**Software Requirements Specification**

**for**

**Attendance Auditor +**

**Version 1.0.0 approved**

**Prepared by:  
 Robert Airth  
Skye Gibney  
Adrian Haro  
Cristian Puac**

**GAHP Dev Group**

**Last Edited:9/30/2020**

**Table of Contents**

**Table of Contents [ii](#_gjdgxs)**

**Revision History [ii](#_30j0zll)**

**1. Introduction [1](#_1fob9te)**

1.1 Purpose [1](#_3znysh7)

1.2 Intended Audience and Reading Suggestions [1](#_2et92p0)

1.3 Product Scope [1](#_tyjcwt)

**2. Overall Description 1**

2.1 Product Perspective 1

2.2 Product Functions 1

2.3 User Classes and Characteristics [2](#_3rdcrjn)

2.4 Operating Environment [2](#_26in1rg)

2.5 Design and Implementation Constraints 2

**3. External Interface Requirements 2**

3.1 User Interfaces 2

3.2 Hardware Interfaces 3

3.3 Communications Interfaces [3](#_1y810tw)

**4. System Features 3**

4.1 Lead Event Organizer 3

4.1.1 Description and Priority 3

4.1.2 Stimulus/Response Sequences 3

4.1.3 Functional Requirements 3

4.2 Event Organizer 3

4.2.1 Description and Priority 3

4.2.2 Stimulus/Response Sequences 3

4.2.3 Functional Requirements 4

[4.3 Event Attendee 4](#_1ci93xb)

4.3.1 Description and Priority 4

4.3.2 Stimulus/Response Sequences 4

4.3.3 Functional Requirements 4

**5. Other Nonfunctional Requirements [4](#_3whwml4)**

5.1 Performance Requirements [4](#_2bn6wsx)

5.2 Safety Requirements [5](#_3as4poj)

5.3 Software Quality Attributes 5

**Appendix A: Software Process Model 5**

**Appendix B: Roles for Sprint 1 6**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# **1. Introduction**

## **1.1 Purpose**

The product described in this SRS is the Attendance Auditor Plus (v1.0.0). This SRS pertains to the entirety of the system.

## **1.2 Intended Audience and Reading Suggestions**

This document is intended for developers, testers, and project managers. It contains information on user and system requirements as it pertains to creating the system itself and subsequent testing for system correctness. The document will be used during the development process, and so there is no single way to proceed through this document. Instead, it will be used as a reference and a goal marker to guide development.

## **1.3 Product Scope**

The software described here is intended for event organizers looking for a simple, easy-to-implement solution for event management that allows them to keep track of event attendees. The software should provide a way of generating reports for specific attendees, listing the workshops they have attended during a given time period.

# **2. Overall Description**

## **2.1 Product Perspective**

The motivational scenario for creating this software was as a solution for attendance auditing for professional development workshops at a community college. Event organizers needed a way to measure interest in a workshop, register who actually attended, and later would need to create reports for the attendance of a specific faculty member to give to the dean of their department. The software is intended to be a standalone system that can be easily hosted on the client’s servers.

## **2.2 Product Functions**

1. User RSVPs for an event to show interest
2. Event organizer registers attendees
3. Event organizer compiles a report of a specific attendee and all the workshops they have visited during a given time period
4. Event organizer compiles a report of attendees for a set of workshops and/or series of workshops
5. Attendee views calendar of upcoming events that have been input by the event organizer
6. Attendee views report of past events attended

## **2.3 User Classes and Characteristics**

1. Lead organizer
   1. Most privileges, able to view logs of actions made by individual organizers
2. Event organizers
   1. Have higher privileges to be able to manage events and audit attendees
3. Event attendees
   1. Can view past and future general event information and view their own past-attendance data.

## **2.4 Operating Environment**

Backend will be hosted on the client’s servers, which will also be used to host the database. Front-end will be implemented in react. Client’s server environment will vary.

## **2.5 Design and Implementation Constraints**

Client organization will maintain the delivered software once implemented.

# **3. External Interface Requirements**

## **3.1 User Interfaces**

1. Logical Characteristics(projected) -- Hosts and Attendees
   1. Hosts will receive special registration links
   2. Hosts will use web browser to access the web application
   3. Hosts will be presented with a GUI
   4. GUI will take user input from BIOS
   5. GUI will be able to display audit info or a form for Attendees
   6. GUI will have big but decent sizes buttons
   7. GUI error messages will be displayed from invalid links or access

## **3.2 Hardware Interfaces**

1. If camera with a QR reader can’t be implemented
2. Events will have a camera that can read QR images from phones to SIGN-IN
3. Hosts can email links to possible Attendees
4. Will be multi-platform to work with browsers on computers or smartphones

**3.3 Communications Interfaces**

1. HTTP and or HTTPS between all GET’s and POSTs

# **System Features**

## **4.1 Lead Event Organizer (LEO)**

4.1.1 Description and Priority

1. LEO will be able to view logs of event organizer activities (high priority)
2. LEO will be able to view past events (medium priority)
3. LEO can audit the event attendance data (medium priority)

4.1.2 Stimulus/Response Sequences

When a LEO decides to host an event, the system will ask the user to enter the name of the event, the time the event will happen, and also the place that this event will take. Once this information is entered, the system will ask the LEO if everything they have entered is accurate to the information the event is supposed to have.

Once the event is published onto the system, the LEO will be able to view the event, as well as any past events. For the current event, LEO are able to audit the attendance data to see who is coming, as well as getting information of who exactly attended on the day of the event.

4.1.3 Functional Requirements

4.1.3.1. Event Management

4.1.3.2. Auditing Event Data

4.1.3.3. Event Registration

**4.2 Event Organizer (EO)**

4.2.1 Description and Priority

1. EO takes notes of who exactly is at event via QR code or unique ID number from Event Attendee (high priority)
2. EO can traffic Event Attendees to make sure that their presence is taken at the event. (high priority)
3. EO sends verification email to confirm the Event Attendee's attendance. (high priority)

4.2.2 Stimulus/Response Sequences

On the day of the event, EO took note of which Event Attendees showed up through unique identification provided by Event Attendees. If the Event Attendee is proven to be at the event, an email verification will be sent to the Event Attendee to prove their attendance.

4.2.3 Functional Requirements

4.2.4.1. Event Auditing

4.2.4.2. Attendee Auditing

4.2.4.3. Verification of Attendance for Attendee

## **4.3 Event Attendee (EA)**

4.3.1. Description and Priority

1. EA will be able to “join/decline” an already created event (high priority)
2. If already attending an event, EAs can sign in to system (medium priority)
3. EA can view past events, whether they attended or declined it (low priority)
4. EA receives a unique randomly generated identifier in order to prove attendance at event (high priority)
5. EA uses email addresses and student ID to identify themselves at the event. (high priority)

4.3.2. Stimulus/Response Sequences

When EAs decide to sign up for an event, they are presented with different upcoming events they could sign up for. EAs will be able to sign-in by providing an email school address and their unique campus identification number. Once signed up, EAs will be reserved a space in the event. EAs can view any past events that they have attended as well, or events that they did not attend even though they had said they were.

4.3.3 Functional Requirements

4.3.3.1. Partake or Decline an Event

4.3.3.2. View Current/Past Events

4.3.3.3. Event Registration

# **Other Nonfunctional Requirements**

## **5.1 Performance Requirements**

## Fast response time under few seconds

## Handles errors gracefully

1. Can’t be offline longer than 5 seconds

## **5.2 Safety Requirements**

1. *Sensitive information can be kept safe by limiting some access to information for users to take a look at.* 
   1. *Attendee has the most limited view of the database, they can only see their own past-attendance information, future-attendance information, and general event information*
   2. *Event Organizer has access to event management functions, future-attendance information, attendance check-in functions, and general event information.*
   3. *Lead Event Organizer has access to everything the Event Organizer has access to plus the ability to view past-event attendance information and attendee attendance information.*

## **5.3 Software Quality Attributes**

## Robust implementation/modular and integratable

## Cheap to implement

## High acceptability/availability (web-browser implementation)

**Appendix A: Software Process Model**

*The development of the software will be implemented using an* ***Agile*** *methodology.*

**Appendix B: Roles for Sprint 1**

1. *Team Leader (Cristian & Robert)*
   * *coordinating team effort during sprints*
   * *clarifying project sprint objectives*
   * *maintaining morale*
   * *scheduling team meetings*
2. *Lead Programmer (Skye)*
   * *Coordinating coding efforts*
   * *Refining design approaches*
   * *Coordinating software development*
   * *Oversight of software-interface implementations*
   * *Oversight of hardware-interface implementations*
3. *Lead Designer(Cristian)*
   * *Designing program structure*
   * *Designing user interfaces*
4. *Quality Assurance Leader(Testing)(Robert)*
   * *Frequent review of all software-engineering processes*
   * *Ensuring software meets SRS requirements before, during, and after each sprint.*
   * *Creating testing methods to verify usability and functionality of each software functional area.*
   * *Creating testing methods to verify usability and functionality of the completed software.*
5. *Assistant QA (Skye)*
6. *Analyst/Requirements Engineer(Adrian)*
   * *Oversees the development of the SRS and milestones*
   * *providing reports on your analysis to the lead engineer or other stakeholders in the project.*
7. *They/them (Skye)*