Technical Design Document

AIMS - Apprentice Internship Management System



# Version History

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Description** |
| 2016/05/25 | Cole Chambers | Initial Creation |
| 2016/06/15 | Jeff Pickett | Initial Creation |
|  |  |  |
|  |  |  |

# Acronyms

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| SOW | Statement of Work |
| TDD | Technical Design Document |
|  |  |
|  |  |
|  |  |
|  |  |

# References

1. AIMS SOW, EFC Salesforce / Tinderbox

# Table of Contents

[1 Version History 2](#_Toc453702327)

[2 Acronyms 2](#_Toc453702328)

[3 References 2](#_Toc453702329)

[4 Table of Contents 3](#_Toc453702330)

[5 Introduction 4](#_Toc453702331)

[6 Scope 5](#_Toc453702332)

[6.1 Mobile App Features 5](#_Toc453702333)

[6.2 Web App Features 5](#_Toc453702334)

[7 Mobile Wireframes 6](#_Toc453702335)

[7.1 Login 6](#_Toc453702336)

[7.2 Register 7](#_Toc453702337)

[7.3 Schedule 8](#_Toc453702338)

[7.4 Daily Check In 9](#_Toc453702339)

[7.5 Weekly Survey 10](#_Toc453702340)

[7.6 Raise Hand / Ask Question 11](#_Toc453702341)

[7.7 Profile 12](#_Toc453702342)

[8 Web App Wireframes 13](#_Toc453702343)

[8.1 Login 13](#_Toc453702344)

[9 Database Diagram 14](#_Toc453702345)

[9.1 User / Group Data Model 14](#_Toc453702346)

[10 API Specifications 15](#_Toc453702347)

[11 Technical Architecture (Placeholder) 16](#_Toc453702348)

# Introduction

The purpose of this document is to expand upon the SOW and define the technical architecture and high level design required to fulfill project scope. Anything not included in this document is considered out of scope and may incur additional time and costs.

The Apprentice Internship Management System (AIMS) is an application that works to resolve pain points that plague the academy administrators, instructors, and students. The system will provide for tracking of apprentice profile information, attendance and hours, course scheduling with times and locations, and provide a mechanism for a student to ask questions anonymously within a classroom setting.

Upon completion, the participating apprentices and all future apprentices will have a functional architecture that demonstrates best practices in communicating with a back-end API powered by a custom relational database with multiple user interfaces for the web, iOS, and Android. The completed project will be an adaptable legacy product that benefits all parties involved. Future apprentices can expand upon this foundation to address future needs of the academy while greatly enhancing their portfolio with a full stack product.

# Scope

This document is structured as follows:

* Mobile Wireframes
* Web Wireframes
* Database Diagram
* API Specifications
* Technical Architecture

## Mobile App Features

The native mobile apps will include the following features:

* Login / Forgot Password / Logout
* Registration / Account Creation - Basic username / password only
* Course Schedule Calendar Interface with times & locations
* Daily Check In - Utilize geo-fences to check people in / out
* Weekly Survey – Allow a student to be sent a template survey to gather feedback
* Raise Hand / Ask Question - Allows students to ask questions, anonymously if desired
* User Profile – View and update user information

## Web App Features

The administrative web app will include the following features:

* Stuff
* More stuff

# Mobile Wireframes

Below are proposed wireframes for the mobile application

## Login

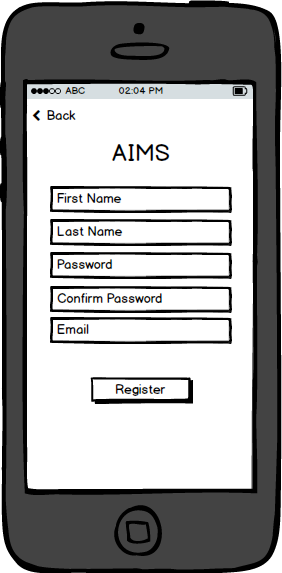
The login page is a standard login page that passes the username and credentials to the server via SSL returning an authentication token.

Also on the page is a Forgot Password button that will send the server a API call that will initiate a password change email.



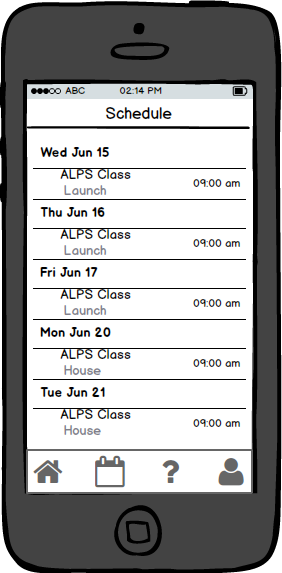
## Register

The register page is the page that allows a new user to create an account on the server. While this does create the account, it is expected that an administrator will still need to assign roles to the user after the user has registered and created their account.



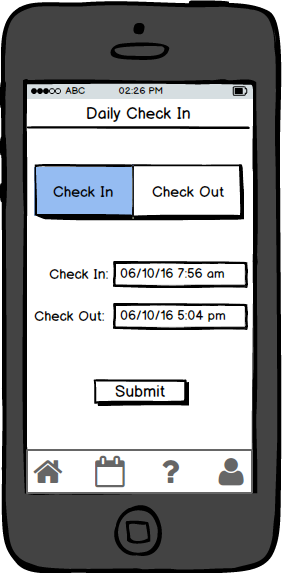
## Schedule

The schedule screen will allow the user to view the date, time and location of each of the events in their schedule stored on the server.



## Daily Check In

The daily check in will allow the user to log when they arrive and leave for the day. This will automatically log the time and location of each event, and verify if the user is within established geo-fences. The first time this screen is used, the user will be asked for permission to access their location.



## Weekly Survey

This page will allow a user to take a quick survey on a regular basis. The questions are stored on the server which will allow for the questions to be dynamic. The answers will be in two parts. The first part is a score (e.g. 1-5 stars) and the second will be an optional note if the user desires to provide context around the score.



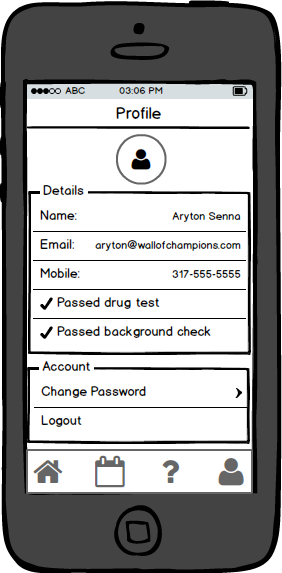
## Raise Hand / Ask Question

This screen allows a student to ask a question during a class which will be seen by a teacher and the rest of the class. The first page shows the questions currently being asked in the class by the student and other students, the second page is where the question is asked. Asking the question anonymously is an option.



## Profile

This screen is where a user can view and edit their profile information. This screen also allows for the user to change their avatar and password. Additionally, there are read-only fields that are shown only if indicated by the server.



# Web App Wireframes

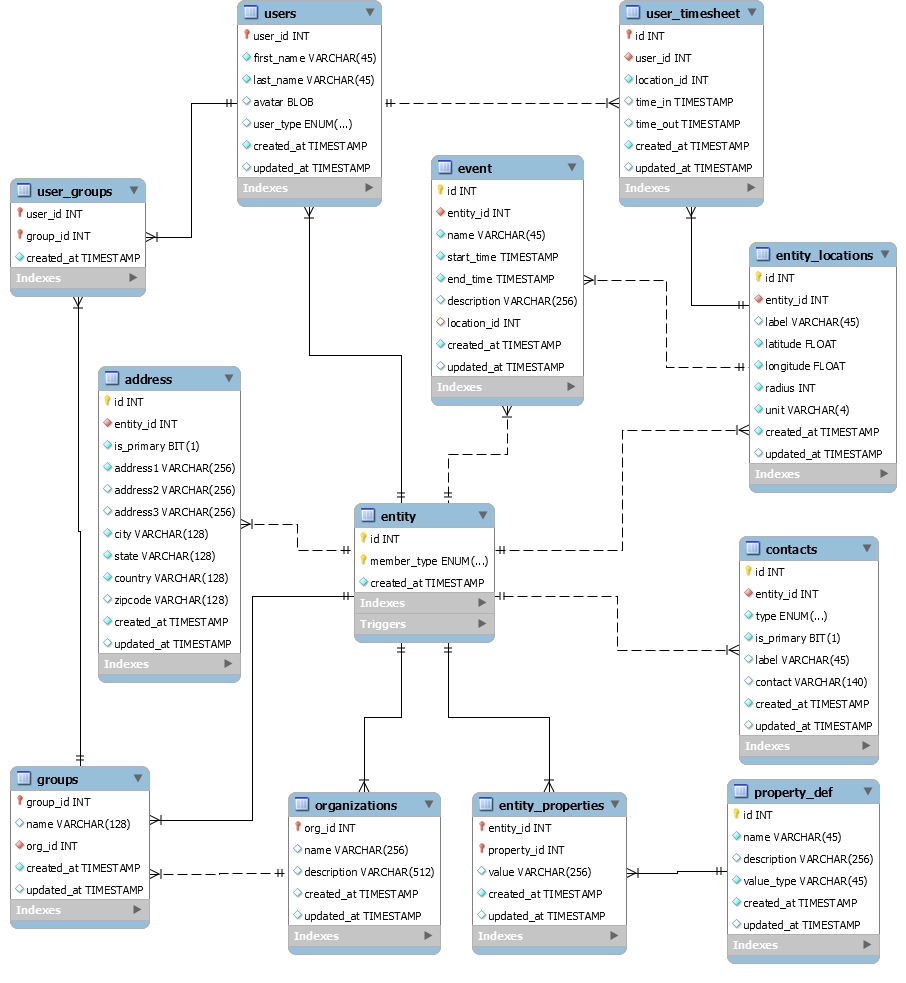
Below are proposed wireframes for the admin application

## Login

Text here…

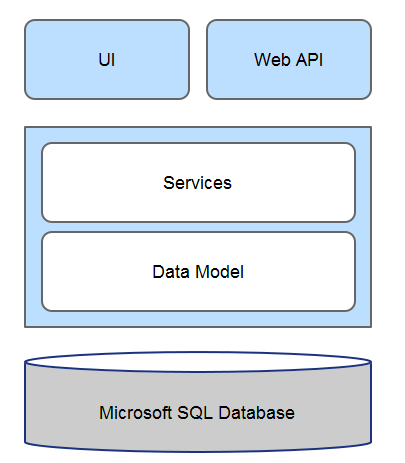
# Database Diagram

## User / Group Data Model



# API Specifications

# Technical Architecture (Placeholder)



**SQL Database**: This component is the database where backend data will be stored.

**Data Model**: This component is the data model that represents the database tables.

**Services**: This component is the assembly that encapsulates the business logic of the app.

**UI**:

**Web API:**