**­­Requirements Analysis Document**

**Event Management System**

CITS3200 Group 9

Semester 2 2022

University of Western Australia

35 Stirling Hwy, Crawley WA 6009

**Revision History:**

Version 0.1 10/08/2022 Jakob Kuriata. Created

Version 0.2 16/08/2022 Joshia Nambi. Formatting and Additions

**Preface:**

This document addresses the requirements of the Event Management System. The intended audience of this document are our client, mentor, and designers of the project.

**Target Audience:**

Ruby Chan

**CITS3200 Group 9 Members:**

* Alian Haidar (22900426)
* Jakob Kuriata (23278189)
* Matt Dai (22546998)
* Joshia Nambi (22976423)
* Oliver Lynch (22989775)
* Luke Antoncich (21108905)

**INSTRUCTIONS**

Creation of a secure web-based shop front for event attendees:

1. to book his/her attendance (including partner, children);
2. to register contact details;
3. to indicate no fees involved;
4. to enable uploading images/text; and
5. to interface with printer to generate badges

Security of data is an important consideration.

**MILESTONES**

* 17/08 Sprint 1 Deliverables Due
* 21/09 Sprint 2 Deliverables Due
* 17/10 All work on project stops, Handover to client of project deliverables, system demonstration & client retrospective.
* 19/10 Sprint 3 System and Retrospective Due

# 1.0 General Goals

- To provide a suitable alternative to the current system in place that will be hosted via a web application that attendees can interact with using their mobile devices as an alternative to paper in order to be more environmentally friendly.

# 2.0 Current System

Currently the client uses paper invitations for her events.

# 3.0 Proposed System

### 3.1 Overview

- Our team will create a web application to manage these events.

- This application will be able to book/mark an attendee’s attendance (including their partner /or children).

- It will also register their contact details and be able to indicate whether there are fees involved for that event.

- Our system will be able to take images to be used as attendees profile picture for their contact details.

- Our system will also be able to generate individual badges for each event as a form of ID for that event.

- Our system will be built using the Flask framework.

### 3.2 Functional Requirements

- Storing/marking attendance booking into a database

- Registering contact details into a database.

- To generate badges for a form of ID.

- A secure online payment system.

- UI testing, ensuring that the application feels good to use over the current system.

- API interface to check if fees are required.

### 3.3 Nonfunctional Requirements

- Being able to print badges as an alternative.

- Being able to upload images/text to our database/system.

- Speed/performance of the website.

- Security testing.

- Scalability.

- User feedback.

### 3.3.1 User Interface and Human Factors

- Novice users/ordinary people should be able to use our system with ease.

- Should be easy as it just filling out a simple form (name, email, mobile number, emergency contact, dietary requirements).

- There will be error detection for example, a user cannot have a number in their name or characters cannot be in a mobile number, this is to ensure that the database doesn’t get confused with incorrect data.

- The input device will be a keyboard (physical or digital).

- The output device will be either a monitor or a mobile device screen as the application will be accessible on both.

### 3.3.2 Documentation

- Deployment documentation - Demonstrate how the client can deploy our web application.

- Usage documentation – Details how a user can use and interact with our web application.

- Troubleshoot documentation – Go through the common problems and solutions an ordinary user may have with our web application.

### 3.3.3 Hardware Consideration

- As our system will be a web application, it can either be accessible via a web address or hosted locally, both of which can be accessible on a mobile device/computer or any system with a modern internet browser.

### 3.3.4 Performance Characteristics

- The speed of the application should be relatively fast as checking in should be quick and convenient to the user.

### 3.3.5 Error Handling and Extreme Conditions

- Our system should be able to detect input errors such as invalid names/phone numbers to prevent them from entering our system and causing any further problems.

### 3.3.6 System Interfacing

- Data for attendees will entering our system.

- Details for events will be going into our system.

- A secure payment system will be going into our system.

### 3.3.7 Quality Issues

- The system is portable as it is a web application they can be deployed locally or hosted on a domain.

- As the system is portable it can easily be deployed on multiple systems if needed.

### 3.3.8 System Modifications

- The badging system could be modified in the future (QR codes or Apple Wallet Passes)

- The system should ideally be always running; however, it should be fine to only turn it on when there is an upcoming event, then reset it after that event.

- The design/theme of the website is also something that can be updated in the future.

- The web framework used could be changed early in development if necessary.

### 3.3.9 Physical Environment

- The system can be hosted locally on a single computer, or it can be hosted on a domain which can be accessed via its domain address.

- The target equipment will be multiple devices as it is a management system interacting with many users.

- The system should not be affected by environmental conditions as it is just a web application being deployed, however, if the machine that is being deployed is affected by something like a bandwidth issue the system may slow down.

### 3.3.10 Security Issues

- Security is an important factor of our system as we are dealing with each user’s personal information.

- This personal information will be stored in a database that can only be accessed on the computer/server the system is being deployed on.

### 3.3.11 Resource Issues

- As the data for this system will be contained in a database located on the computer hosting the system, it can only be backed up with access to that system if needed.

## 3.4 Constraints

- Flask is not as popular as other web frameworks.

- Flask is dependent on libraries for decent functionality (login/authentication).

- Less secure than other similar web frameworks (Django).

## 3.5 System Model

### 3.5.1 Dynamic Models

- Updatable emblem regarding specific events

- Animated Buttons

### 3.5.2 User Interface – Navigational Paths and Screen Mockups

