HYBRID ATHLETE FOUNDATION
ADMIN PORTAL

Matthew Murray

A Capstone Project Submitted to the
University of North Carolina Wilmington in Partial Fulfillment
of the Requirements for the Degree of
Master of Science

Department of Computer Science
Congdon School of Business Analytics, Supply Chain and Information Systems

University of North Carolina Wilmington

2024

Approved by

Advisory Committee

Laurie J Patterson	Albert Steed

Jeffrey Cummings, Chair

TABLE OF CONTENTS

ABSTRACT

Hybrid Athlete Foundation Capstone Admin Portal. Murray, Matthew, 2024. Capstone Paper, University of North Carolina Wilmington.

The Hybrid Athlete Foundation (HAF) administrative portal was developed to provide gym owners and administrators with a web-based tool to manage their facilities, members, and access control. As HAF has grown to over 89,000 users and 657 gyms nationwide, the need for an administrative portal separate from the mobile applications became clear. This capstone project sought to create a web-based admin portal that could be accessed from any device, providing gym owners and the HAF support team with similar functionality to the existing mobile apps while also adding new capabilities. The admin portal was developed using C#, .NET, and Bootstrap, integrating with HAF's existing API and database. Key features include membership management, guest access control, gym information updates, access event logging, and the ability for administrators to set door schedules and access restrictions. The portal also includes a “Support Settings” section for super administrators to manage door assignments and gym interface integrations. The end result is a comprehensive administrative tool that gives gym owners full control over their facilities, members, and access, all from a centralized web-based platform. The portal’s intuitive design and seamless integration with existing HAF systems allows for a smooth transition from the mobile apps. Future plans include separating the portal from the core services, migrating to .NET Core, and eventually incorporating the admin functionality directly into the mobile applications.

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

The concept of Hybrid Athlete Foundation (HAF) was to give gym owners the ability to grant access to members at times when no administrators are present, and the gym is essentially empty. It also has the ability to connect gyms, so that when people travel, they are able to find a place to work out without having to pay the usual drop-in fees. The concept is based on the belief that this capability should be done from a mobile application that can be used from anywhere.

The original HAF application was designed by Albert Steed and developed by Nguyen Phuoc Vinh who created the original website, iOS, and Android apps. The first apps were deployed in the summer of 2017. Since then, HAF has grown to over 89,000 users and 657 gyms across the nation. Currently, HAF is made up of a two-person development team. One developer's responsibilities revolve around web and iOS development. The second developer is responsible for Android tasks and also provides support for web development when needed.

With the rapid growth of HAF, numerous gym owners have requested the ability to manage members and perform other administrative tasks outside of the existing iOS and Android application. This along with a need to give increased tools to the support team, has led to the determination that a web-based administration portal is needed that can be accessed from any web-based device without needing to have access to a mobile device. For this capstone, an administration portal is designed for HAF that includes similar admin features to the mobile apps, such as the ability to manage members and their access, manage gyms, and view access events. The new admin portal also includes new features such as the ability to set door schedules, assign doors to a gym, and set a gym's interface parameters used to connect to member various management systems.

CHAPTER 2: REVIEW OF LITERATURE REVIEW AND ANALYSIS

The Hybrid Athlete Foundation (HAF) app is an application designed for gyms that are members of the HAF network and their clients. Any person that is a member of a gym inside the HAF network can use an Android or iOS app to access their gym 24 hours a day, seven days a week (24/7) without a coach or administrator present. The app also allows users who are traveling to find in-network gyms, and request/gain access to other gyms. Finally, for gym owners, it is a customer/facility management system that allows them to manage access to their facility. For the new admin portal, the existing mobile app architecture needs to be explored to better capture the functional requirements of the new portal.

Current Architecture

Currently, the Android app is coded in java, and the iOS app is coded in objective-c. They both utilize HAF's in-house API written in C#, which is deployed as a Microsoft service. This service then accesses HAF's Microsoft Azure database.

In order for users to open a gym door through the app, a Brivo box needs to be installed at each gym location. By utilizing an internet connection, these boxes allow users to unlock doors through various API calls. The Id/key for each Brivo box is stored in an Azure database that is called whenever a user tries to unlock a gym door.

A Brivo box is an internet connected access control system seen in Figure 1 below. In order to interact with the Brivo control unit (Figure 1), HAF uses Microsoft services to communicate with Brivo's API whenever an access request is made by a user.

The current Hybrid Athlete Foundation (HAF) mobile applications have three types of users in mind. The first of which are gym goers or users, who have the ability to search for a gym within the HAF network, request access to a gym, view workouts posted

by their home gym, edit their accounts, and unlock the door(s) to any gym within a 5-mile radius. The second type of user is gym administrators (admins) who are able to do everything normal users can do, but can also create new members, generate guest passes, manage existing members, and grant admin privileges to members. The last type of user is a super admin, who has all the privileges of an admin, but can manage and access every gym's information. Super admin privileges are typically reserved for HAF employees.

Figure 1: Brivo Access Control Unit

User Privileges

Functional Requirements – Login

When a user first opens the app on their mobile device, they are brought to the login screen (shown in Figure 2: HAF Login Screen). Here a user can choose to login with their user credentials, create a new account, or reset their forgotten password.

Functional Requirements - Administrator Menu

If the user is an admin or super admin, they can click the admin menu bar at the bottom of the home menu to bring up the admin menu shown in

Figure *3*. In the new app, the user will start on the admin menu. In the current mobile apps, the Admin menu has three tiles, a My Members tile for managing members, an Access Events tile to see who has been entering the gym, and a My Gym tile for

updating the gym's information. In the new admin portal, there will only be two of these tiles "Manage Members" (My Members) and "Manage Gym" (My Gym). Access Events will be moved to a tab under the "Manage Gym" section. There will also be a tile for logging out as well as a tile Called "Support" which will take users to a new support page.

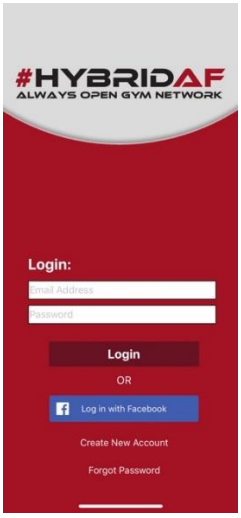

Figure 2: HAF Login Screen

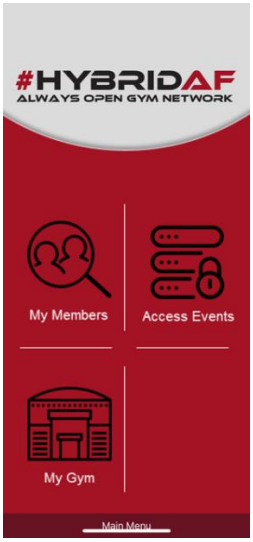

Figure 3: Admin Menu

Functional Requirements - Display Gym Members/Guests

Selecting the My Members (Manage Members in new portal) tile will bring a user to the My Members (Manage Members) screen shown in Figure 4. From here, they can filter users by various criteria as well as switch between the members tab and the guests tab. The look for each tab is the same in that they can see who has an active membership as well as how many times they have accessed the gym during the current and previous month. When Changing tabs, the button "Add Members" changes to "Generate Pass" to distinguish between adding a new member or creating a guest pass. All of this functionality needs to be available for the new portal.

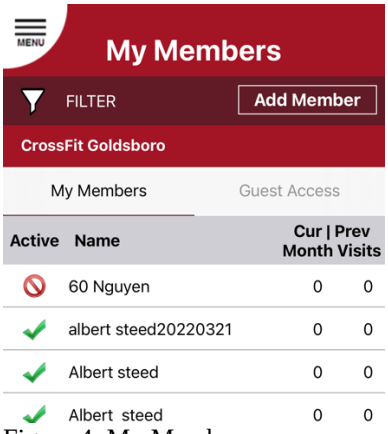

Figure 4: My Members

Functional Requirements - Update Gym Member Information

Upon selecting a member or guest, the user is taken to the Member Details screen which has two tabs (as shown in Figure 5) either a the membership tab (for members) or the Guest Access tab (for guest) with the Profile tab being the default displayed. Common across all tabs is that the admins can give or revoke a user's admin status. This is done by tapping the button on the top right seen in Figure 5, 6, and 7. The button will say Set Admin if they are not an admin and Set User if they are an admin.

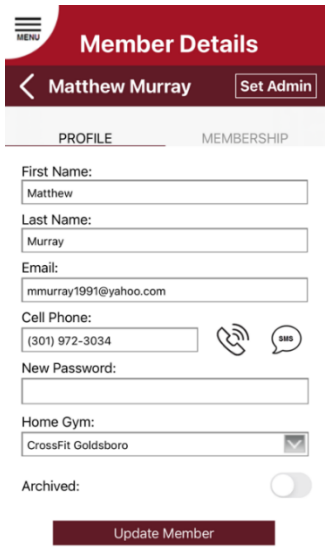

Figure 5: Member Details

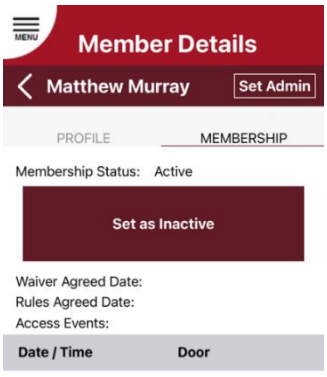

Figure 6: Membership Tab

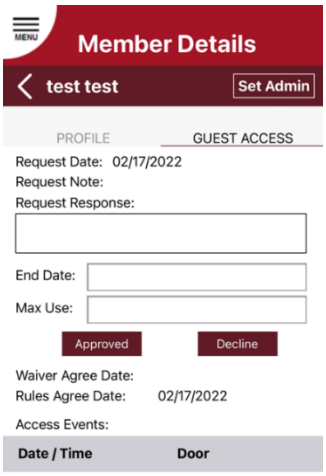

Figure 7: Guest Access

On the Profile tab, admins can edit members' information, and even reset a member's passwords for them. When a guest is selected, the same details appear. However, in the case of guests the admin does not have the ability to edit or update the guest's details.

In the Membership tab, admins can change a person's membership status by tapping the button in the middle (in this example, "Set as Inactive") shown in

Figure 6 above. If the user is active, the button will say "Set as Inactive," and if they are inactive, the button will say "Set as Active."

In the case of guests, the Guest Access Tab can be used. Here an admin can change the expiration date, and usages remaining for their guest pass. If the fields are left blank, the user's guest pass will not have an expiration date or Max usages. To save changes, the user must tap the Approved or Extend button, while to revoke access, the admin must tap the decline button.

Functional Requirement - Display Gym Access Events

If the user selects the Access Events tile from the admin menu, they are brought to the Access Events screen shown in Figure 8. Here admin can see who entered the gym, what time they entered, and what door they used. They can also look back at previous dates to see who accessed the gym in the past. In the new admin portal this will be moved to a tab that will fall under the "Mange Gym" tile.

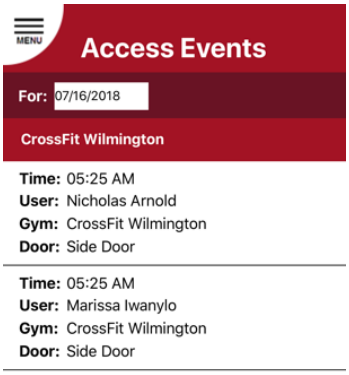

Figure 8: Access Events

Functional Requirement - Update Gym Information

A third tile an admin can select is the “Manage Gym” tile. which will bring the user to the “My Gym” screen which has two tabs. The first and default tab is the “Manage Gym” screen shown in **Error! Reference source not found.** which has all the gym’s details. Here, an admin can edit and update basic information about their gym. In the apps there is the ability for admins to tap a button to update the gym’s location using the current device’s GPS coordinates. This feature will not be available in this iteration.

The second tab is the collateral tab shown in Figure 10. Here admins can set their gym’s logos, as well as rules and waivers. They can do these by uploading files or typing in URL links to the respective files. For rules and waivers, admins can also set if they want users to have to acknowledge/review them either once or once daily which will cause them to have to review them the first time they try to access the door each day.

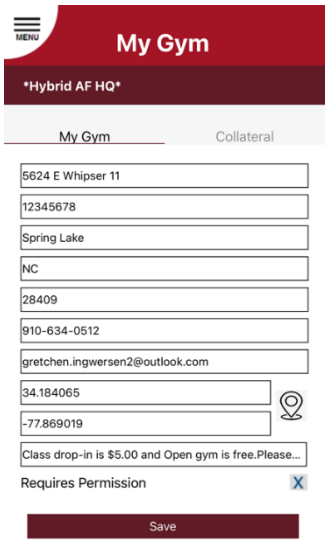

Figure 9: My Gym

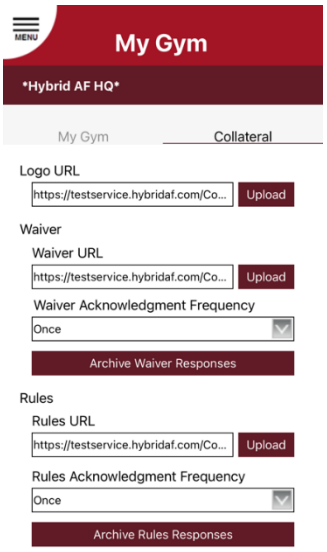

Figure 10: My Gym Collateral

Additional Functional Requirements

Additional functional requirements for this project are functions HAF wants to add to the new admin portal to provide gym admins as well as the support team with increased functionality. The first is to include a way for admins to set access schedules for their doors. This will allow gym admins to allow or deny access at times they set. The second is to add extra functionality for the support team so they do not have to directly interact with the database. The two-support main support functionalities HAF wants are the ability to assign doors to gyms, as well as the ability to set a gym's interface credentials. Interface credentials are what HAF uses to integrate with various member management system that gym owners use.

Additional Functional Requirements - Gym Door Access Schedules

By default, doors are locked 24/7 and can only be unlocked by holding the unlock button on the app. However, there are situations where admins want the door to remain

unlocked, or do not want the non-admins to be able to unlock the door through the app. This is where the new door access schedule system will come into play. The door access schedule functionality will be accessible by admins and super admins and will be built as its own page. This will allow the new door schedule system to be accessed from the new admin portal, as well as framed into the existing mobile apps. It will have two sections (Door Schedule and App Disable) which will function similarly in that they allow admin to set schedules for different accessibility.

The first section, door schedule, deals will present a list view of days of the week as well as periods. By clicking/tapping into a day, an admin will be allowed to set start and end times for when their door will remain unlocked. Upon saving this new unlock period will be set as a reoccurring time every week on the selected day. The next functionality in this section are door exceptions. Door exceptions are one-time exceptions that can be set to hold a door open, or keep a door locked at the day and time set. These exceptions will override anything that is set in any existing door schedule. Currently, door schedules and expectations are set in Brivo's online portal. Utilizing Brivo's API's, HAF plans to change this so that this can all be done from the admin portal.

The second section app disable, has a recurring weekly schedule as well as exceptions just like door schedule does. However, app disable is focused solely on keeping the door inaccessible at the set dates and times. On the day and time of an app disable, a non-admin member will not be able to see the unlock button in the app and thus not be able to unlock the door. Like with door schedules, there can be one time app disables or one-time overrides done by adding an exception to the app disable section.

Additional Functional Requirements - Support Setting

The new admin app will also feature a new Tile on the admin menu. This tile will

appear only for super-admin and will be labeled Support Settings. Clicking on this new tile will take them to the support page which will feature a doors tab and an interfaces tab.

Under the doors tab there will be a table that has all the door records that were received from Brivo's API. To make sure the data is up to date, a button will be added called Sync Brivo doors. Clicking this button will cause the Brivo sync job to run that gets all new doors from Brivo's API and stores them in our database so they can be displayed in the table. The data that will be displayed in the table are the Site Name (information HAF gets from Brivo), the door name, the name of the gym it is assigned to (blank if unassigned), if the door is enabled (accessible to normal users), admin enabled (accessible to admin), and external schedule which indicates if the door has an access schedule setup in Brivo. By clicking on one of the doors, a modal will be brought up that will allow the user to change all the information shown in the table except for the site name.

Under the Interface tab there will be a dropdown to select which gym's interface you want to update, as well as an input box that takes in the interface parameters (usually an xml string). When a gym is selected, the interface parameters input will populate for gyms that have existing interface parameters. Upon saving, the interface parameter for the gym will be updated in the database.

Non-Functional Requirements - Security

There are a number of security controls in place in the application. First and foremost are the two user types that will have access to the new admin portal (admin, super admin). Each user type has its own privileges associated with it to prevent unwanted access to management systems. To make sure that only our user has access to

gym doors, HAF uses an Open Authentication (OAuth) token to access Brivo's API. Without access to Brivo, none of the doors will work. Finally, just like with Brivo's API, the HAF API uses an OAuth token to verify access to our API. All of these tokens are stored in the config files of the apps so that they can be referenced by the apps, without users being able to see them.

To keep users' credentials secure, the users' password are encrypted using MD5 hash encoding to encrypt the password before they are sent over the network to the database. In order to verify a password, when the user attempts to login, the entered password is encrypted, and the encrypted password is compared to the previously stored encrypted password in the database. In order to keep these credentials secure, they must re-enter their credentials each time they open the app.

Look and Feel

The main goal of the app when it comes to design is to have a sleek design while still being intuitive enough that a user can use the app without training. HAF wants the new admin portal to have a look and feel similar to the mobile apps. HAF believes this will allow users to transition into the new app with virtually no training. HAF believes that the current design accomplishes this. However, HAF is always open to making it more intuitive through feedback from users.

Portability and Maintainability

The idea behind this project is to give gym owners the same control they currently have over their facility, but with the convenience of utilizing any device with a web-browser.

Constraints

Web Access. The first constraint that needs to be addressed is that the app must

work on most web-based browsers. This includes Google Chrome, Firefox, Edge, and Safari.

Current Architecture. Next, the app must be able to use/access the current web services and database that is already in place. This means being able to access the HAF API, and HAF Azure database.

Preserve Existing Data. Finally, the app must be able to utilize existing data. The data stored in the database is essential to the operations of the company and would take too long to reproduce.

Problem Statement

The Hybrid Athlete Foundation (HAF) is now an established access control system company dedicated to providing our customers with the features and ease of use they need to manage access to their facilities. In the last few years, there have been requests from many of our clients to create an administration (admin) portal to allow them to utilize all the administration functions they enjoy in the app but from the convenience of any web-based browser. With this, clients have requested the ability to manage their own access schedules. Previously, owners would need to email support when they would want to create a time exception. Support would then have to login to Brivo's management portal and adjust the schedule manually. So, with a functional admin portal, HAF needs to allow gym owners the ability to manage their own door schedules.

As HAF has grown, they have also employed the services of outside support. The support team not only needs to have all the abilities of a super admin, but they also need to perform some extra tasks that can currently only be done through database access. These two tasks are linking gyms to doors and setting up gym interfaces with other member management systems (Wodify, Push Press, ZenPlanner, Mindbody, etc.).

Thus, the goal is the creation of an admin portal for owners/admin as well as the support team that can be accessed from a web-based browser. This will allow owners/admin the same admin functionality they currently have in the mobile apps, while empowering them with the new ability to manage their door unlock schedule. The admin portal will also give the support team the ability to manage facilities doors as well as set interface parameters which currently can only be accomplished through manual database manipulation.

CHAPTER 3: METHODOLOGY

With the needs/requests of gym owners and the support team, the goal of this project is to create an admin portal and new scheduling system that can be accessed from a web-based browser. This will allow owners/admin the same admin functionality they currently have in the mobile apps, while empowering them with the new ability to manage their door unlock schedule. The admin portal will also give the support team the ability to manage facilities doors as well as set interface parameters which currently can only be accomplished through manual database manipulation.

Development Language

Since HAF has few developers, this development will need to be done in a language that all employees understand. That, paired with the fact that the API is built in C# using dotnet framework version 4.8, means the decision was made to build the new features using the existing C# and dotnet framework. This will enable fast delivery and maintainability.

Security

No matter which platform is used for cross platform development, they will all provide security the same way HAF's current apps do. This will be done by utilizing HAF's API allowing only specific requests to be made to third party venders such as Brivo. The way access will be granted to the API will be through the current user authentication system which was described earlier.

Stress Testing

In order to perform stress tests, HAF's test Azure DB and test services were used. The structure of the DB mimics that of the production DB. However, when it comes time to test functionality relating to gym access.

Additionally, user testing will be conducted on the new application. Members of HAF, as well as a limited number of outside users who are familiar with the current app, will work through and evaluate the main functions of the admin portal.

Plan

Deliverable - Administration Portal. The final product is a web browser-based administration portal created for the Hybrid Athlete Foundation. It has the same admin functions as the current iOS and Android mobile applications, as well as the ability for owners to manage their door schedules, and super admin to manage doors and gym integrations. This is done by utilizing C#, MVC, bootstrap, data tables, Razor pages, inhouse and Brivo's API's. A new front-end UI was created as well as adjusting our backend to support new and existing functionality.

CHAPTER 4: OUTLINE OF COMPLETED PROJECT

At the start of this project, HAF wanted to ensure the new web app would display properly on any browser. To accomplish this, Bootstrap 5 was used as the main library for building the user interface. It was also decided that HAF could combine the front-end tool with a mix of MVC and Razor pages to allow for dynamic pages.

The first page built out was the login page. The login page features an MVC form with two inputs, one for the user's email address and the other for the user's password. There is also a button labeled login, and a link that says forgot password. Clicking forgot password takes the user to the forgot password page which has an input for the user's email and a submit button. Upon clicking submit, the email is validated, and if valid, a new password is emailed to the provided email address. If the user clicks login, HAF sends the information to the home controller that validates the provided email/password combination, and then validates if they are either an admin or a super admin. If they are an admin or super admin, a session is created and data such as their permissions are stored in the session. HAF then redirects to the index section of the admin controller which redirects the use to the index page.

Index Page

The Index page (see Figure 11) was built using a grid layout to mimic what exists in the mobile app. If the user is an admin, 3 clickable tile buttons appear on the screen: Manage Members, Manage Gym, and Logout. If the user is a super admin, they will see a fourth tile labeled Support Settings. Clicking/tapping on these buttons takes the user to the respective pages apart from the logout button which will end the session and redirect the user back to the login screen.

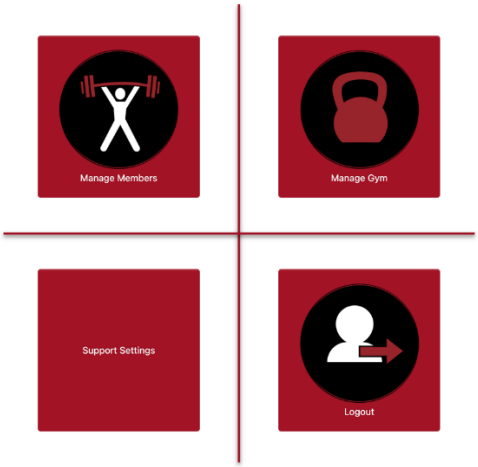

Figure 11: Index Page

Manage Members Page

Clicking/tapping the Manage Members button will take the user to the Manage Member Page (see Figure 12). This page has two tabs, members and guests, with each having a similar layout. The main part of the page is a large table, generated using the datatables plugin. This plugin provides built in functions such as searching, sorting, and exporting to the clipboard, excel, or PDF. Some custom CSS was needed to allow for spacing and labels to show up properly.

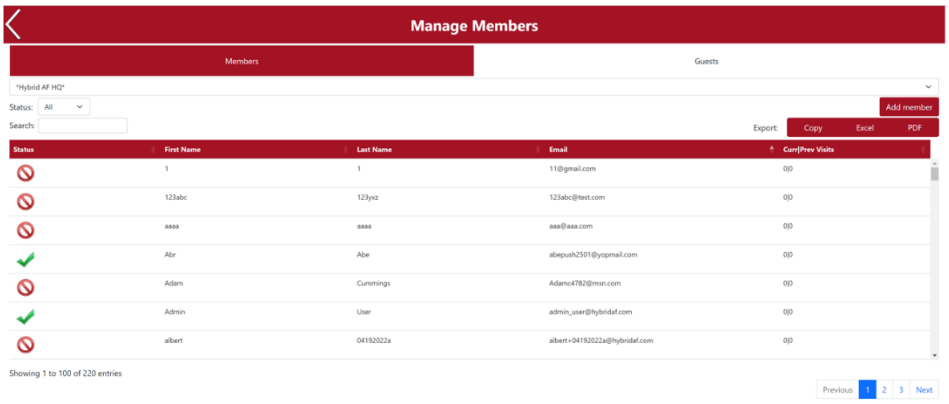

Figure 12: Manage Members Page

Additionally, two dropdowns are generated using razor with one containing a list of all the gyms the user is an admin of while the other dropdown contains a list of gym

statuses such as Active (default), Suspended and Archived. Anytime one of the dropdowns is changed, a function called renderMemberView is called that uses an ajax call to communicate with the admin controller. The admin controller takes in the data and builds a partial view that gets returned to the UI. The UI then re-renders the table and then re-initializes the table.

Another item on the page is a button to add member/generate guest pass depending on what tab the user is on. Clicking this button causes the UI to make an AJAX call to get the partial view that has an MVC form in it. The partial view is then injected into a modal using jQuery and for guest users, the expiration date field gets sets as a bootstrap datetimepicker. When the form in the add member/guest modal is submitted, the controller does some verification to make sure the user or guest record does not exist, and if it does it will create one.

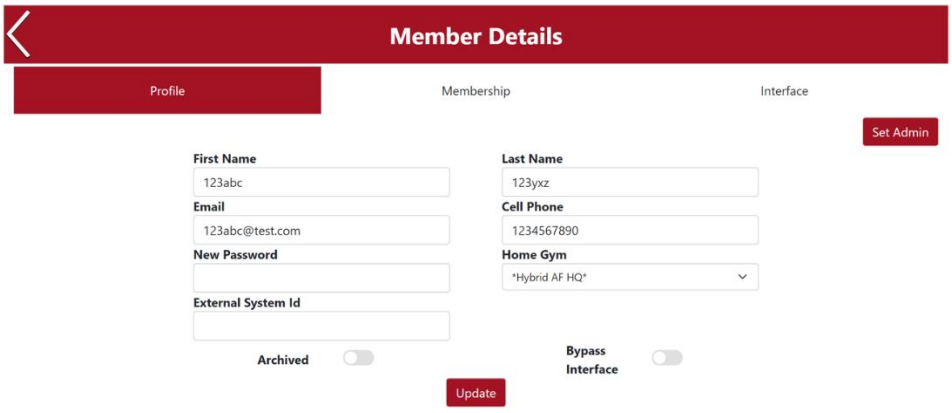

Figure 13: Member Detail Profile Tab

If the user taps or clicks on a member/guest, it will redirect them to the MemberGuestDetail page (see Figure 13 above). This page is dynamically generated using razor and contains three tabs for members and two tabs for guests. The partial view is converted to a string using a RenderViewToString method that takes the view name

and the model the view is going to use. Then, it uses a StringWriter to build an html string using the ViewEngine and ViewContext from the System.Web.Mvc assembly. The first tab is the profile tab that has a secure form and allows admin to update a member's info. If the admin tapped on a guest, the details in the profile section are not editable.

The second tab, membership/guest access (Figure 14 and Figure 15, respectively) has buttons for setting members active, or setting guest pass remaining usages and the guest pass end date.

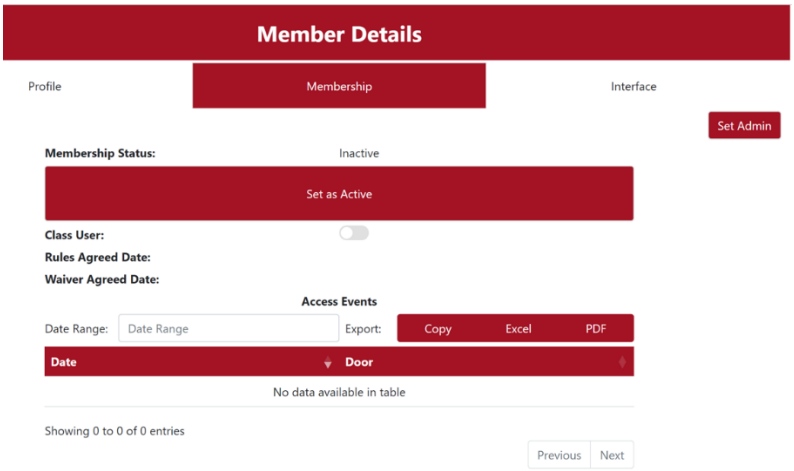

Figure 14: Member Detail Membership Tab

Something in common for this page is the Access Events section that uses the data table plugin to render the table, as well as a date range search bar. The date range input is created using a JavaScript plugin called Tempus Dominus. On change the date range is passed to the controller which builds a partial view as an html string and passes it to the front end where the access event table gets re-rendered/reinitialized.

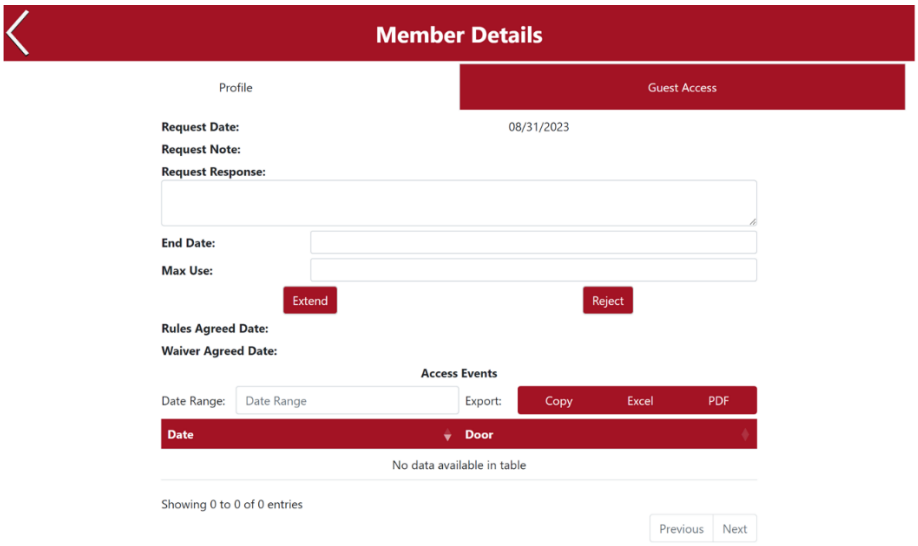

Figure 15: Member Detail Guest Access Tab

The final tab, which is only visible when looking at a member, is the Interfaces tab (see Figure 16). This shows the number of times the selected member showed up in HAF's interface table. It is made using the datatables plugin and allows the user to search the table as well as export the table to clipboard, excel, or pdf. Something that persists on each tab is an admin button that when clicked either removes or grants the user admin rights to the gym. The button says Set Admin when the user is not an admin and Set User when they have admin access.

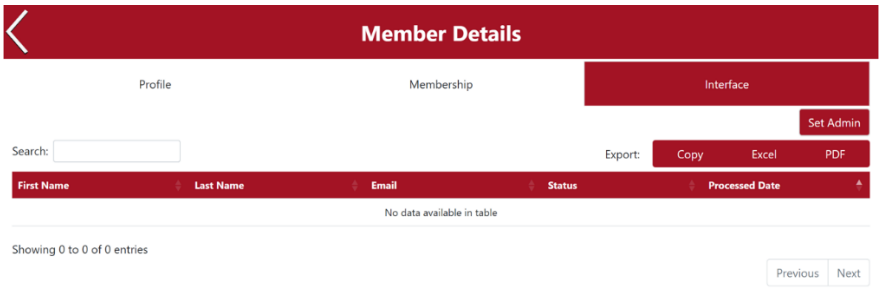

Figure 16: Member Detail Interfaces Tab

Manage Gyms Page

Clicking on the Manage Gyms tile on the main menu will take the admin to the My Gyms page (see Figure 17) which has 4 tabs: My Gym, Collateral, Schedule, and

Access Events. My Gym is the default tab, and gym's information such as address, notes, and various permissions toggles to control things like if permission is required, if the schedule is enabled or if the gym is active.

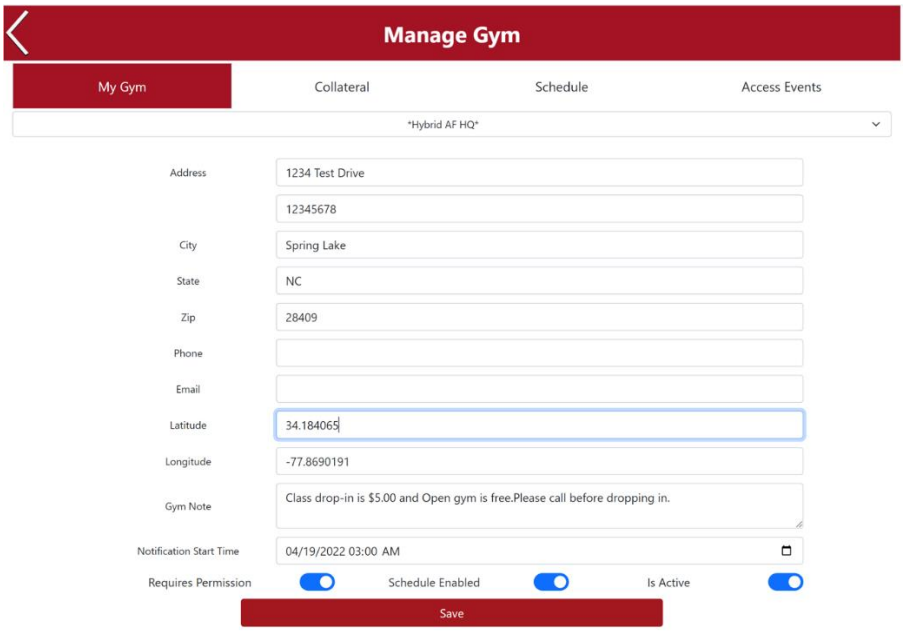

Figure 17: Manage Gym My Gym Tab

The second tab (see Figure 18) is the Collateral tab. Here, an admin can upload a logo, rules, and waivers. The inputs for these are read only to show files that already exist. Additionally, there is an upload button next to each that calls ajax functions to send the files to the container where they are stored in the proper folders on the server. The other inputs on this tab are selects which allow admins to set waivers/rules to be reviewed once or once daily.

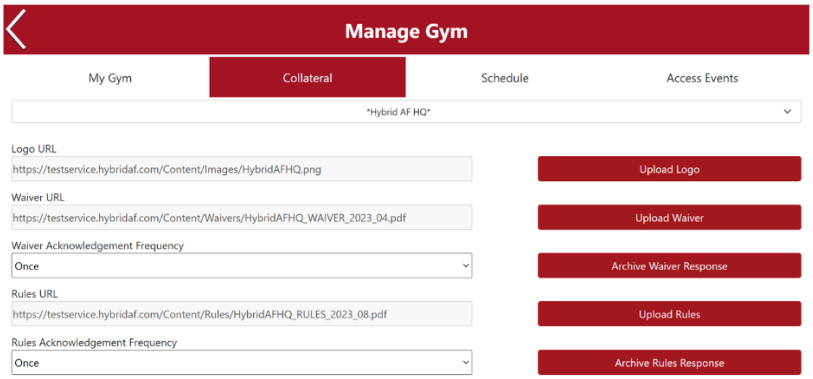

Figure 18: Manage Gym Collateral Tab

The next tab (Figure 19) is the Schedule tab which is a completely new feature. Since the functionality for the schedule tab is completely new, it needed to be built so that it can either be framed in by the mobile apps or framed into this project. Because of this, the schedule system was actually built with its own URL that uses an encrypted key as a parameter to tell the controller which user and gym the schedule is for.

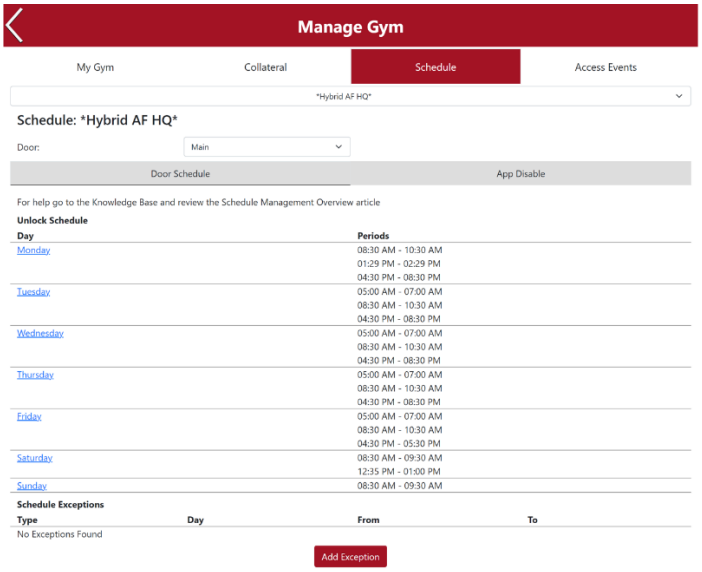

Figure 19: Schedule Door Schedule Tab

Next, the provided data from the key is used to build the model for the view. Half the data that goes into the model is grabbed from our third-party partner's API and parsed, and the other half is grabbed from HAF's database. Once the model is put

together, it is passed to a partial view which is rendered as HTML and added to the model. The model is then passed to the view which contains two tabs and renders the partial view that was passed from the model. The view has a lot of JavaScript functions. A function for converting from military time to standard time was needed as the database and the third-party API are based in military time. There is also a function to convert from military time to standard time. These functions make sure the offset is correct for each gym since the date/times are in UTC and need to be shown as local times. When a day in the list view is clicked, the main page is hidden and the periods for that day are dynamically created with JavaScript (see Figure 20).

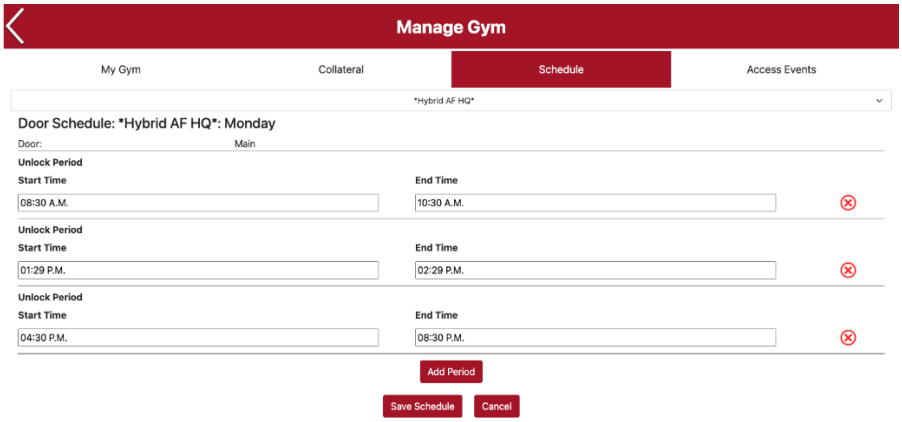

Figure 20: Schedule Periods

For setting dates, the Tempus Dominus plugin was used. For time inputs, Material Design RTL Time Picker (MDTP) JavaScript plugin was used. This MDTP plugin had to be modified to properly account for morning times and afternoon times. A validation function was created that makes sure times do not overlap and that start times occur before end time. When saving, the front-end makes an ajax call to the back-end with the day (date for exceptions) and the list of start times and end times. If the user was on the Door Schedule tab, the controller will send the data to HAF's third party API. If the User

was on the App Disable tab, the data is saved to HAF's database. After the save, the controller regenerates the partial view where it is sent back to the front-end which dynamically rebuilds the page.

Selecting an exception (or the Add Exception button), loads an exceptions page (Figure 21) in the same way normal periods are loaded. The only difference on loading is that all schedule exceptions are stored in HAF's database. The only difference when saving is that door schedule exceptions are saved to the database. Then a job in azure runs every minute to grab any unprocessed exceptions who's date is the current date and who's start time and end time are between the current time. The grabbed exceptions are then sent to HAF's third party to trigger a door exception override till the specified end time.

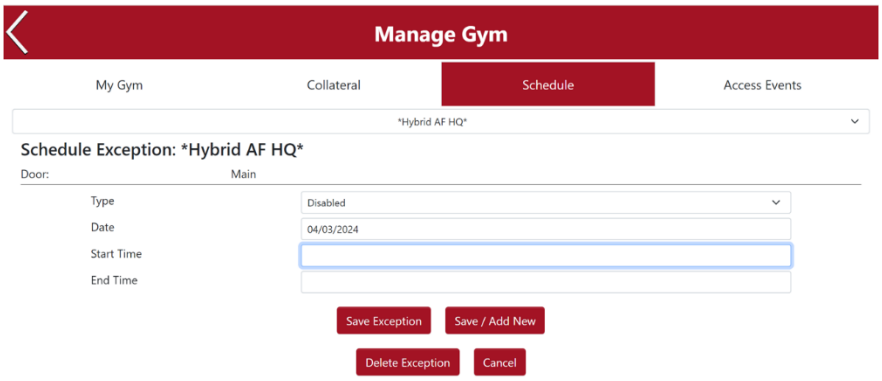

Figure 21: Schedule Exceptions

Other functionality in the schedule are a few toggles on the app disable tap such as "Allow Emergency access - when disabled" and "Send Push Notifications on Emergency Access" which can be seen in Figure 22. When changed, these toggles trigger functions which make ajax calls to the back end to update their respective fields in the HAF database.

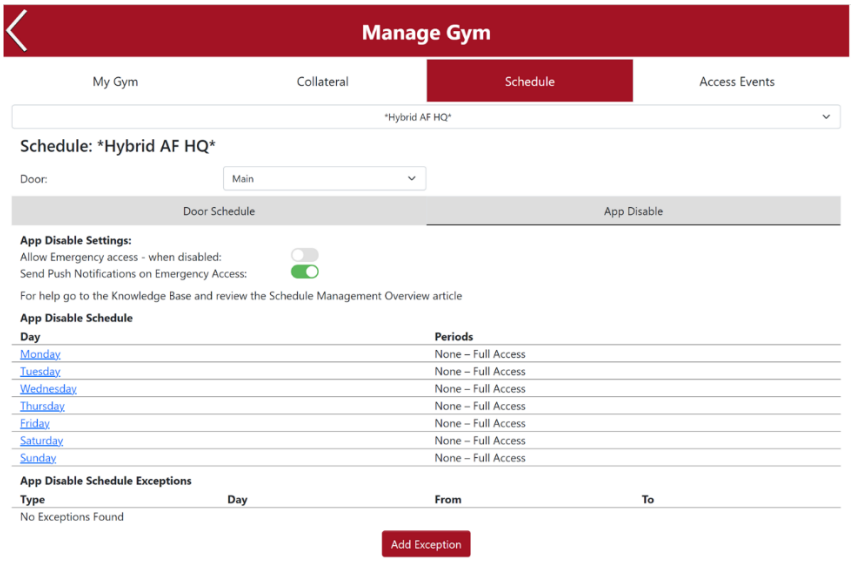

Figure 222: Schedule Door App Disable Tab

The final tab in the Manage Gym section is the Access Events tab (see Figure 23). The Access Events tab functions similarly to the Access Events in the Members section. It is pulled in using an HTML string that was generated by the controller using a partial view and an access events model. It uses the data tables plugin to enable searching, ordering, and exporting. It also uses the Tempus Dominus plugin for setting the date ranges.

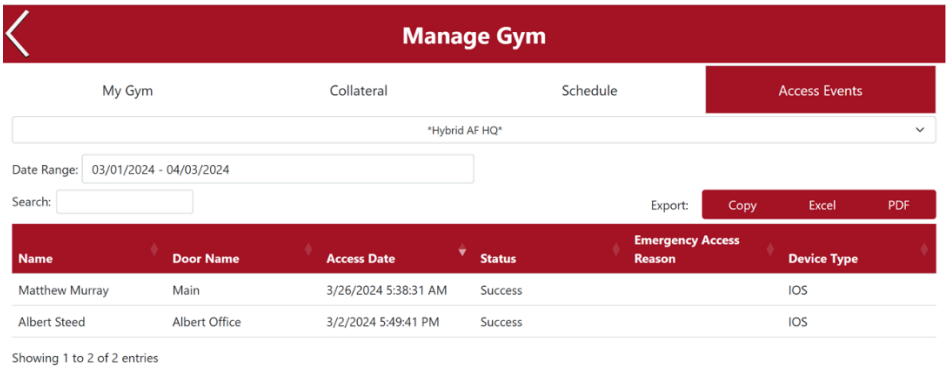

Figure 23: Manage Gym Access Events Tab

Support Settings Page

If the user is a super-admin, the user will see a third tile on the admin menu called "Support Settings" shown here in Figure 24:

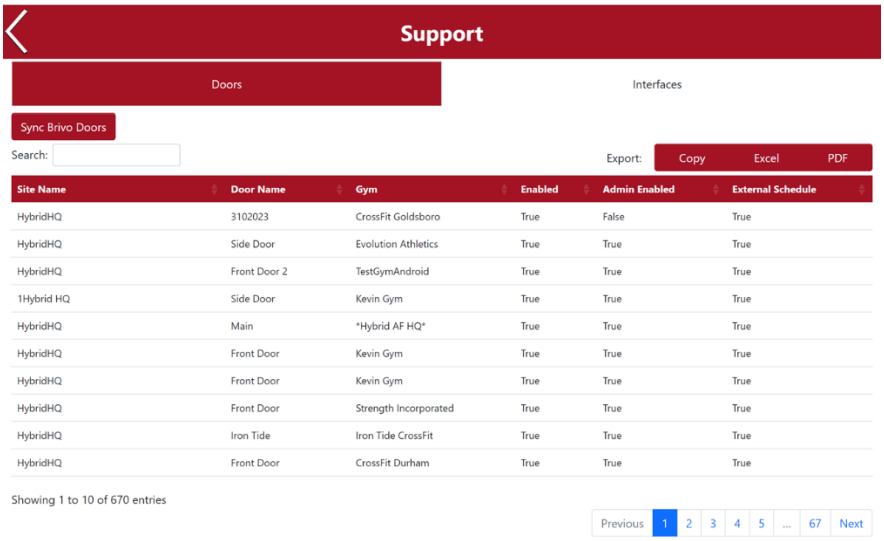

Figure 24: Support Setting Doors Tab

The main portion of the support page is built using a partial view that is rendered to an HTML string which gets rendered on the view. The view has two tabs, “Doors” and “Interfaces”. The doors tab shows a table that is built using the data tables plugin so that it can be ordered, filtered, and exported. On this tab there is also a button called Sync Brivo Doors which grabs the list of doors from our third party’s API and inserts the new ones into a table in the HAF database. Clicking/Tapping on a row launches a modal (see Figure 25), where all the data from the table can be edited apart from “Site Name”. On clicking save in the modal, an ajax call is made to the controller, where the data is updated in the database, and an updated version of the table is passed back to the front end where the modal gets closed and the updated table is dynamically reloaded.

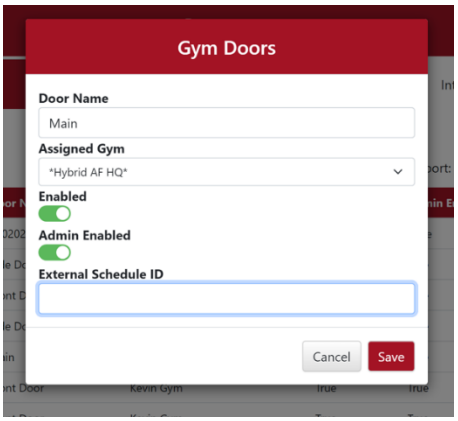

Figure 25: Support Setting Doors Modal

The other tab in the support section is the Interfaces tab (see Figure 26). This tab contains a dropdown of gyms, and upon selecting a gym, a text area appears. If the gym already has interface parameters, the text area will be populated with the interface data (usually an xml string). Saving on this tab makes an ajax call to the controller where the interface parameters for the selected gym get updated.

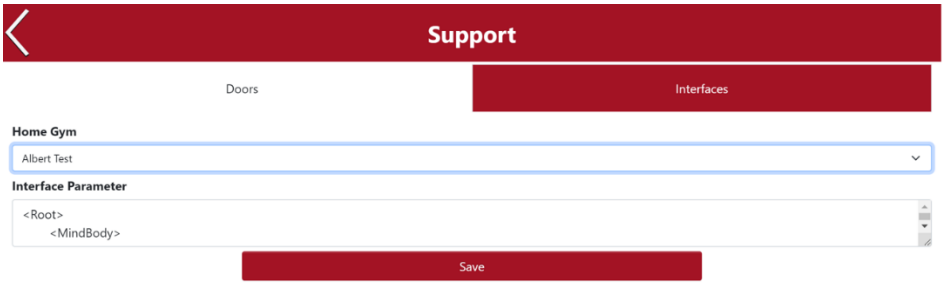

Figure 26: Support Setting Interfaces Tab

The final tile on the main menu is the Logout tile. Clicking/Tapping this tile ends the session and redirects the user back to the login screen.

CHAPTER 5: CONCLUSIONS AND FUTURE WORK

Conclusion

The development of the Hybrid Athlete Foundation (HAF) administrative portal has successfully provided gym owners and administrators with a powerful web-based tool to manage their facilities, members, and access control. By integrating with HAF's existing infrastructure, the portal delivers a familiar and intuitive user experience while introducing new capabilities such as door scheduling, and support settings for super administrators.

The portal's flexible architecture, built on C#, .NET, and Bootstrap, allows for easy maintenance and future expansion. Key security measures, including OAuth authentication and password hashing, ensure the protection of sensitive members and facility data.

Future Work

Future work is planned for the admin portal which HAF hopes will improve it as well as make it more secure. The first is to separate it from the services. Currently the admin portal is built on top of the existing API. What HAF hopes to do in the future is to separate it out so that it can stand on its own, and then use more API calls via requests instead of directly calling the service. While the Admin portal gets separated out, the plan is to also migrate the code from .net framework 4.8 to .net core 4 or 5.

Another plan for the admin portal is to eventually make it so that it can be pulled into the mobile apps, much like the new door schedule system that was built in this project. This will make it so that functionality will be identical between the admin portal and the mobile apps. Meaning any new features only need to be created in one place.