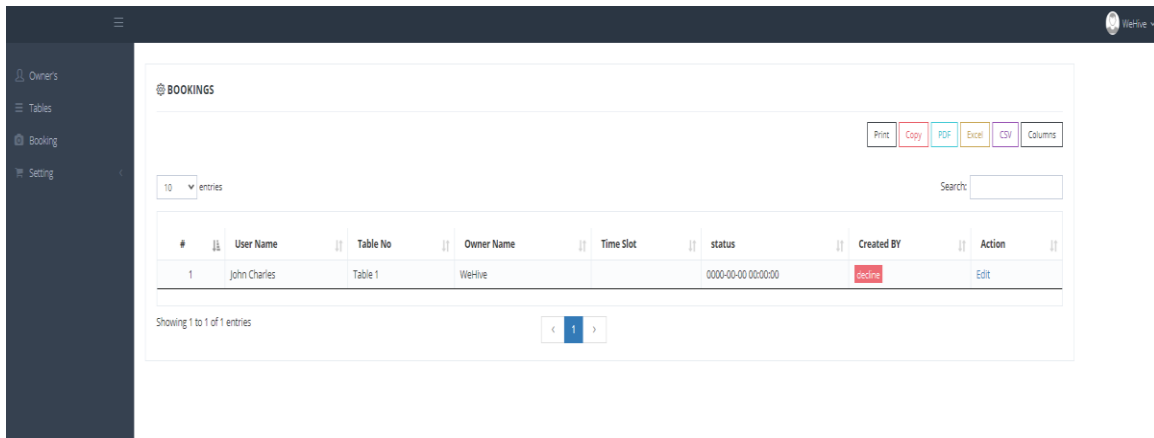


Figure A.13: Bookings page

A.14 Admin Settings Page

Admin can update his details that he entered in the signup page by clicking the settings page.

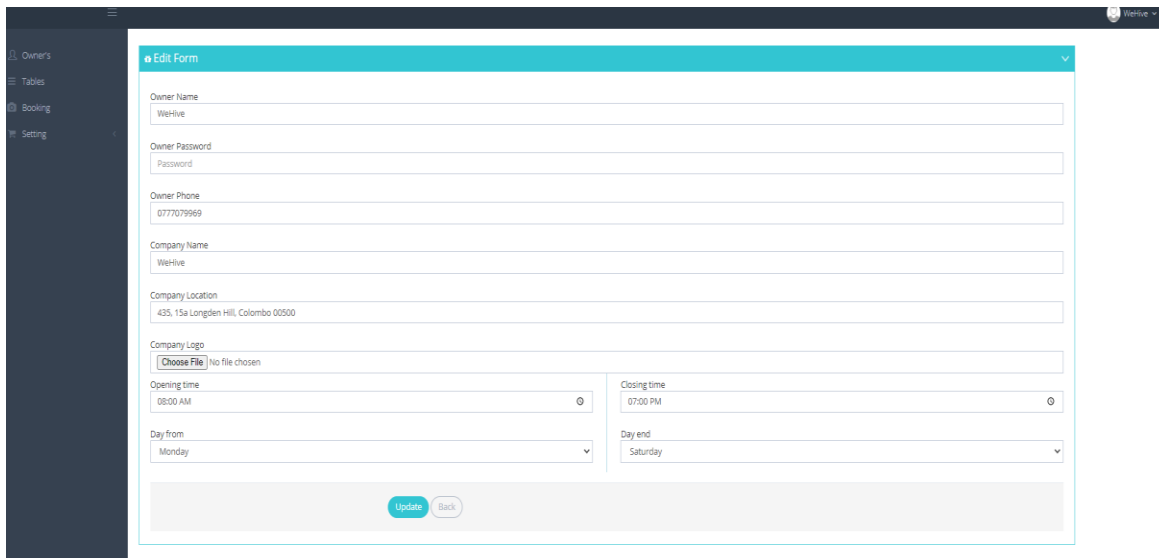
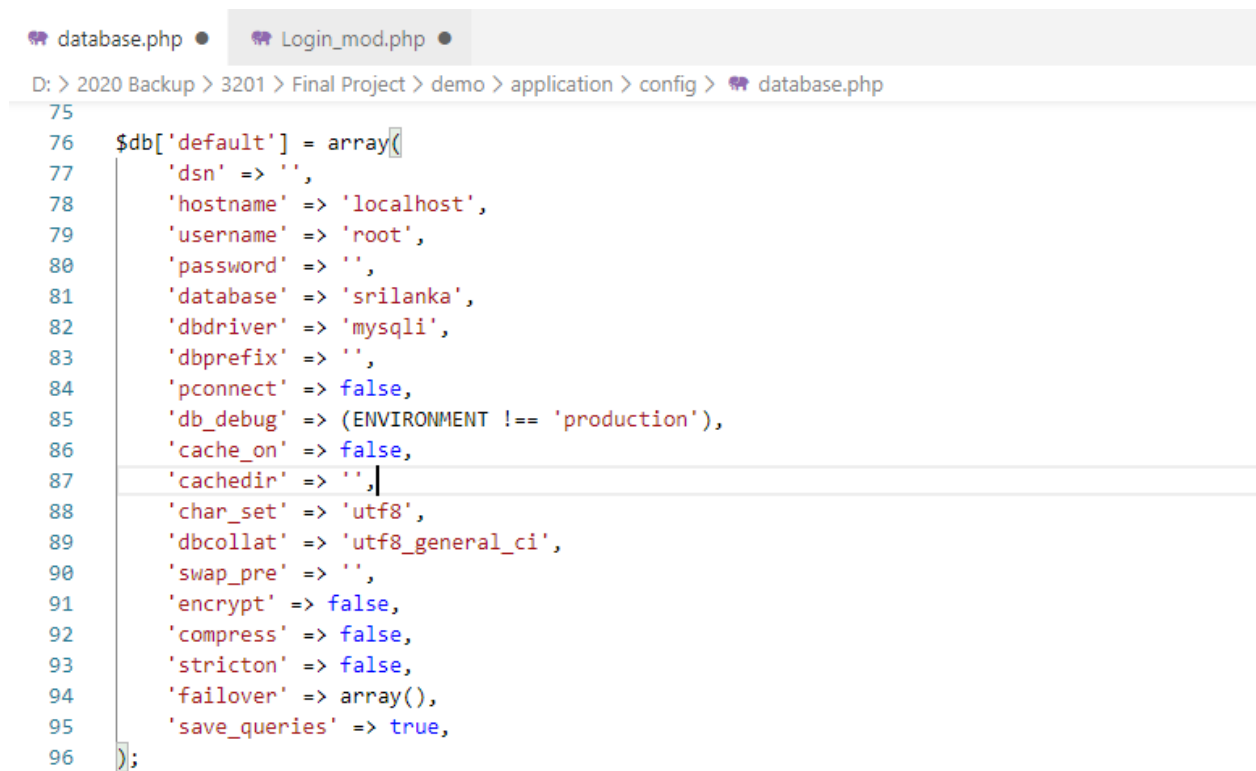


Figure A.14: Admin settings page

Appendix B: Important Code Segments

B.1 Code segment for database configuration settings

Following code segment displayed from database.php file shows the database configuration settings that is used to connect to the database. These values are stored in a multi-dimensional array.

The image shows a code editor window with two tabs: 'database.php' and 'Login_mod.php'. The 'database.php' tab is active, showing a code segment for database configuration. The code is as follows:

```
75
76 $db['default'] = array(
77     'dsn' => '',
78     'hostname' => 'localhost',
79     'username' => 'root',
80     'password' => '',
81     'database' => 'srilanka',
82     'dbdriver' => 'mysqli',
83     'dbprefix' => '',
84     'pconnect' => false,
85     'db_debug' => (ENVIRONMENT !== 'production'),
86     'cache_on' => false,
87     'cachedir' => '',
88     'char_set' => 'utf8',
89     'dbcollat' => 'utf8_general_ci',
90     'swap_pre' => '',
91     'encrypt' => false,
92     'compress' => false,
93     'stricton' => false,
94     'failover' => array(),
95     'save_queries' => true,
96 );
```

Figure B.1: Database configuration settings in database.php file

B.2 Code segment for admin registration and create spaces

Following code segments display the controller class for the admin registration and create space process.

Figure B.2 displays the controller class of the admin registration and create space process.

```
Admin.php x
D:\> 2020 Backup > 3201 > Final Project > demo > application > controllers > Admin.php
65 public function registered()
66 {
67     if (empty($_FILES['company_logo']['name'])) {
68         $file_name = $_FILES['company_logo']['name'];
69         $file_size = $_FILES['company_logo']['size'];
70         $file_tmp = $_FILES['company_logo']['tmp_name'];
71         $file_type = $_FILES['company_logo']['type'];
72         move_uploaded_file($file_tmp, "assets/upload/company_logo/" . $file_name);
73         $file_name = "assets/upload/company_logo/" . $file_name;
74     } else {
75         $file_name = '';
76     }
77 }
78
79 $dataOwner = array(
80     'email' => $this->input->post('email'),
81     'password' => sha1($this->input->post('password')),
82     'name' => $this->input->post('owner_name'),
83     'phone' => $this->input->post('owner_phone'),
84     'company_name' => $this->input->post('company_name'),
85     'company_location' => $this->input->post('location'),
86     'opening_time' => $this->input->post('opening_time'),
87     'closing_time' => $this->input->post('closing_time'),
88     'day_from' => $this->input->post('day_from'),
89     'day_end' => $this->input->post('day_end'),
90     'company_logo' => $file_name,
91     'status' => 'active'
92 );
93
94 $insertId=$this->Owner_m->insertOwner($dataOwner);
95
96 if($insertId){
97     $time_from = $this->input->post('time_from');
98     $time_to = $this->input->post('time_to');
99     foreach($time_from as $key=>$value){
100         $time_slots[] = array(
101             'time_from'=>$time_from[$key],
102             'time_to' => $time_to[$key],
103             'owner_id'=>$insertId,
104         );
105     }
106     $this->Owner_m->insertTimeSlots($time_slots);
107     redirect('Admin');
108 }else{
109     $this->session->set_flashdata('msg', 'Something is wrong try again');
110     redirect('Admin/signup');
111 }
112
113
114
115
116 public function check_email()
117 {
118     $this->Admin_model->email_check($this->input->post('email'));
119 }
120
121 }
```

Figure B.2: Admin registration and create space controller class

B.3 Code segment for create table process

Following code segments display the controller class for the create table process.

Figure B.3 displays the controller class of the create table process.

```
Dashboard.php x
D:\2020 Backup > 3201 > Final Project > demo > application > controllers > Dashboard.php
1350     public function addTable()
1351     {
1352         $data['categories'] = $this->table_m->getTablesCategory();
1353         $data['header'] = $this->load->view('Admin/header', null, true);
1354         $data['footer'] = $this->load->view('Admin/footer', null, true);
1355         $this->load->view('Admin/tables/add-table', $data);
1356     }
1357     public function save_table()
1358     {
1359     }
1360
1361     if ($this->input->server('REQUEST_METHOD') == 'POST' || !empty($_REQUEST)) {
1362
1363         $table_no = trim($this->input->post('table_no'));
1364         $description = trim($this->input->post('description'));
1365         $category_id = $this->input->post('category_id');
1366         $facility = $this->input->post('facility');
1367         $charge = $this->input->post('charge');
1368         $charges = trim($this->input->post('charges'));
1369         $targetDir = "assets/upload/tables/";
1370         $data = array(
1371             'table_no' => $table_no,
1372             'description' => $description,
1373             'created_at' => date('y-m-d h:m:s'),
1374             'change' => $charge,
1375             'owner_id' => $this->session->userdata('admin_id'),
1376             'category_id' => $category_id,
1377             'status' => 'active',
1378         );
1379
1380         $data = $this->security->xss_clean($data);
1381         $insertId = $this->table_m->addTableInfo($data);
1382         if($insertId) {
1383             foreach($_FILES['image']['name'] as $key => $val) {
1384                 // File upload path
1385                 $fileName = basename($_FILES['image']['name'][$key]);
1386                 $targetFilePath = $targetDir . $fileName;
1387                 // Upload file to server
1388                 if (move_uploaded_file($_FILES['image']['tmp_name'][$key], $targetFilePath)) {
1389                     // Image db insert sql
1390                     $table_shoot[] = array(
1391                         'image' => $targetFilePath,
1392                         'table_id' => $insertId,
1393                         'owner_id' => $this->session->userdata('admin_id'),
1394                     );
1395                 }
1396             }
1397
1398             foreach ($facility as $key => $value) {
1399                 $table_facilities[] = array(
1400                     'facility' => $facility[$key],
1401                     'charge' => $charge[$key],
1402                     'table_id' => $insertId,
1403                 );
1404             }
1405
1406             $this->table_m->insertTableShoot($table_shoot);
1407             $this->table_m->insertTableFacilities($table_facilities);
1408
1409             $alert = "<div class='alert alert-success'>
1410                 <strong>Save!</strong> Data has been inserted Successfully. </div>";
1411             $this->session->set_flashdata('Alert', $alert);
1412             redirect('Dashboard/addTable');
1413         } else {
1414             $alert = "<div class='alert alert-danger'>
1415                 <strong>Error!</strong> Data failed to inserted. Please Try again. </div>";
1416             $this->session->set_flashdata('Alert', $alert);
1417             redirect('Dashboard/addTable');
1418         }
1419     } else {
1420         redirect('Dashboard/addTable');
1421     }
1422 }
1423 }
```

Figure B.3: Create table controller class

B.4 Code segment for update booking status

Following code segments display the controller class for the update booking status process.

Figure B.4 displays the controller class of the update booking status process.

```
Dashboard.php x
D:\> 2020 Backup > 3201 > Final Project > demo > application > controllers > Dashboard.php
1624 public function updateBooking()
1625 {
1626     if ($this->input->server('REQUEST_METHOD') == 'POST' || !empty($_REQUEST)) {
1627         if (isset($_POST['updatecategory'])) {
1628             $booking_id = $this->input->post('booking_id');
1629             $status = $this->input->post('status');
1630
1631             $dataOwner = array(
1632                 'status' => $status,
1633                 'updated_at' => date('y-m-d')
1634             );
1635
1636             $data = $this->security->xss_clean($dataOwner);
1637
1638             $result = $this->Booking_m->updateBooking($data, $booking_id);
1639
1640             if ($result) {
1641                 $alert = '<div class="alert alert-success">
1642                     <strong>Updated!</strong> Data has been updated Successfully. </div>';
1643                 $this->session->set_flashdata('Alert', $alert);
1644
1645                 redirect('edit-booking/' . $booking_id);
1646             } else {
1647                 $alert = '<div class="alert alert-danger">
1648                     <strong>Error!</strong> Data failed to update. Please Try again. </div>';
1649                 $this->session->set_flashdata('Alert', $alert);
1650
1651                 redirect('edit-booking/' . $booking_id);
1652             }
1653         }
1654     }
1655 } else {
1656     redirect('Dashboard');
1657 }
1658 }
1659 }
1660 }
1661 }
```

Figure B.4: Controller class of update booking process

Appendix C: Use case diagram

Figure C.1 displays the overall use case diagram of the system

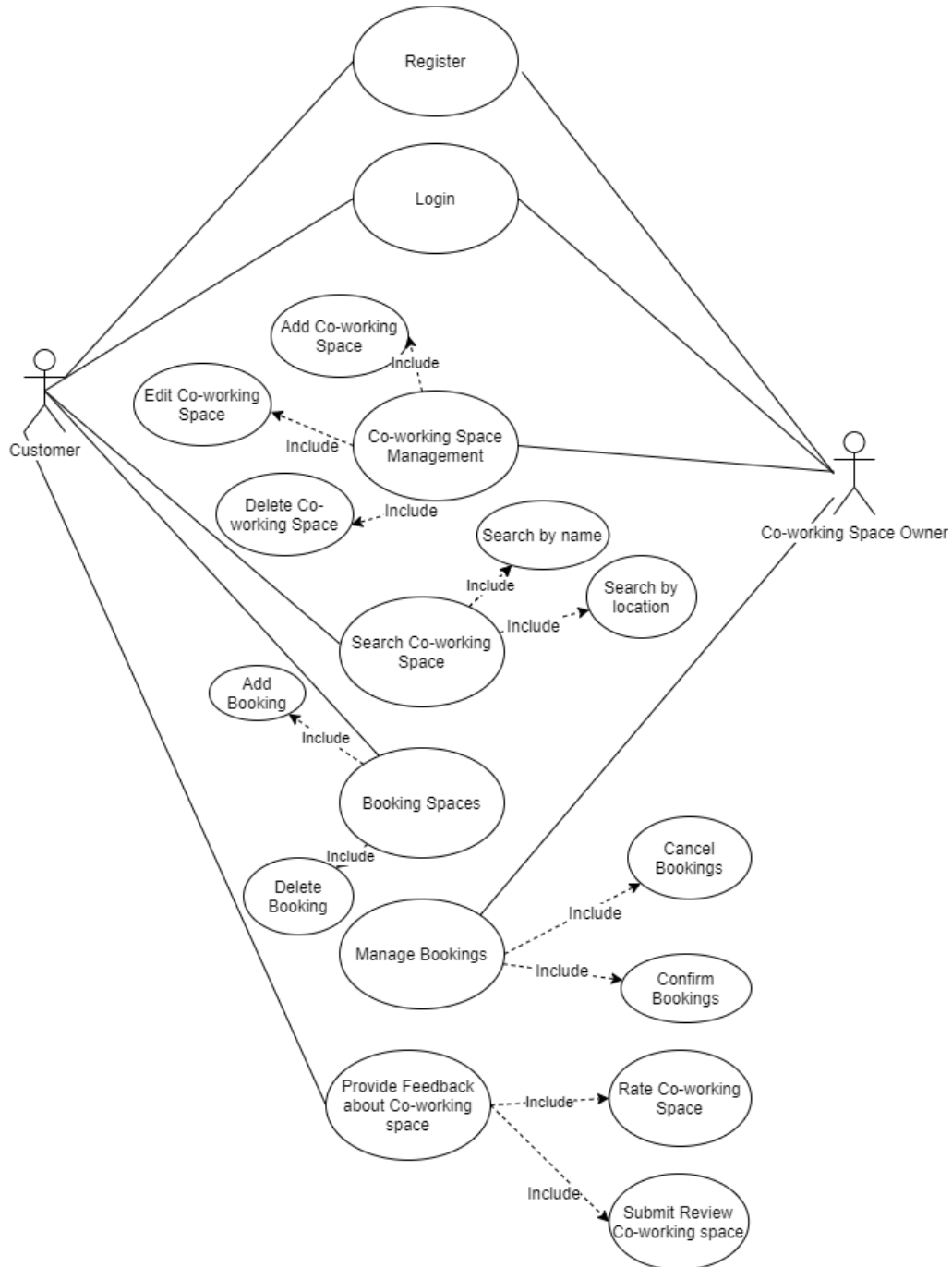


Figure C.1: Overall use case diagram

Appendix D: Activity Diagram

D.1 Activity diagram of Coworking Space Admin

Figure D.1 displays the activity diagram of the coworking space admin

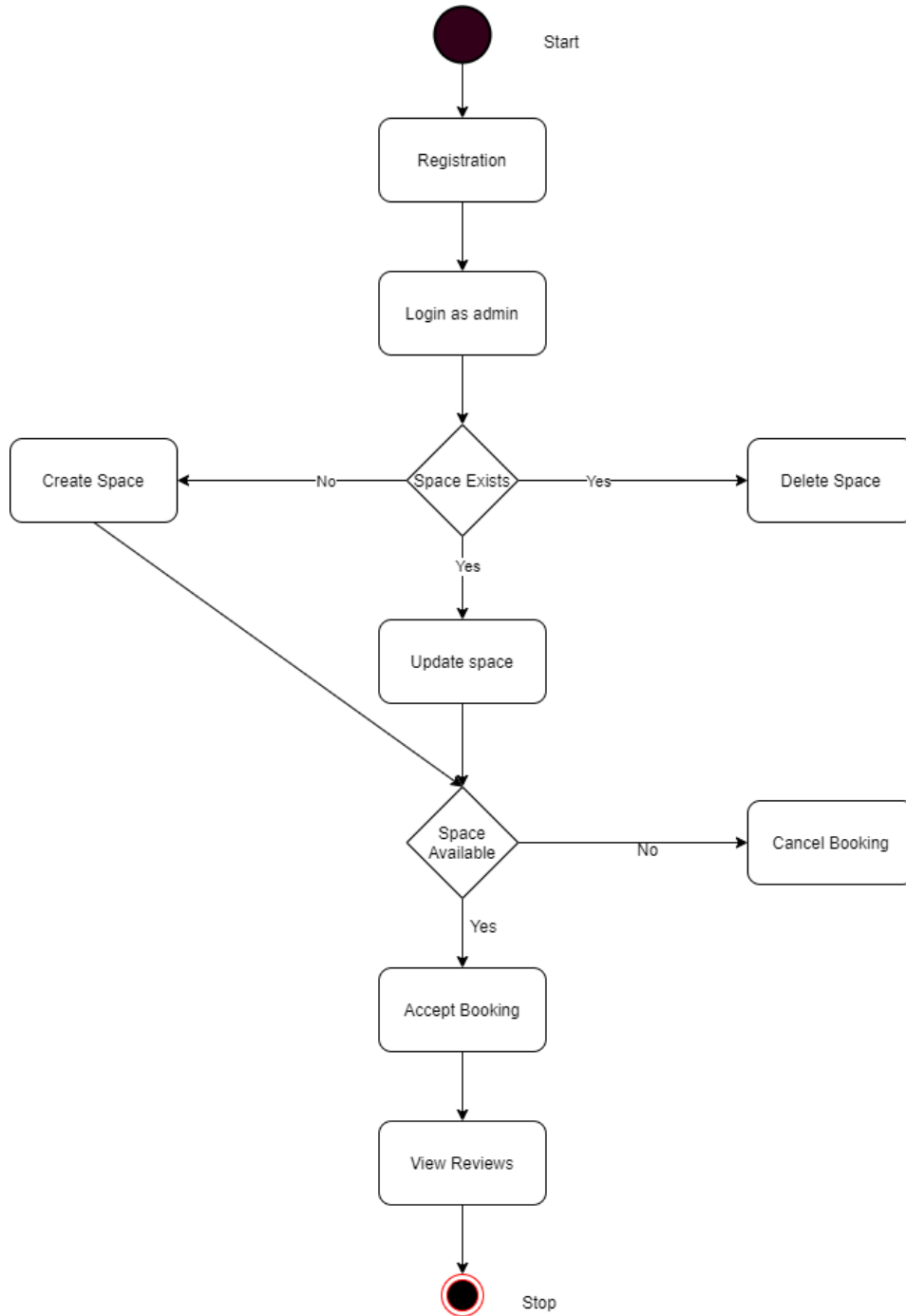


Figure D.1: Activity diagram of space admin

D.2 Activity diagram of Customer

Figure D.2. displays the activity diagram of the customer

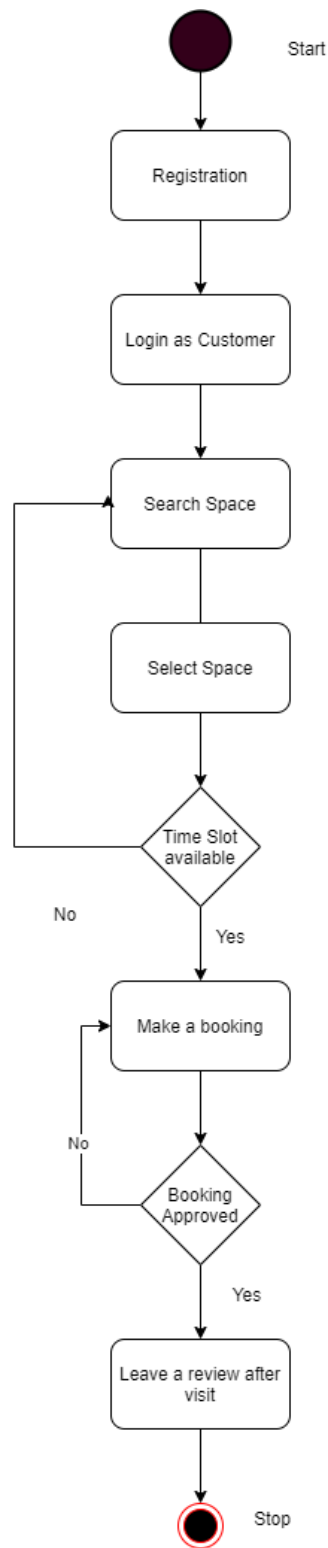


Figure D.2: Activity diagram of customer

Appendix E: Test case Document

Test Case ID	Scenario	Test Steps	Test Data	Expected Result	Actual Result Pass / Fail
Signup & Login related Test cases					
A001	Signup to the system as an Admin	Go to the Admin Signup page Enter e-mail Enter Password Enter admin details Enter Company details Click on Submit Button	E-mail : admin@mailinator.com Password : admin	User should be able to login to the system ONLY Admin privileges should be displayed	Pass
A002	Signup to the system as a Customer	Go to the Public Signup page Select User role as Customer Enter User Name Enter Password Click on Login Button	User Name : dist Password : dist123	User should be able to login to the system ONLY Customer privileges should be displayed	Pass
A003	Login to the System as an Admin	Go to the login page Enter User Name Enter Password Click on Login Button	User Name : admin Password : admin	User should be able to login to the system ONLY admin privileges should be displayed	Pass
A004	Login to the System as a Customer	Go to the login web page Enter User Name Enter Password Click on Login Button	User Name : app Password : app123	User should be able to login to the system ONLY customer privileges should be displayed	Pass
A005	Unsuccessful Login	Go to the login web page Enter User Name Enter Password Click on Login Button	Invalid User Name Valid Password	User should NOT be able to login to the system	Pass
A006	Verify Admin user can logout from the system	Login as an Admin Click Logout Button		Admin should be able to log out of the system	Pass
A007	Verify Customer user can logout from the system	Login as a customer Click Logout Button		Customer should be able to log out of the system	Pass
Admin related Test cases					
AD001	Verify Space Company is created after admin creates his account	Precondition : Signup to the System as an admin Test Steps: Go to the Owners page in admin dashboard		Admin should be able to see the company created in the owners page All details he provided about the company in the signup page should be displayed Customer should be able to see the newly created space in the public site	Pass

AD002	Verify admin can update the details of the space he created	<p>Precondition : Login to the System as an admin Navigate to the owners page Click Edit button next to the space name under actions</p> <p>Test Steps: Update all space related details Update logo images with new images Click on "Update" button to continue</p>		<p>Admin should be able to see space details are updated correctly</p> <p>Admin should be able to see newly uploaded images are updated correctly</p> <p>Customer should be able to see the updated details in the Admin site</p>	Pass
AD003	Verify admin can navigate to the tables menu	<p>Precondition : Login to the System as an admin</p> <p>Test Steps: Navigate to the tables menu</p>		<p>Admin should be able to see tables page with all the tables details listed</p> <p>Admin should be able to see "Add new table" button</p>	Pass
AD004	Verify admin clicks "Add new table" button	<p>Precondition : Login to the System as an admin Navigate to the tables page</p> <p>Test Steps: Click "Add new table" button</p>		Admin should be navigated to Add new table page	Pass
AD005	Verify admin fills the Add new table form	<p>Precondition : Login to the System as an admin</p> <p>Test Steps: Click "Add new table" button Enter Details about the table Click Submit button</p>		<p>Admin should be able to see a new table created with all details he entered and images he uploaded</p> <p>Customer should be able to see the newly created table in the public site of the company page</p>	Pass
AD006	Verify admin can update the details of the table he created	<p>Precondition : Login to the System as an admin</p> <p>Test Steps: Click "Edit" button under actions column in the tables page Update the details about the table Click Submit button</p>		<p>Admin should be able to see the table details updated in the tables page</p> <p>Customer should be able to see the table details updated in the public site</p>	Pass
AD007	Verify admin can delete the details of the table he created	<p>Precondition : Login to the System as an admin</p> <p>Test Steps: Click "Edit" button under actions column in the tables page Update the details about the table Click Submit button</p>		<p>Admin should be able to see the table deleted in tables page</p> <p>Customer should be able to see the table deleted from the public site</p>	Pass

AD008	Verify admin can generate a report about all the tables he created	Precondition : Login to the System as an admin Test Steps: Click "report" button in the tables page		Admin should be able to generate a report with the details of the tables he created	Pass
AD009	Verify admin navigates to the "Bookings" menu	Precondition : Login to the System as an admin Test Steps: Navigate to the bookings menu		Admin should be able to see all the booking details for his space listed	Pass
AD010	Verify admin can approve a booking	Precondition : Login to the System as an admin Navigate to the bookings menu Test Steps: Click edit button next to a booking Select "Approve" option and click update button		Admin should be able to see the approved status appear next to the booking in the bookings page	Pass
AD011	Verify admin can approve a booking	Precondition : Login to the System as an admin Navigate to the bookings menu Test Steps: Click edit button next to a booking Select "Decline" option and click update button		Admin should be able to see the declined status appear next to the booking in the bookings page	Pass
AD012	Verify admin can generate a report about all the bookings he got for his space	Precondition : Login to the System as an admin Test Steps: Click "report" button in the bookings page		Admin should be able to generate a report with the details of the bookings he received for his space	Pass
AD013	Verify admin can navigate to the settings menu	Precondition : Login to the System as an admin Test Steps: Navigate to the settings menu		Admin should be able to see all the details entered in the signup page	Pass
AD014	Verify admin can update the details in the settings page	Precondition : Login to the System as an admin Test Steps: Navigate to the settings menu Update all the details in the settings page Click Update button		Admin should be able to see all the details updated	Pass

Customer related Test cases					
C001	Verify customer can browse all the spaces in the site	Test Steps: User should navigate to the spaces page		User should be able to see all the spaces listed in the website	Pass
C002	Verify Customer goes clicks a space page	Test Steps: User should navigate to the spaces page Click a space listed		User should be able to see all the tables available in the space	Pass
C003	Verify Customer clicks a table listed	Test Steps: User should navigate to the spaces page Click a table listed		User should be able to see 1. Location, Opening Time, Closing time, opening days listed 2. Images of the tables 3. Facilities provided for the table 4. Reviews for the table 5. Booking section	Pass
C004	Verify customer can book a table listed	Precondition : Login to the System as a customer Test Steps: Click a table listed Enter Time slot Enter Name, Last Name and phone number		User should be able to see a booking created for that table	Pass
C005	Verify booking made by the customer is created	Precondition : Login to the System as a customer Test Steps: Go to My account section Click My listings tab		User should be able to see all the bookings he made listed	Pass
C006	Verify customer can see the status of the booking he created when the admin approves his booking	Precondition : Login to the System as a customer Test Steps: Go to My account section Click My listings tab		User should be able to see the status of his booking listed as approved next to his booking	Pass
C007	Verify customer can see the status of the booking he created when the admin declined his booking	Precondition : Login to the System as a customer Test Steps: Go to My account section Click My listings tab		User should be able to see the status of his booking listed as declined next to his booking	Pass
C008	Verify customer can cancel a booking he made	Precondition : Login to the System as a customer Test Steps: Go to My account section Click My listings tab Click the cancel button		User should be able to see the booking he made cancelled	Pass

C009	Verify customer can see all the bookings he made listed	<p>Precondition : Login to the System as a customer</p> <p>Test Steps: Go to My account section Click My listings tab</p>		User should be able to see the all bookings he made in the past and future listed	Pass
C010	Verify customer can rate a space	<p>Precondition : Login to the System as a customer</p> <p>Test Steps: Click a table listed Go to reviews tab Select the stars field where you can rate</p>		User should be able to rate a table by clicking the star rate fields and selecting the number of stars	Pass
C011	Verify customer can review a space	<p>Precondition : Login to the System as a customer</p> <p>Test Steps: Click a table listed Go to reviews tab Enter review in the review field Click submit</p>		User should be able to submit a review to the table by entering a review in the review field	Pass
C012	Verify customer can search spaces	<p>Test Steps: Enter search term in the search bar</p>		User should be able to see spaces listed according to the search made	Pass
C013	Verify customer can see spaces according to types. E.g.: Trending spaces	<p>Test Steps: Navigate to Trending spaces section</p>		User should be able to see all the trending spaces listed	Pass

Appendix F: Gantt chart

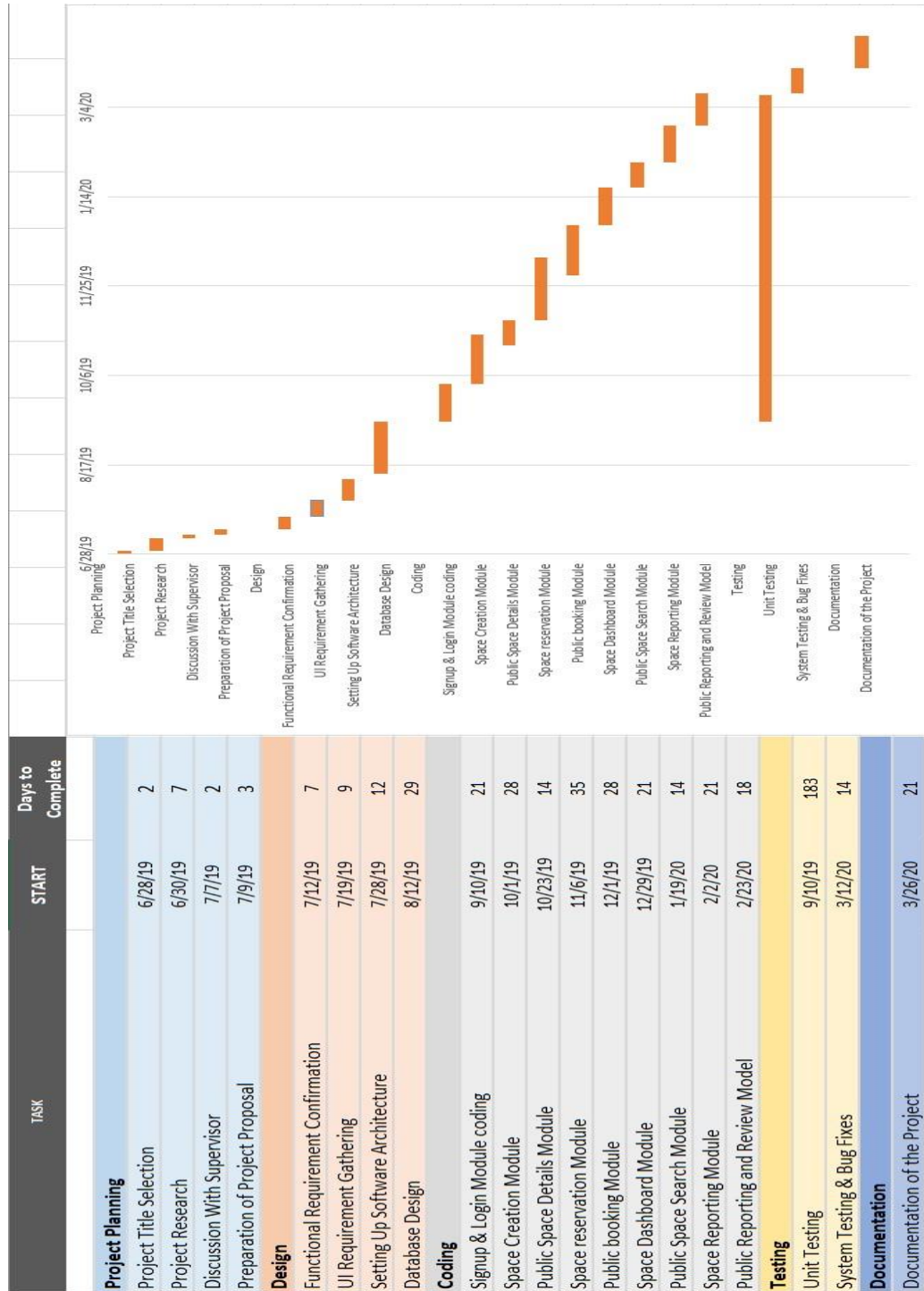


Figure F.1. Gantt chart

