**Programmer’s Manual**

**for**

**Attendance Auditor Plus**

**Version 1.0 approved**

**Prepared by:**

Robert Airth

Adrian Haro

Skye Gibney

Cristian Puac

**December 12th, 2020**

[**https://canvas.humboldt.edu/courses/46023/files/2332858?module\_item\_id=846543**](https://canvas.humboldt.edu/courses/46023/files/2332858?module_item_id=846543)

**^Reference Above**

**Table of Contents**

1. **Introduction**

**1.1 Purpose**

**1.2 Scope**

**1.3 Overview**

1. **Getting Started**

**2.1 Architectural Design**

**2.2 Component Decomposition & Data Flow Diagram**

**2.3 Design Rationale**

1. **Coding Standards**
   1. **HTML**
   2. **CSS**
   3. **JavaScript**
2. **Modules**

**4.1 Data Flow**

**4.2 User Interface**

**4.3 Description of Modules**

1. **Introduction**

**1.1 Purpose**

The purpose of this document is to explain the functionalities and usage of the Attendance Auditor Plus (AAP) web application. The goal is to allow users to experience all of the features that AAP has to offer to give the best experience.

**1.2 Scope**

AAP is a web application that aids in the process of auditing the attendance of an event in a college campus setting. Event organizers could use this application to create an event, as well as audit the people that decide to show up to some event. This application also provides event information for students/guests on campus that may be searching for events to attend.

**1.3 Overview**

Users of Attendance Auditor Plus have to create a profile in order to access the features of the application. Some information that is required are:

* Email address (used for authentication)
* First, Last Name
* Phone Number
* Organization/Department (if they are part of the same campus, as where the event is taking place)
* Category (guest, student, faculty, staff, classified)
* Password

Visible information for information email address, first, last name,

phone number, and organization/department.

**Getting started** - signing up and logging into the application, as well as understanding some certain privileges based on the category chosen: (guest, student, faculty, staff)

* Creating an event/viewing events
* RSVP to an event
* Managing inventory and checking out items

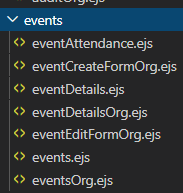
**2. Getting Started**

**2.1 The Source Code**

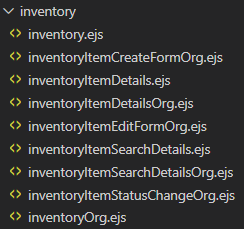
* To use the current version of Attendance Auditor Plus, the user can visit the Github Repository at <https://github.com/CS458-Capstone-Group/Attendance-Auditor-Plus> and download the file as a zip. After downloading
* **2.2 File Descriptions**

**2.2.1 HTML EJS FILES [FIX]**

**2.2.1.1** **Events**



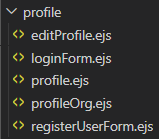
**2.2.1.2 Inventory**



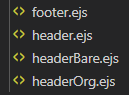
**2.2.1.3 Audit**



**2.2.1.4 Profile**



**2.2.1.5 Headers**



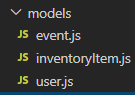
**2.2.2 CSS Files:**



**2.3 JavaScript Files:**

**2.3.1 index – server logic:**





**2.3 JSON package dependencies Files:**



**3. Coding Standards**

**3.1 HTML , CSS is unrestricted**

1. **Headers of every file**
   1. **Name of module**
   2. **Synopsis of module**
   3. **Notable functions**
   4. **Modification history and authors are taken care of by version control**
2. **Variable naming conventions**
   1. **Local - camel case (e.g.- localVar)**
   2. **Global - preceded with and underscore + camel case (e.g.- \_globalVar)**
   3. **Class - Pascal Case (e.g.- ClassName)**
   4. **Constant - Upper Case (e.g. CONSTANT)**
   5. **Variable names should be unique**
3. **Indentation**
   1. **Must be a space after commas in function arguments and parameter**
   2. **Nested blocks must be properly indented**
   3. **Braces for functions, classes, looping structures, and conditional structures are put on the same line as the definition**
4. **Comments**
   1. **Each function should be preceded by a synopsis**
   2. **Each comment should be preceded by a blank line**
   3. **The subject of the comment should directly succeed the comment**
   4. **Comments should be readable and unobtrusive**
5. **Function Constraints**
   1. **Functions should have one job**
   2. **The name of the function should define the expected behavior**
   3. **If any part of a function seems to obscure the readability, that part should be Refactored to be a stand-alone function which is then linked and called within The original function.**
   4. **Functions should return new objects instead of modifying existing objects.**
6. **Line length of 100 characters**

**3.3 Javascript**

**3.3.1**

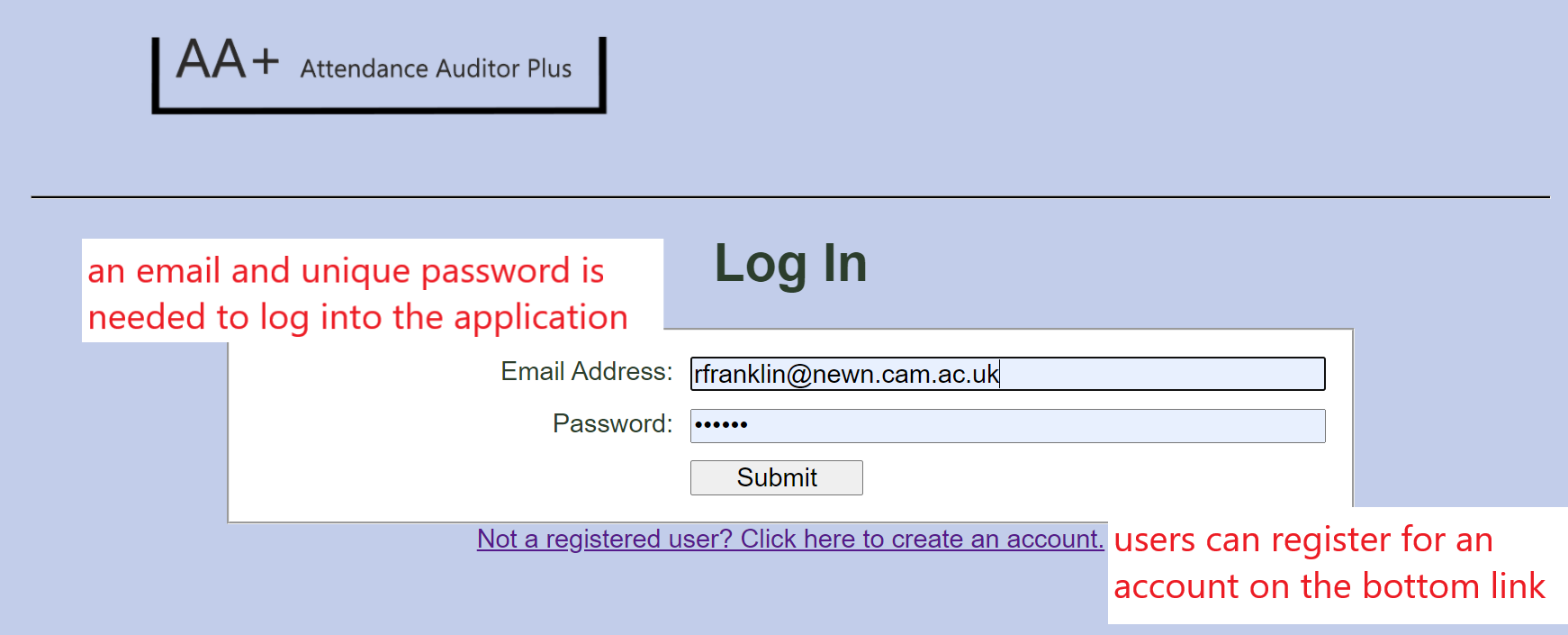
1. **Variable naming conventions**
   1. **Local - camel case (e.g.- localVar)**
   2. **Global - preceded with and underscore + camel case (e.g.- \_globalVar)**
   3. **Class - Pascal Case (e.g.- ClassName)**
   4. **Constant - Upper Case (e.g. CONSTANT)**
   5. **Variable names should be unique**
2. **Indentation**
   1. **Must be a space after commas in function arguments and parameter**
   2. **Nested blocks must be properly indented**
   3. **Braces for functions, classes, looping structures, and conditional structures are put on the same line as the definition**

**3.3.2**

**4. MODULES**

**Refer to User’s Manual for more info on usage**

**4.2 User Interface**

****

*Figure 2.3.1 Login Page*

**2.3.1.1 Email Address and Password Fields**

* Users can log into the their accounts by providing an email and unique password and click

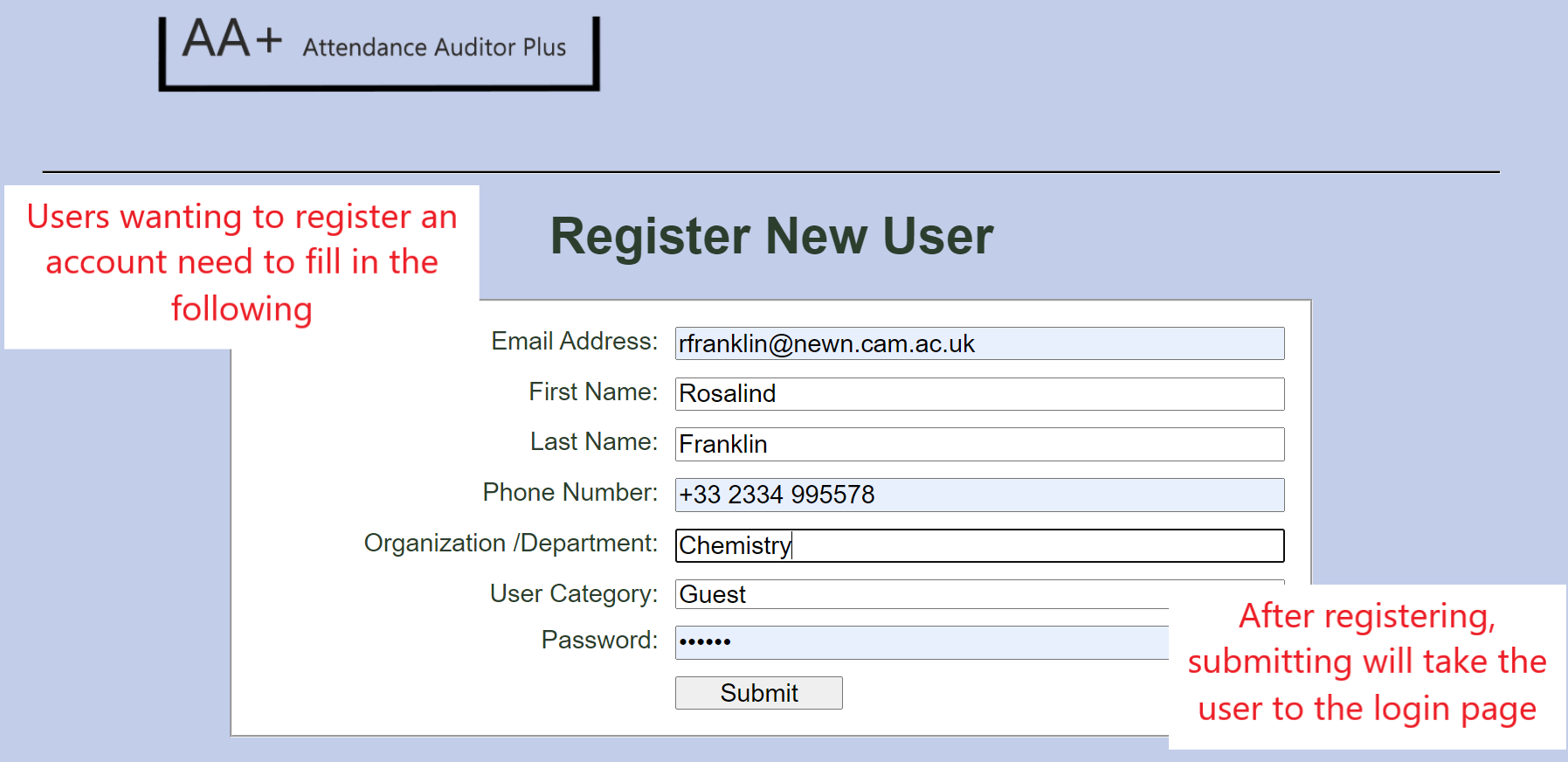
**2.3.1.2 Submitting**

* Once the needed information is inputted, the submit button send you back to the homepage **(Refer to Section 2.2)**

**2.3.1.3 Register Link**

* Users that do not have an account can click the link below the login field to create an account through the Registration Page **(Refer to Section 2.3.2)**

**2.3.2 Register Page**

****

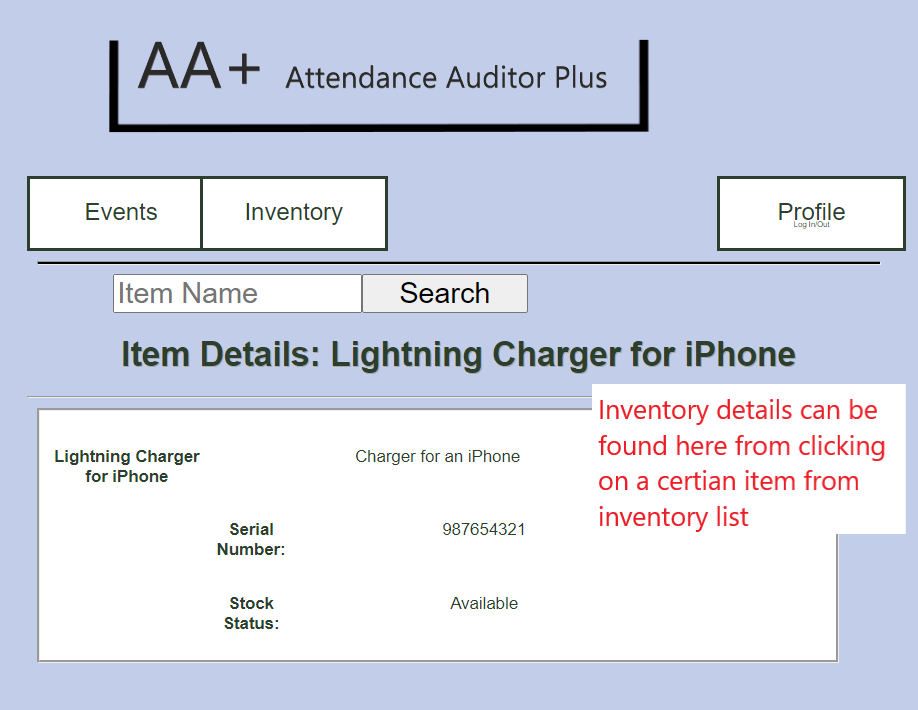
*Figure 2.3.2 Registration Page*

**2.3.2.1 Registering**

* Fields provided need to be filled out in order to register an account. Information needed from user includes:
  + ***Email Address:*** Unique to user; used for authentication
  + ***First Name***
  + ***Last Name***
  + ***Phone Number***
  + ***Organization/Department***: refers to a department on campus or any club/organization user is from.
  + ***User Category***: choose between ***guest, student, faculty, staff, classified***
  + ***Password:*** Unique to user

**2.3.2.2 Submit Button**

* Once the user submits the proper information, they are directed back to the login page **(Refer to 2.3.1: Login Page)**

****

*Figure 4.2 Inventory Details Page*

**4.2.1 Inventory Details**

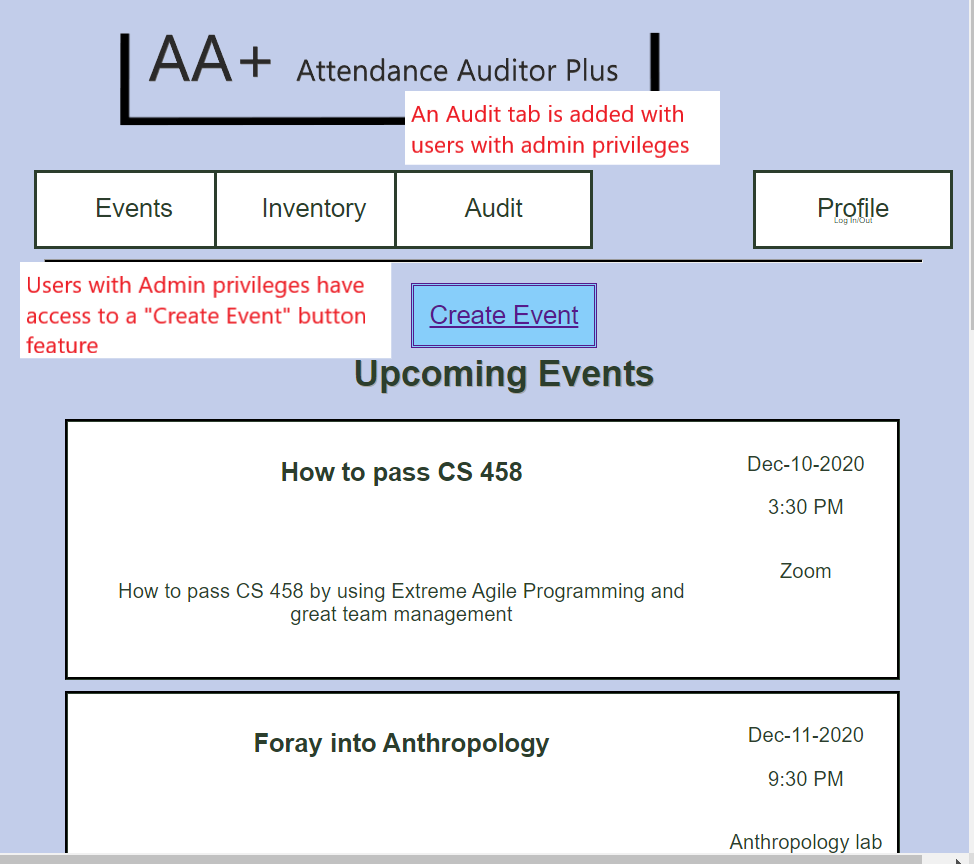
* Inventory details include the title and description of the item. A unique serial number is given to each item to make them distinct. There is also a status detail that provides the availability of the inventory item.

**5. Admin Privileges**

* A user can have admin privileges, which allows for some different privileges for admins to do. Some of the privileges include creating/editing events, adding/editing items to the inventory, checking items from the inventory, and auditing events.

**5.1 Admin User Homepage**.

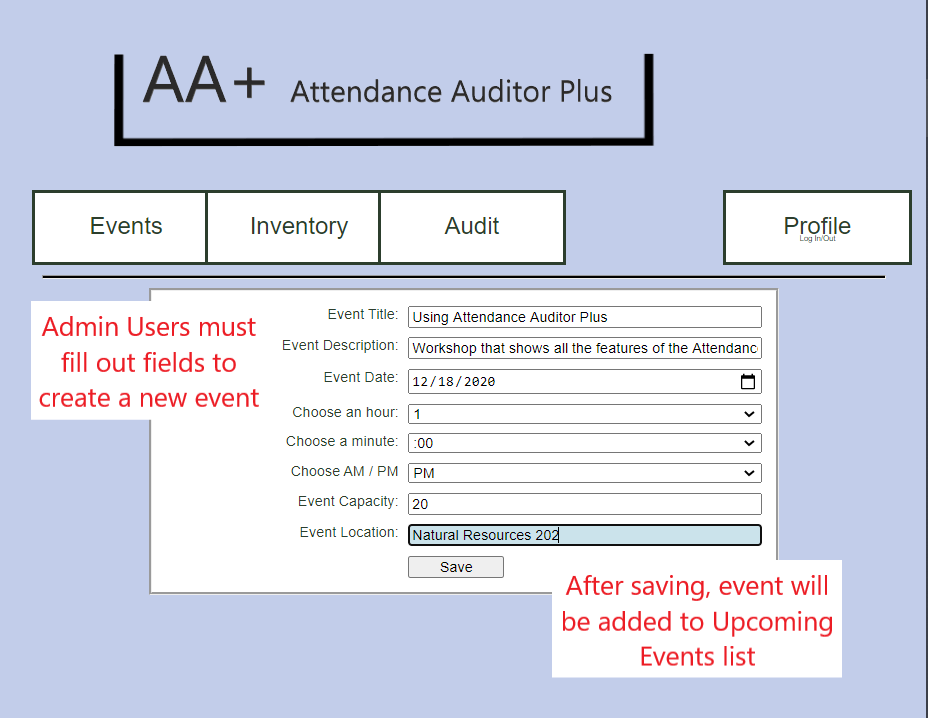
* Similar to the Upcoming Events page **(Refer to Section 2.2)**, but with two major differences. Create Event and Audit buttons are added as part of admin privileges
* The Events and Inventory tabs serve the same function, but now are able to edit/delete items as well.

****

*Figure 5.1 Homepage for Admin Users*

**5.1.1 Create Event button**

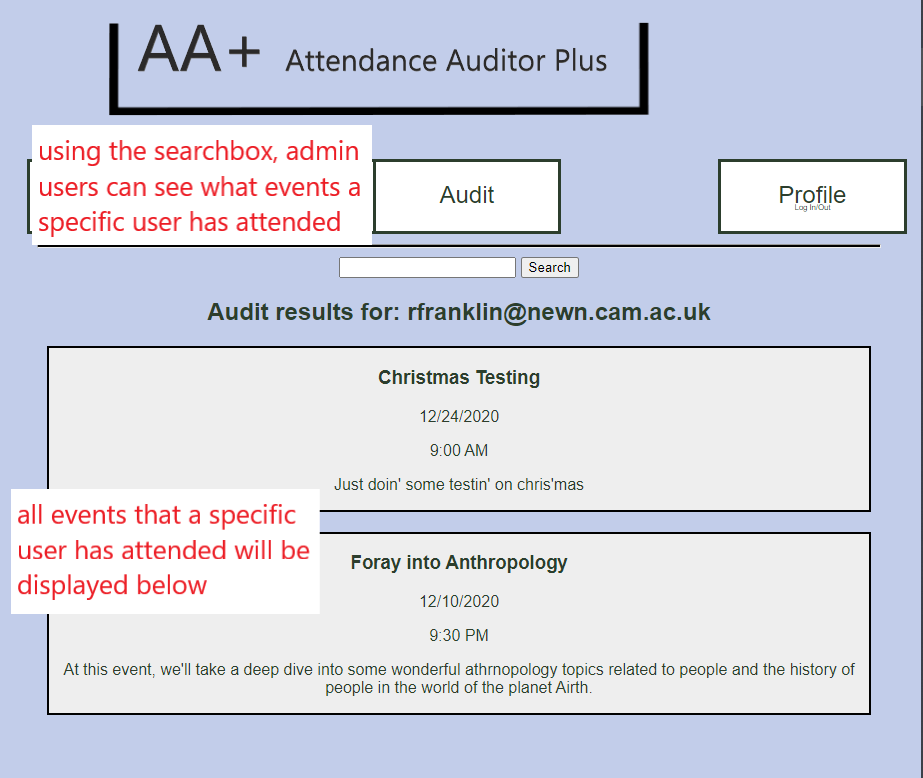
* Events can be created by clicking this button, which requires the user to fill in required fields to create the event
* After saving, the users are redirected back to the Admin User homepage **(Refer to Section 5.1)**



*Figure 5.1.1 Creating an Event page for Admin Users*

**5.1.2 Audit Button**

* Admin Users can use this button in order to find what events some users have actively attended. An email is needed to find what events a user has attended.



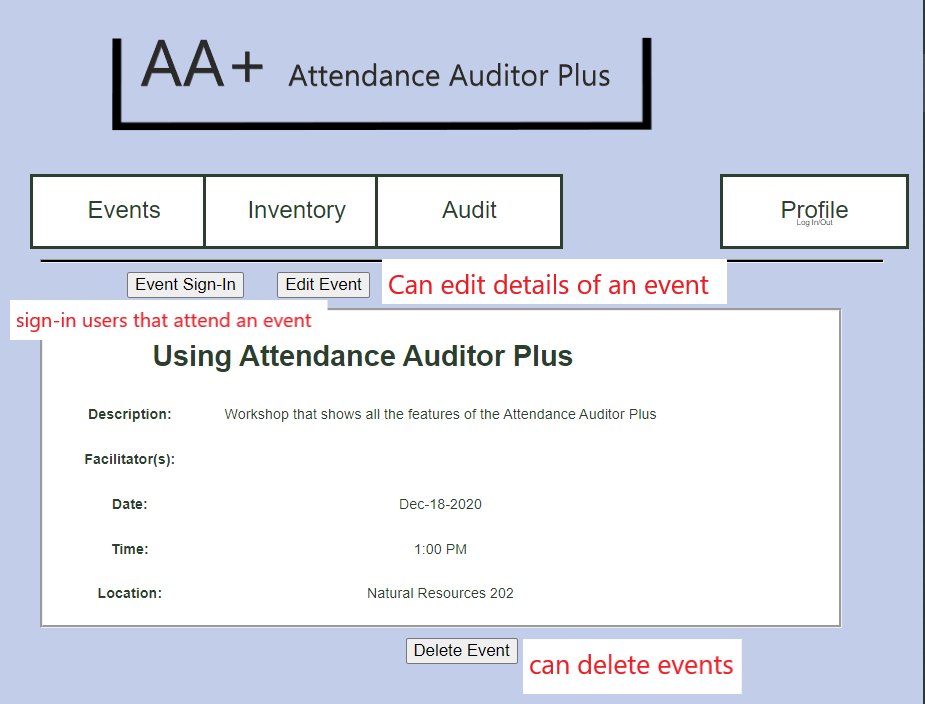
*Figure 5.1.2 Audit Button Page for Admin Users.*

**5.1.2.1. Search box**

* Typing a user member’s email address will show what events that user has attended

**5.2 Event Details Page (Admin User)**

* When viewing an Event Details page **(Refer to Section 3.2)**, Admin Users have the option to edit or delete events as they wish. There is also an Event Sign-In button that admin users can use to check off Users who are present at the event.



*Figure 5.2 Event Details Page for Admin Users*