**Tee Time Reservation System**

|  |  |
| --- | --- |
| **Capstone Proposal Project Name:** | Tee Time Reservation System |
| **Student Name:** | Brennen Ellison |

Table of Contents

[Application Design and Testing 4](#_Toc187342001)

[Class Design 4](#_Toc187342002)

[UI Design 6](#_Toc187342003)

[Unit Test Plan 8](#_Toc187342004)

[Introduction 8](#_Toc187342005)

[Purpose 8](#_Toc187342006)

[Overview 8](#_Toc187342007)

[Test Plan 8](#_Toc187342008)

[Items 8](#_Toc187342009)

[Features 9](#_Toc187342010)

[Deliverables 9](#_Toc187342011)

[Tasks 9](#_Toc187342012)

[Needs 10](#_Toc187342013)

[Pass/Fail Criteria 10](#_Toc187342014)

[Specifications 11](#_Toc187342015)

[Procedures 13](#_Toc187342016)

[Results 14](#_Toc187342017)

[Hosted Web Application Details 17](#_Toc187342018)

[Hosted Web Application Link: 17](#_Toc187342019)

[User Guide for Installation 17](#_Toc187342020)

[Introduction 17](#_Toc187342021)

[Installation and Using the Application 17](#_Toc187342022)

[Pre-requisites 18](#_Toc187342023)

[Okta Application Set up 18](#_Toc187342024)

[Docker setup 19](#_Toc187342025)

[Repository setup 20](#_Toc187342026)

[Container setup 20](#_Toc187342027)

[Debugging Container setup 20](#_Toc187342028)

[User Guide for Operating the Application 21](#_Toc187342029)

[Introduction 21](#_Toc187342030)

[Tee-Time Search 21](#_Toc187342031)

[Tee-Time Reservation 23](#_Toc187342032)

[Login 26](#_Toc187342033)

[Tee-time Editing and Cancelation 26](#_Toc187342034)

# Application Design and Testing

## Class Design

The diagram above is a representation of all entity classes within the Tee Time Reservation System (TTRS) Java Spring Boot application. The entities create, update, and delete methods are automatically implemented through the usage of the Java Hibernate framework and do not appear in this diagram. Additional business logic is handled through a “service” implementation for each entity, which abstracts these operations into dedicated methods. As shown in the second image, service implementation objects are derived from an interface, ensuring flexibility and modularity. This modular design enhances the application’s structure and supports future growth and development.

A screenshot of a computer program

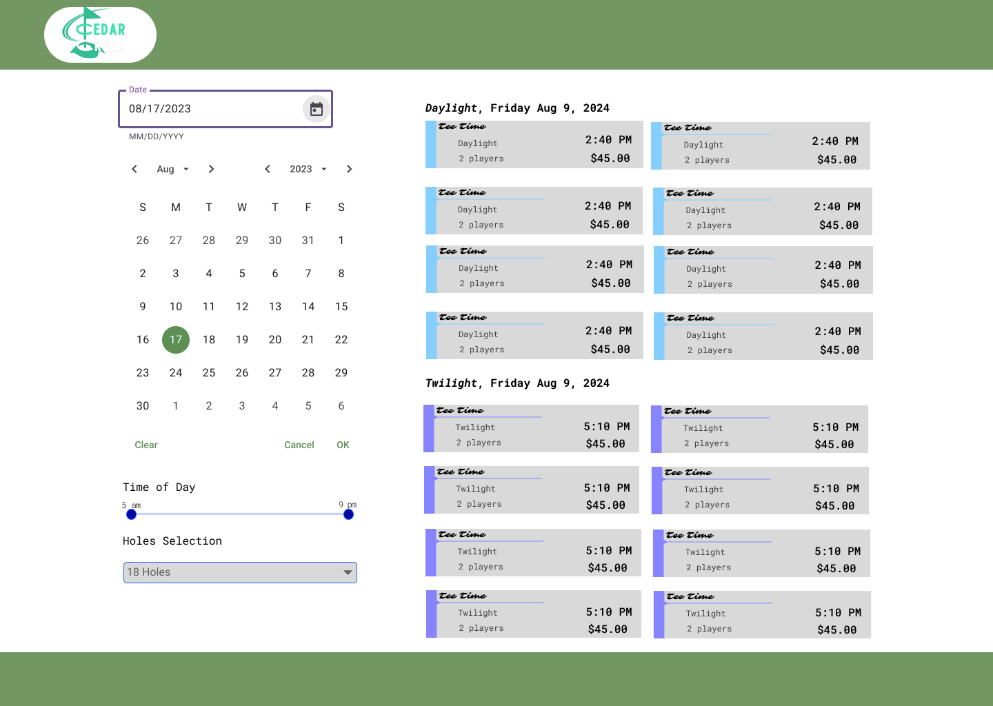
Description automatically generated

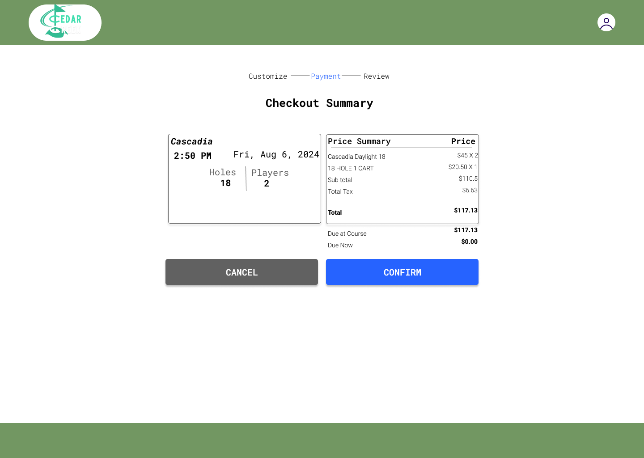
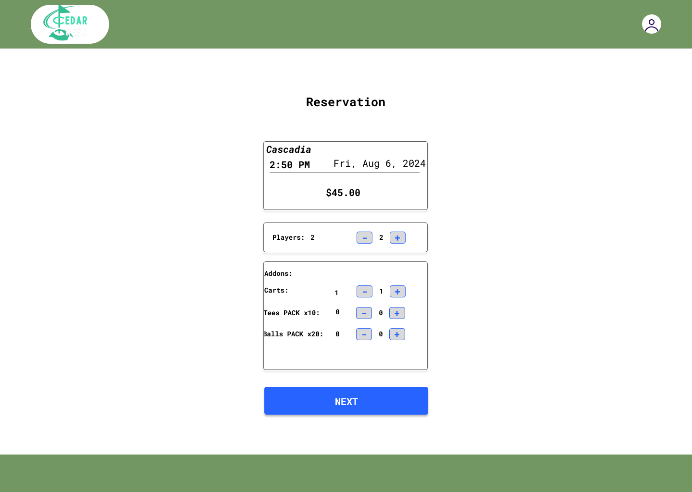
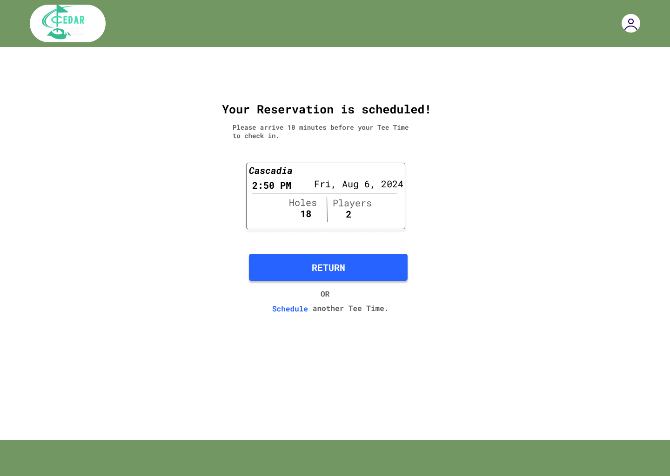
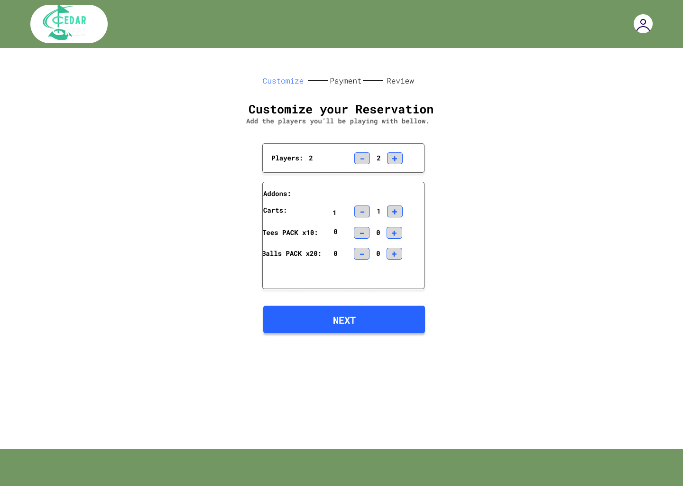
## UI Design

The TTRS User Interface (UI) is designed to provide users with an intuitive and easy experience, enabling them to search for and reserve tee times efficiently. Upon accessing the application, users are immediately presented with the option to search and sort through a list of available tee times. This structure ensures that users can promptly view available options and proceed with their reservation.

Once a tee time is selected, users are guided through a process to provide personal information and select optional addon-ons. These add-ons provide the business with an opportunity to offer additional merchandise.

While users have the option to create an account, it is not mandatory to complete a reservation. However, registered users can review, edit, or cancel existing tee time reservations. This approach ensures new customers can engage with the system and complete reservations without barriers.





# 

# Unit Test Plan

## Introduction

### Purpose

The purpose of the unit test plan is to document and strategize the testing of components and ensure the TTRS application upholds proper functionality.

### Overview

These unit tests consist of the most critical aspects of the application such as tee-time searching and generation, adding, updating, and deleting tee-time reservations, and user authentication.

* Tee-Time management: Tests for tee-time management involve testing the process of tee-time generation and searching.
* Reservation Management: Tests for reservation management contain testing for adding, updating and deleting reservations from the database.
* User Authentication: Involves end to end testing to ensure a user can properly log into the application, unauthorized users are blocked from certain aspects of the application and verifying error messages display correctly.

## Test Plan

### Items

* Development Environment: A development environment with NodeJS installed.
* Database: The application requires a MySQL database with the proper schema configured.
* Testing Frameworks: Cypress for E2E testing.
* Application source code: Test scripts and application code provided in the GitLab repository.

### Features

* Tee-Time Management: Testing tee-time generation upon searching filters and booking status.
* Reservation Management: Testing the ability to add, update, and delete tee-time reservations created by the user.
* User-authentication: Test user’s ability to login to the application with proper and false credentials and access control to protected routes from unauthorized users.

### Deliverables

* Test Scripts: A collection of test scripts written for the E2E testing with Cypress.
* Test Results: The test results for the given test scripts.

### Tasks

* Set up the development environment in NodeJs by installing the dependencies.
  + npm install
* Open the Cypress Interface by running this command within the NodeJs directory
  + npm run cypress:open
* Run each spec file by clicking the file in the Cypress Interface.
* Review test results and identify unexpected behavior.

### Needs

To perform the tests the application should be running locally on a development machine. This can be configured by either pulling the docker images from the general repository, or by setting up the environment locally by installing the bellow dependencies:

* Node.JS
* MySql
* Java
* Maven
* Angular & Angular CLI tool.

The testing tool required is Cypress, this is included as a development dependency within the Node.JS project and can be installed running “npm i” within a terminal window in the NodeJs project directory. Additionally, the tester will need access to the source code and test scripts for the application in the Git repository.

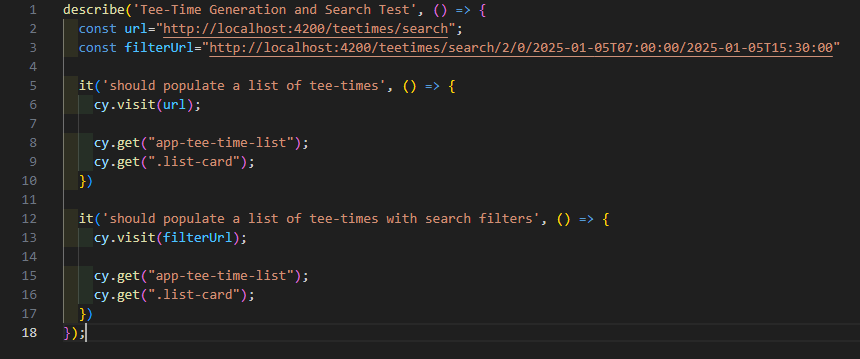
### Pass/Fail Criteria

* Tee-Time management:
  + Pass: A list of tee-times is successfully generated when searching.
  + Fail: A list of tee-times is not generated when searching.
* Reservation Management:
  + Pass: The user can successfully add a reservation.
  + Fail: The user is unable to add a reservation.
* User Authentication:
  + Pass: The user can log into the application with valid email and password.
  + Fail: The user is unable to log into the application with valid email and password.

If a test failed during the testing process the issue will be identified through a thorough investigation of the error messages, logs, and related code. The issue will then be documented and reported as a bug within the Jira project to later be corrected and retested.

## Specifications

Sample code shown in screenshots taken from the TTRS test scripts.





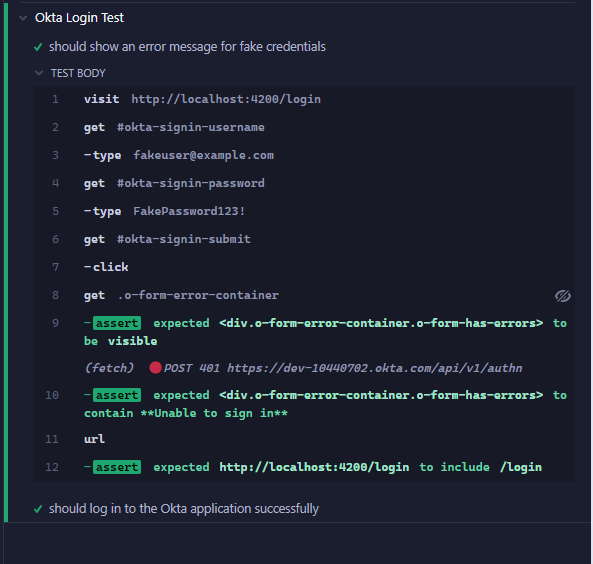


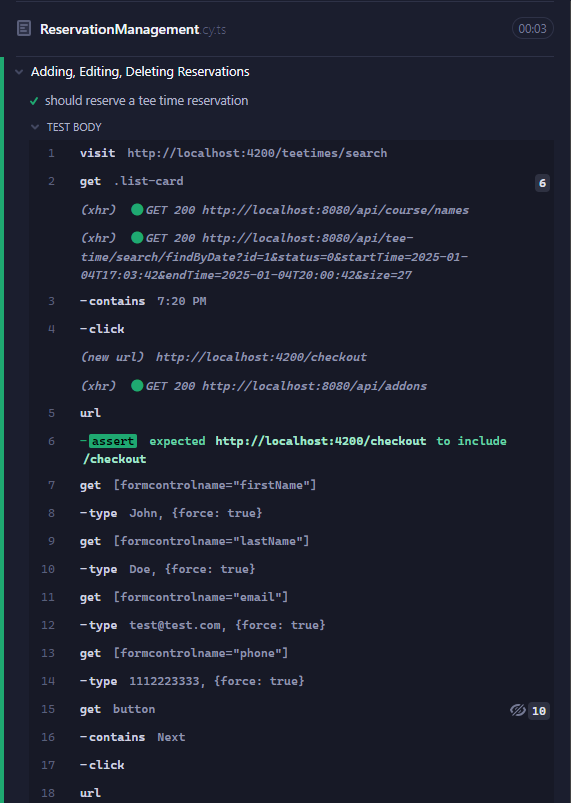
## Procedures

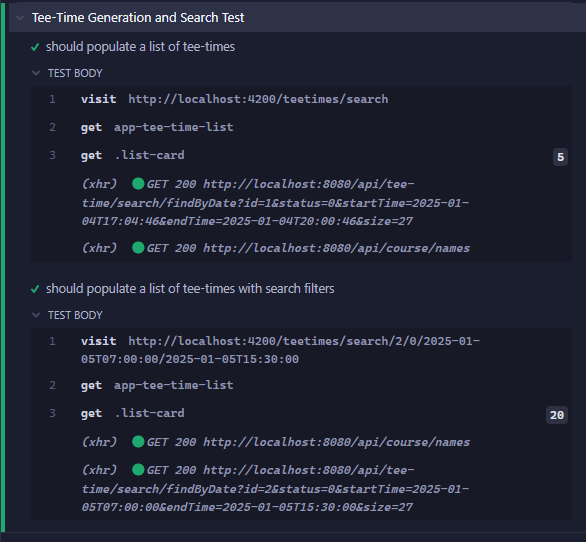
The testing process involved several key steps. First, critical functionalities and features of the application were identified. Then test cases were created to verify each function works as expected. Next, a testing environment was created to ensure each test could properly be created and analyzed for errors. Finally, test results were documented detailing pass/fail outcomes and modifications were made.

## Results

The following is the results from the previous test scripts using the Cypress testing framework.

* Tee-Time management: The tests passed when the user was successfully able to generate a list of tee-times by searching with random search filters.
* Reservation-Management: The test passed when the user was successfully able to complete the full process of selecting and reserving a tee-time.
* User Authentication: The test passed when the user was able to log into the application with correct credentials and was unable to log into the application with false credentials.





# Hosted Web Application Details

## Hosted Web Application Link:

https://www.ellisonware.com/

**GitLab Repository & Branch History**

## GitLab Link:

https://gitlab.com/wgu-gitlab-environment/student-repos/bell188/d424-software-engineering-capstone/-/tree/working\_branch

## A white background with black text Description automatically generatedGitLab Branch History:

A screenshot of a computer

Description automatically generated

# User Guide for Installation

## Introduction

This guide will demonstrate how to run the application on your local machine.

## Installation and Using the Application

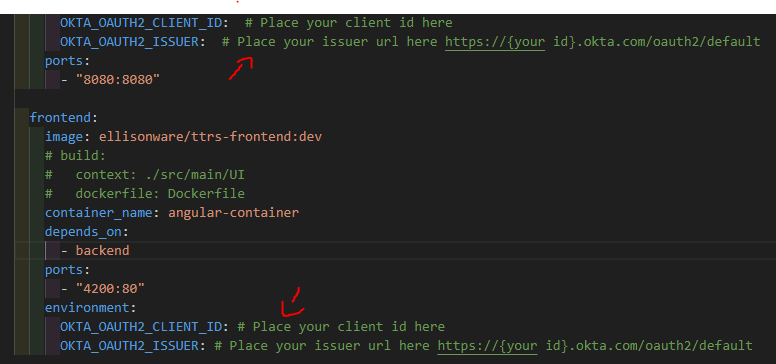
The set up for this application in a development environment is simple and intuitive. The guide bellow will provide the steps required to get the application running.

## Pre-requisites

The TTRS application uses a third-party authentication system. To setup the application locally you will need to visit https://developer.okta.com/signup/ and setup a free Workforce Identity Cloud Okta developer account and create your own authentication application. After setting up your account, follow the instructions provided bellow to create the Okta application and connect it to TTRS.

## Okta Application Set up

1. Login to the Okta development console.
2. On the left side navigation window select *Application>applications.*
3. On the Applications screen click the “Create App Integration” button.
4. Select the “Sign-in method” to be “OIDC – OpenID Connect”.
5. Select the “Application type” to be “Single-Page Application”.
6. From the “New Single-Page App Integration” follow these steps
   1. Name your application under the “App integration name” any name will work.
   2. Locate the “Sign-in redirect URIs” click “Add URI” and input:
      1. <http://localhost:8080/login/callback>
   3. Locate the “Sign-out redirect URIs” click “Add URI” and input:
      1. <http://localhost:8080>”
   4. Under the “Assignments” select “Allow everyone in your organization to access”.
   5. From the bottom of the form click “Save”.
7. Take note of the application “Client ID” credentials.
8. Open the user credentials by clicking where your email is listed from the right of the top navigation menu.
9. Take note of the issuer URL listed under the email in the format of:
   1. dev-###.okta.com
10. Follow The **Repository Setup** guide.
11. Open the docker-compose.yml file and place the application Client ID and issuer URL in the following environment variables for the frontend and backend configurations.
    1. OKTA\_OAUTH2\_CLIENT\_ID:
    2. OKTA\_OAUTH2\_ISSUER:



## Docker setup

1. Visit <https://docs.docker.com/get-started/get-docker/> and follow the directions for your specific OS for installing docker desktop on your machine.
2. In a terminal window pull the latest docker images from the general repository. Using the commands bellow.
   1. docker pull ellisonware/ttrs-frontend:development
   2. docker pull ellisonware/ttrs-backend:development

## Repository setup

1. Using git, clone the main repository. You can install the zip file or in a terminal window use the command bellow.
   1. git clone url https://gitlab.com/wgu-gitlab-environment/student-repos/bell188/d424-software-engineering-capstone.git

## Container setup

1. Within the ttrs/ repository directory open a terminal window and input the command bellow.
   1. docker-compose up -d

## Debugging Container setup

If the containers fail to start correctly follow the bellow steps.

1. Review whether the container is running via
   1. docker ps
2. Run the individual containers in order, giving each 10 seconds to initialize before running the next.
   1. docker start mysql-container
   2. docker start springboot-container
   3. docker start angular-container

Each container is dependent on the other to be running for the application to work successfully. For example, the spring boot application will not run without connecting to the mysql instance. You can check the logs for errors with the bellow command.

docker logs <container-name>

# User Guide for Operating the Application

## Introduction

This user guide will provide a tutorial for the main functionality of the application. This tutorial will take the user through searching and reserving a tee time, logging into the application, and editing or cancelling a reserved tee time.

## Tee-Time Search

Initially upon viewing the application you’ll be greated with a screen that looks like this:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

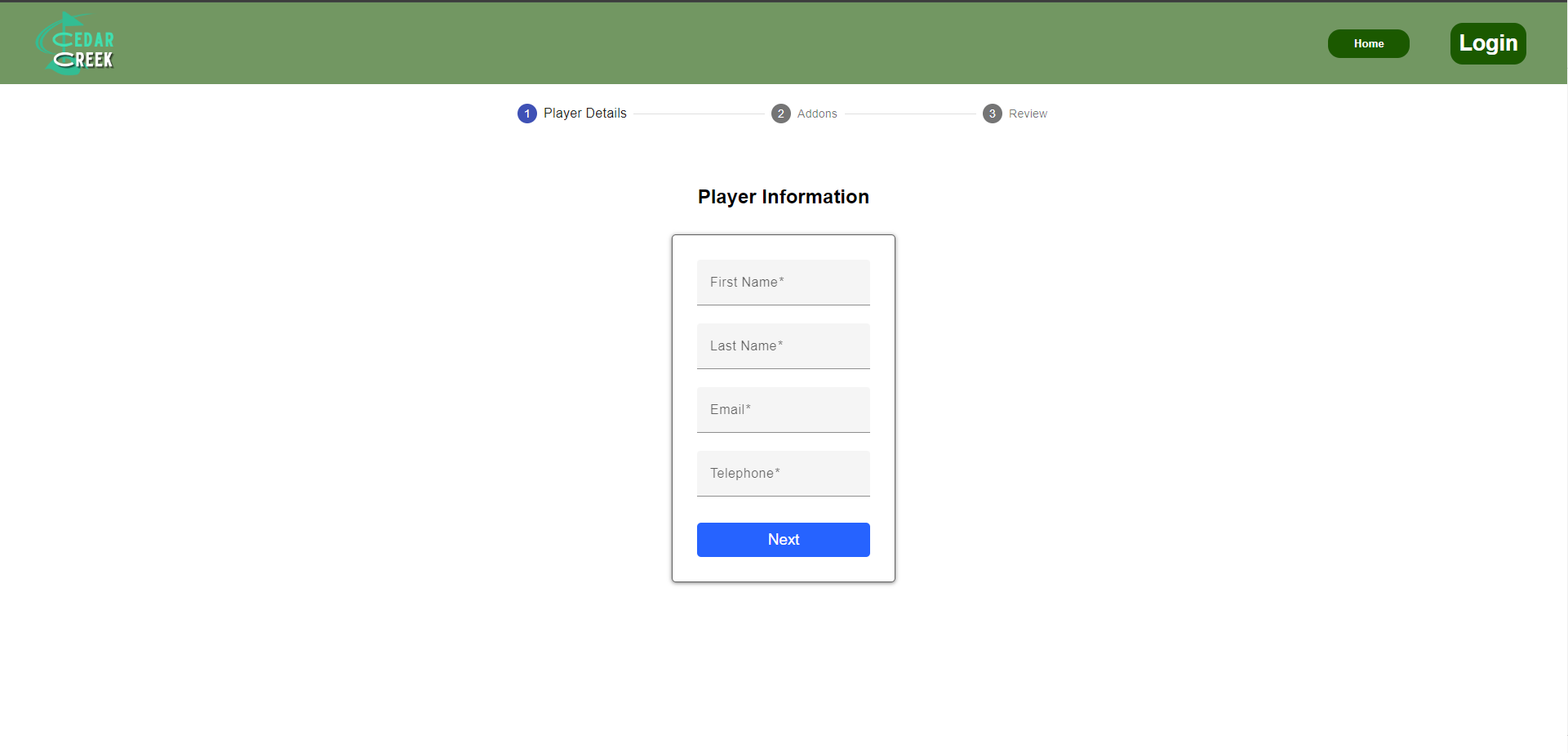
Description automatically generatedOn the left-hand side under filters, you can select the date, time of day, or course from the search form. Clicking the date widget will provide a drop-down menu with available dates highlighted, and unavailable dates greyed out. The Time-of-Day selector can be controlled via the two sliders to produce a range of dates of your choice. The course dropdown field allows the user to choose which course to produce the list of tee-times from.

Follow the steps below to search for tee-times using the Filter form.

1. Select a highlighted date from the date widget.
2. Move the selector bar to choose a range of times.
3. Select a course from the course dropdown menu.
4. Click the search button with your preferences implemented.

After searching for tee-times a buffer animation will display while the application is loading. After loading, the top header will display the selected date, and a list of tee times will populate within the search window. Each tee-time will have a cover banner indicating the category the tee time is in. There are two categories for daylight and twilight times. The daylight tee-times have the normal green fee for the course, and the twilight times have a $10 discount.

## Tee-Time Reservation

Select a Tee Time from the list of times to be redirected to the reservation customization portal. You will be prompted to input some personal information; first and last name, email address, and phone number. This is used to track your reservation to you in the database.

This next slide gives you the ability to purchase addons for your reservation online. You can pick from golf carts, balls, and tees for your reservation.

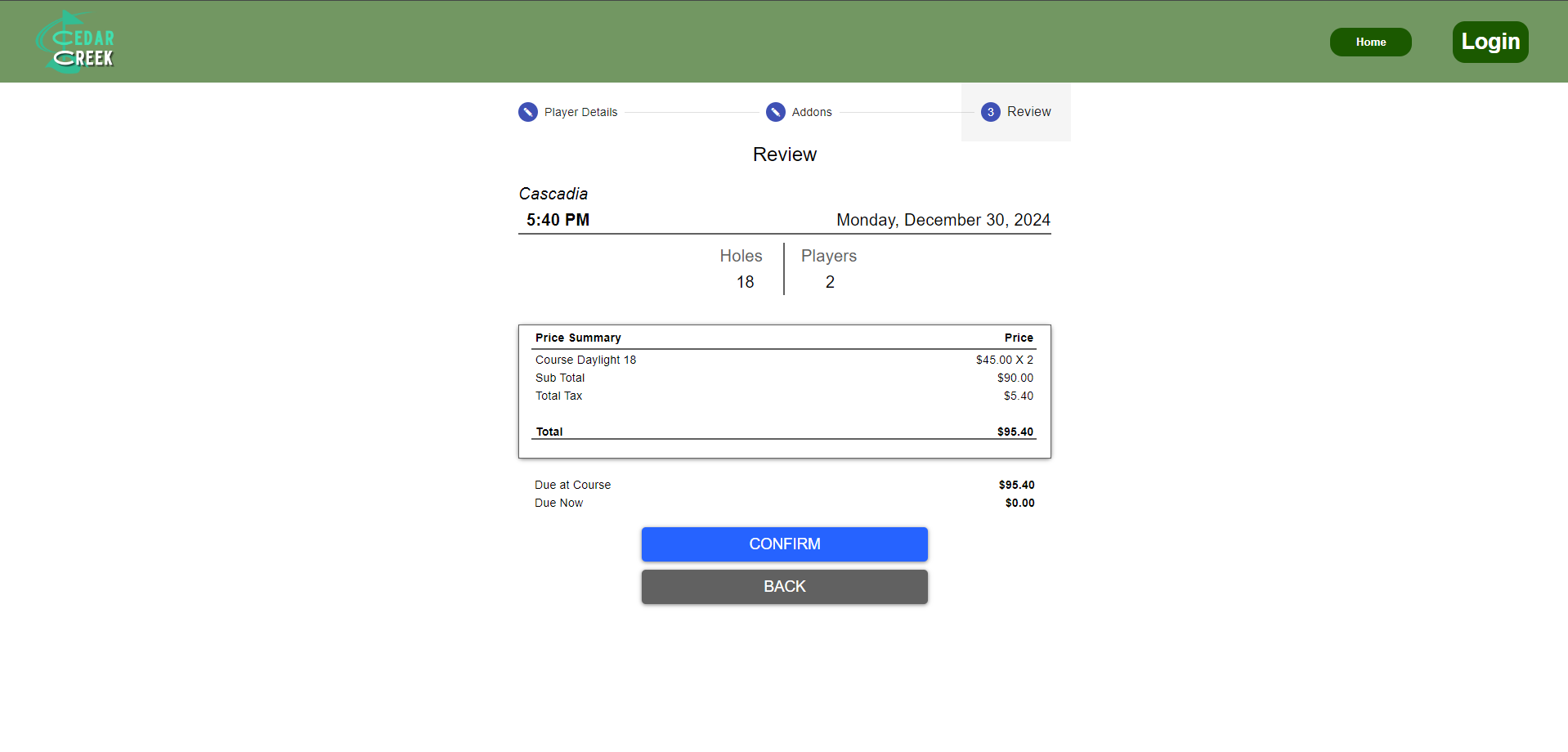
A screenshot of a computer

Description automatically generated

To add additional players to your reservation you can toggle the left and right arrows. Each reservation has a max of four players. Including yourself, you can add three additional players to the tee-time.

To add addons to the reservation you click the blue “+”. Next click the dropdown menu and select from the available options. You can increment and decrement the quantity of the specific item you chose with the “<” and “>” buttons next to the numerical value that represents the addons quantity. To remove an addon from the list click the blue “- “button on the left of the item. Select the “Next” button to continue to the next slide.

The final page will display the totals and allow you to review your reservation before confirming your purchase. On the top of the page, you can review the course, tee-time information, all addons, and purchases within the “Price Summary” box. No payments are completed online, once you arrive to the golf course you will be charged for your reservation.



## Login

Logging into the application is simple. There is a login button on the right-hand side of the top navigation bar. Select the login button, you will be redirected to the login page. A screenshot of a login form

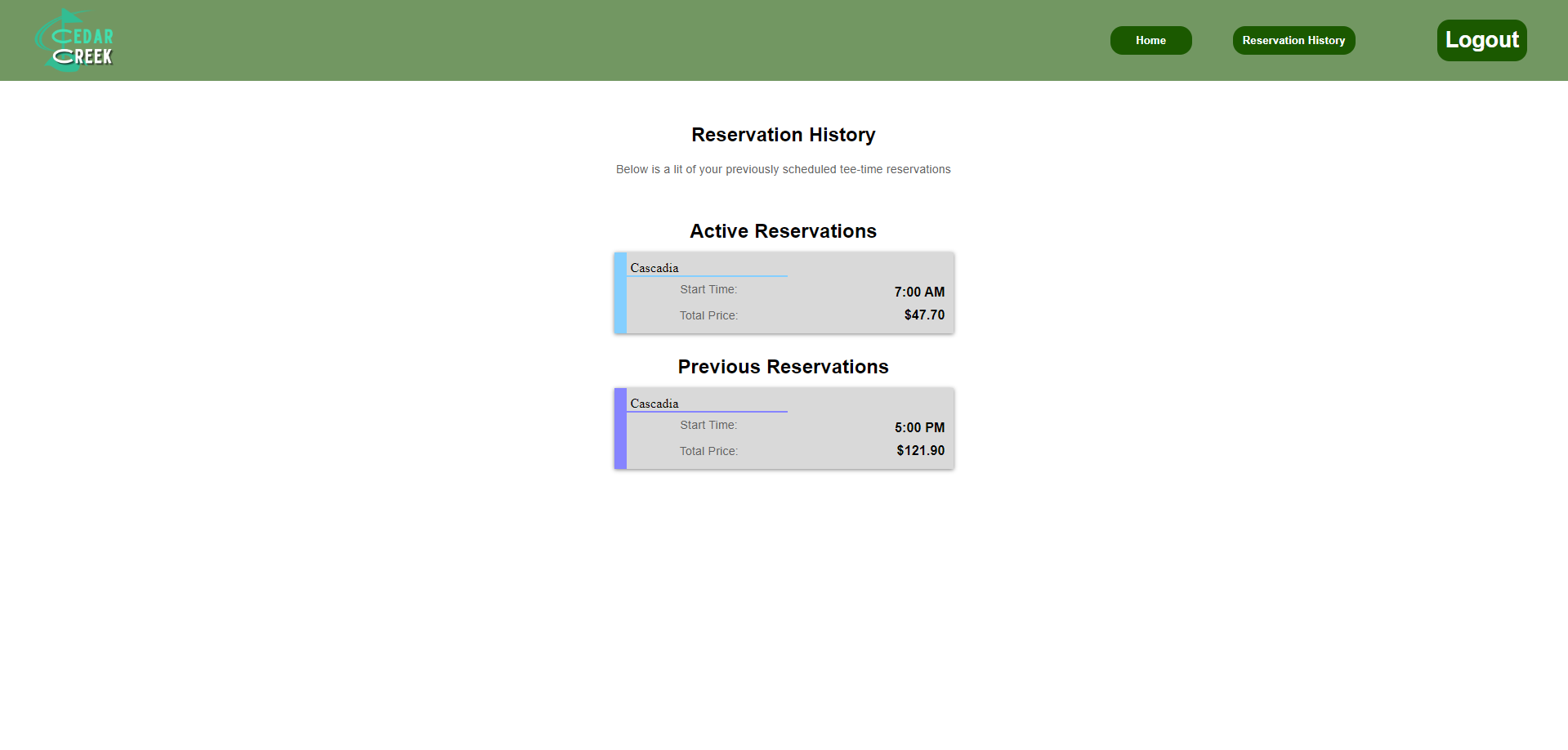
Description automatically generated

For testing purposes, this account will log you into the application.

* Username: [bell188@wgu.edu](mailto:bell188@wgu.edu)
* Password: testingadmin

After logging in you will be redirected back to the tee-time search screen. There will be an additional navigation option “Reservation History” that allows you to review previous reservations you have made, as well as edit the reservation addons or cancel the reservation entirely. Additionally, when logged into the application, your information will auto fill on the first page when reserving a tee-time.

## Tee-time Editing and Cancelation

To access this addition of the application you must be signed in. Review the **Login** tutorial for a demo account you can use to access the application. After logging into the application, you will get access to the “Reservation History” page. You can access this page by clicking the link in the navigation bar. 

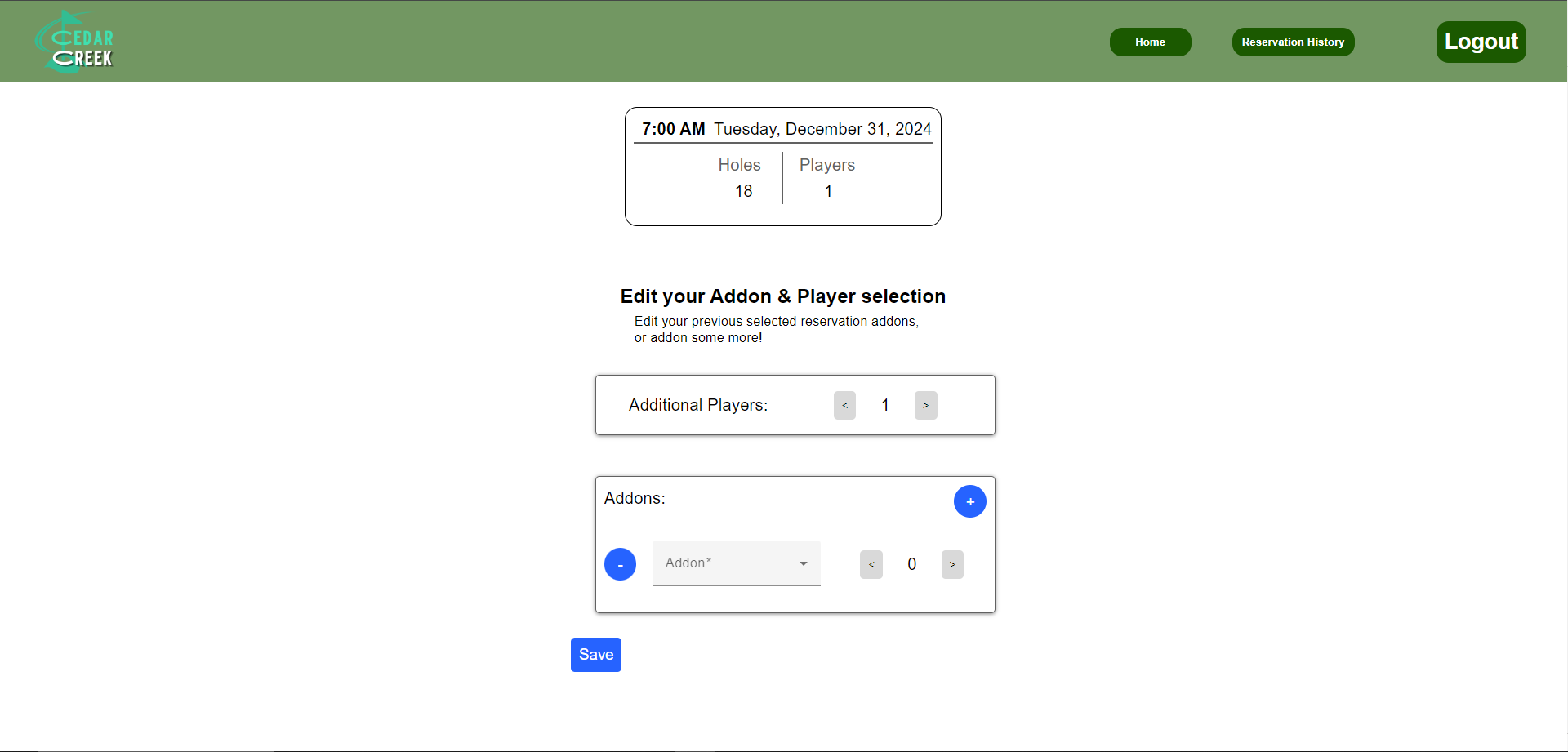
The Reservation History page will show you active reservations, reservations that have not been completed yet, and passive reservations; a reservation that was previously completed.

To edit or cancel a reservation, click on an active reservation to display the “Cancel” and “Edit” buttons:

A screenshot of a computer

Description automatically generated

A reservation can only be cancelled two hours before the start time. In the last image the tee-time start time shows as 7:00am, in this example the reservation can only be cancelled any time before 5:00am.

Clicking on the “Edit Addon” button will direct you to the edit page:

The edit addon widget is almost identical to the previous addon customization widget. The same usability applies.

* + - 1. Click the “<“ and “>” buttons in the Additional Players window to add or remove additional players.
      2. In the Addons window click the “+” blue button to add an addon to the list.
      3. Click the drop down menu and select an addon from the menu.
      4. Click the “<” and “>“ to increment and decrement the addon quantity.
      5. Click the “-“ button on the left of an addon to remove it from the list.
      6. Click the save button to save the tee-time and return to the Reservation History page.