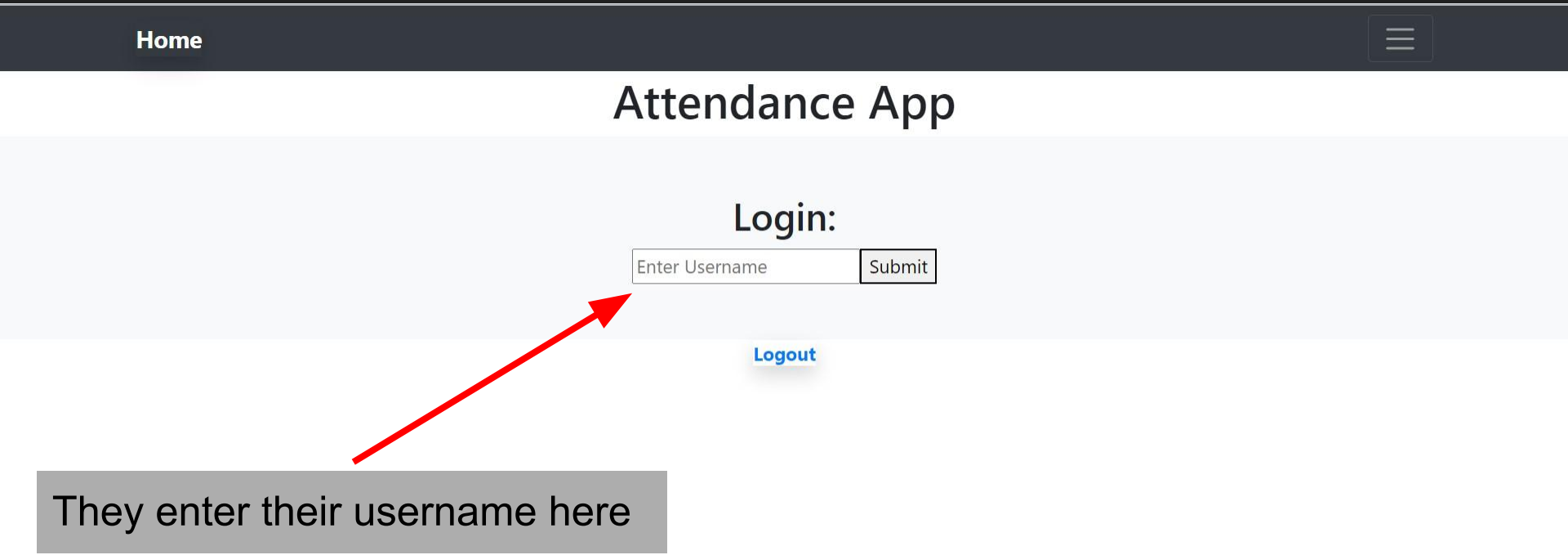
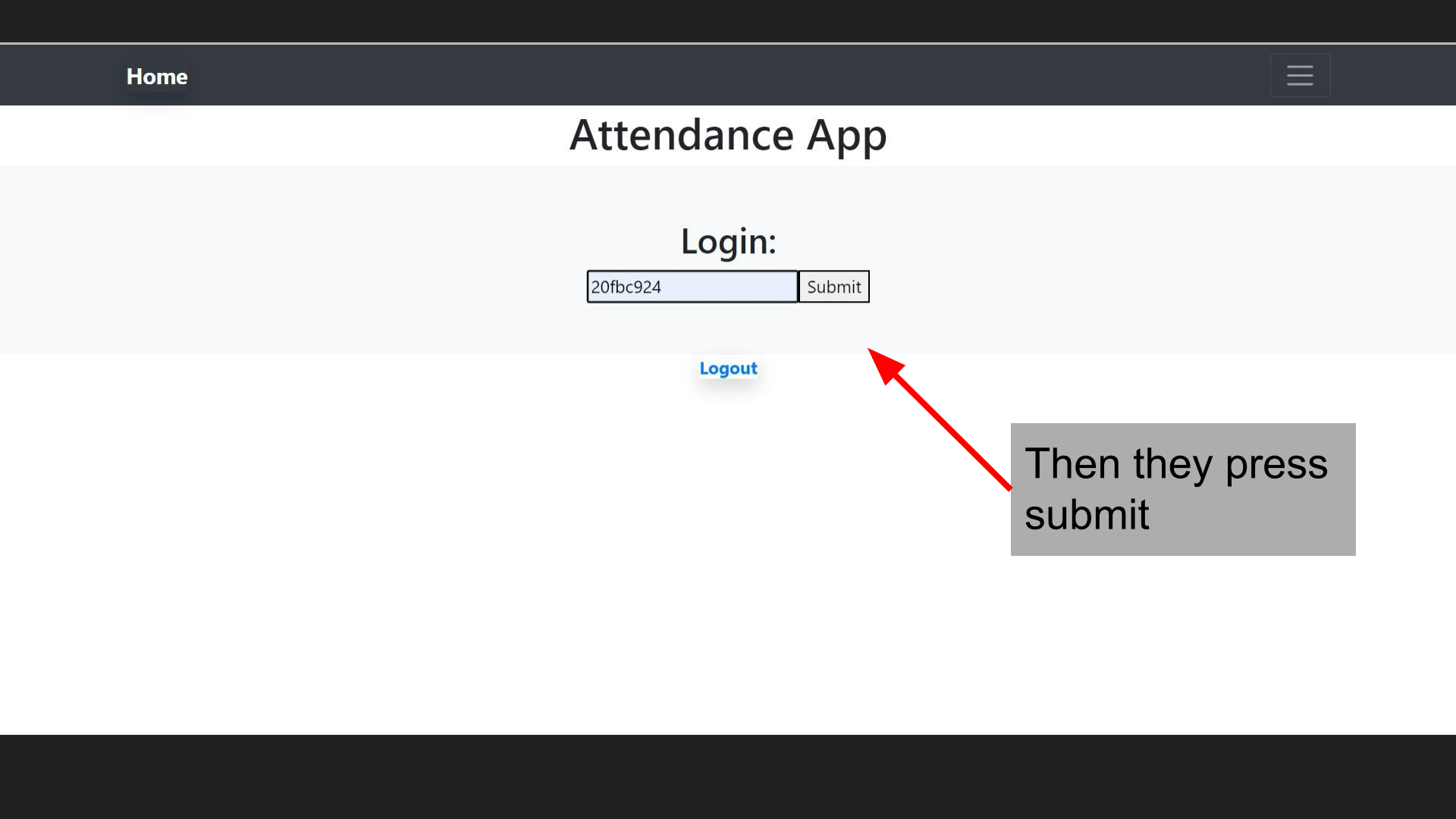
Attendance App
Final Presentation

By Cal Colistra, Adrienne Loc, Chris Nolan,
Naresh Kalluri, and Alyssa Chatman

Scenario #1

- It is the first day of the semester and a student doesn’t know what time their class meets
- They pull up the attendance app …


Home
Attendance App
Login:
Enter Username
Submit
Logout
They enter their username here


Home
Attendance App
Login:
20fbc924
Submit
Logout
Then they press submit

Home

Attendance App

20fbc924's Classes

Calculus

See More

Software Engineering

See More

testClass

See More

Logout

Home

Attendance App

Then they would choose the class they are worried about.

In this case it is **testClass**

20fbc924's Classes

Calculus

See More

Software Engineering

See More

testClass

See More

Logout

Home

Attendance App

testClass

Todays's Date: **12-13-2022**
You're next class is: **Tue Dec 13 2022** at **1:30**

Class Date	Time of Swipe	Attendance Mark
Tue Dec 13 2022	**TBD**	**TBD**
Thu Dec 15 2022	**TBD**	**TBD**
Tue Dec 20 2022	**TBD**	**TBD**
Thu Dec 22 2022	**TBD**	**TBD**

Attendance Statistics

Present	0 days
Late	0 days
Absent	0 days
Classes Left	4 classes

Here it shows when the students next class meeting is and at what time

Attendance App

testClass

Todays's Date: **12-13-2022**

You're next class is: **Tue Dec 13 2022** at **1:30**

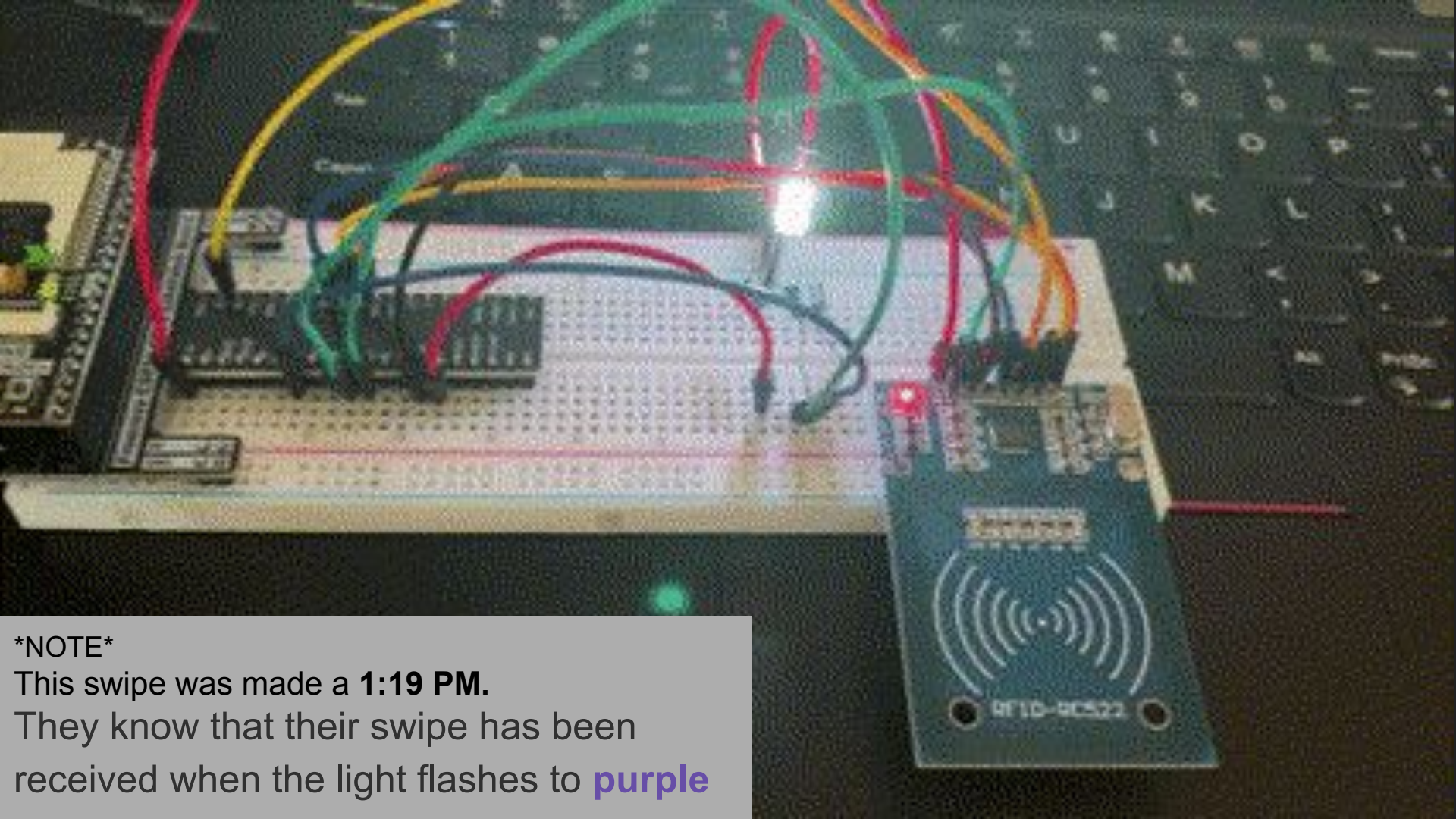
Class Date	Time of Swipe	Attendance Mark
Tue Dec 13 2022	**TBD**	**TBD**
Thu Dec 15 2022	**TBD**	**TBD**
Tue Dec 20 2022	**TBD**	**TBD**
Thu Dec 22 2022	**TBD**	**TBD**

Attendance Statistics

Present	0 days
Late	0 days
Absent	0 days
Classes Left	4 classes

Present Late Absent

They show up to class and swipe their ID card …



NOTE

This swipe was made a **1:19 PM.**

They know that their swipe has been received when the light flashes to **purple**

Home

Attendance App

testClass

Todays's Date: **12-13-2022**

You're next class is: **Tue Dec 13 2022** at **1:30**

Class Date	Time of Swipe	Attendance Mark
Tue Dec 13 2022	**1:19:03 PM**	**Early**
Thu Dec 15 2022	**TBD**	**TBD**
Tue Dec 20 2022	**TBD**	**TBD**
Thu Dec 22 2022	**TBD**	**TBD**

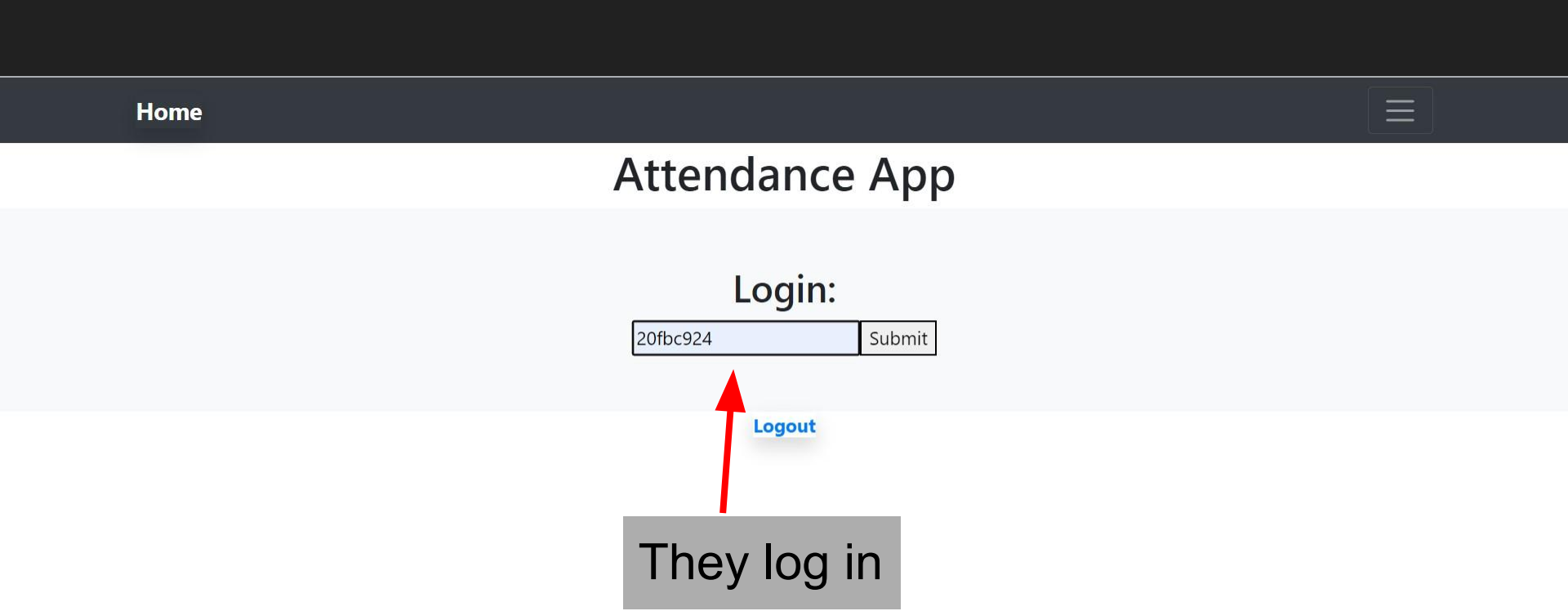
Attendance Statistics

Present	1 days
Late	0 days
Absent	0 days
Classes Left	3 classes

They log back in and check their attendance history

Scenario #2

- A student realizes they have missed a lot of classes
- They know they are only allowed 4 absences or else they fail the course
- The problem is that they can’t remember exactly how many classes they have missed


Home
Attendance App
Login:
20fbc924
Submit
Logout
They log in

Then they would choose the class they are worried about.

In this case it is testClass

Home

Attendance App

20fbc924's Classes

Calculus

See More

Software Engineering

See More

testClass

See More

Logout

Home

Attendance App

testClass

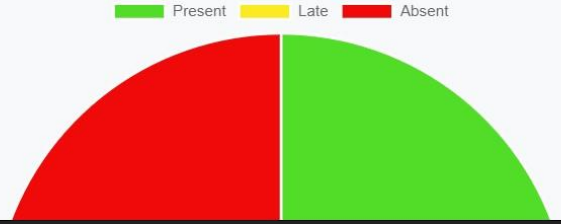
Todays's Date: **12-13-2022**

You're next class is: **Wed Dec 14 2022** at **1:32**

Class Date	Time of Swipe	Attendance Mark
Mon Nov 28 2022	**1:31:00 PM**	**Early**
Wed Nov 30 2022	**1:22:00 PM**	**Early**
Fri Dec 02 2022	**1:34:00 PM**	**Present**
Mon Dec 05 2022	**1:47:00 PM**	**Late**
Wed Dec 07 2022	**No Swipe**	**Absent**
Fri Dec 09 2022	**No Swipe**	**Absent**
Mon Dec 12 2022	**No Swipe**	**Absent**
Wed Dec 14 2022	**TBD**	**TBD**
Fri Dec 16 2022	**TBD**	**TBD**

Attendance Statistics

Present	3 days
Late	1 days
Absent	3 days
Classes Left	2 classes



Here it shows how many times they were marked as absent

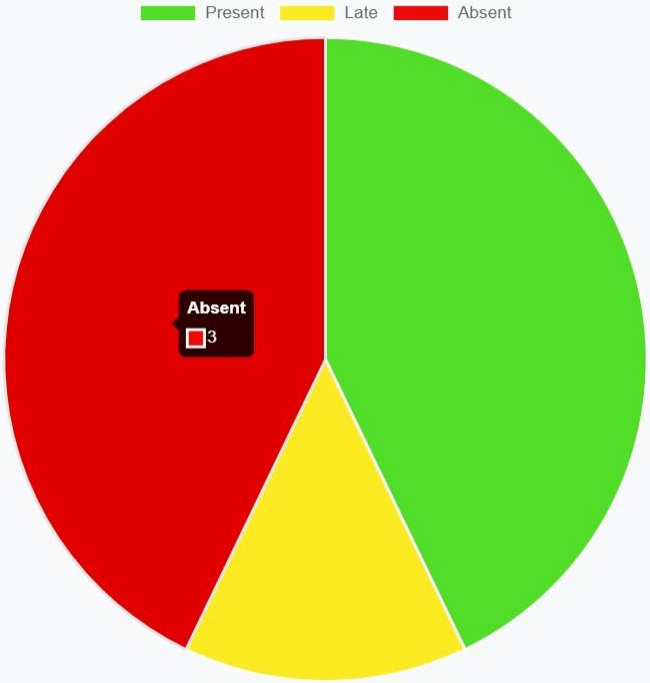
Now they know that they can only miss 1 more class before exceeding the limit and failing the course

Fri Dec 09 2022	**No Swipe**	**Absent**
Mon Dec 12 2022	**No Swipe**	**Absent**
Wed Dec 14 2022	**TBD**	**TBD**
Fri Dec 16 2022	**TBD**	**TBD**

Absent 3 days

Classes Left 2 classes

They may also scroll down and hover their mouse over the pie chart

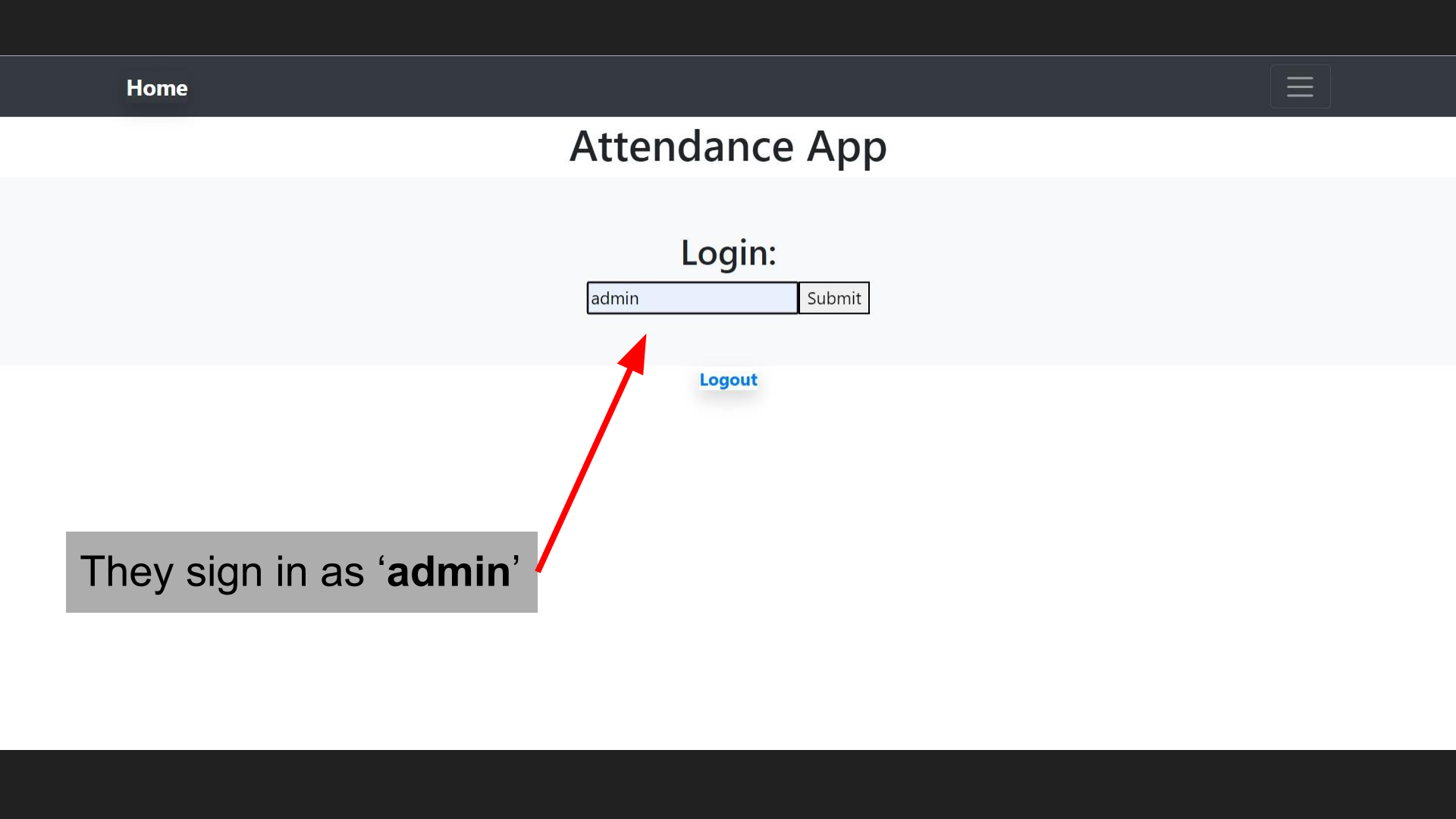


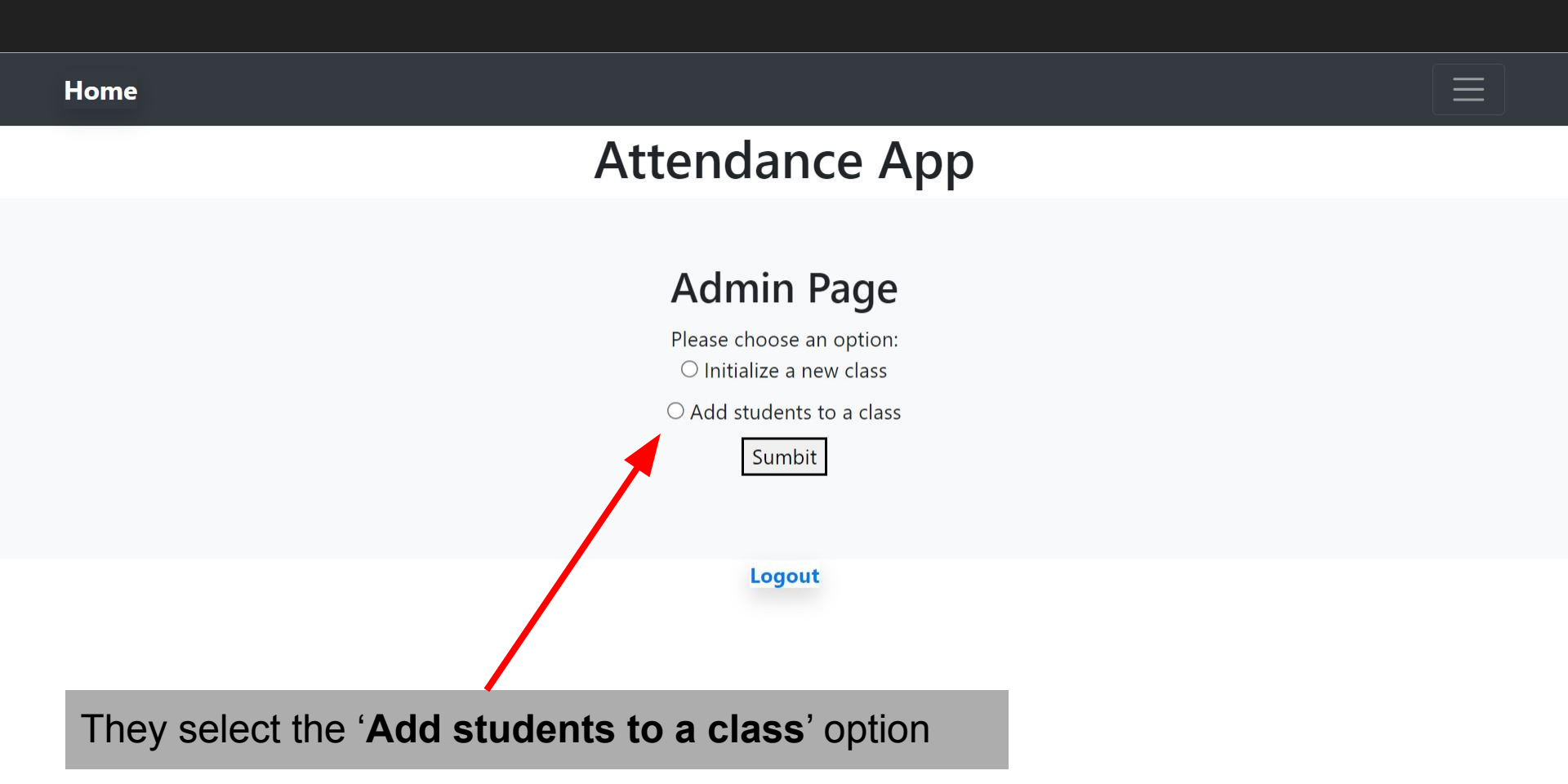
Back

Logout

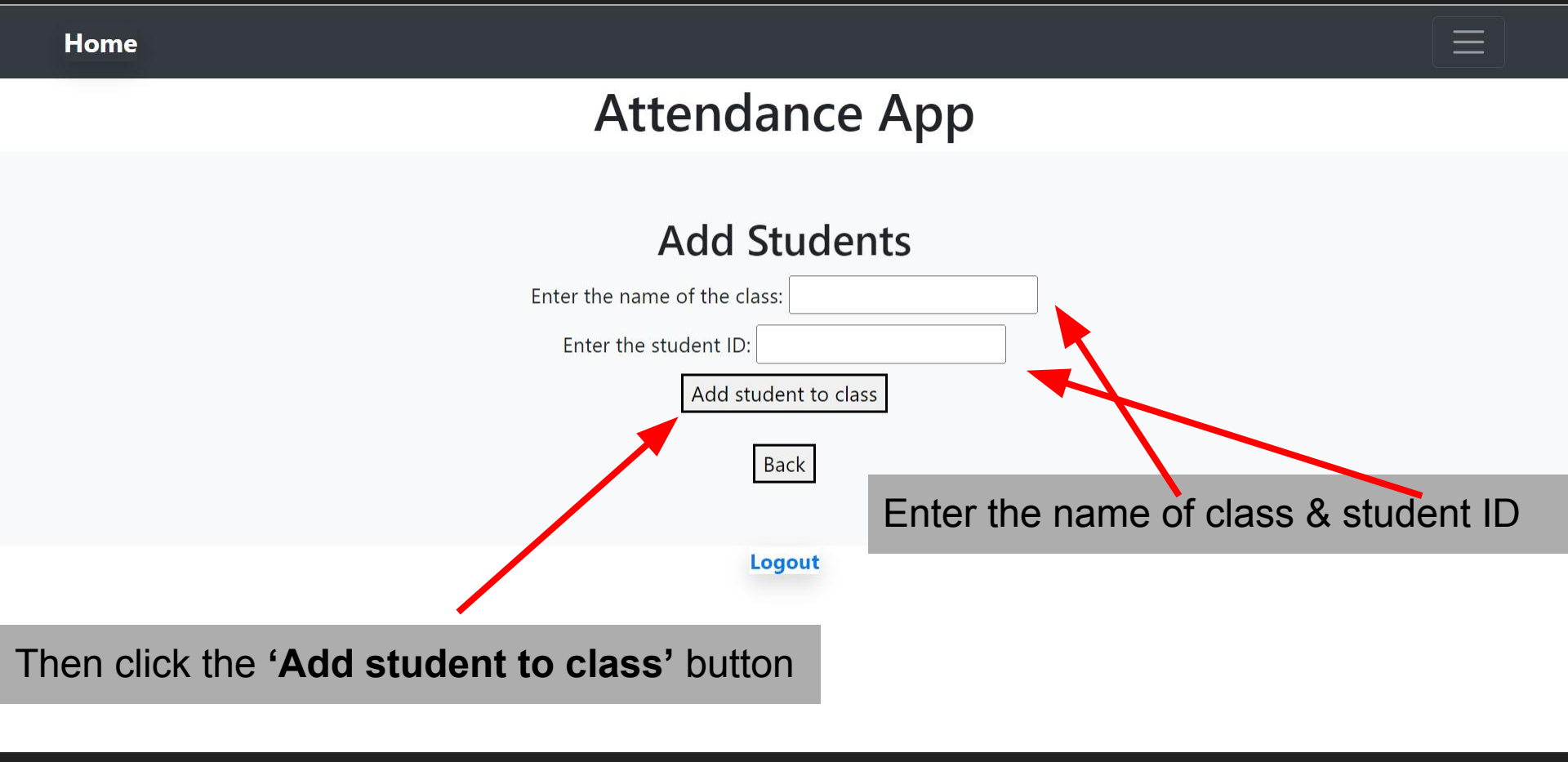
Scenario #3

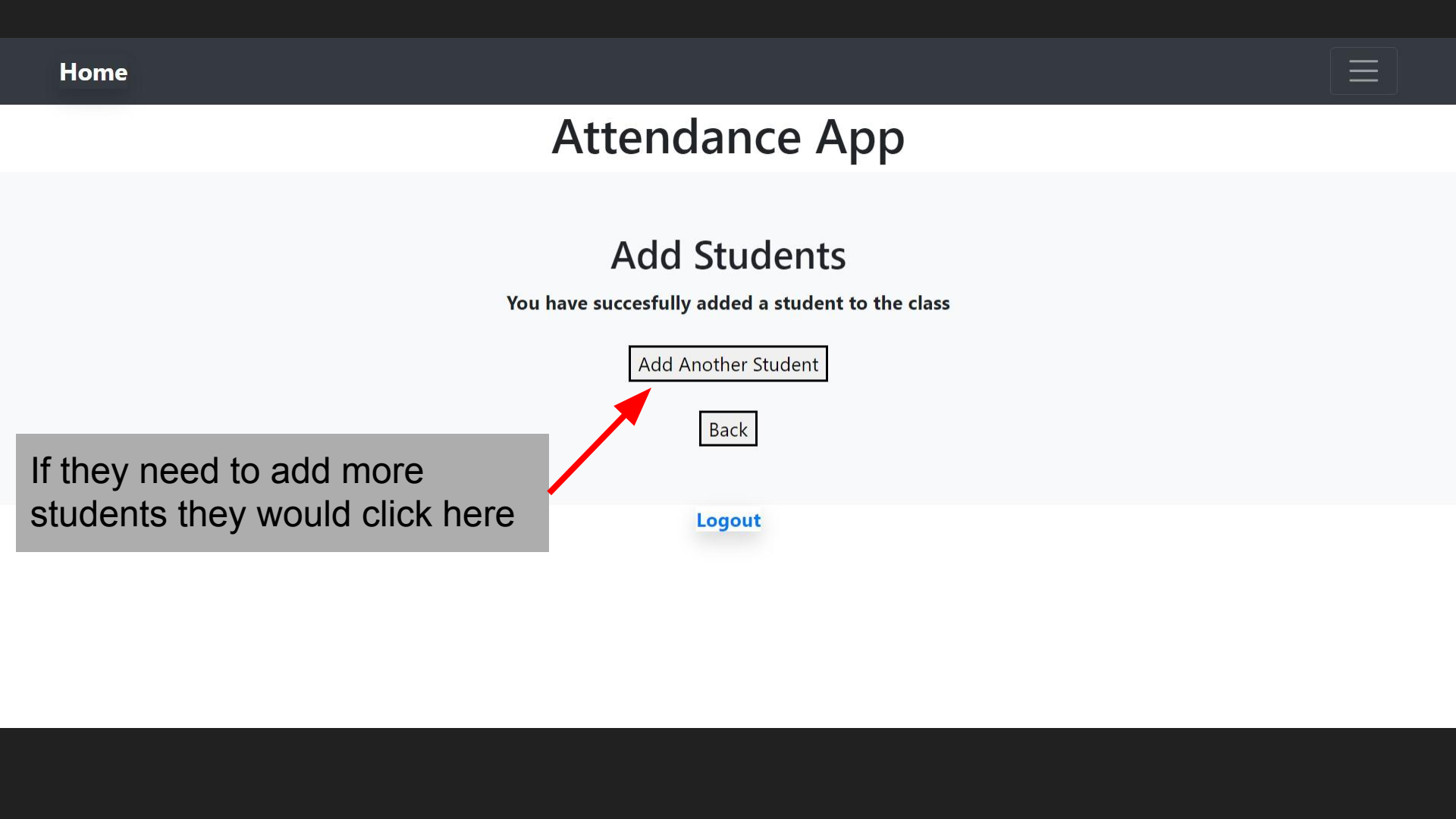
- A student has registered for a class after the semester has already started
- A teacher or administrator wants to add them to the class


Home
Attendance App
Login:
admin
Submit
Logout
They sign in as ‘admin’



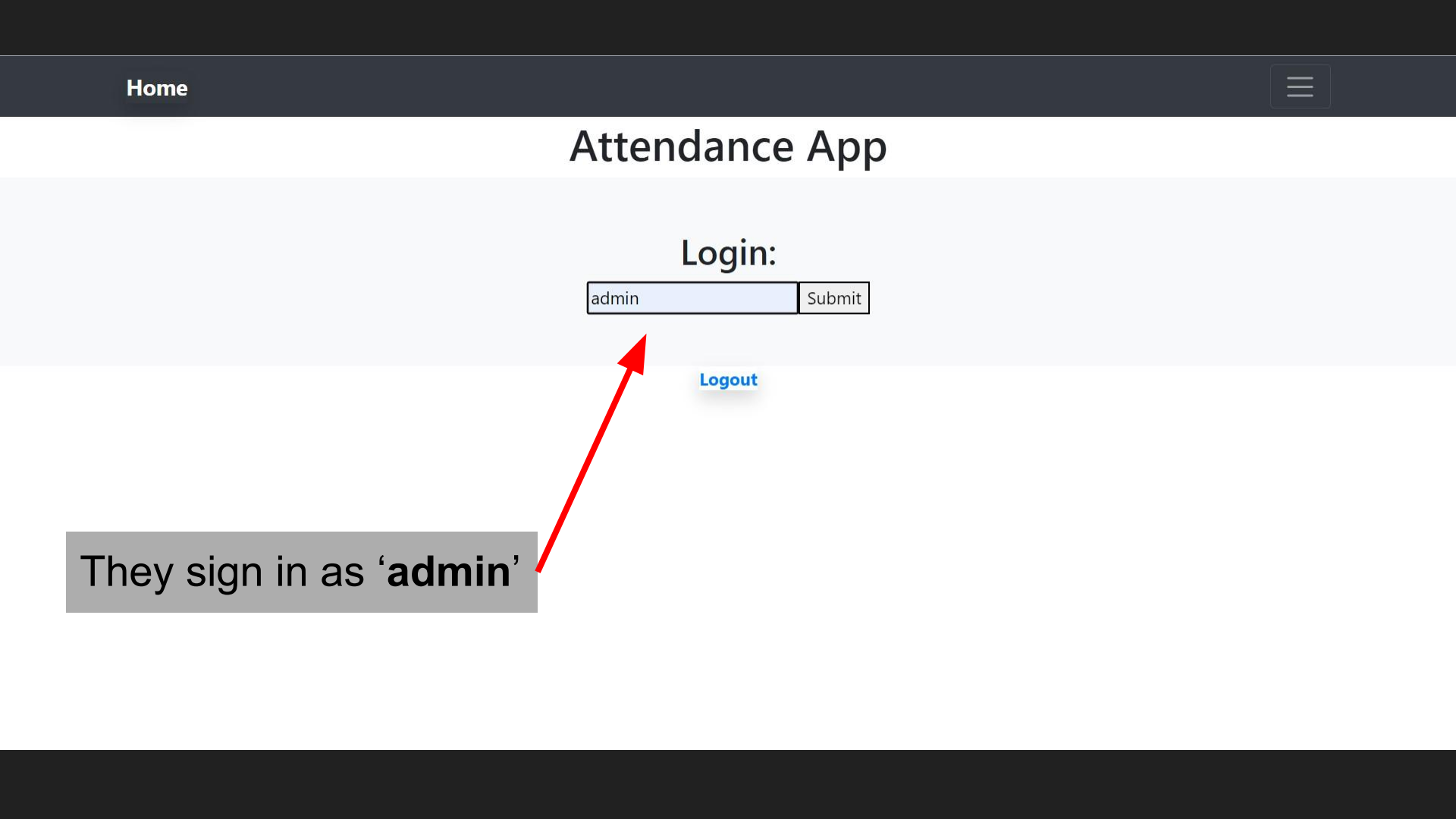
They select the '**Add students to a class**' option

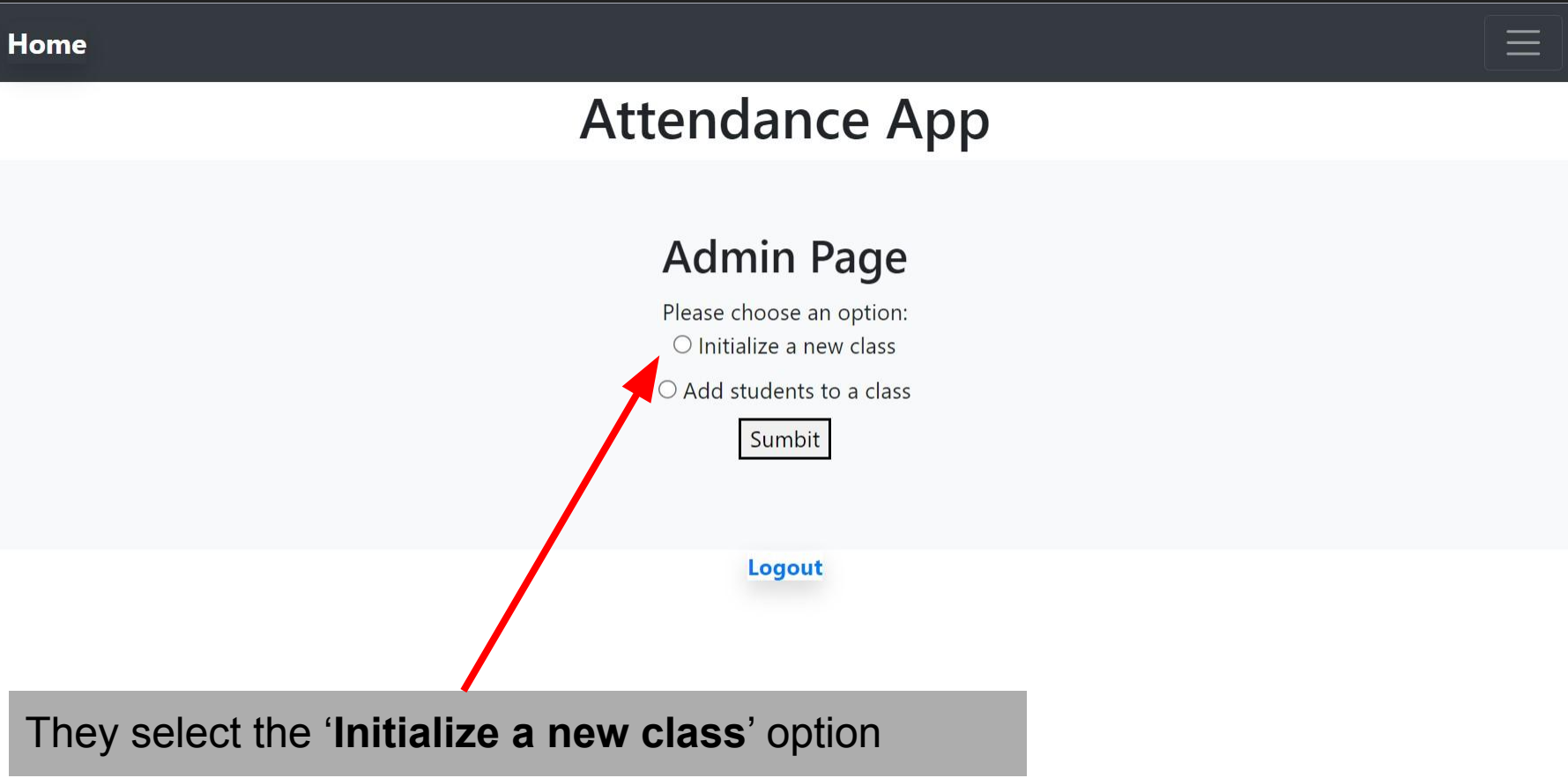

Home
Attendance App
Add Students
Enter the name of the class:
Enter the student ID:
Add student to class
Back
Logout
Enter the name of class & student ID
Then click the ‘Add student to class’ button


Home
Attendance App
Add Students
You have succesfully added a student to the class
Add Another Student
Back
Logout
If they need to add more students they would click here

Scenario #4

- A teacher or administrator just got word from the department that they are offering a new class this semester
- They want to create the class in the attendance app …


Home
Attendance App
Login:
admin
Submit
Logout
They sign in as ‘admin’



They select the '**Initialize a new class**' option

Attendance App

They would fill out all of the information for the new class here

Create a Class

Enter the name of the class:

Enter the room number:

Which days of the week does the class meet?:

Monday ☐

Tuesday ☐

Wednesday ☐

Thursday ☐

Friday ☐

Beginning of semester: mm/dd/yyyy

End of semester: mm/dd/yyyy

Enter the time slot for this class:

Start Time: --:-- -- End Time: --:-- --

Submit this class to the database

Back

Then press the '**Submit this class to the database'** button

Home

Attendance App

You have succesfully added a new class to the database

Back

Logout

Successfully added to the firebase database

Responsive to all browser sizes

From google:

What is the best screen size to design for in 2022?

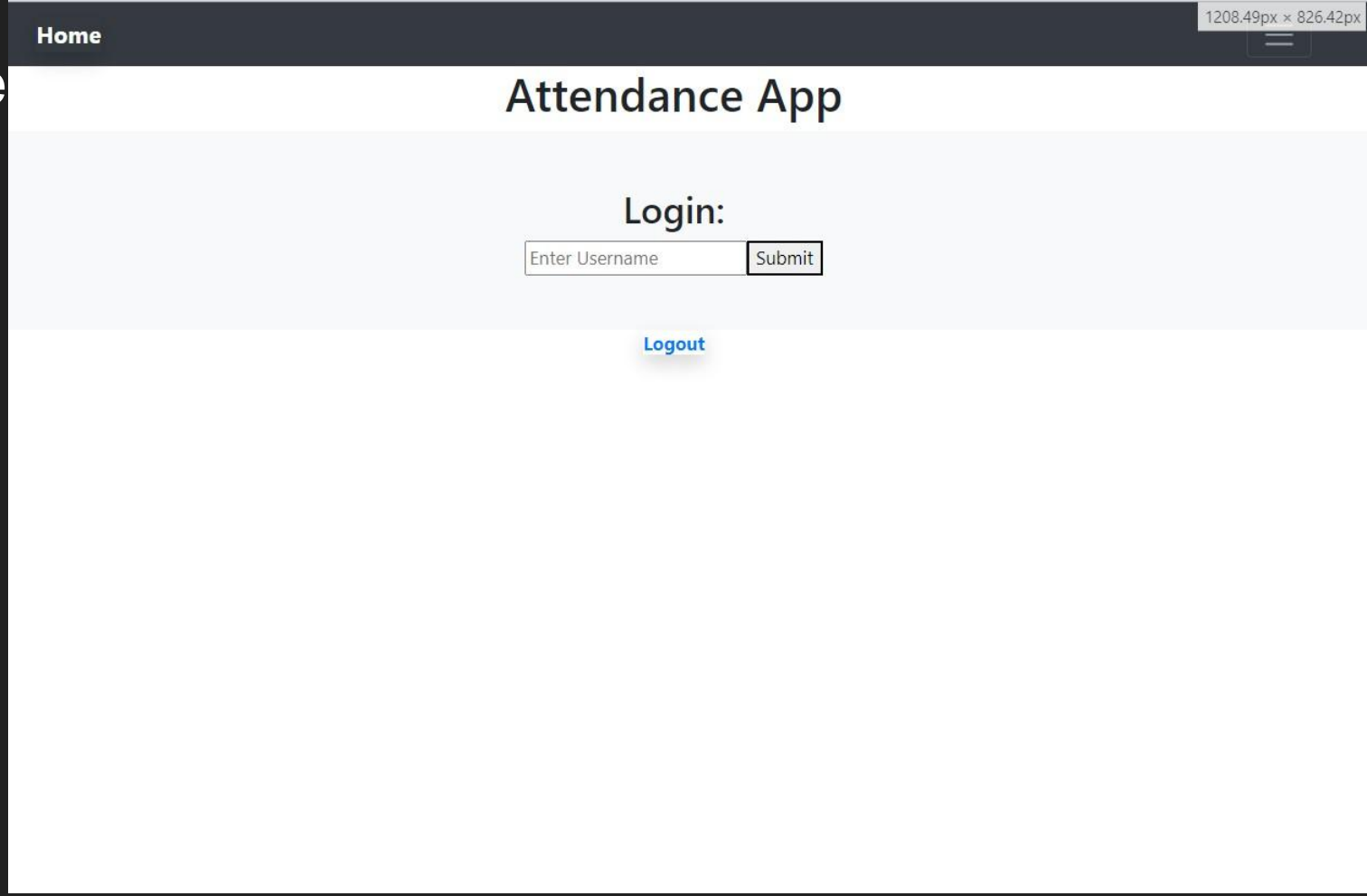
There's **no one best screen size** to design for. Websites should transform responsively and fast at all screen resolutions on different browsers and platforms. Accessible. Mobile-friendly. Design for your audience, first. Design from 360×640 through 1920×1080.

Also:

- Design for desktop displays from 1024×768 through 1920×1080
- Design for mobile displays from 360×640 through 414×896
- Design for tablet displays from 601×962 through 1280×800

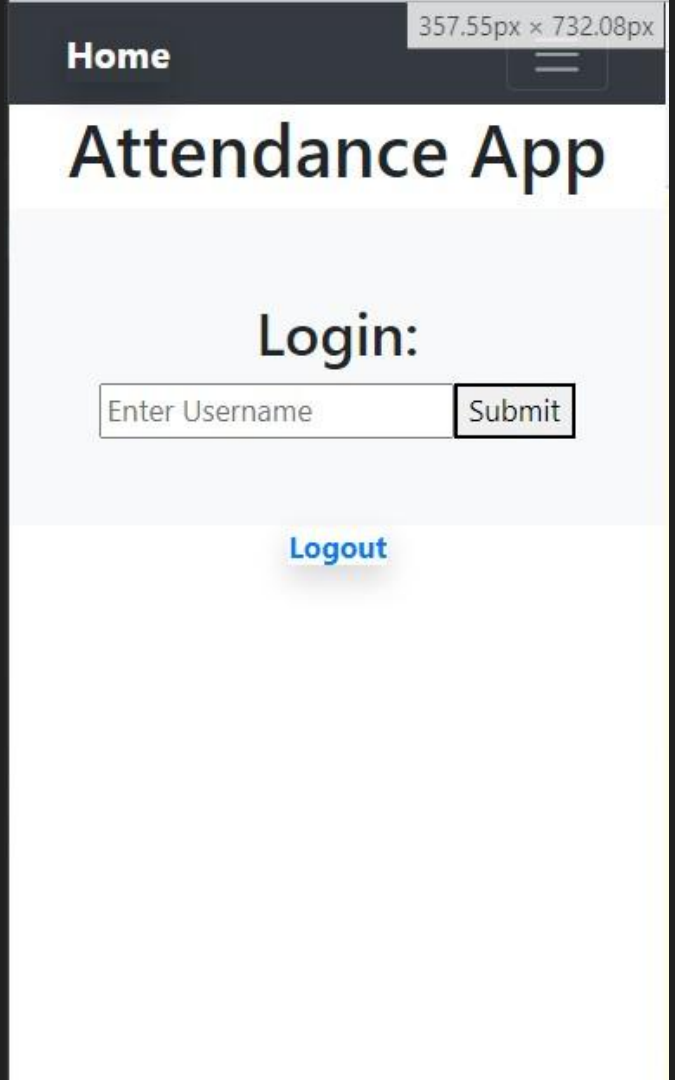
Login page

Desktop view:
1208px x 826px



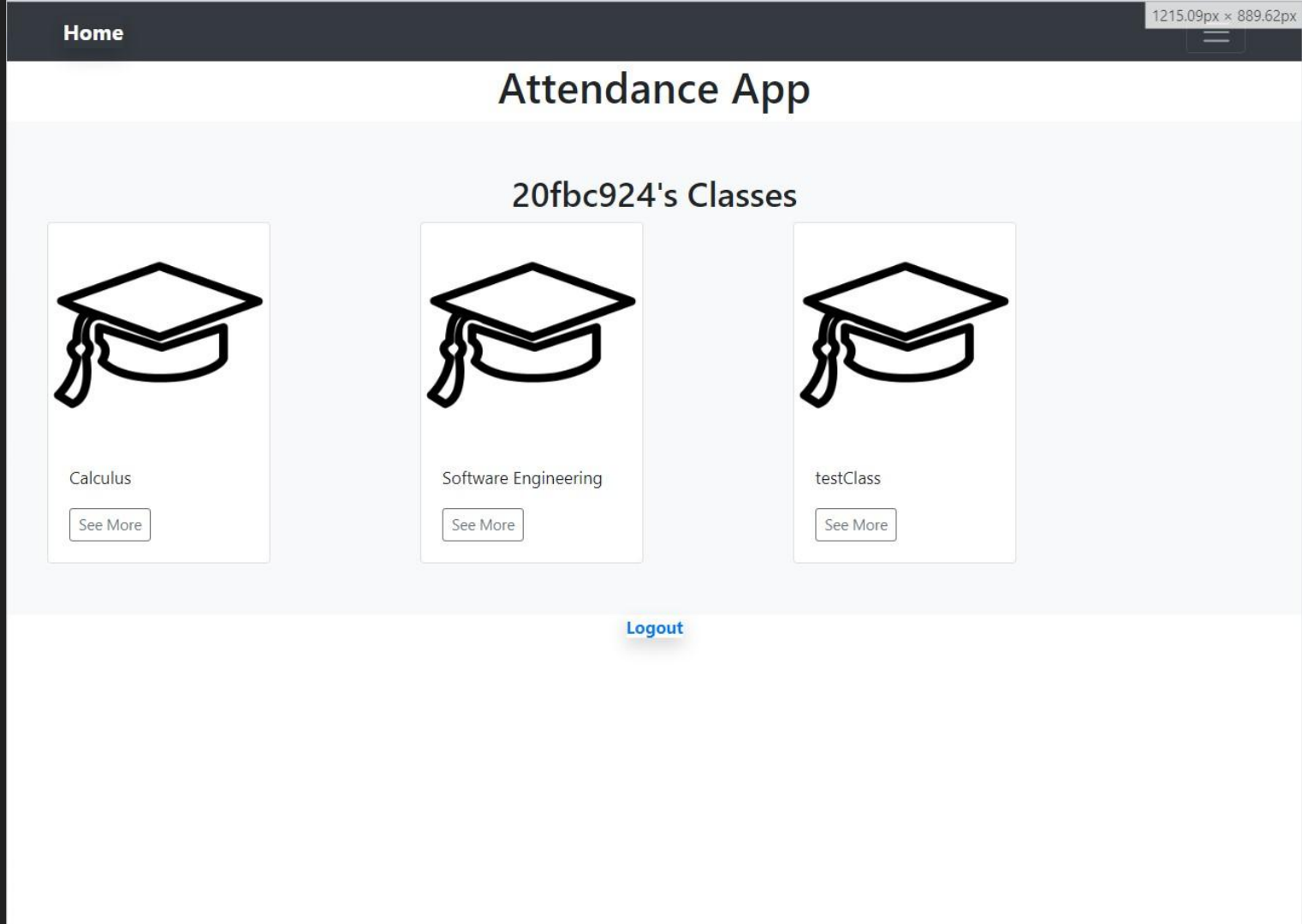
Login page

Mobile view: 357px x 732px



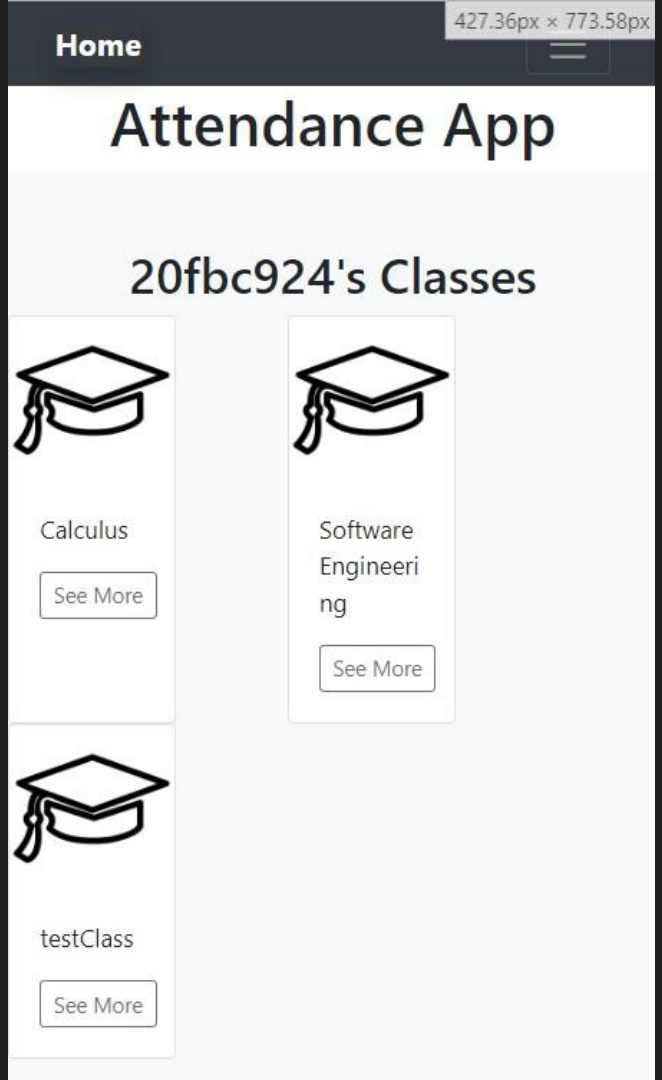
Class page

Desktop view:
1215px x 889px



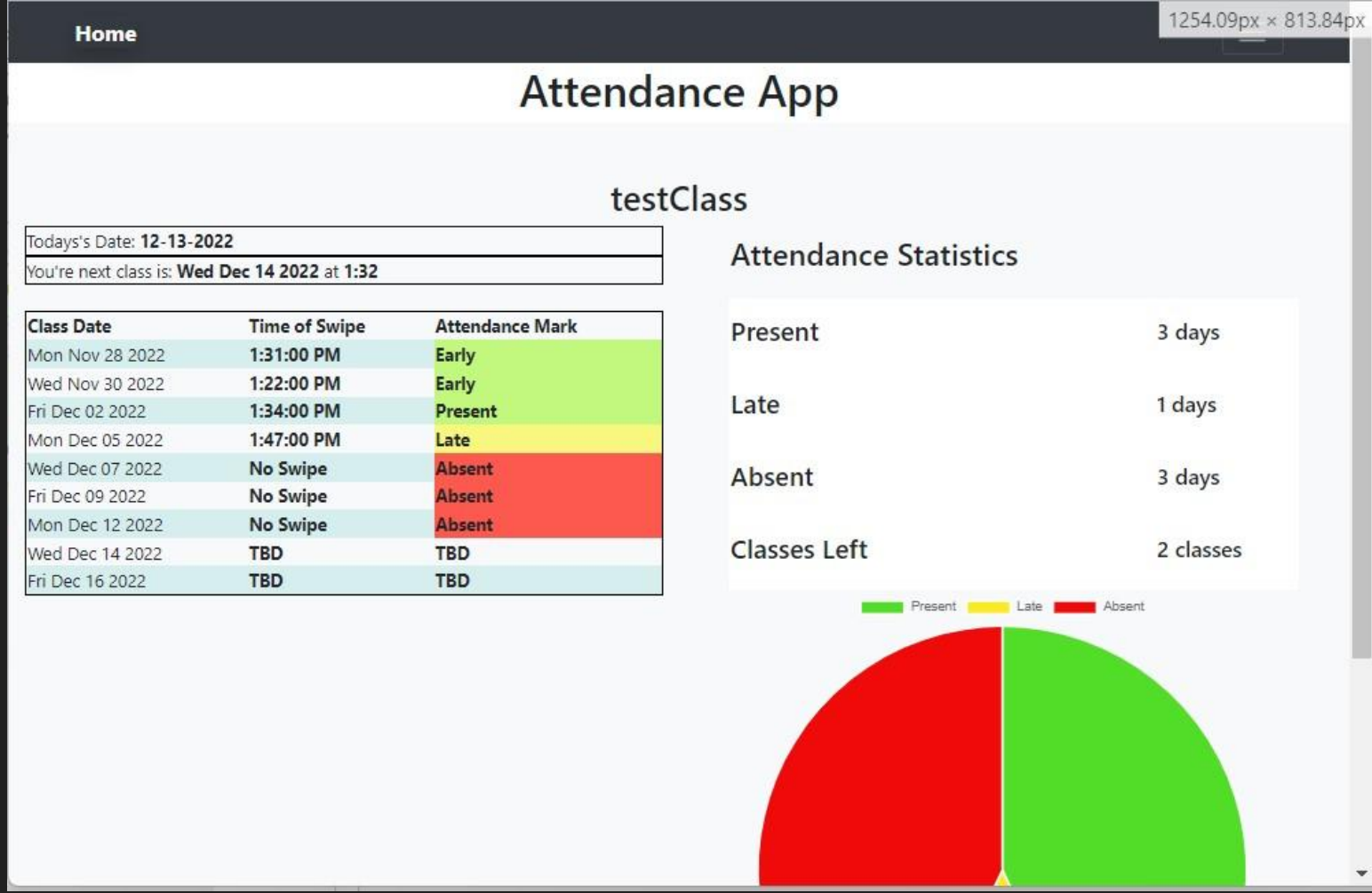
Class page

Mobile view: 427px x 773px



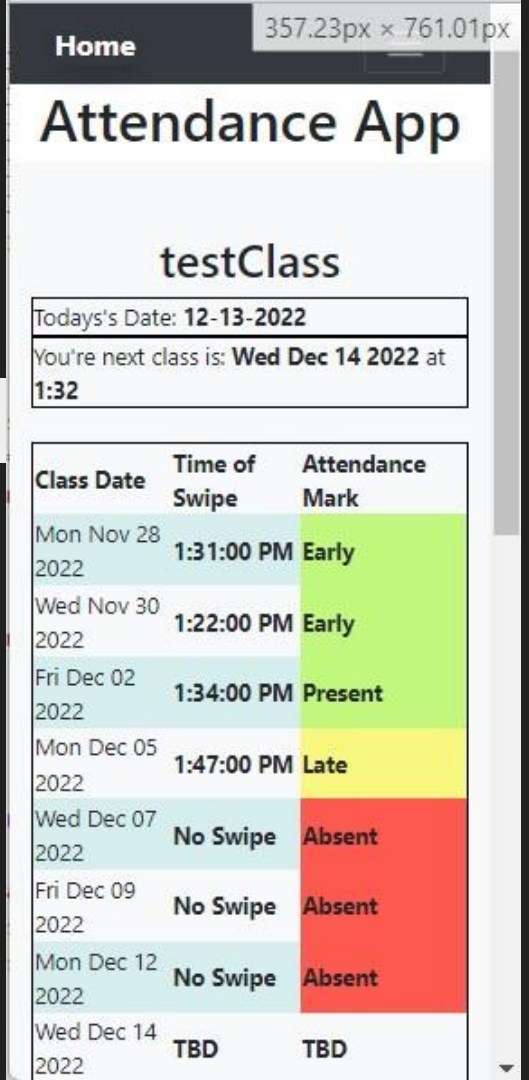
Attendance history page

Desktop view:
1254px x 813px

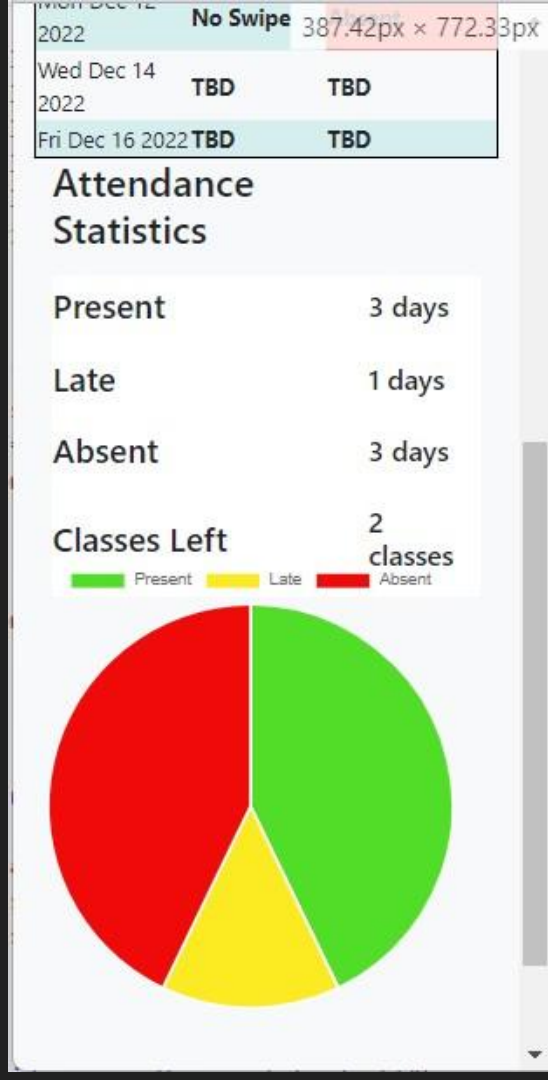


Attendance History Page

Mobile view: 357px x 761px

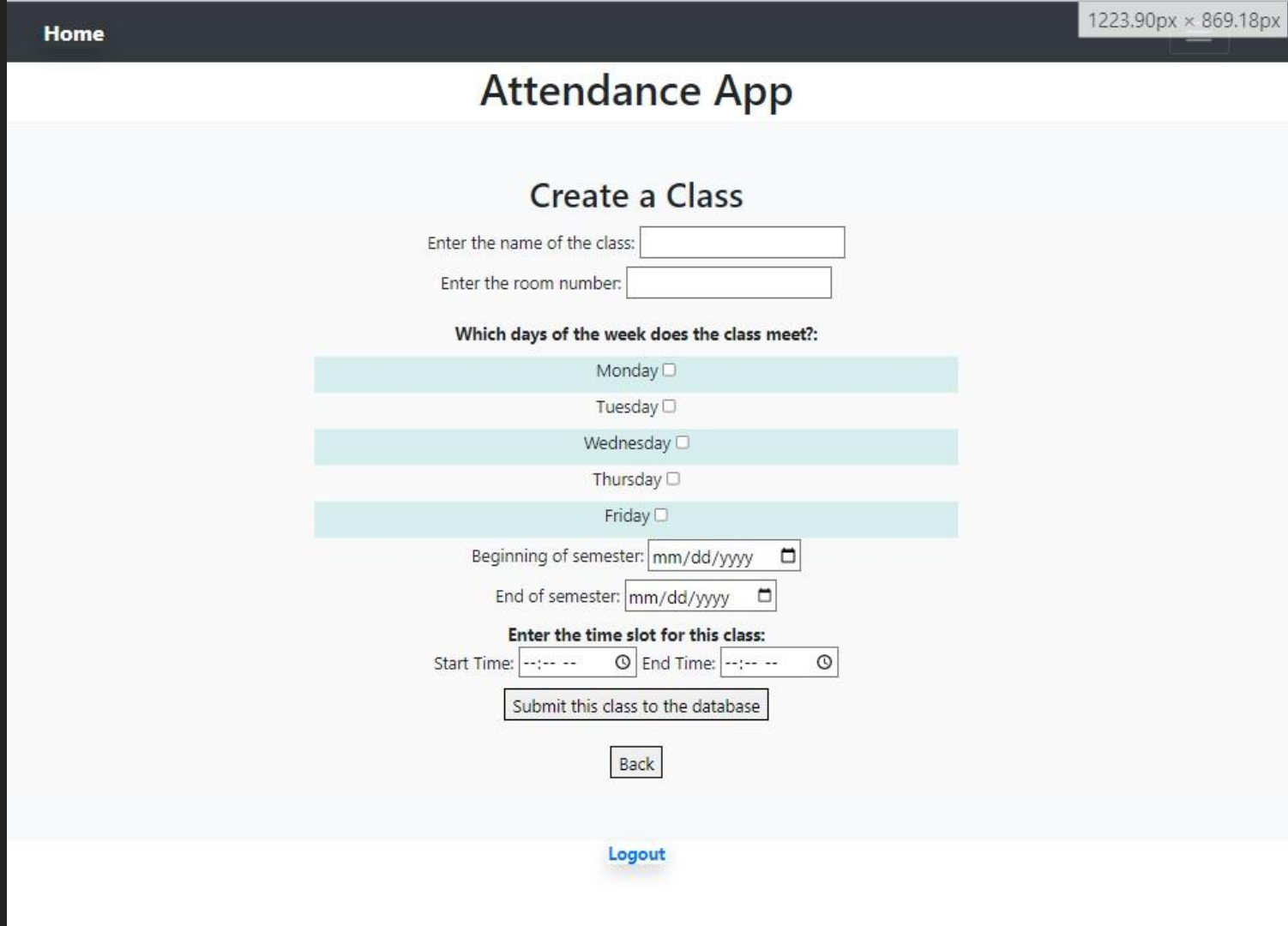


Scroll down



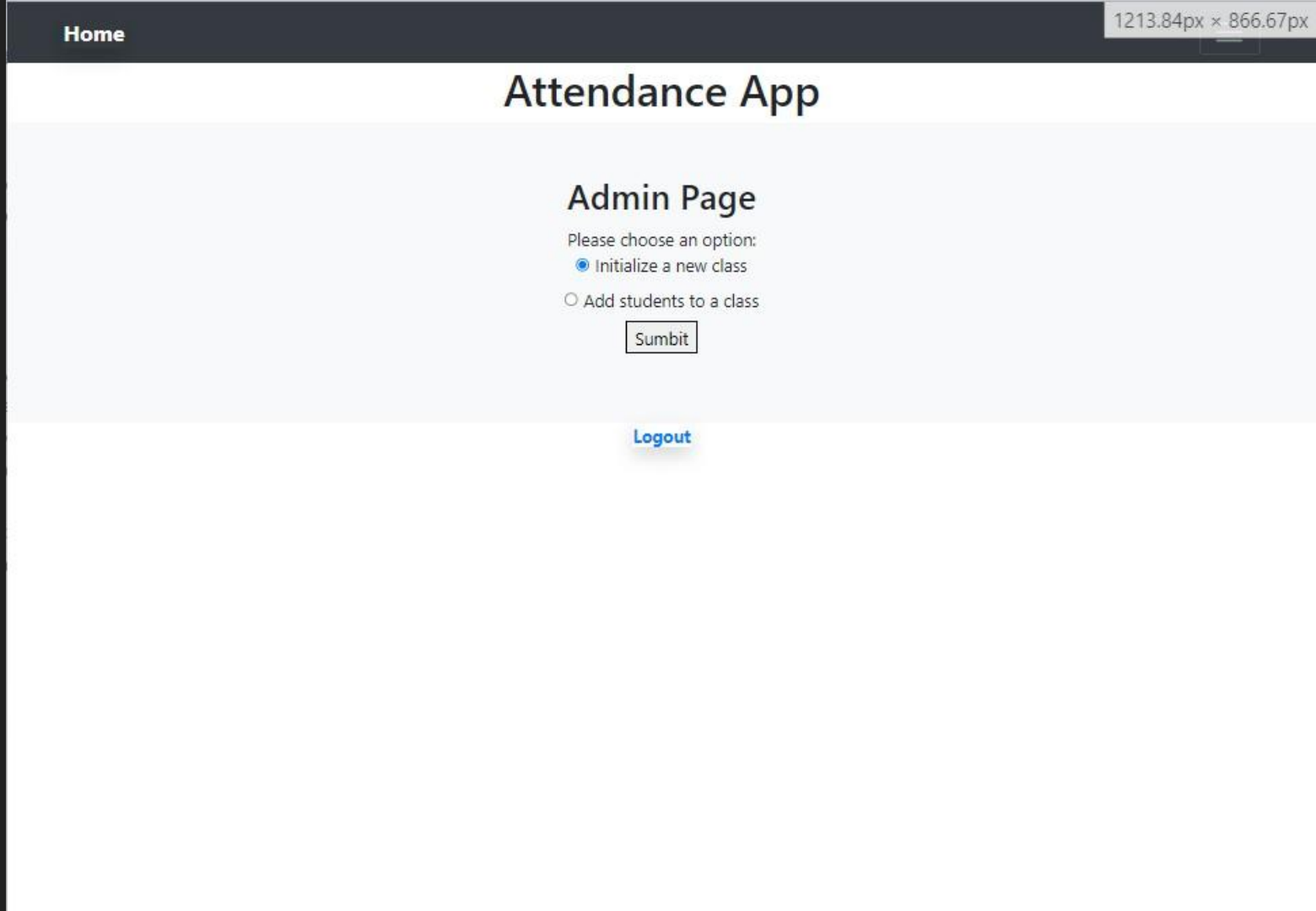
Admin - create a class page

Desktop view:
1213px x 866px



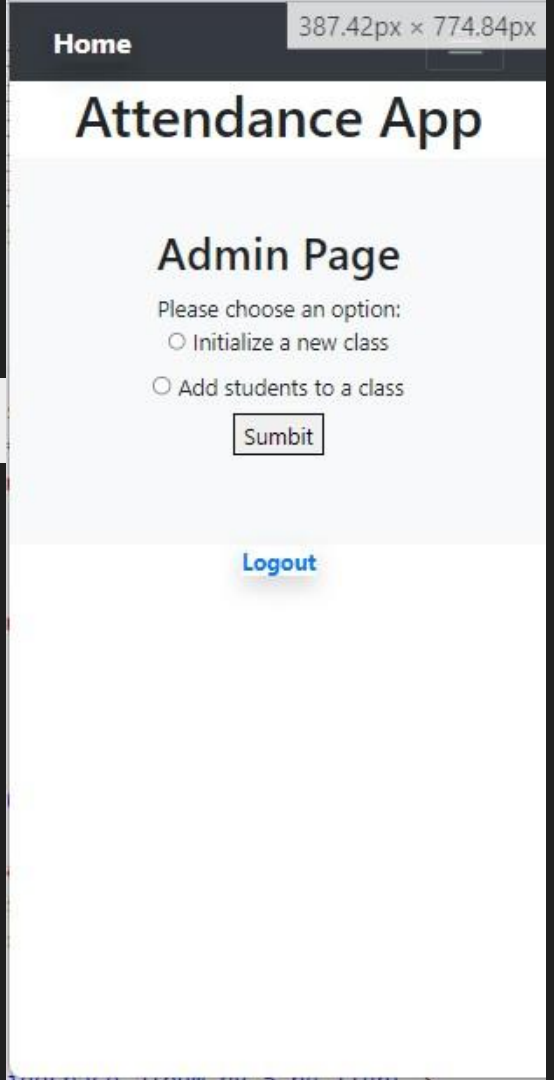
Admin page

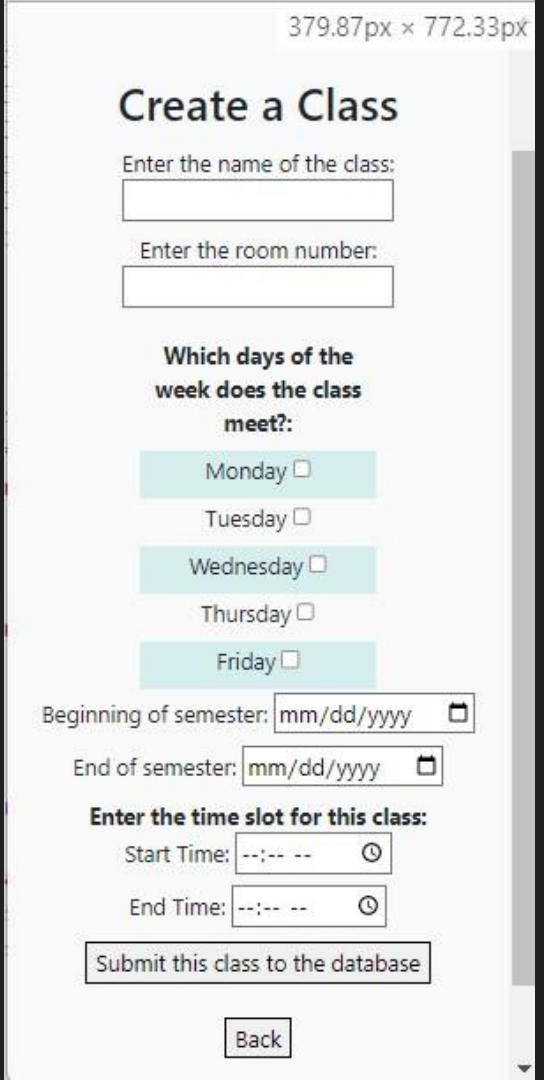
Desktop view:
1213px x 866px



Admin Pages

Mobile view: 387px x 774px





Java Backend

- All database calls are done in the javascript in the website
- This can lead to people editing the code and getting access to the database
- We wanted the database calls to be handled by the server and the website asks data
- Visible API keys and URLs are a big security risk

```
String apiKeyLocation = "C:\\Users\\micha\\IdeaProjects\\AttendanceAppDatabase\\adminsdk_firebase.json";
// Replace databaseURL with the url of YOUR database
String databaseURL = "https://attendanceapp-se-default-rtdb.firebaseio.com";
```

Main.java vs databaseEditor.java

```
// Setting up database for access (ONLY EDIT VARIABLE CONTENTS)
databaseEditor dbEditor = new databaseEditor();
// Replace apiKeyLocation with YOUR individual API Key since file path will be different
String apiKeyLocation = "C:\\Users\\micha\\IdeaProjects\\AttendanceAppDatabase\\adminsdk_firebase.json";
// Replace databaseURL with the url of YOUR database
String databaseURL = "https://attendanceapp-se-default-rtdb.firebaseio.com";

try {
    dbEditor.updateInitialize(apiKeyLocation, databaseURL);
}
catch(Exception e) {
    System.out.println("Exception " + e + " occurred" );
}
// End of database setup (Edit Away Below)
```

Main.java vs databaseEditor.java

```java
public void updateInitialize(String apiKey, String databaseURL) throws IOException {

    // Store apikey
    FileInputStream serviceAccount = new FileInputStream(apiKey);

    // Ask google's api for access to our database
    options = FirebaseOptions.builder()
            .setCredentials(GoogleCredentials.fromStream(serviceAccount))
            .setDatabaseUrl(databaseURL)
            .build();

    // Create the link between our program and the firebase
    FirebaseApp.initializeApp(options);
    db = FirestoreClient.getFirestore();
}
```

```java
import com.google.api.core.ApiFuture;
import com.google.auth.oauth2.GoogleCredentials;
import com.google.cloud.firestore.*;
import com.google.firebase.FirebaseApp;
import com.google.firebase.FirebaseOptions;
import com.google.firebase.cloud.FirestoreClient;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import java.util.concurrent.ExecutionException;
```

Main.java vs databaseEditor.java

```java
public <T> void basicSearch(String collection, String field, T value) throws ExecutionException, InterruptedException {

    // Create a reference to the cities collection
    CollectionReference cities = db.collection(collection);
    // Create a query against the collection.
    Query query = cities.whereEqualTo(field, value);
    // retrieve  query results asynchronously using query.get()
    ApiFuture<QuerySnapshot> querySnapshot = query.get();

    for (DocumentSnapshot document : querySnapshot.get().getDocuments()) {
        System.out.println(document.getId());
    }
}
```

```java
try {
    dbEditor.basicSearch( collection: "Classes",  field: "teacher",  value: "Dr. Nolan");
}
catch (Exception e){
    System.out.println("Exception " + e + " occurred");
}
```