University of Regina

ENSE 477

Software Design Project

SpaceBook Final Document

Authors:

Connor Meredith
Demitri Kourles
Zain Chowdhary

Advisors:

Christine Chan

April 9th, 2019

Table of Contents:

List of Figures:

1 Introduction

1.1 Purpose

The purpose of this application is to allow athletic facility managers to advertise their facilities and availability so that interested users can make bookings at facilities without experiencing the hassle of the current methods of facility booking and coordination.

1.2 Description

SpaceBook is a web-based athletic facility booking application. Many facilities today still use email or phone communication to coordinate bookings and communicate booking information. Information such as facility availability is often not easily accessible to those interested in booking athletic spaces, making the booking process more difficult for those seeking a facility. Adding to the complication, any forms must often be exchanged in person and processed manually when booking a facility. SpaceBook aims to provide the ability to advertise athletic spaces for rent and as well as the ability to browse and book available spaces all through the web. It will serve both facility owners and managers as well as potential users of those facilities by making the booking process more efficient for all involved.

2 Requirements and Specifications

2.1 Application Features

The following features are planned to be delivered by project day:

Login/Logout:

Users that have an existing SpaceBook account will be able to login to the application with their credentials and logout when they are finished using the application.

Recover Password:

Users will be provided a method of recovery if they have forgotten their password.

Registration:

Users will be able to register for a SpaceBook account that will allow them to login to the application.

Browse Facilities:

Users will be able to browse a list of available facilities filtered by a set of criteria such as facility type or features.

Location/Map Functionality:

Users will be able to view a facility's location on a map.

Post a Facility:

Facility owners and managers will be able to post a facility's details and availability.

View Facility Details:

Users will be able to view a facility's details and availability.

Edit Facility Details:

Facility owners and managers will be able to edit facility postings.

Browse Facility Availability:

Users will be able to view facility availability for a given date and time.

Rate a Facility:

Users will be able to rate a facility after using it.

View User Profile:

A user profile page will contain details about a given user. Users will be able to view their user profiles.

Edit User Profile:

Users will be able to customize and edit their user profiles.

Book a Facility:

Users will be able to book an available facility at a specific date and time.

View Upcoming Bookings:

Users and facility owners and managers will be able to view upcoming bookings.

Cancel a Booking:

Users and facility owners and managers will be able to cancel upcoming bookings.

Alerts and Notifications:

Users will receive alerts and notifications for important events such as reminders of an upcoming booking or a booking cancellation.

2.2 Release Criteria

Functionality:

The minimum viable product will include the following features:

- Login/Logout
- Registration
- Browse Facilities
- Post a Facility
- Edit Facility Details
- View Facility Details
- Browse Facility Availability
- Book a Facility
- Cancel a Booking
- View Upcoming Bookings

Usability:

The usability of the application will be determined through user testing. The tests to be used will be defined in the testing plan. Changes to the user interface will be made if the user tests indicate that changes are necessary.

Reliability:

The reliability of the software will be determined through testing. The tests to be used are defined in the testing plan. All tests must be passed to deliver the minimum viable product.

Performance:

The application must support simultaneous use by multiple users without impacting performance. Pages should load within a matter of seconds given a sufficient internet connection and a moderate load on the server.

Supportability:

The application will be supported in the Google Chrome and Mozilla Firefox web browsers.

3 System Design

3.1 System and Architecture Design

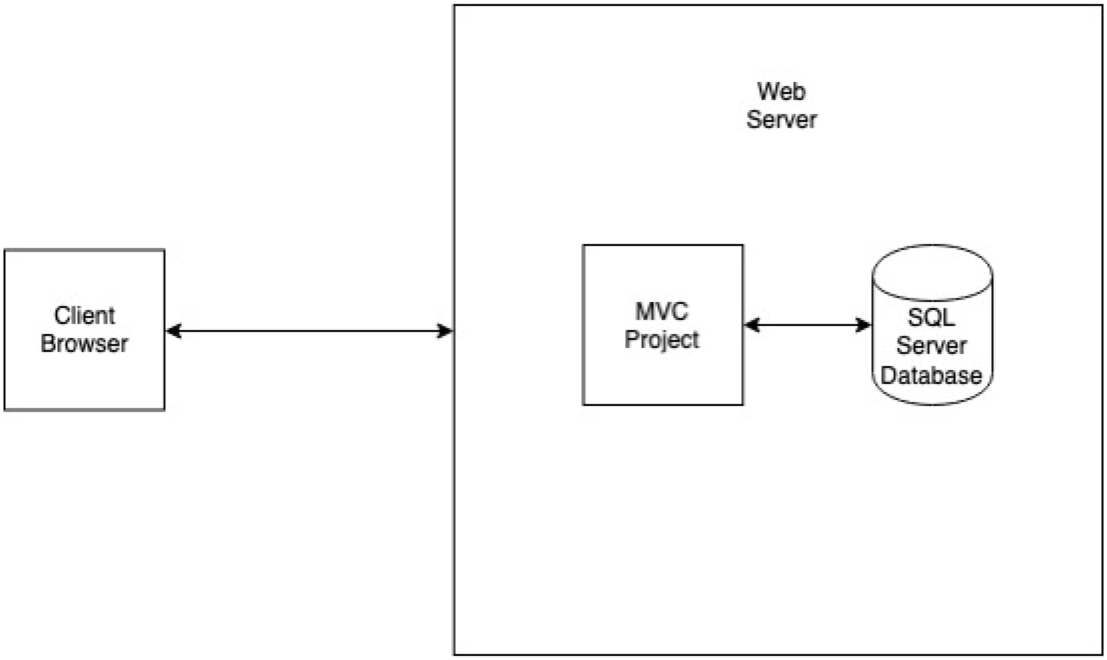


Figure 3.1.1: Client-Server Model

The application uses a basic client-server model in which the server is accessed via a client browser in order to use the application. A rented web server will be used to run the application and store the database. The diagram above (Figure 3.1.1) shows the described structure.

The reason a rented web server was chosen over independently hosting and maintaining a web server is due to a number of factors. Passing the setup and upkeep of the server to a 3rd party frees up time to focus on development. The availability of free hosting options from Amazon or Microsoft for the duration of the project period is another important factor in the decision. As well, the application's database storage needs are unlikely to be high enough to exceed the storage allowed under the free services. Altogether, this means that renting a web server provides a number of benefits without adding any restrictions to the use of the application.

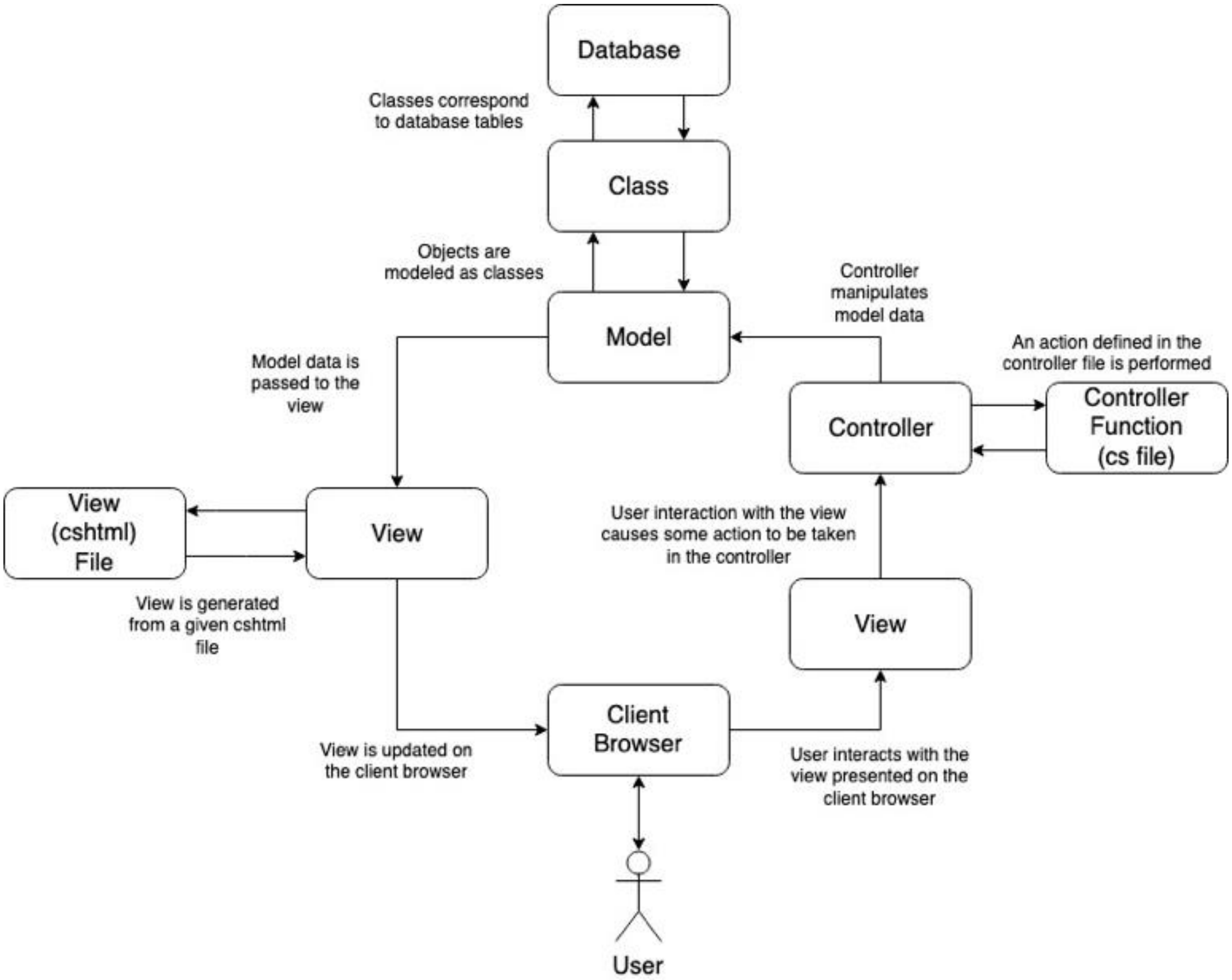

Database
Classes correspond to database tables
Class
Objects are modeled as classes
Model
Controller manipulates model data
An action defined in the controller file is performed
Model data is passed to the view
Controller
Controller Function (cs file)
View (cshtml) File
View
User interaction with the view causes some action to be taken in the controller
View is generated from a given cshtml file
View
Client Browser
View is updated on the client browser
User interacts with the view presented on the client browser
User

Figure 3.1.2: MVC Diagram

The application uses the Model View Controller (MVC) architecture as the basis of the application. It is implemented through Microsoft's ASP.NET MVC framework. Through the client browser, the user interacts with a given view. That interaction triggers an action by the controller on the server side. The controller acts upon data through the model and the model data is passed to the view. The view is updated with the provided data and served to the user via the client browser. The diagram above (Figure 3.1.2) shows the described process.

Microsoft's ASP.NET and MVC framework was chosen due to the familiarity of the group members with the technology through work terms and other class projects. As well, the availability of built in templates and functionality means that not everything needs to be built from scratch, allowing for focus to be shifted towards higher level functionality. This technology is widely used which makes the skills learned in the project likely to be applicable to a career in the industry. As well, this means there are many resources to pull from when encountering issues or deciding how to best implement features so they are in line with best practices.

3.2 Tools Used

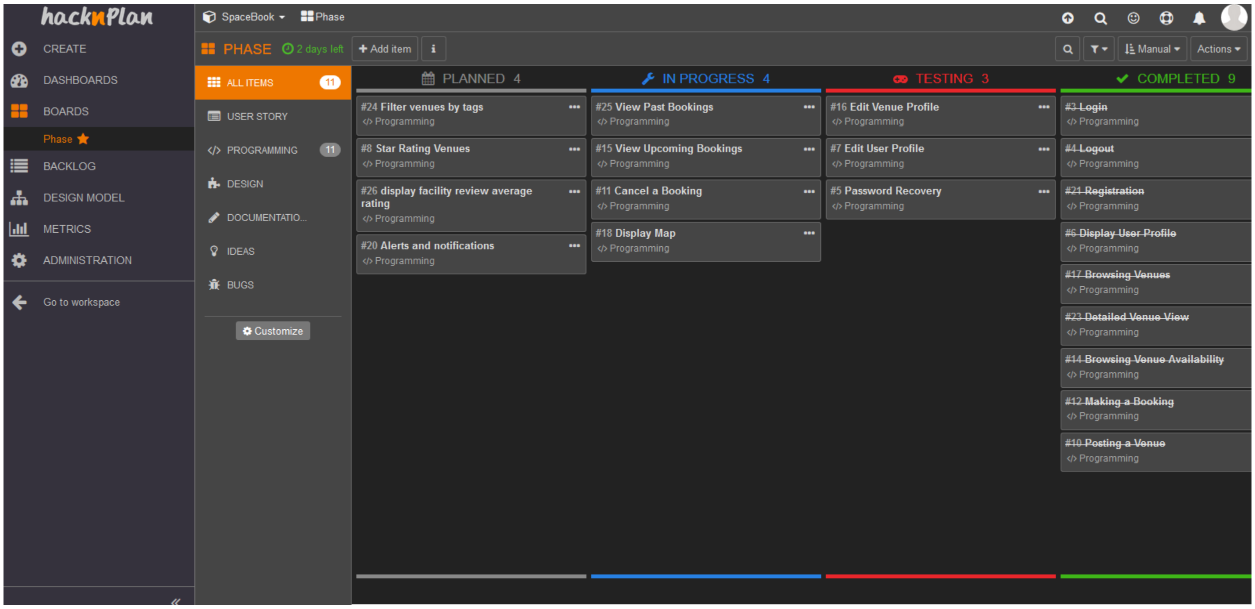
Below is the list of tools used for the project:

- Microsoft Visual Studio
    - IDE for ASP.NET MVC project
    - Visual Studio was chosen because it is the standard IDE for working with ASP.NET.

- SQL Server Management Studio (SSMS)
    - Database management
    - SSMS was chosen because it provides an easy way to generate scripts and manage a SQL database.

- Git
    - Version control
    - Git was chosen due to the need for collaboration on the implementation of the application as well as the familiarity of group members with git specifically.

- hacknplan
    - Release tracking
    - Hack n Plan was chosen because a method of tracking planned features was needed to improve organization for the project and facilitate the assignment of tasks to group members.



PLANNED 4	IN PROGRESS 4	TESTING 3	COMPLETED 9
#24 Filter venues by tags </> Programming	#25 View Past Bookings </> Programming	#16 Edit Venue Profile </> Programming	~~#3 Login~~ </> Programming
#8 Star Rating Venues </> Programming	#15 View Upcoming Bookings </> Programming	#7 Edit User Profile </> Programming	~~#4 Logout~~ </> Programming
#26 display facility review average rating </> Programming	#11 Cancel a Booking </> Programming	#5 Password Recovery </> Programming	~~#21 Registration~~ </> Programming
#20 Alerts and notifications </> Programming	#18 Display Map </> Programming		~~#6 Display User Profile~~ </> Programming
			~~#17 Browsing Venues~~ </> Programming
			~~#23 Detailed Venue View~~ </> Programming
			~~#14 Browsing Venue Availability~~ </> Programming
			~~#12 Making a Booking~~ </> Programming
			~~#10 Posting a Venue~~ </> Programming

Figure 3.2: hacknplan Example Board

- Google Drive
    - Collaborative work, planning, documentation, slide shows
    - Drive was chosen since it allowed collaboration and information sharing for all aspects of the project that were not directly related to the implementation of the application. It provided slide shows for the group's bi-weekly updates to the class, shared documents for referencing and adding notes relating to specific meetings, and spreadsheets for tracking meeting minutes.

- What's App
    - Communication
    - What's App was chosen since all group members were familiar with the application and it allowed communication across iOS and Android devices.

4 Object Design

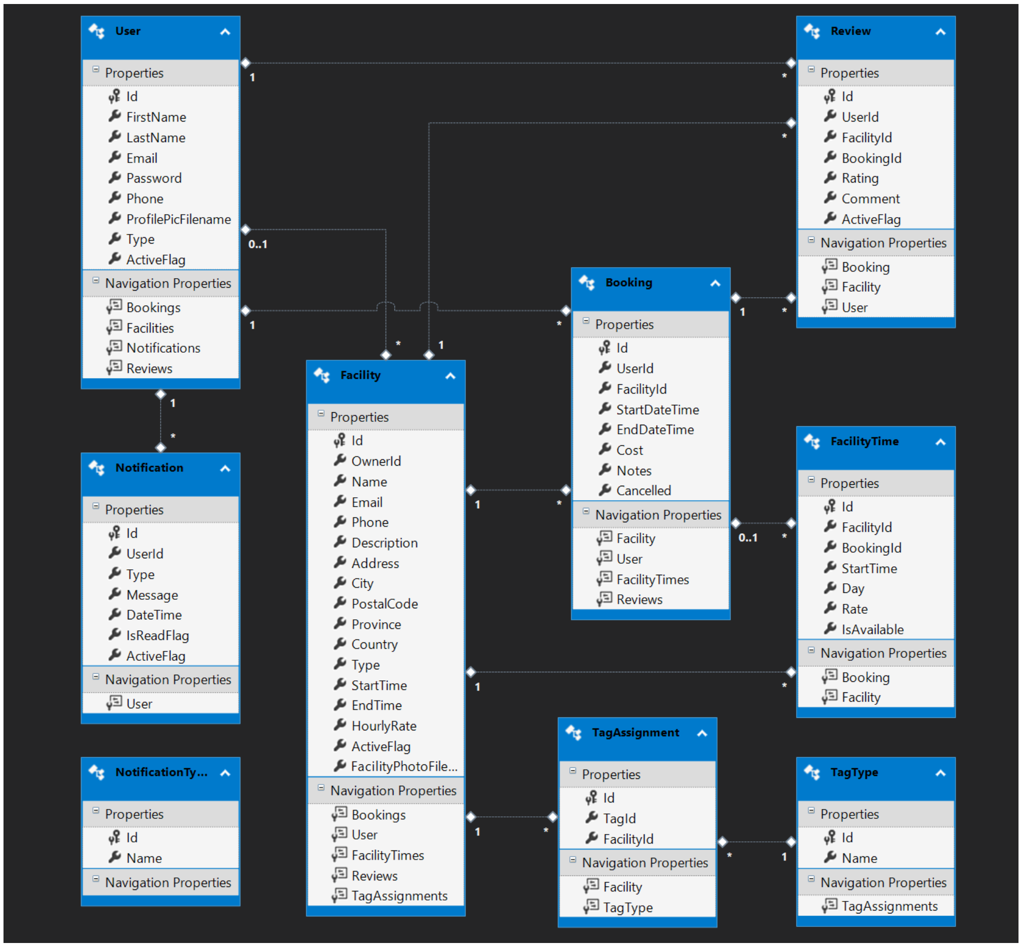


Figure 4.0: Database Tables Diagram

The data for the application is stored in a SQL Server Database. Classes are automatically generated based on the database tables. The diagram above (Figure 4.1) shows the current database tables and columns and the relationships between the tables.

In general, all database objects have a unique ID and most have an active flag. The unique ID of each object is used to reference and identify that object. Active flags are used to essentially delete objects from the database without having to actually remove them to avoid any issues with missing records. Objects that are provided with active flags are those that may need to be deleted in the future such as users and facilities.

4.1 Facility and Facility Time

Objects of type “Facility” hold the basic information about a given facility. It is provided by the facility owner/manager upon posting the facility. It was decided that contact information would be included with in the basic information about the facility since a facility’s contact information may be different from that of the user that created it. “Facility Time” objects are used to indicate weekly availability of a specific time slot in a given day. Combined with bookings, these are used to display to users times available to be booked

4.2 Tag Assignment and Tag Type

“Tag Type” objects describe various details not included in the basic information about facilities such as number of change facilities, if any. These are used to associate various details with a facility when displaying facility details or searching for facilities with specific characteristics. “Tag Assignment” objects simply assign relevant tags to a given facility. The reason the tag method was chosen to categorize additional details about facilities was to allow maximum flexibility when describing a facility. This accommodates a wide variety of types of athletic facilities that the application can cater to.

4.3 User

“User” objects correspond to user accounts, containing the login information for a given user. In addition to that, these objects also contain various user details that are provided by the user upon registration. These details are displayed on the user’s profile page and can be modified after registration as long as the user is logged in.

4.4 Notification and Notification Type

“Notification” objects contain the details about a given notification served to the user. They also keep track of whether the notification has been read or not. “Notification Type” objects classify a notification based on what triggered the notification (new booking, cancellation, etc.). The type of notification will determine the template used when providing the details of the notification to the user.

4.5 Booking

“Booking” objects represent specific time periods that have been booked by a user at a facility. These objects provide detailed start and end date/time information along with the total cost for the booking which is calculated once the booking is confirmed based on the hourly rate at the date/time of the booking as specified by the facility manager.

4.6 Review

“Review” objects contain review information for facilities. A user can rate and comment on their experience at a facility. Users can only review a facility in the 30 days after a booking has passed. Reviews are displayed on the facility details page for a given facility.

5 Test Plan and Execution

5.1 Objectives

For the testing of our application we intend on using Visual Studio’s integrated testing tools. Visual Studio’s integrated testing tools will be used for the unit testing. Our testing strategy is to perform acceptance tests, usability tests and unit tests for all actions in our controller.

For acceptance testing we have written acceptance tests based off of our previously defined user stories. These tests will be pass / fail and will be performed manually. For usability testing we plan on getting volunteers to perform a variety of tasks using our application. We will use our acceptance tests to direct what tasks we ask our volunteers to perform. We may collect some quantitative data from our usability testing such as how long each task takes, but the primary goal of our usability testing is to gather qualitative data to help refine our front end.

5.2 Acceptance Tests

Login/Logout:

Users that have an existing SpaceBook account will be able to login to the application with their credentials and logout when they are finished using the application.

Test Criteria	Failed	Passed
[Positive] Given that a user enters the correct credentials and clicks the login button, that user will be logged in to their account.		*
[Positive] Given that a user successfully logs in, the user will be redirected to the home page.		*
[Negative] Given that a user clicks the login button without filling out all the fields, the user will not be logged into the site.		*

[Negative] Given that a user enters the incorrect credentials and clicks the login button, the user will not be logged into the site.		*
[Positive] Given that a user is logged in, if the user clicks the logout button, they will be logged out.		*
[Positive] Given that a user successfully logs out, the user will be redirected to the login page.		*

Recover Password:

Users will be provided a method of recovery if they have forgotten their password.

Test Criteria	Failed	Passed
[Positive] Given the user clicks the password recovery link, they will be redirected to the password recovery page.		*
[Positive] Given the user provides their account email address their password will be reset.		*

[Negative] Given the user fails to fill out their email address and clicks the password reset button, the user will not be sent a password recovery email		*
[Positive] Given the user has entered an existing email address, they will receive a password recovery email.		*
[Positive] Given the user has reset their password, they will be able to login with the given credentials.		*

Registration:

Users will be able to register for a SpaceBook account that will allow them to login to the application.

Test Criteria	Failed	Passed
[Positive] Given a user has filled out all the necessary fields and clicks the register button, a new user account with the given details and credentials will be created.		*
[Negative] Given a user has not filled out all the necessary fields and clicks the register button, an account will not be created.		*

[Positive] Given a user has filled out all the necessary fields and clicks the register button, the user will be redirected to the login page.		*
[Positive] Given a user has successfully created a new account, they will be able to login with the given credentials.		*

Browse Facilities:

Users will be able to browse a list of available facilities filtered by a set of criteria such as facility type or features.

Test Criteria	Failed	Passed
[Positive] Given the user has entered a set of criteria, they will be able to view a list of facilities relevant to that criteria.		*

Location/Map Functionality:

Users will be able to view a facility’s location on a map.

Test Criteria	Failed	Passed

[Positive] Given the user is viewing a specific facility’s details, the user will be able to view the facility location on a map.		*

Post a Facility:

Facility owners and managers will be able to post a facility’s details and availability.

Test Criteria	Failed	Passed
[Positive] Given a facility owner/manager has filled out all the necessary fields and clicks the post facility button, a new facility with the given details and credentials will be created.		*
[Negative] Given a facility owner/manager has not filled out all the necessary fields and clicks the post facility button, a facility will not be created.		*
[Positive] Given a facility owner/manager has posted a new facility, the facility will be able to manage that facility.		*
[Positive] Given a facility owner/manager has successfully created a new facility, it will be available for viewing by other users.		*

View Facility Details:

Users will be able to view a facility’s details and availability.

Test Criteria	Failed	Passed
[Positive] Given a user clicks on a facility details link, they will be redirected to the facility details page for a given facility.		*
[Positive] Given a user clicks on a view availability link, they will be redirected to the facility availability page for a given facility.		*

Edit Facility Details:

Facility owners and managers will be able to edit facility postings.

Test Criteria	Failed	Passed
[Positive] Given a facility exists and a given user is the owner/manager of that facility, then that user will be able to edit the facility details for that facility.		*

[Negative] Given a facility exists and a given user is not the owner/manager of that facility, then that user will not be able to edit the facility details for that facility.		*

Browse Facility Availability:

Users will be able to view facility availability for a given date and time.

Test Criteria	Failed	Passed
[Positive] Given the user has selected a date, the user will be able to view the available time slots for that date.		*

Rate a Facility:

Users will be able to rate a facility after using it.

Test Criteria	Failed	Passed
[Positive] Given a user has had a booking at a given facility within the past 30 days, that user can rate and comment about their experience with the given facility.		*

[Negative] Given a user has not had a booking at a given facility within the past 30 days, that user cannot rate and comment on the facility.		*

View User Profile:

A user profile page will contain details about a given user. Users will be able to view their user profiles.

Test Criteria	Failed	Passed
[Positive] Given a user has clicked on a their user profile link, the user will be redirected to their user profile page.		*

Edit User Profile:

Users will be able to customize and edit their user profiles.

Test Criteria	Failed	Passed
[Positive] Given a user is logged in to their account, then that user will be able to edit the user details for their account.		*

Book a Facility:

Users will be able to book an available facility at a specific date and time.

Test Criteria	Failed	Passed
[Positive] Given a user specifies a specific date/time and facility and that date/time period is available for the given facility, then the user will be able to book the specified date/time at that facility.		*
[Negative] Given a user specifies a specific date/time and facility and that date/time period is unavailable for the given facility, then the user will not be able to book the specified date/time at that facility.		*

View Upcoming Bookings:

Users and facility owners and managers will be able to view upcoming bookings.

Test Criteria	Failed	Passed
[Positive] Given a user has booked at least one facility in the future, they will be able to view upcoming bookings.		*
[Positive] Given a facility has at least one booking in the future, the facility owner/manager will be able to view upcoming bookings at that facility.		*

Cancel a Booking:

Users and facility owners and managers will be able to cancel upcoming bookings.

Test Criteria	Failed	Passed
[Positive] Given that a booking exists, a user or facility owner/manager will be able to cancel that booking.		*

Alerts and Notifications:

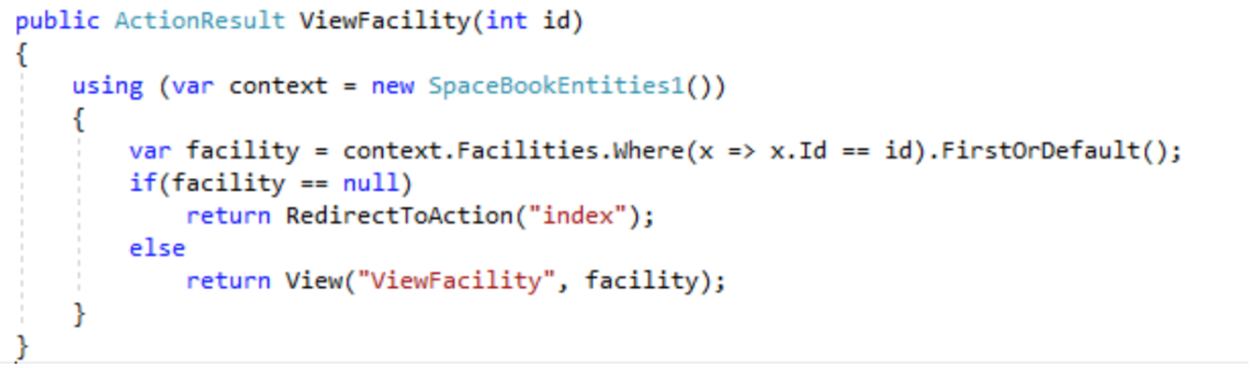
Users will receive alerts and notifications for important events such as reminders of an upcoming booking or a booking cancellation.

Test Criteria	Failed	Passed
[Positive] Given an event triggers a notification, the user will be able to see that the count of unread notifications increased by 1.		*
[Positive] Given the user clicks on the notification link, the user will be able to read all their notifications		*

[Positive] Given the user dismisses a notification, the user will be able to see that the count of unread notifications decreased by 1.		*

5.3 Unit Tests

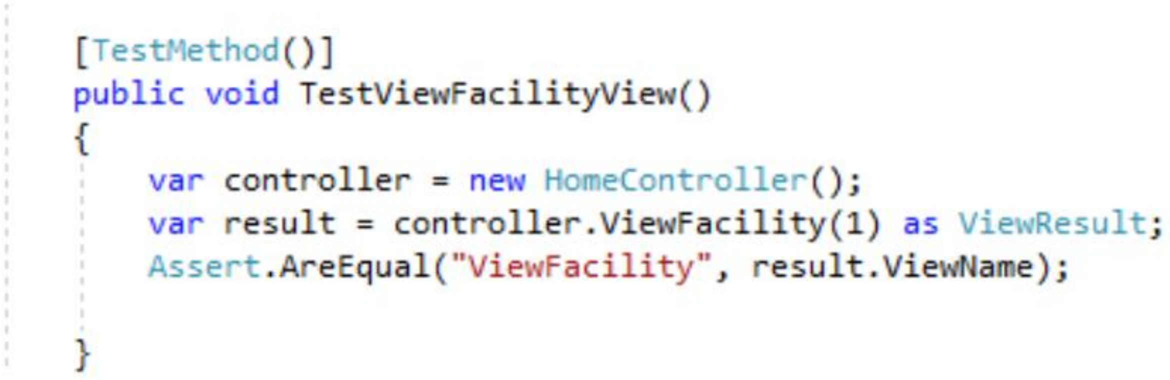
There are three types of unit tests we intend to carry out. The first test type is a test which simply asserts that an action returns the correct view. The second test type is a test which asserts that an action passes the correct view data to the corresponding view. The third test type will test that an action calls the correct action under certain circumstances. Please note that all three of our testing types will not be applicable to every single action; however, at least one of our testing types will be used for every action in our application. To further understand our testing strategy please see the action below (Figure 5.1).



```
public ActionResult ViewFacility(int id)
{
    using (var context = new SpaceBookEntities1())
    {
        var facility = context.Facilities.Where(x => x.Id == id).FirstOrDefault();
        if(facility == null)
            return RedirectToAction("index");
        else
            return View("ViewFacility", facility);
    }
}
```

Figure 5.3.1: View Facility action

This particular action will be called whenever a user attempt to view an individual facility using our application. This serves as a good example because all three of our test types are applicable to this action. The first test we would run would assert that the correct view is returned when this action is called. In this case, the correct view to be returned would be the “ViewFacility” view which also happens to share the same name as the action. This first test would look something like the test shown below (Figure 5.2).

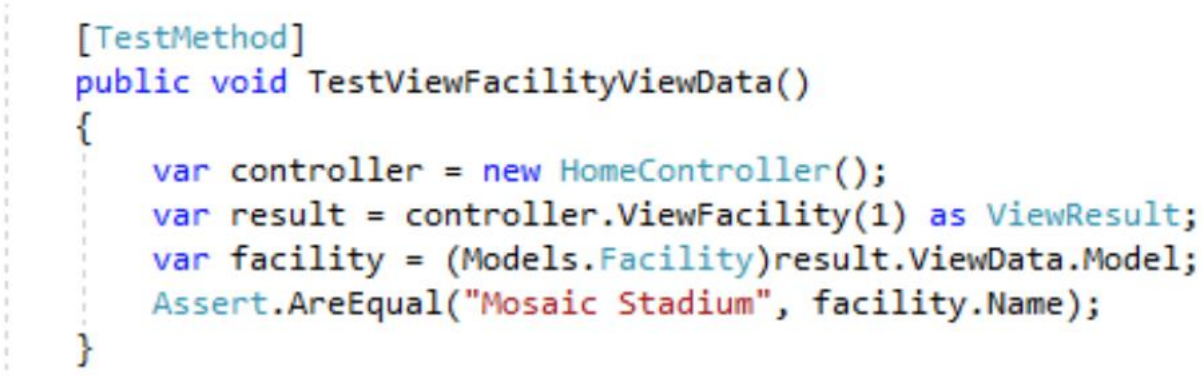


```
[TestMethod()]
public void TestViewFacilityView()
{
    var controller = new HomeController();
    var result = controller.ViewFacility(1) as ViewResult;
    Assert.AreEqual("ViewFacility", result.ViewName);

}
```

Figure 5.3.2: Test Correct View

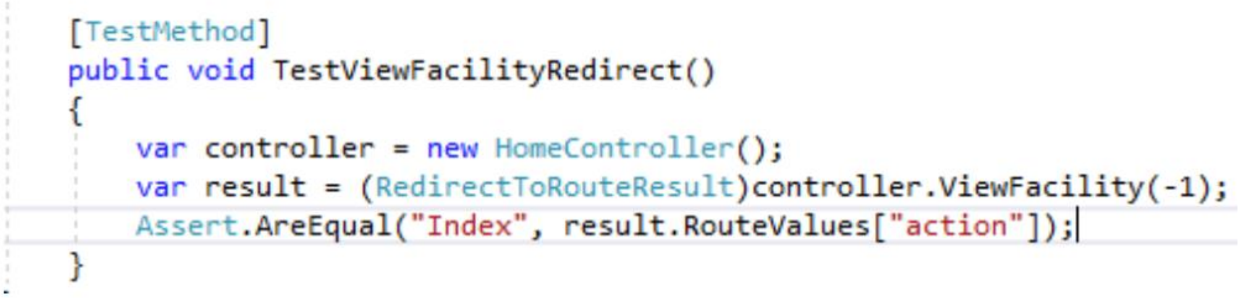
The second test type that would be applicable to this action, would be a test which asserts that the view data passed to the “ViewFacility” view is correct. This would be accomplished with a test like the test shown below



```
[TestMethod]
public void TestViewFacilityViewData()
{
    var controller = new HomeController();
    var result = controller.ViewFacility(1) as ViewResult;
    var facility = (Models.Facility)result.ViewData.Model;
    Assert.AreEqual("Mosaic Stadium", facility.Name);
}
```

Figure 5.3.3: Test Correct View Data

The third test type applicable to this action, would be a test which asserts that the action calls the correct resulting action. Please notice how the “Index” action is called in the “ViewFacility” action if the result of the query is null. This particular case will only occur if a user attempts to access the “ViewFacility” view without passing the action a valid facility ID, but this is still a case that we will need to test to ensure our application is robust. We would test this with a unit test just like the one shown below.



```
[TestMethod]
public void TestViewFacilityRedirect()
{
    var controller = new HomeController();
    var result = (RedirectToRouteResult)controller.ViewFacility(-1);
    Assert.AreEqual("Index", result.RouteValues["action"]);
}
```

Figure 5.3.4: Test Correct Redirect

In addition to our unit test, acceptance testing and usability testing will be used to do manual testing. Overall the variety of testing methods we will use will ensure that our application is robust.

5.4 Test Execution

All unit tests and acceptance tests that were designed were confirmed to pass successfully. Usability testing was done less formally but feedback from volunteer users was taken into consideration throughout the development process to better cater the front end design of the application to the typical user.

6 User Manual

6.1 User Registration

1. From the login screen, click the link to register for the site.

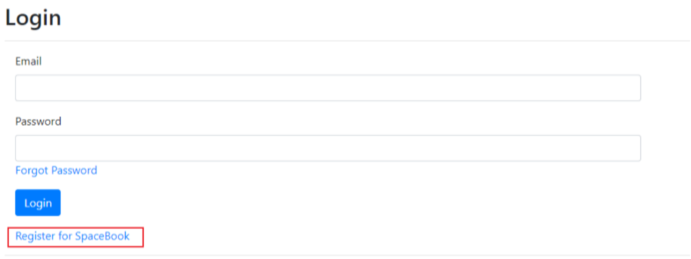


Figure 6.1.1: Registration link

2. Fill out each field with the required information.

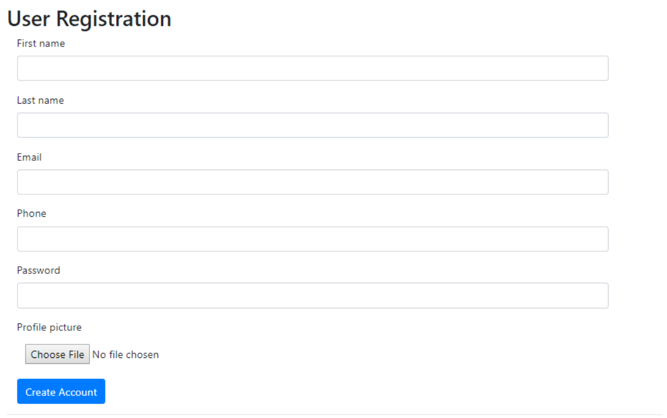


Figure 6.1.2: Registration form

3. Optionally, upload a photo of yourself to use on your profile.
4. Submit the registration for by clicking “Create Account” .

6.2 Login

1. On the login screen, enter your email and password into the provided fields.

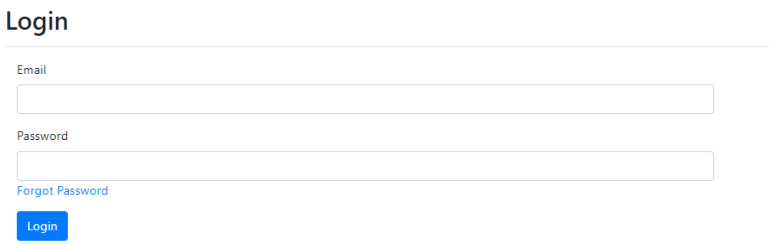


Figure 6.2: Login form

2. Click the “Login” button.

6.3 Logout

1. Login with an existing user account.
2. From the home page, click the logout option from the navigation bar at the top of the window.

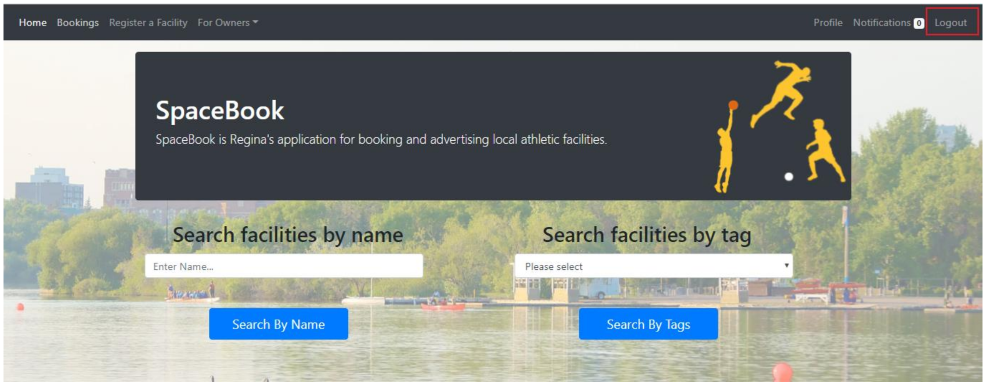


Figure 6.3: Logout navigation link

6.4 Register a Facility

1. Login with an existing user account.
2. From the home page, click the register a facility option from the navigation bar at the top of the window.

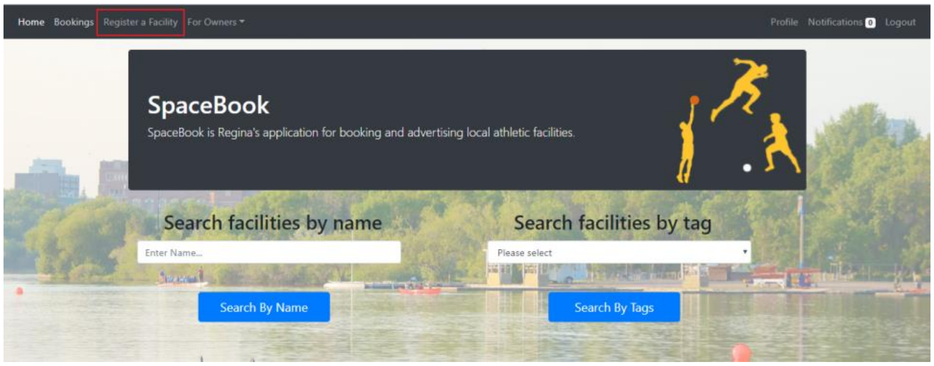
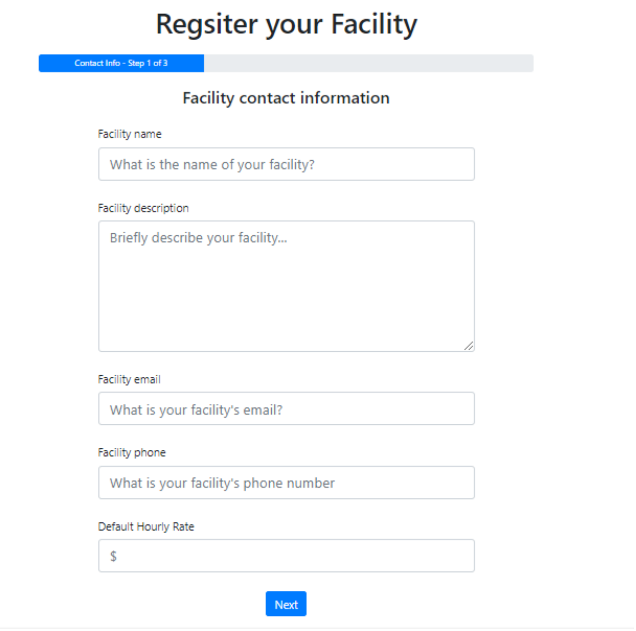


Figure 6.4.1: Facility registration navigation link

3. Fill in the contact information as it pertains to your venue in the fields provided.



Regsiter your Facility

Facility contact information

Facility name

What is the name of your facility?

Facility description

Briefly describe your facility...

Facility email

What is your facility's email?

Facility phone

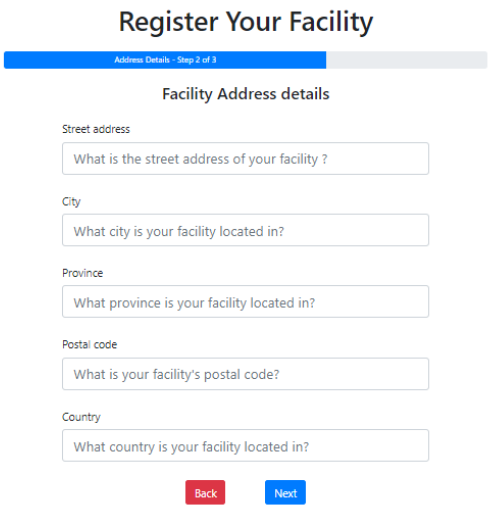
What is your facility's phone number

Default Hourly Rate

$

Figure 6.4.2: Facility registration contact information

4. Press next and fill out the address information in the fields provided.



Register Your Facility

Facility Address details

Street address

What is the street address of your facility ?

City

What city is your facility located in?

Province

What province is your facility located in?

Postal code

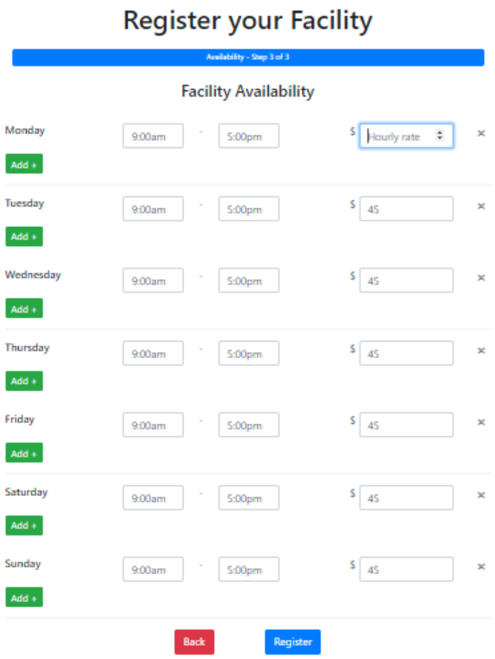
What is your facility's postal code?

Country

What country is your facility located in?

Figure 6.4.3: Facility registration address information

5. Press next and you will be brought to the facility availability page. Set the hours your facility is available for each day of the week. Make sure to also set the hourly rate for each set of time slots.



Register your Facility

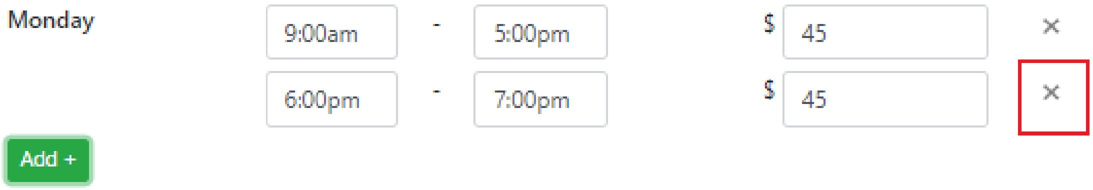
Availability - Step 3 of 3

Facility Availability

Monday	9:00am	-	5:00pm	$ Hourly rate	×
Add +					
Tuesday	9:00am	-	5:00pm	$ 45	×
Add +					
Wednesday	9:00am	-	5:00pm	$ 45	×
Add +					
Thursday	9:00am	-	5:00pm	$ 45	×
Add +					
Friday	9:00am	-	5:00pm	$ 45	×
Add +					
Saturday	9:00am	-	5:00pm	$ 45	×
Add +					
Sunday	9:00am	-	5:00pm	$ 45	×
Add +					

Figure 6.4.4: Facility registration availability

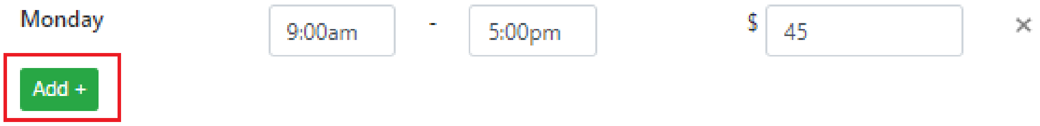
a. To remove a time slot in which the facility is available, press the X beside the set of time slots you wish to remove.



Monday	9:00am	-	5:00pm	$ 45	×
	6:00pm	-	7:00pm	$ 45	×
Add +					

Figure 6.4.5: Delete time slot

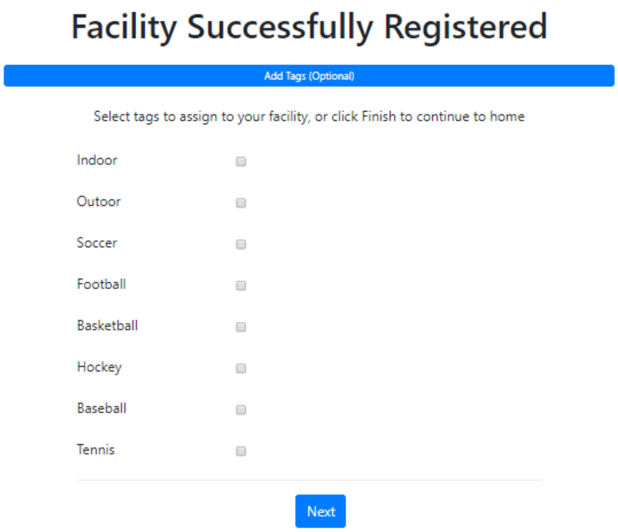
b. To add a set of time slots, press the add button and adjust the hours accordingly.



Monday	9:00am	-	5:00pm	$ 45	×
Add +					

Figure 6.4.6: Add time slot

6. Press next. Optionally add tag identifiers to associate with your facility. This will allow users to view amenities your facility has once posted.



Facility Successfully Registered

Select tags to assign to your facility, or click Finish to continue to home

Indoor □

Outoor □

Soccer □

Football □

Basketball □

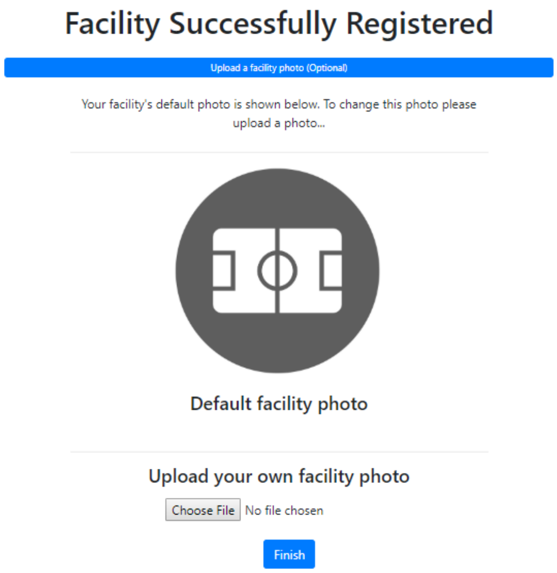
Hockey □

Baseball □

Tennis □

Figure 6.4.7: Add facility tags

7. Optionally, upload a photo of the facility to use on the facility profile



Facility Successfully Registered

Your facility's default photo is shown below. To change this photo please upload a photo...

Upload your own facility photo

Choose File No file chosen

Figure 6.4.8: Add facility photo

8. Submit your facility by clicking “finish”.

6.5 Edit a Facility

1. Login with an existing user account.
2. From the home page, click the “For Owners”option from the navigation bar at the top of the window, and select “View facilities”

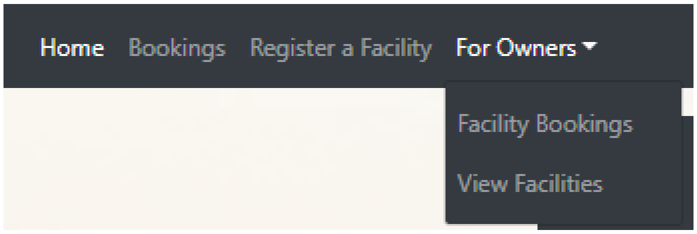
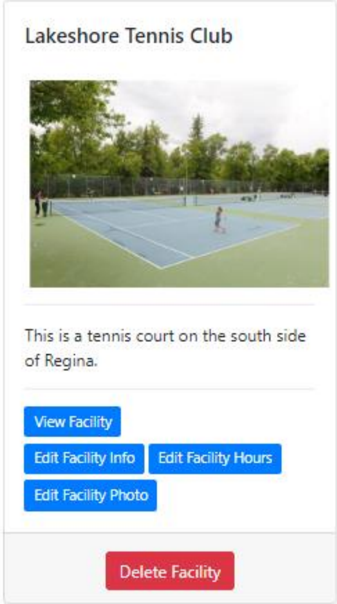


Figure 6.5.1: View your facilities

3. Find the facility of yours you wish to edit. Select whichever edit button pertains to the facility information you would like to edit.



Lakeshore Tennis Club

This is a tennis court on the south side of Regina.

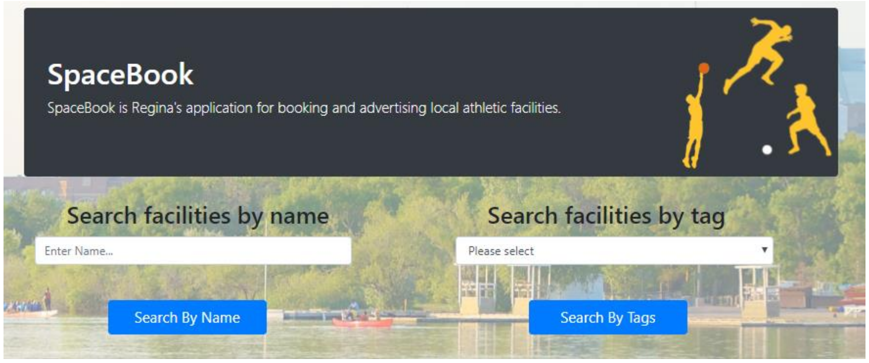
Figure 6.5.2: Edit facility options

4. Once you have edited any facility information, be sure to submit the form so that the changes are saved.

6.6 Search Available Facilities

1. Login with an existing user account.
2. From the home page, you will need to select a facility tag to search by or a facility name. Either enter the a facility name and click “Search By Name” or select a facility tag and click “Search By Tags”

3.



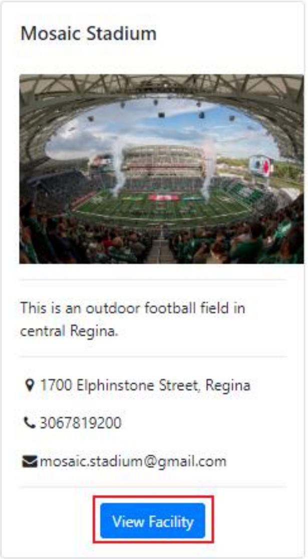
SpaceBook

SpaceBook is Regina's application for booking and advertising local athletic facilities.

Figure 6.6: Searching facilities

6.7 View Facility Details

1. Login with an existing user account.
2. From the home page, search for the facility you would like to view by searching by name or tags.
3. Once you have found the facility you would like to view, click the view facility button on your desired facility.



Mosaic Stadium

This is an outdoor football field in central Regina.

1700 Elphinstone Street, Regina

3067819200

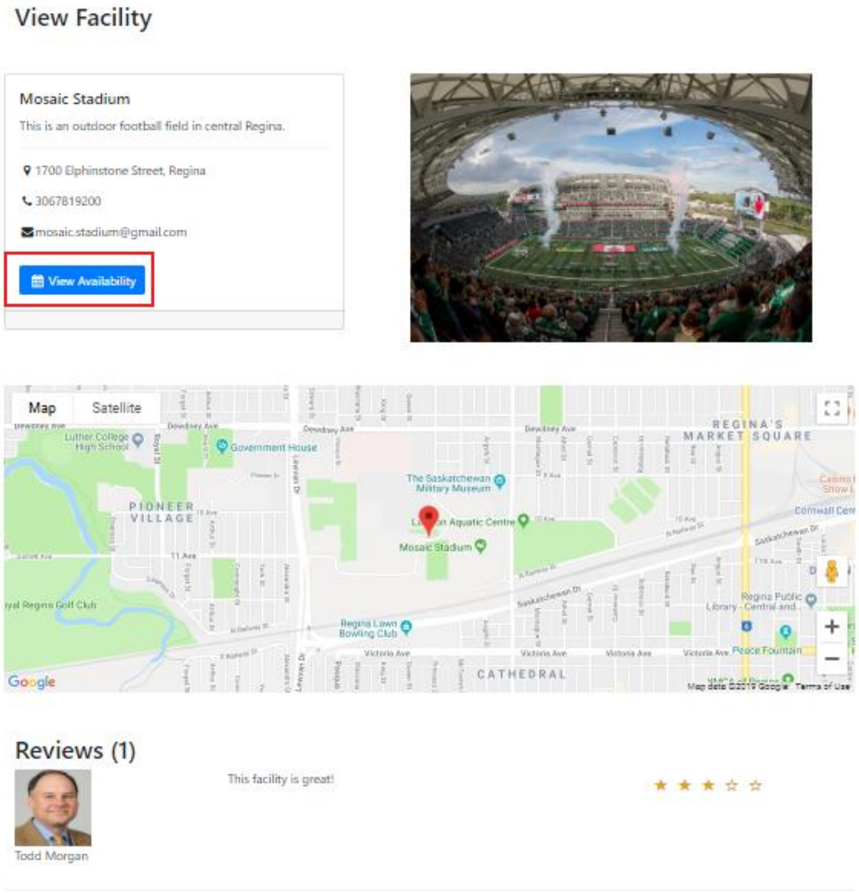
mosaic.stadium@gmail.com

Figure 6.7: View Facility

6.8 View Facility Availability and Schedule a Booking

1. Login with an existing user account.
2. From the home page, search for the facility you are looking for.
3. Click the view facility button on your desired facility.

4. Click the view facility availability button on the facility details page.



View Facility

Mosaic Stadium

This is an outdoor football field in central Regina.

1700 Elphinstone Street, Regina

3067819200

mosaic.stadium@gmail.com

Reviews (1)

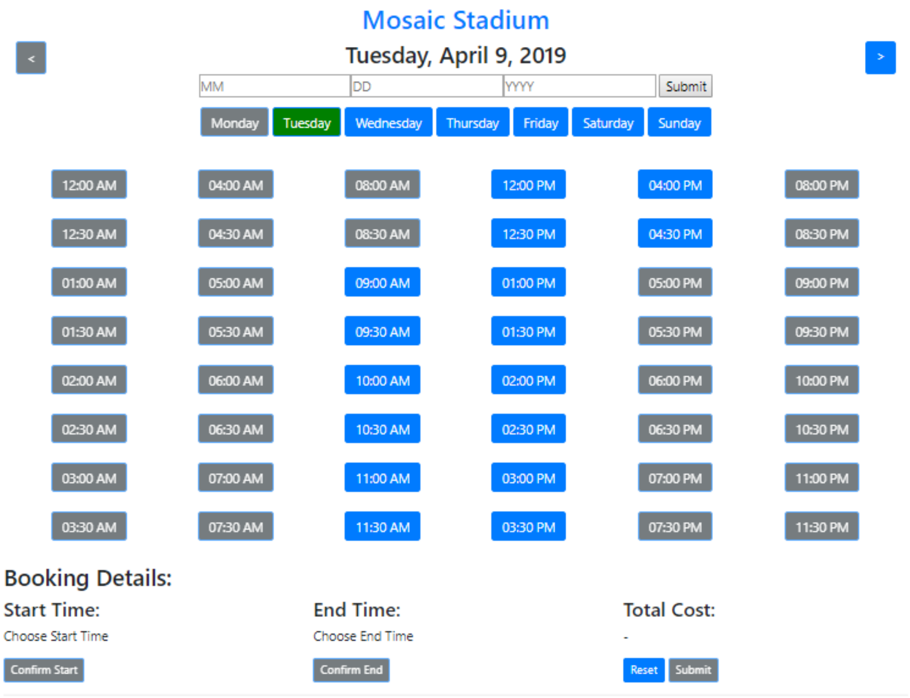
This facility is great!

★ ★ ★ ☆ ☆

Todd Morgan

Figure 6.8.1: View availability button

5. Select the desired week in which you would like to schedule your booking.



Booking Details:

Start Time: Choose Start Time

End Time: Choose End Time

Total Cost: -

Figure 6.8.2: Scheduling a booking

a. This can be done using the arrow buttons to navigate through the available weeks.
b. This can also be done by inputting a specific date and submitting it.

6. Click on your desired start time and click “Confirm Start”.
7. Click on your desired end time and click “Confirm End”.
8. Submit the booking by clicking submit ”Submit”.
    a. You can cancel the booking using the cancel button.

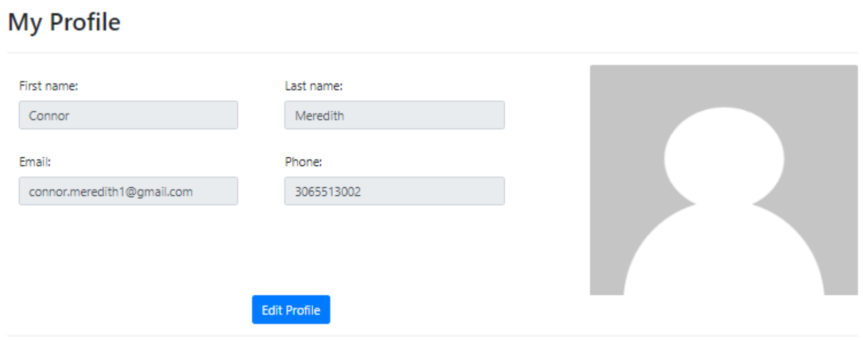
6.9 View User Profile

1. Login with an existing user account.
2. From the home page, click the profile option from the navigation bar at the top of the window.

6.10 Edit User Profile

1. Login with an existing user account.
2. From the home page, click the profile option from the navigation bar at the top of the window.

3. Click the “Edit Profile” button.



My Profile

First name: Connor

Last name: Meredith

Email: connor.meredith1@gmail.com

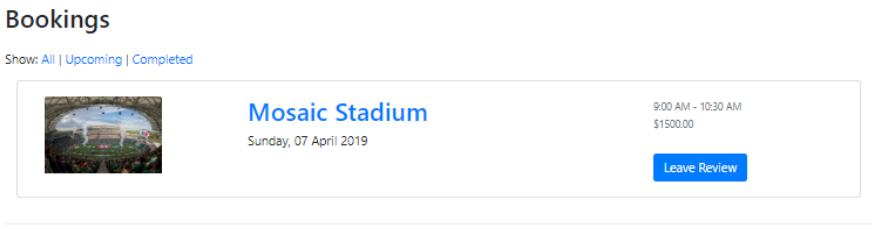
Phone: 3065513002

Figure 6.10: User profile

4. Edit the fields you wish to be changed.
5. Submit your changes.

6.11 Browse List of Existing Bookings as a User

1. Login with an existing user account.
2. From the home page, click the bookings option from the navigation bar at the top of the window.
3. Use the tabs up top to filter your bookings based off whether or not they have occured yet.



Bookings

Show: All | Upcoming | Completed

Mosaic Stadium

Sunday, 07 April 2019

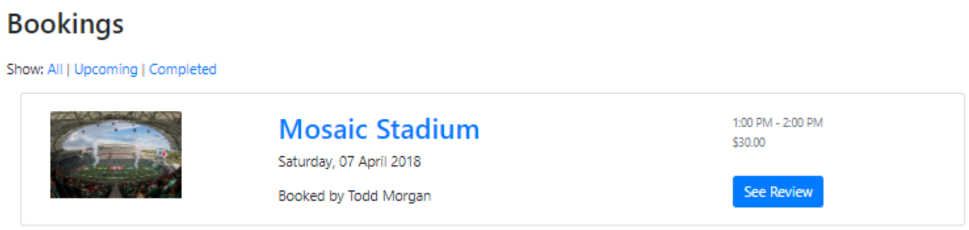
9:00 AM - 10:30 AM

$1500.00

Figure 6.11: View bookings

6.12 Browse List of Existing Bookings as a Facility Manager

1. Login with an existing user account.
2. From the home page, click the “For Owners”option from the navigation bar and then select “View Facilities”.
3. On the my facility bookings page, filter by your desired parameter to see upcoming r completed bookings.



Bookings

Mosaic Stadium

Saturday, 07 April 2018

Booked by Todd Morgan

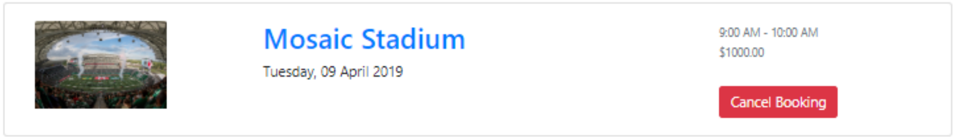
1:00 PM - 2:00 PM

$30.00

Figure 6.12: View my facility bookings

6.13 Cancel a Booking as a User

1. Login with an existing user account.
2. From the home page, click the bookings option from the navigation bar at the top of the window.
3. On the bookings page, filter by upcoming bookings.
4. Press the “Cancel Booking” button to cancel the booking.



Mosaic Stadium

Tuesday, 09 April 2019

9:00 AM - 10:00 AM

$1000.00

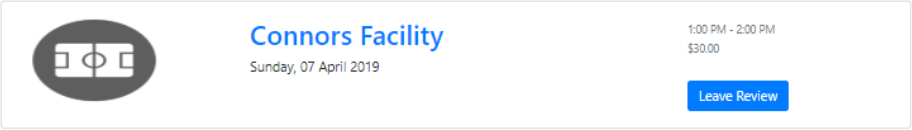
Figure 6.13: Cancel a booking

6.14 Cancel a Booking as a Facility Manager

1. Login with an existing user account.
2. From the home page, click the my facility bookings option from the navigation bar at the top of the window.
3. On the my facility bookings page, filter by upcoming bookings.
4. Press the cancel button and confirm to cancel the booking.

6.15 Review a Venue

1. Login with an existing user account.
2. From the home page, click the bookings option from the navigation bar at the top of the window.
3. On the bookings page, filter by “Completed” bookings.
4. Click “Leave Review” on a booking pertaining the facility you would like to review.



Connors Facility

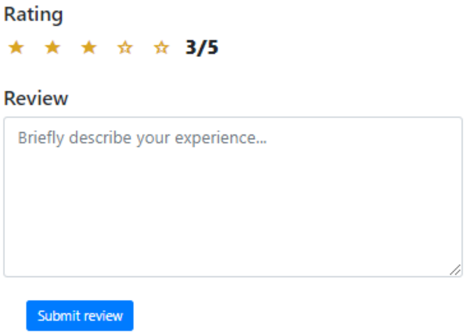
Sunday, 07 April 2019

1:00 PM - 2:00 PM

$30.00

Figure 6.15.1: Leave a review

5. Enter a star rating for the review.



Rating

★ ★ ★ ☆ ☆ 3/5

Review

Briefly describe your experience...

Figure 6.15.2: Review page

6. Optionally, write a review in the text field provided.
7. Submit the review.

6.16 View Notifications

1. Login with an existing user account.
2. From the home page, click the notifications link from the navigation bar at the top of the window. This link will also display the number of notifications you have.

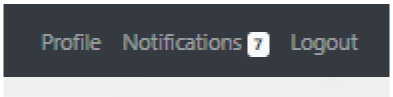
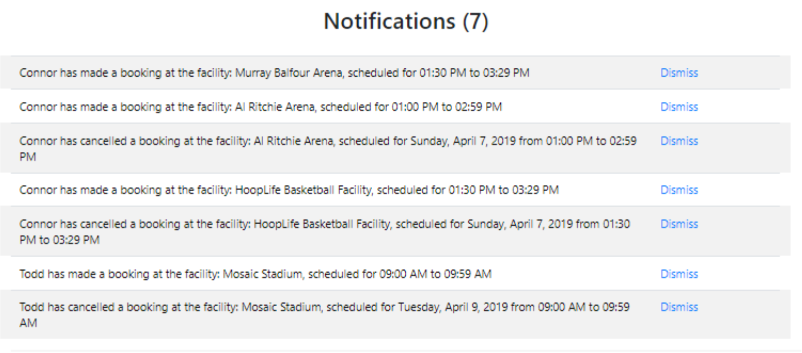


Figure 6.16.1: Notifications on navigation bar

3. Once you are on your notifications view, you can dismiss notifications by clicking the “Dismiss” link beside each notification.



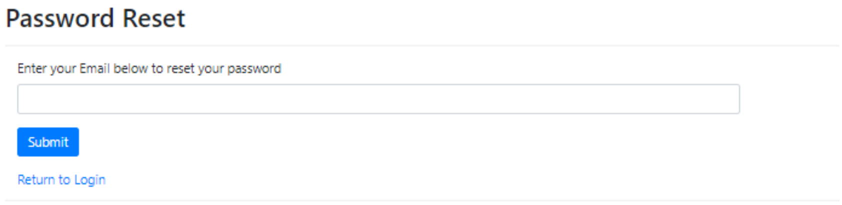
Notifications (7)

Connor has made a booking at the facility: Murray Balfour Arena, scheduled for 01:30 PM to 03:29 PM	Dismiss
Connor has made a booking at the facility: Al Ritchie Arena, scheduled for 01:00 PM to 02:59 PM	Dismiss
Connor has cancelled a booking at the facility: Al Ritchie Arena, scheduled for Sunday, April 7, 2019 from 01:00 PM to 02:59 PM	Dismiss
Connor has made a booking at the facility: HoopLife Basketball Facility, scheduled for 01:30 PM to 03:29 PM	Dismiss
Connor has cancelled a booking at the facility: HoopLife Basketball Facility, scheduled for Sunday, April 7, 2019 from 01:30 PM to 03:29 PM	Dismiss
Todd has made a booking at the facility: Mosaic Stadium, scheduled for 09:00 AM to 09:59 AM	Dismiss
Todd has cancelled a booking at the facility: Mosaic Stadium, scheduled for Tuesday, April 9, 2019 from 09:00 AM to 09:59 AM	Dismiss

Figure 6.16.2: View and dismiss notifications

6.17 Reset Forgotten Password

1. From the login page, click the forgot password button.
2. Enter the email associated with your account and select “Submit”



Password Reset

Enter your Email below to reset your password

Figure 6.17: Reset password

3. If there is an account associated with the provided email, an email will be sent to reset the password.
4. Use the temporary password to login to the account.
5. Change the password.

7 Future Work:

This section will address some future work and potential future features which could be useful for this application.

- In-app payment
  - Currently all bookings are associated with a cost, but there are not actually any transactions taking place within our application. It is just assumed that users are paying facility owners the amount of money they owe them.

    - It would be useful if the application could also process payments as this would make for a more convenient booking experience.
    - This would also reduce the hassle for facility owners as they would have to put in less effort towards collecting the money for any bookings at their facility.
- Comparing facilities and their availability
    - Currently users can compare facilities just by finding the facilities they would like to compare and then just looking at them each individually.
    - It would be useful if instead users could select multiple facilities at a time and compare them based off criteria such as their reviews, their availability, or their location.
    - Most of this can be achieved just by switching back and forth between facilities, but it would just be more convenient if users could just view all this information at once.
- More filtering and sorting
    - Currently the application supports filtering and searching using tags and facility names, but it would be useful if users could filter, sort, and search based off more criteria. This criteria includes things such as:
        - Sorting facilities by proximity to a user
        - Sorting or filtering facilities by average hourly rate
        - Sorting or filtering facilities by average rating
        - Filtering facilities based off availability
- Email and phone notifications
    - Currently the application delivers notifications to users through the application itself, but users may not always be logged into the application or in a position where they can check their notifications within the application.
    - It would be more useful to users if they could receive notifications through more reliable communication channels such as phone or email.

8 Business Case

8.1 Developer Budget

The total time spent developing and implementing this project would be comprised of all meeting minutes between developers as well as all hours spent individually coding. This also includes time spent working on documentation and preparing for bi-weekly presentations from January onwards. As described in the log book document, the total time spent in meetings was 91.5 hours when considering each developer separately. The meetings were split between project management and architectural design. The time spent writing and testing code is estimated to be 60 hours per developer for a total of 180 hours. Assuming a hourly pay of $30 per hour, the total cost for 271.5 hours is $8145.

8.2 Expenses

If this application were to be deployed in a production environment, web hosting server and database storage costs, as well as any other maintenance costs would need to be taken into account. Upgrading to a standard Microsoft Azure hosting plan would cost $122 per month. The application uses a Google API for location based services, which currently is free as the traffic of the application is below the maximum free threshold. The application can handle 25000 map load requests per day. Once this threshold is exceeded, map loads would cost $0.50 USD per 1000 map load requests. The Google API is also used to automatically fill out address information. 2500 free requests are allowed per day. If exceeding this threshold, it would cost $0.50 USD per an additional 1000 requests. General application and database maintenance could be estimated at about 8 additional developer hours per month.

8.3 Potential Yearly Business Models

Over the course of the first year, we would be looking to break even which would require that the application generates approximately $820 per month. This accounts for the development, hosting, and maintenance costs. After looking at a number of existing booking software systems we have concluded that these are the most suitable approaches to take.

Flat Rate Pricing

For this pricing model, we would charge facility owners a monthly fee of $15 to advertise their facilities for booking. This would allow for an unlimited amount of facilities to be advertised by an owner. This plan would be ideal for owners with multiple facilities to advertise as there is no additional costs for additional facility postings. To break even with this model, we would require about 55 paying facility owners.

Incremental Pricing

For this model, we would charge a base fee of $5 per month. Paying facility owners would be entitled to list one facility. Each additional facility posting would cost $2 per month. This model is better suited if the majority of facility owners are only advertising a single facility. If the average owner was posting 2 facilities, this would cost $7 per month. To break even with this model, we would require about 118 paying facility owners.

Hybrid pricing

This model would offer incremental pricing for users with a smaller amount of facilities to post, or flat rate pricing for users wanting to post more than six facilities. To break even with this model, a combination of 25 users paying the flat rate fee of $15 per month, and 64 users paying the incremental fee for two facilities which would be about $7 per month.

9 Conclusion

The initial scope of the project was narrowed during development from a general facility booking application to a booking application focused on athletic facilities. Given this scope, the project was a success. By April 6, 2019 (Project Day), the following key aspects of the project were delivered upon:

- A platform was created to allow facility managers to advertise facilities available for booking to interested users
- Facility managers can define their hours in a flexible way
- Facility managers can define different rates for different ranges of hours
- Users can access information about facilities quickly and easily
- Users can view available times at a facility for a given day
- Users can view the rate for specific time slots and calculate the total cost of a booking
- Users and facility managers can view and manage bookings through the site

The project allowed for the application of concepts learned throughout the course of our degrees. We were able to apply our acquired knowledge of the software development cycle, project management and programming concepts to the project, resulting in a successful end product.