iAttend:

Design Document 1, Draft ii

Three Furious Locomotives

March 19, 2020

iAttend Design Document 1

DD1-ThreeFuriousLocomotives.tex
Draft ii
March 19, 2020
Three Furious Locomotives

Revisions

Version	Primary Author	Description of Version	Date Completed
i	Matthew Dutt and Michael Jarman	Initial creation of document.	3/1/20
ii	TFL	Edits suggested by Prof. Cindric.	3/18/20

Contents

1 Introduction

1.1 Overview

The development team affiliated with Mount Union's 2020 CSC 492 Software Engineering course known as the Three Furious Locomotives is building a system for the Music department that will replace the current manual method of recording attendance with a software solution that will simplify and speed up the process of recording attendance. The product, which will be known as iAttend, will also allow the designation of requirements, which will allow progress to be tracked, and reports to be easily generated. Using the system will be as easy. After the easy account creation and setup process, all the admin and professor roles need to do is scan QR codes to record attendance. This will be akin to scanning goods at a self-checkout at a grocery store. Those who are less confident will find iAttend easy to learn and use, since its learning curve is low. Those concerned about security need not worry, since security has been a top priority of the development team since the project's conception.

1.2 List of Deliverables, with delivery dates

Name	Date
Project Selection	01/15/2020
Create Blog	02/05/2020
Requirements Specification Document and Press Release 1	02/10/2020
Work Schedule	02/17/2020
Presentation 1	02/19/2020
Design Document 1	03/18/2020
Design Document 2	04/01/2020
Presentation 2/Prototype	04/06/2020
Press Release 2	04/15/2020
User Manual	04/27/2020
Project Due	04/29/2020

1.3 Dates of scheduled reviews of plan and project schedule

The development team will meet at various points before the project deadline to review the progress of the project and the plan to develop it. The group will meet bi-weekly to review the project starting on March 16. As the due date draws closer, it will move to weekly meetings. This means that the team will review their schedule and project design on March 16 and 30, and April 13, 20, and 27.

1.4 Glossary of Terms

- "The Software" shall refer the iAttend system in its entirety, including all devices involved.
- "The Team" shall refer the software development team in charge of creating The Software, known as Three Furious Locomotives.
- "The Administrator (Admin)" shall refer the owner of an iAttend Administrator account, which will have permission change configuration of their course as set up in the iAttend portal. For example, administrators would be department chairs or a professor configuring iAttend for their own classroom use.
- "The Moderator (Mod)" shall refer a user with elevated permissions. This user does not have permission change configuration of an established course but is able scan users and manually change attendance records. Professors assisting The Administrator with scanning would be examples of moderators.
- "The User (User)" shall refer a user with an ID that has no permissions beyond viewing their own attendance records. E.g. students, attendees.

- “The Portal” shall refer the web site that is accessible via a web browser. It will allow all types of users log in and view/edit records depending on their permission levels.
- “Course” shall refer the environment set up by The Administrator. It includes a roster of users record attendance for and has its own set of attendance records for each user for each event that takes place in that course. This could be a Math 101 course in Spring of 2020, for instance.
- “Check-In” shall refer the act of a user being scanned by a moderator or administrator, thereby recording attendance. This process involves the scanner scanning the QR code, identifying The User, and recording that attendance affirmation the database for the course.
- “Event” shall refer the specific event within the course for which a set of check-ins are grouped by. This could be attendance for the third lecture of a course, for instance.
- “The App” shall refer the mobile application for iOS primarily responsible for scanning QR codes and sending the check-in the database.
- Personally Identifiable Information (PII) refers information about a user that makes her/him identifiable in the real world, such as name, address, birthday, etc.
- Secure Sockets Layer (SSL) refers a way that data is securely sent across the Internet using encryption.
- Public Key Encryption refers a way of encrypting data where one key is publicly available while another key is privately held by the software owner. Together, these keys allow for encryption and decryption of data.
- Simple Mail Transfer Protocol (SMTP) is a way of transmitting emails.

2 Project Organization

2.1 Software Life Cycle Model chosen

The Team shall use the Kanban SLCM. There may be slight variations to the default model to better keep track of who is working on what and how far along they are in the process. For example, each team member may have his own "In Progress" board, and there may be additional boards for testing or items that need additional work. Kanban was chosen due the nature of the project, where there are many different types of services that need be implemented. Therefore, The Team will be using best practices for development with micro-services in order facilitate thorough testing and the best system integration. Kanban was seen as a good option where The Team can freely pull items from the backlog without worrying about sprints or other restraints put in place by other SLCMs.

2.2 Identification of responsible parties for major project functions and activities

Much of the programming workload will be shared between the members of the project. Using Kanban boards makes this easy since each member can simply pull an issue and work on it if they believe that they can at least partially complete it. Each member will have additional responsibilities outside of the coding of the project. Alec Barker will be in charge of testing various aspects of the project to detect flaws, Matthew Dutt will create the architecture for the code areas of the project, and Michael Jarman will control the configuration management and version control of the project.

3 Managerial Process

3.1 Team Reporting/Monitoring Mechanisms (how will the team judge its progress?)

The Team is using GitHub Projects boards and GitHub issues to create a list of desired features that need implemented. As the issues are completed, the GitHub Projects boards will visually display the progress of the project and its completion. A Gantt chart was created to create a rough schedule of each part of the project, which should also help with keeping track of progress.

4 Technical Process

4.1 Tools (compilers, editors, Configuration Management software, ...)

- Github–Configuration management and version control
- Visual Studio–Integrated Development Environment
- ASP.NET Core–A framework for developing web applications with a C# code back-end
- Xamarin–A library for developing cross-platform mobile applications
- PostgreSQL–The object-relational database system

4.2 Documentation strategy (location, update frequency, Internet usage)

The program files will feature documentation to specify what chunks of code do. Documentation documents will be stored on GitHub and will be publicly accessible on the Internet. These documents will be updated if needed as progress is made.

5 Risk Analysis

5.1 Risk identification

Probability:

Low

Medium

High

Severity:

Minimal

Moderate

Critical

Catastrophic

Hazards:

- Illness or burnout of one or more team members
  - Probability: High
  - Severity: Critical
- Schedule conflicts
  - Probability: High
  - Severity: Moderate
- University Networking and Security shooting it down for being insecure
  - Probability: Medium
  - Severity: Catastrophic
- Music department shooting it down last minute
  - Probability: Medium
  - Severity: Minimal
- Azure being difficult to implement, develop and deploy
  - Probability: Low
  - Severity: Catastrophic
- Xamarin is difficult to program with or doesn't work right
  - Probability: Low
  - Severity: Catastrophic
- Intercept of PII
  - Probability: Low
  - Severity: Catastrophic
- Team members' development environments stop working

  - Probability: Low
  - Severity: Moderate

- GitHub starts subscription model
  - Probability: Low
  - Severity: Moderate

- Coronavirus devastates campus
  - Probability: Low, but here we are.
  - Severity: Moderate

- Some other software existing that does it better and for free
  - Probability: Low
  - Severity: Minimal

- The Music Department becomes too large for professors to recognize their students
  - Probability: Low
  - Severity: Minimal

5.2 Risk management strategies for high-likelihood risks as well as catastrophic risks that have a reasonable chance of occurring

- Illness of one or more team members
  - Probability: High
  - Severity: Critical
  - Prevention: Take care of ourselves, eat and sleep well, and exercise.
  - Solution: Redistribute workload until team member(s) recover.

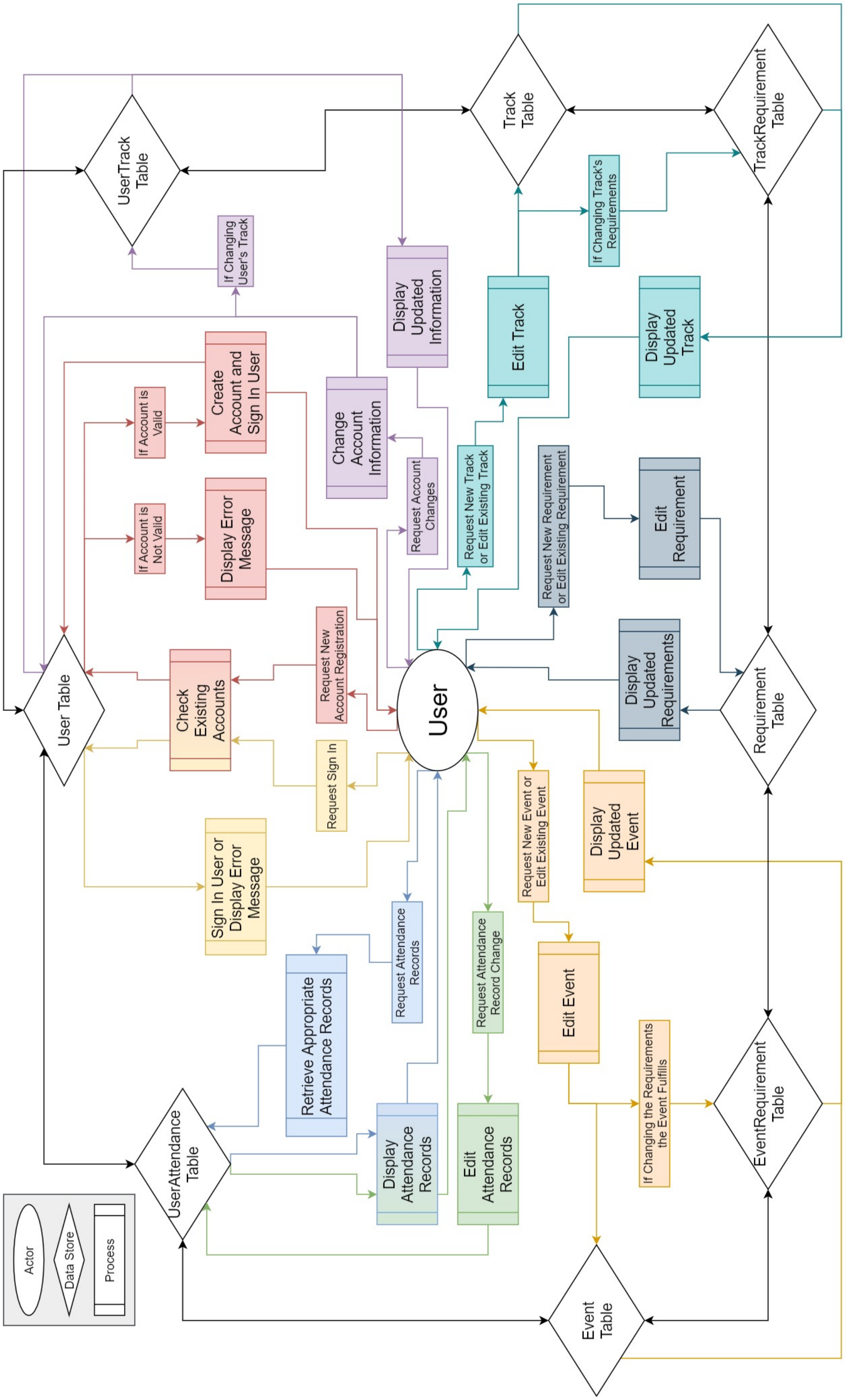
- Burnout of one or more team members (but not all)
  - Probability: High
  - Severity: Critical
  - Prevention: Balance the workload, work together, try to make it fun.
  - Solution: Have an intervention, do a fun/bonding activity to help.

- Schedule conflicts
  - Probability: High
  - Severity: Moderate
  - Prevention: Communicate schedules, stay connected with technology.
  - Solution: Work separately.

- Everyone burning out at the same time
  - Probability: Medium
  - Severity: Catastrophic
  - Prevention: Balance the workload, work together, try to make it fun.
  - Solution: Have an intervention, do a fun/bonding activity to help.

No matter what, the Three Furious Locomotives will just keep on chugging.

6 Architectural Design Specification


UserTrack Table
Track Table
TrackRequirement Table
If Changing User's Track
If Changing Track's Requirements
Display Updated Information
Edit Track
Display Updated Track
Create Account and Sign In User
Change Account Information
If Account is Valid
If Account is Not Valid
Display Error Message
Request Account Changes
Request New Track or Edit Existing Track
Request New Requirement or Edit Existing Requirement
Edit Requirement
Request New Account Registration
User
Check Existing Accounts
User Table
Display Updated Requirements
Requirement Table
Request Sign In
Sign In User or Display Error Message
Request New Event or Edit Existing Event
Display Updated Event
Request Attendance Records
Request Attendance Record Change
Edit Event
Retrieve Appropriate Attendance Records
If Changing the Requirements the Event Fulfills
EventRequirement Table
UserAttendance Table
Display Attendance Records
Edit Attendance Records
Event Table
Actor
Data Store
Process