Critical Labs Alarm Clock Challenge Requirements Document

I. Scope

A. Problem Statement

"

Johnny keeps oversleeping and missing class! He needs your help to build an alarm clock that can keep track of his schedule and make sure he's up in time. He's heard of this cool Javascript framework called ReactJS and asked you to build a web frontend using it.

He also heard that Javascript can be used for the backend, too, and was thinking that you could develop a simple API to manage his schedule using NodeJS and ExpressJS. Johnny's schedule is as follows

Monday - 7am
Tuesday - 9am
Wednesday - 9am
Thursday - 7:45am
Friday - 8:15am
Saturday - Sleep in! (no alarm)
Sunday - Sleep in! (no alarm)

"

B. Goal

The goal of this change is to deliver a web application to the company Critical Labs that allows for scheduling and alarm notifications to ensure that Johnny is able to wake up on time.

C. Objectives

The Application will fulfill the following features

1. Creation, Reading, Updating, and Deleting of alarms
2. Text message based notifications upon alarm date and time
3. Web-based push notifications upon alarm date and time

D. Not in Scope

1. Full day scheduling of events other than alarms
2. Native Mobile application

II. Requirements

A. Functional

#	Requirement	Description
1	CRUD functionality on alarms	The application will allow for basic CRUD functionality (Create, Read, Update, and Delete) on each alarm set.
2	Text notifications	The application will optionally be able to send text-based notifications to a provided number in order to remind the user of alarms when away from their computer.
3	Web notifications	The application will supply notifications when an alarm goes off and the browser window is open.

B. Non-Functional

1. The web application must not have reasonable response times with low latency.
2. All notifications must be sent/delivered in a reasonable timeframe so alarms remain effective.

C. Usability

1. Web application must have consistent functionality across devices (Responsive Design).
2. If web application encounters errors they must be either handled internally or displayed to the user in a friendly and informative way.

D. Technical

1. All dependencies must be documented and explained
2. All personal information and any identifying information given or automatically collected on the application must not be disclosed publicly without electronic consent by the user.

III. System Interaction

A. Twilio

1. Phone Number Verification
2. Alarm Text Messaging Service

B. Firebase

1. Authentication
2. Database

C. Apache

1. Hosting Platform

IV. Assumptions

A. The end user is able to either
   1. Keep their browser window open and volume up when expecting any wake-up alarm.
   2. Or is willing and able to give their mobile phone number to the application so messages can be sent to alert of alarms when the computer is not close or able to be kept open.
B. The end user will be using either a modern browser or a smartphone to access the application

V. Constraints

A. Monetary

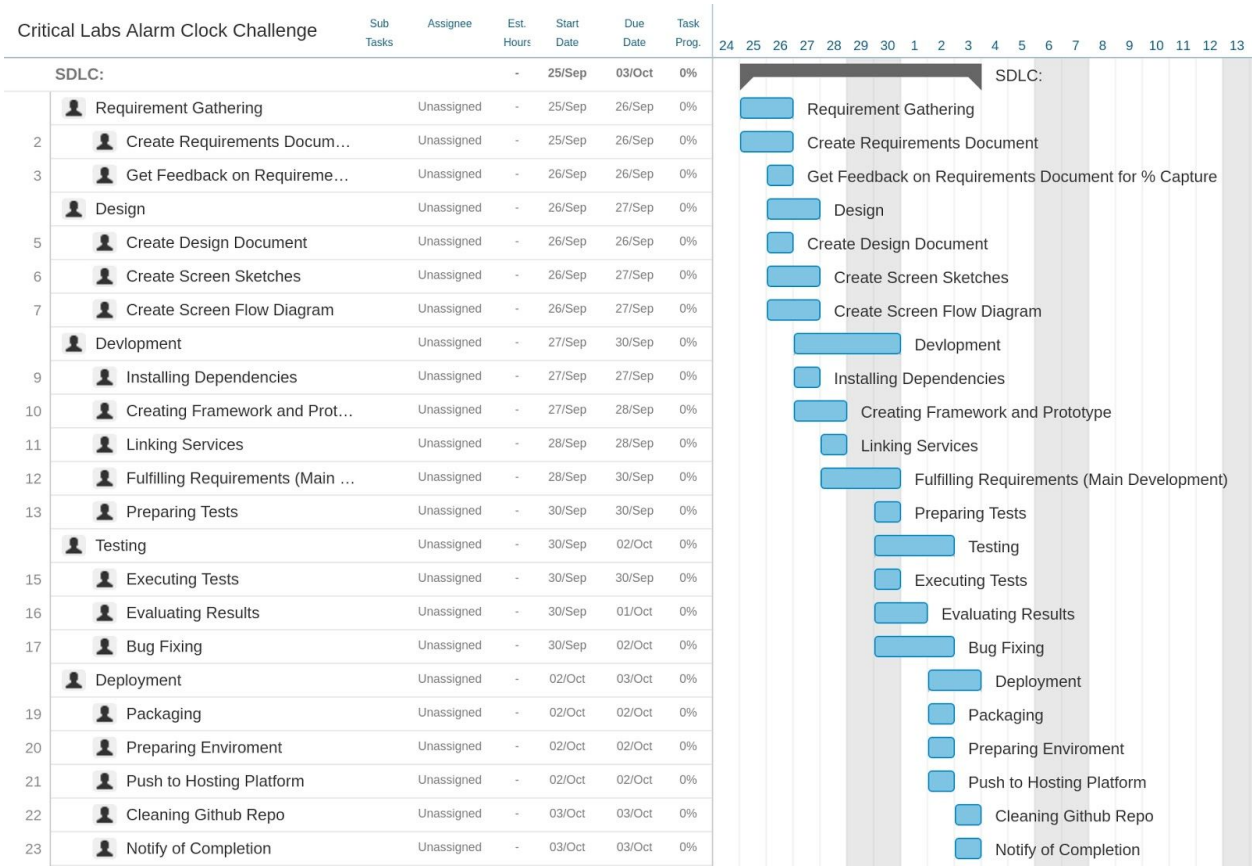
There will be no funds delegated to this project. Thus, with a budget of $0, all dependencies and supporting systems must be free or kept on the free trial limits of other systems.

B. Time

This project timeline is extremely short (approximately 9 days) from absolute start to finish. During that time I plan to step through the entirety of the Software Development Life Cycle (SDLC). Additionally, school work will not slow with exams approaching. Thus, time is at a premium and will need to be managed effectively to accomplish the project requirements by Wednesday October, 3rd.

VI. Timeline

What follows is a tentative timeline of the steps I will be taking to complete the project on time and on budget. The Gantt chart is ideal for this short time frame as it illustrates the waterfall nature of this project well (waterfall vs agile as there is little time to go back and correct mistakes made initially).



	Critical Labs Alarm Clock Challenge	Sub Tasks	Assignee	Est. Hours	Start Date	Due Date	Task Prog.
	SDLC:			-	25/Sep	03/Oct	0%
	Requirement Gathering		Unassigned	-	25/Sep	26/Sep	0%
2	Create Requirements Docum…		Unassigned	-	25/Sep	26/Sep	0%
3	Get Feedback on Requireme…		Unassigned	-	26/Sep	26/Sep	0%
	Design		Unassigned	-	26/Sep	27/Sep	0%
5	Create Design Document		Unassigned	-	26/Sep	26/Sep	0%
6	Create Screen Sketches		Unassigned	-	26/Sep	27/Sep	0%
7	Create Screen Flow Diagram		Unassigned	-	26/Sep	27/Sep	0%
	Devlopment		Unassigned	-	27/Sep	30/Sep	0%
9	Installing Dependencies		Unassigned	-	27/Sep	27/Sep	0%
10	Creating Framework and Prot…		Unassigned	-	27/Sep	28/Sep	0%
11	Linking Services		Unassigned	-	28/Sep	28/Sep	0%
12	Fulfilling Requirements (Main …		Unassigned	-	28/Sep	30/Sep	0%
13	Preparing Tests		Unassigned	-	30/Sep	30/Sep	0%
	Testing		Unassigned	-	30/Sep	02/Oct	0%
15	Executing Tests		Unassigned	-	30/Sep	30/Sep	0%
16	Evaluating Results		Unassigned	-	30/Sep	01/Oct	0%
17	Bug Fixing		Unassigned	-	30/Sep	02/Oct	0%
	Deployment		Unassigned	-	02/Oct	03/Oct	0%
19	Packaging		Unassigned	-	02/Oct	02/Oct	0%
20	Preparing Enviroment		Unassigned	-	02/Oct	02/Oct	0%
21	Push to Hosting Platform		Unassigned	-	02/Oct	02/Oct	0%
22	Cleaning Github Repo		Unassigned	-	03/Oct	03/Oct	0%
23	Notify of Completion		Unassigned	-	03/Oct	03/Oct	0%

VII. Evaluation Plan

Following the project delivery will come the evaluation period. During that time I will look back at all of my documents once again and compare what I promised to what I made. I will call this project a success if I deem that I passed all of my functional and non-functional requirements. A failure would mean that I did not complete or adequately complete one or more type of requirement.

VIII. Conclusion

This Document was written by
Jameel Kelley
9/27/18