Problem Statement

Attendance is an important part of the educational process. Because attendance is so closely linked to academic success it is often a requirement in order to receive credit for the class. Even in instances where attendance is not required academically, the last two years of the COVID-19 pandemic have taught the world that it is important to track attendance in order to monitor potential exposure to viruses and protect the health and welfare of both students and staff. The purpose of this project is to create an application which can track a student's attendance based on their GPS location or a predetermined passcode without requiring manual input from the professor. This will save time and allow professors to focus on beginning the class and getting the most out of the allotted time. It will also give students an easy way to track their absences and be sure that they are not about to FA (Failure due to Absences) without the fear that the records are not yet up to date. Finally, it provides a valuable resource when combined with other metrics to track virus exposure and protect against another pandemic.

Background

During my freshman year of college, I took a few classes in programming which helped me to learn fundamental techniques and how to think logically. However, since then, I have not engaged in any programming and the skills I learned became rusty. In my job I work with IT and cybersecurity, and the training I received was tailored to those roles, so I did not face any exposure to programming in my 2 years of work experience or extracurricular studies. Because of this my skill level was not adequate at the beginning of the project and I would have to learn almost every aspect of it through research and trial and error.

Research

My project was originally meant to use Apache Cordova which is meant to convert existing HTML and JavaScript into a native application for iOS and Android. This would allow me to write the code once and deploy it on multiple platforms. I started by researching the components of a Cordova compatible web application. I made a very simple prototype in HTML to figure out how to set up a web server. I started researching how to connect to a database using JavaScript, and during this process I decided to switch to jQuery because it makes database connections easier. One thing I learned early on was I had a very flawed understanding on how a webserver interacts with a browser. It took me a while to fully understand the connection between the web server, the database, and what the user sees in the browser. I also did research in Bootstrap for the user interface. I put together a prototype out of HTML, Bootstrap, and a small amount of JavaScript with the intention of testing Cordova's capabilities. Through this prototype, and an increased understanding of how web servers work, I found that Cordova applications do not interact with web servers in the same way that a conventional website does, and because of this, would not be able to support the kinds of database interactions that my project required. This forced a drastic redesign of the project into a purely web-based format and I began research in web frameworks to facilitate this change. I chose Laravel as my framework and started researching how to create a login system. I created a new prototype with a login system I created from scratch only to realize that Laravel contains an optional plugin which can create a login system automatically. I started over again with the new login system and began researching the Model View Controller principle. Prior to working with Laravel I had no knowledge of a model view controller. This was difficult

for me to understand as I often have trouble understanding object-oriented languages, and it was not until much later in development that this concept really made sense. I also did extensive research into PHP as it is the backbone of the Laravel framework and Blade in order to make the front-end pages. Finally, I researched SQL queries and how to utilize the Laravel query builder in order to facilitate database access. In the end, although I had a basic introduction into programming from my freshman year, the final version of this project is made entirely using tools, languages, and techniques researched exclusively for use in this project.

Project Language(s), Software, and Hardware:

The project uses the Laravel framework consisting of PHP and HTML. The project was built using Visual Studio Code and phpMyAdmin but does not require any specific outside software to run. The project does need a webserver in order to function but this server does not need to have any specific hardware requirements.

Project Requirements

Requirement #1

Description: App can determine if a student is inside a given room to within 10 yards of the building.

Rationale: This is to determine a student's location in order to see if they are located at the class location

Fit Criterion: If a student is inside the building the app should count them as present. If they are more than 10 yards outside the building they should be counted as absent

Dependencies: None

Conflicts: None

Completed: Yes

Test Case: #3

Priority: 1

Requirement #2

Description: App can compare a user's location to the location of their class to determine if they are present

Rationale: This is to determine not only a student's location but to be able to verify the student is in the same location as the class.

Fit Criterion: If the location of the student matches the location of the class, they should be marked present. If they are not located within the accepted location of the class, they are marked absent.

Dependencies: Requirement #1

Conflicts: None

Completed: Yes

Test Case: #3

Priority: 1

Requirement #3

Description: App captures the time of check in and compares it to the time of the class

Rationale: This is to determine that a student has checked into a class at the correct time for the class.

Fit Criterion: If the time of the check in is within 15 minutes of the start of the class, the student is counted as present. If the check-in is after 15 minutes but before the end of the class the student is tardy. If the check in is not within the time period of the class the student is not counted present or tardy.

Dependencies: None

Conflicts: None

Completed: Yes

Test Case: #3

Priority: 1

Requirement #5

Description: A student should only be able to view their own attendance record

Rationale: This is to ensure the security of a student's personal information

Fit Criterion: If a student views their attendance record, they should only see entries which are their own attendance

Dependencies: Requirement #8

Conflicts: None

Completed: Yes

Test Case: #5

Priority:1

Requirement #6

Description: A Student should be able to view their attendance record for only the current semester

Rationale: This is to allow students to see how many days they have missed in that semester

Fit Criterion: If a student views their attendance record, they should only see entries which are from the current semester

Dependencies: Requirements #8

Conflicts: None

Completed: Yes

Test Case: #5

Priority: 1

Requirement #7

Description: A professor should only be able to see the attendance records for classes which they teach

Rationale: This is to protect the personal information of the students in the class

Fit Criterion: If a professor views the attendance records for their classes, they should not see entries from any other professors

Dependencies: Requirement #9

Conflicts: None

Completed: Yes

Test Case: #7

Priority: 1

Requirement #8

Description: A Student should be able to view their attendance record

Rationale: This is to allow student to view the status of their attendance.

Fit Criterion: If a student requests to view their attendance information they should be able to view an attendance report within the app.

Dependences: None

Conflicts: None

Completed: Yes

Test Case: #5

Priority: 1

Requirement #9

Description: A professor should be able to see a report of attendance

Rationale: This is to allow professors to review the attendance of their classes

Fit Criterion: If a professor requests a report of attendance, they should be able to see a report of the attendance of their classes within the professor's portal website.

Dependencies: None

Conflicts: None

Completed: Yes

Test Case: #7

Priority: 1

Requirement #10

Description: The app will be available on Android and IOS devices

Rationale: This is to allow a wide range of students to access the app

Fit Criterion: The app will run on the Android and iOS operating systems

Dependencies: None

Conflicts: None

Completed: This requirement longer applies to project

Test Case: N/A

Priority: N/A

Requirement #11

Description: The professor portal will be verified to work on Google Chrome and Firefox browsers

Rationale: This will allow professors to access the portal on commonly used modern browsers

Fit Criterion: The professor portal website will be fully functional on Firefox and Google Chrome

Dependencies: None

Conflicts: None

Completed: Yes

Test Case:

Priority: 1

Requirement #12

Description: If a student has arrived 15 or more minutes late, this will be counted as one third of an absence

Rationale: This will enforce the school's policy on late arrivals

Fit Criterion: If the student checks in 15 or more minutes after the start of the class the app will give them a tardy.

Dependencies: Requirement #3

Conflicts: None

Completed: Yes

Test Case:

Priority: 1

Requirement #13

Description: Professors will be able to see the total absences for students in their classes in the professor's portal.

Rationale: This will allow professors to easily see how many absences each student has in their class

Fit Criterion: If a professor uses the professor portal to view attendance records, they will be able to see a list of total absences for student in their class.

Dependencies: Requirement #9

Conflicts: None

Completed: Yes

Priority: 1

Requirement #14

Description: Students should receive a notification if they are 2 absences or less from receiving an FA.

Rationale: This is to give students a warning when they are close to failing a class

Fit Criterion: If the students absence count is two or less from the amount which constitutes an FA, they will receive a notification.

Dependencies: None

Conflicts: None

Completed: No

Test Case: N/A

Priority: 2

Requirement #15

Description: Professors will receive a notification if one of their students is one absence away from failing

Rationale: This is to give professors a heads up that one of their students is close to failing.

Fit Criterion: If a student's absences is one less than the amount to receive an FA then the professor should receive a notification in the browser.

Dependencies: None

Conflicts: None

Completed: No

Test Case: N/A

Priority: 2

Requirement #16

Description: Students should receive a notification if they have accrued enough absences to receive an FA for a class.

Rationale: This is to notify students that they have failed a class.

Fit Criterion: If a student has an absence count high enough to receive an FA, they will receive a notification.

Dependencies: None

Conflicts: None

Completed: No

Test Case: N/A

Priority: 2

Requirement #17

Description: Professors should receive a notification is one of their students has failed that class.

Rationale: This is to notify professors that one of their students has failed the class

Fit Criterion: If a student has an absence count high enough to receive an FA the professor will receive a notification in the browser.

Dependencies: None

Conflicts: None

Completed: No

Test Case: N/A

Priority: 2

Requirement #18

Description: The professor can manually update attendance records for their students

Rationale: This is to allow professors to retain complete control over the attendance records for their classes and to ensure that the attendance records are accurate

Fit Criterion: The professor portal will give professors the ability to change existing attendance records.

Dependencies: None

Conflicts: None

Completed: Yes

Test Case:

Priority: 1

Requirement #19

Description: The professor will be able to manually input attendance rather than use the automated system.

Rationale: This is to allow the professor to input attendance themselves if they do not wish to use the automated system

Fit Criterion: If a professor disables automatic attendance, they will be able to input attendance themselves.

Dependencies: None

Conflicts: None

Completed: Yes

Test Case:

Priority: 1

Requirement #20

Description: The app will support alternate attendance methods which the professor can enable/disable at will.

Rationale: This is to offer alternate attendance methods other than GPS

Fit Criterion: The professor will be able to enable/disable alternate attendance tracking methods through the professor's portal

Dependencies: None

Conflicts: None

Completed: Yes

Test Case:

Priority: 1

Requirement #21

Description: The app will support attendance tracking via a QR code

Rationale: This is an alternate method of tracking attendance

Fit Criterion: If the professor does not want to rely on GPS, they will be able to enable a QR code which can be displayed on a screen for students to scan in order to log their attendance

Dependencies: Requirement #20

Conflicts: None

Completed: No

Test Case: N/A

Priority: 2

Requirement #22

Description: The app will be able to generate a unique QR code each time the QR method is used

Rationale: This will be a QR code containing a randomized identifier used to identify the specific class period/time that the code was generated.

Fit Criterion: When the QR method is used the app will generate a unique QR code for the professor to display.

Dependencies: Requirement #21

Conflicts: None

Completed: No

Test Case: N/A

Priority: 2

Requirement #23

Description: The app will be able to read a QR code and use it to determine a student's attendance

Rationale: The QR code will be read by the app and the student will be counted present for the class period the code is for.

Fit Criterion: If a QR code is scanned the app will be able to determine what class period it is for and mark the user as present if they meet all other present criteria

Dependencies: Requirement #21, Requirement #22

Conflicts: None

Completed: No

Priority: 2

Requirement #24

Description: The app will allow a professor to assign a passcode for the class period

Rationale: This is to give professors a simple way to track attendance without GPS

Fit Criterion: The professor can assign a passcode which, when entered by a student, will mark that student as present so long as they are within the time window of the class.

Dependencies: None

Conflicts: None

Completed: Yes

Test Case:

Priority: 1

Requirements #26

Description: Students will only be required to have the app open and running at the time of sign in

Rationale: This is to reduce the number of errors and to make it easier for students to be counted properly. It also helps to account for students using popular security features regarding location usage. Fit Criterion: If a student signs in they will not be tracked afterward.

Dependencies: None

Conflicts: None

Completed: Yes

Test Case: #3

Priority: 1

Requirements #27

Description: The client-side hardware requirements are that the device has access to the internet and has access to a location API.

Rationale: This is the basic requirements for the program to run as a web application.

Fit Criterion: If a clients device meets these specifications the program will run

Dependencies: None

Conflicts: None

Completed: Yes

Test Case:

Priority: 1

Requirement #28

Description: The server-side hardware requirements are a Linux system with uninterrupted internet access.

Rationale: This is the basic requirements for the program to run as a web application.

Fit Criterion: If a server meets these specifications, it should be able to support the program

Dependencies: None

Conflicts: None

Completed: Yes

Test Case:

Priority: 1

Project Implementation Description & Explanation

When the user navigates to the site URL the first page that they come to is the login page (see fig. 01). The site has no landing page other than the login page because it is not designed to be used by the general public. It is intended for use by an organization internally and should only be accessed by authorized users. The site does have a registration page which can be accessed by clicking the "Register" button in the top right corner of the login page. It is recommended that user accounts be created in advance by administrators and then provided to the users. However, the register page serves as an alternative depending on the client's organizational needs. On the registration page, the user will be prompted to enter their name, email address and password, and to select a user type, Student or Instructor (see fig 02).

Once the user logs in or registers an account they will be directed to their dashboard. The specific dashboard they are provided depends on if they are a student or instructor. Beginning with the student dashboard, the user is presented with three discrete actions: "Check-In with GPS", "Enter Passcode", and "View Attendance" (see fig 03). The first option, Check-In With GPS, will automatically attempt to sign the user into class if one of the classes they are enrolled in is currently in progress, and they are in the correct location. The user will then be shown a message saying whether or not the check in was successful. This is intended as the primary method of submitting attendance and is designed to function with minimal interaction from the instructor. The second option, Enter Passcode, operates very similar to the GPS mode but will prompt the user for a passcode provided by the professor rather than use the student's GPS (see fig 04). This is useful for cases in which 1) GPS is unreliable or not applicable such as classrooms with limited cell service, 2) classes taking place outside of their usual location, or 3) instances where online students are required to attend a live class. To limit misuse, the passcode

will only work during the time period the class is active. In cases where a class period is being held at an irregular time, attendance will need to be taken manually by the instructor.

The final action available to the student is to “View Attendance”. Clicking this button will take the user to a page showing the total absences for each class (see fig 05). This is useful for the student to judge their attendance at a glance, especially as it relates to FA. In order to view more detailed information about their attendance, the user can click on the name of a class to see the status of individual class periods. This will take them to a new page where the date of each class period is displayed along with whether they were Present, Absent, or Tardy (see fig 06). This will only display class periods which have already happened. The system updates as soon as a class ends, if attendance was taken manually this will show up as an absence until the instructor updates the records.

The instructor dashboard contains four discrete actions: View Current Class, View Previous Classes, Change Class Location, and Display Passcode (see fig 07). The first button, View Current Class, will take the instructor to a table showing the status of each student in whichever class is currently occurring. This page will show each student’s name and whether they are currently Present, Absent, or Tardy (see fig 08). The page must be refreshed for changes to be reflected. The Edit button next to the status of each student allows the instructor to update or input attendance manually. Clicking it routes the user to a page where they are asked to choose one of the three possible statuses for the selected student (see fig 09). Once they make a selection they are returned to the previous page. View Previous Classes works very similarly to View Current Class. The main difference is that upon clicking this option, the user must click on a course section and then the date of the class period they wish to edit. Once they are viewing the class period of their choice the UI and manual entry features are the same as the live view.

The next option the instructor has is the ability to change the location of a class period. Upon selecting this option, the instructor is asked to select a section they wish to edit along with the date of the upcoming class. Upon choosing a class period, the user is able to choose a location from a predetermined list of valid locations. This could be specific classrooms, buildings, or other locations based on organizational needs. If the class is going to be held in a nonstandard location the passcode method should be used.

The final action is the ability to display a passcode. If there is a class currently running, the passcode for that class period will display onscreen. This way the instructor can show their screen to the class or distribute the passcode a different way. If there is no current class a message will appear. The passcode will become inaccessible once the class period ends.

a link to the source code repository:

https://github.com/CodyJCain/AttendanceApp

Test Plan

Introduction:

This document covers all testing activities for the uAttend attendance app. The goal is to verify that all requirements for project completion are satisfied. The constraints are a lack of budget for devices and testing equipment, and an end date of April 13th.

Test Items:

uAttend version 1.0

Features to be tested:

- Attendance verification
    - Requirement 1
    - Requirement 2
    - Requirement 3
    - Requirement 12
    - Requirement 24
    - Requirement 25
    - Requirement 26
    - Requirement 27
- Student can see their current attendance in the app
    - Requirement 5
    - Requirement 6
    - Requirement 8
    - Requirement 14
    - Requirement 16
- Professors can monitor the attendance of their classes
    - Requirement 4
    - Requirement 7
    - Requirement 9
    - Requirement 11
    - Requirement 13
    - Requirement 15
    - Requirement 17
    - Requirement 18
- Professors can configure the methods of attendance verification
    - Requirement 11
    - Requirement 19
    - Requirement 20
    - Requirement 21
    - Requirement 24

Approach:

Test cases will be developed prior to the completion of each major feature. A test database will also be populated with a set of test data to represent an instructor with a small class. Each feature will be tested with their respective test plan as soon as they become feature

complete. One the entire project is complete all test cases will be run again to ensure that all features are still operating properly. Once all test cases pass simultaneously then testing will conclude.

Item Pass/Fail Criteria:

Items will be determined to have passed if they match the expected result specified in the test cases. See test cases for more details.

Test Results

My tests revealed a couple areas of improvements to be made. This biggest problem is that the GPS attendance system, which was the first part of the project to be created, had broken after changes were made to the database later in the project. It was also discovered that this system was not accounting for Tardy students. Another issue I found was that the ability to view total absences was not implemented for professors. These things were fixed and the test cases were run again.

Challenges Overcome

As I stated in my background section, I do not consider myself a programmer, which made this project extremely challenging. Lack of experience and knowledge in web applications were the main challenges during this project. I did not have enough knowledge to know what I did not know or how to ask for help. This made researching difficult because due to not knowing a given concept or technique existed so I never thought to research it. I often attacked a problem

with the best solution I could think of from my existing knowledge to limited success only to find in later research that there were much better solutions that were more efficient or more reliable which caused me to do the same work multiple times. The first challenge I faced was in regards to the relationship between a database, web server, and browser. Prior to this project, I thought I knew how these objects interacted with each other. It made working with Cordova extremely confusing because I did not realize at first that Cordova applications do not use traditional webservers in the same way a website does. I would search for things on the internet and ask colleagues at work who were familiar with these concepts but the questions I was asking made no sense to those who actually understood the concepts I was asking about and I could not find the information I was looking for. Eventually, I reached out to the network administrator and database administrator at my job and they explained how the database, web server, and browser interact. This gave me enough of an understanding of how things are supposed to work to realize that Cordova was different and was not suited to my application. If I knew this information beforehand, I could have saved myself months of work and frustration. Lack of experience followed me through the project. When I switched to Laravel, the first thing I started working on was a login system. I spent a few weeks creating a login system because I did not know that Laravel has a built-in login system as an optional plugin. By the time I was informed of this, the other parts of my project were incompatible with the built it system and I had to start over again. I eventually overcame the experience gap by brute force. I did a lot of research to the point that I spent far more time researching than I did actually writing code. On top of that, I took an approach of trial and error until something was successful. Once I found one successful method, I could use that same method all over the site. This took a lot of effort especially in the beginning, but as the project grew, I began to have more and more of a knowledge base to work

from. A great example of this is how I pass data between different pages. This was one of my largest roadblocks when I switched over to Laravel. I tried many different methods in order to send the contents of one variable to a variable on a different page and it was taking a lot of time. Eventually I found that I could reliably send data using formatted URLs. This is far from the best, most efficient, or most secure approach, but it worked and let me move on to other things. I tend to be a perfectionist especially when it comes to school work. I had to overcome that urge in order to get the project done on time. Because of the numerous restarts, the current version of the project was only started in January of 2022. I soon found that any solution, even if it was not the best solution, would let me keep moving forwards and keep making progress. There were times when progress slowed because I got stuck on a single problem or bottleneck and it was a challenge to accept that it was impossible to make the project perfect and finish it in time. This has been the most challenging project thing I have ever done. It was difficult to work on something so far outside of my experience, knowledge level, and general sphere of influence. I had to learn new concepts, learn new applications and tools, perform extensive research, and reach out to work colleagues in order to produces a working final product.

Future Enhancements

Email notification system including alerts for low attendance

Ability for professors to use QR codes rather than a passcode – The passcode controller can already accept a QR code containing a URL. However, there is no way for the system to generate the QR code automatically. Also, it would need to be tested to see if the login system interacts with this.

Ability to set as custom class passcode – This would allow the instructor to set a custom passcode for each class period rather than the preset codes the app currently uses.

Add password reset/recovery options – This relies on the ability to send emails which is currently not configured.

Ability to set all students in a class to a given status – This is a quality of life improvement which would make manual attendance faster and easier.

Known Issues

Security vulnerability from potential ability to manipulate formatted URLs used to pass data between pages.

Screenshots

Figure 01

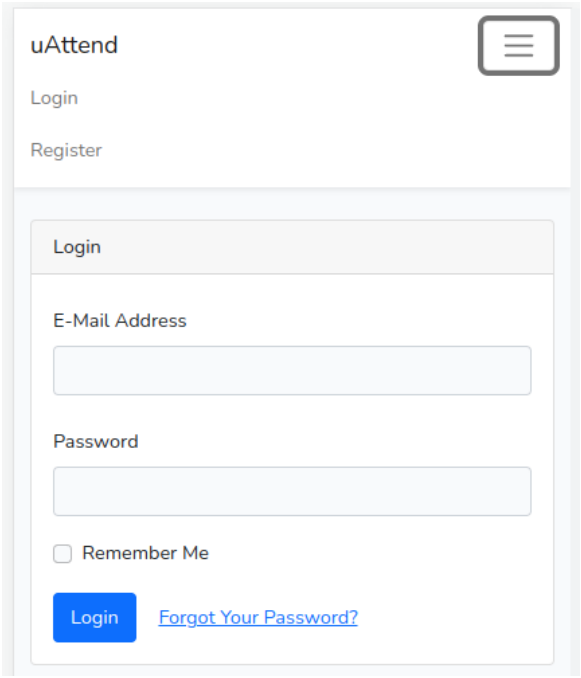

uAttend
Login
Register
Login
E-Mail Address
Password
Remember Me
Login
Forgot Your Password?

Figure 02

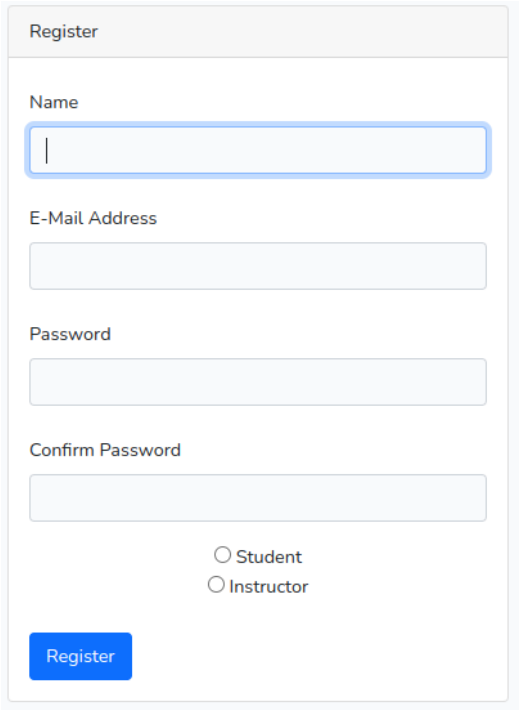

Register
Name
E-Mail Address
Password
Confirm Password
Student
Instructor
Register

Figure 03

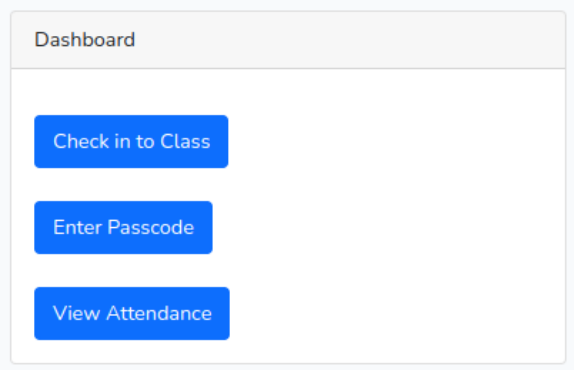


Figure 04

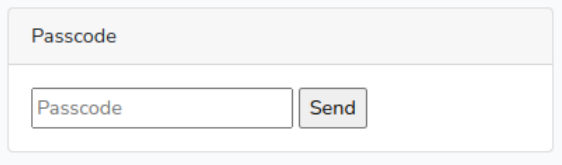


Figure 05

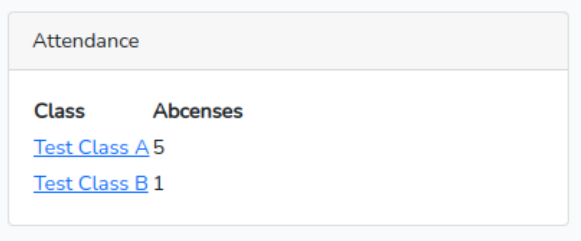


Figure 06

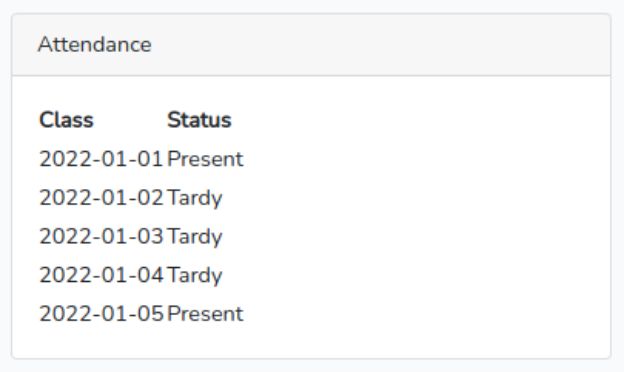


Figure 07

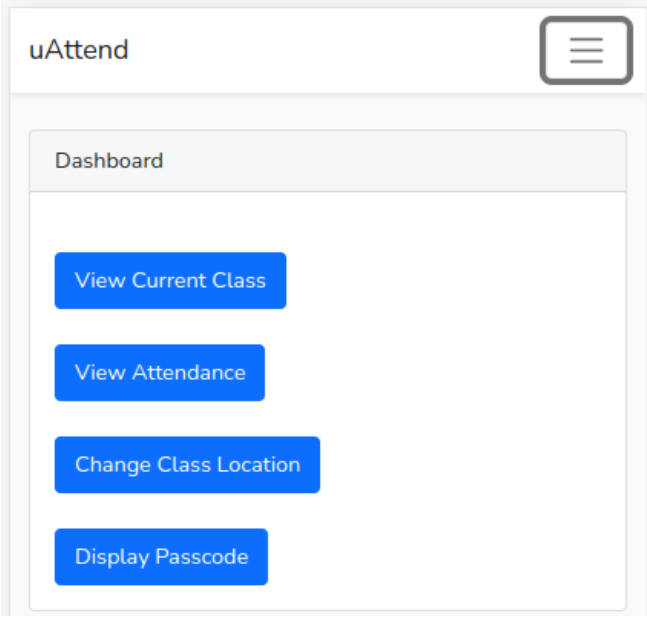


Figure 08

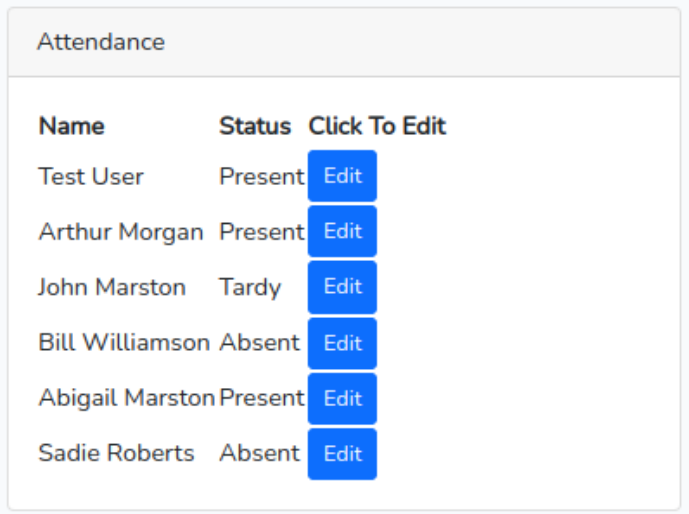


Figure 09

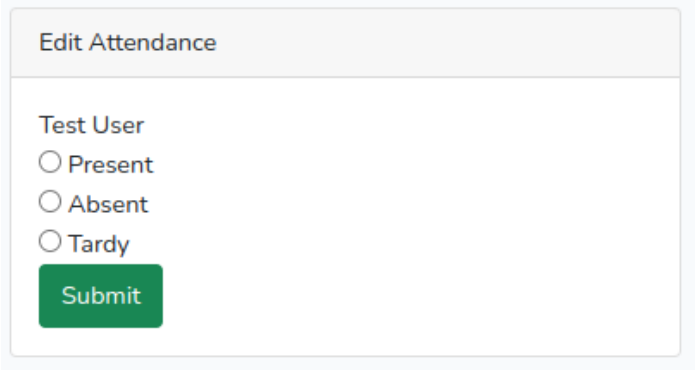


Figure 10

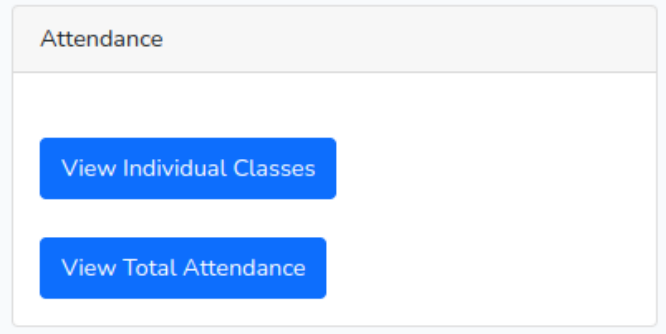


Figure 11

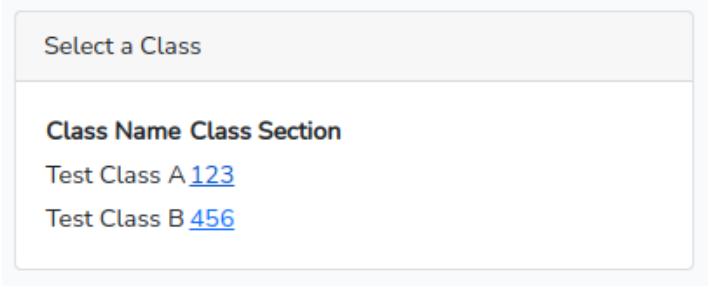
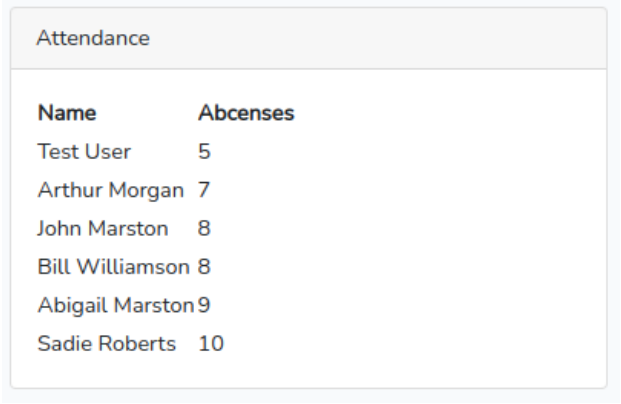

Select a Class
Class Name Class Section
Test Class A 123
Test Class B 456

Figure 12


Attendance
Name Abcenses
Test User 5
Arthur Morgan 7
John Marston 8
Bill Williamson 8
Abigail Marston 9
Sadie Roberts 10