System Requirement Specifications

Darien Cortez, Jakob Lopez, Cory Press

2019-30-04

Contents

1 Revisions **2**

2 Introduction **3**
2.1 Purpose . . . 3
2.2 Scope . . . 3
2.3 Acronyms, Abbreviations and Definitions . . . 3
2.4 References . . . 3

3 Description **3**
3.1 Context of Product . . . 3
3.2 Product Function . . . 4
3.3 User Characteristic . . . 4
3.4 Constraints . . . 4
3.5 Assumptions and Dependencies . . . 4

4 Specific Requirements **4**
4.1 Functional Requirements . . . 4
4.1.1 REQ1. Authentication . . . 4
4.1.1.1 Introduction . . . 4
4.1.1.2 Inputs . . . 5
4.1.1.3 Processing . . . 5
4.1.1.4 Outputs . . . 5
4.1.2 REQ2. Professor Search . . . 5
4.1.2.1 Introduction . . . 5
4.1.2.2 Inputs . . . 5
4.1.2.3 Processing . . . 5
4.1.2.4 Outputs . . . 5
4.1.3 REQ3. Professor Status . . . 6
4.1.3.1 Introduction . . . 6
4.1.3.2 Inputs . . . 6
4.1.3.3 Processing . . . 6
4.1.3.4 Outputs . . . 6
4.1.4 REQ4. Alerts . . . 6
4.1.4.1 Introduction . . . 6
4.1.4.2 Inputs . . . 6
4.1.4.3 Processing . . . 6
4.1.4.4 Outputs . . . 6
4.1.5 REQ5. Schedule Update . . . 7
4.1.5.1 Introduction . . . 7
4.1.5.2 Inputs . . . 7
4.1.5.3 Processing . . . 7
4.1.5.4 Outputs . . . 7
4.1.6 REQ6. Favorite . . . 7
4.1.6.1 Introduction . . . 7

4.1.6.2 Inputs . . . 7
4.1.6.3 Processing . . . 7
4.1.6.4 Outputs . . . 7
4.1.7 REQ7. Keep user signed in . . . 8
4.1.7.1 Introduction . . . 8
4.1.7.2 Inputs . . . 8
4.1.7.3 Processing . . . 8
4.1.7.4 Outputs . . . 8
4.1.8 REQ8. Forgot password . . . 8
4.1.8.1 Introduction . . . 8
4.1.8.2 Inputs . . . 8
4.1.8.3 Processing . . . 8
4.1.8.4 Outputs . . . 8
4.2 Non-functional Requirements . . . 9
4.2.1 External Interface Requirements . . . 9
4.2.1.1 User Interfaces . . . 9
4.2.1.2 Hardware Interfaces . . . 9
4.2.1.3 Software Interfaces . . . 9
4.2.1.4 Communication Interfaces . . . 9
4.3 Performance Requirements . . . 9
4.3.1 Design Constraints . . . 9
4.3.2 Attributes . . . 9
4.3.2.1 Availability . . . 9
4.3.2.2 Security . . . 10
4.3.2.3 Maintainability . . . 10
4.3.3 Other Requirements . . . 10
4.3.3.1 Database . . . 10
4.3.3.2 Operations . . . 10

5 Appendices **10**
5.1 Motivation/Scope of Project . . . 10

1 Revisions

Revision 1.0 : Rough Draft : February 14, 2019 : Darien Cortez, Jakob Lopez, Cory Press
Revision 2.0 : Orignal : Februay 16, 2019 : Jakob Lopez
Revision 2.1 : Finalized Version : April 30, 2019 : Darien Cortez, Jakob Lopez, Cory Press

2 Introduction

2.1 Purpose

The purpose of this document is to provide a detailed overview of the required capabilites Are You Busy must recognize within the system's constraints. This document will define the purpose, scope, acronyms and abbreviations, references used, and the functionality and design of the system.

2.2 Scope

The system will be available as an app on Android mobile devices. Users will have access to store, view and edit their schedule through a digital interface. This information will be stored in a real-time database.

2.3 Acronyms, Abbreviations and Definitions

- AYB - Are You Busy
- Firebase - Google's mobile platform real-time database and backend service
- Ionic - Framework used for hybrid mobile app development
- Cordova - Framework that uses JavaScript bindings for native mobile functionality

2.4 References

https://ionicframework.com/docs/components/
https://firebase.google.com/docs/firestore/
https://cordova.apache.org/docs/en/latest/
https://tex.stackexchange.com/questions/60209/how-to-add-an-extra-level-of-sections-with-headings-below-subsubsection
http://user.ceng.metu.edu.tr/ e1679216/documents/SRS.pdf

3 Description

3.1 Context of Product

This software system will be a tool used at Midwestern State University by the Computer Science department. This system will be designed to maximize the time efficiency of students by providing information to professors' office hours and real-time availability. The development of such system will instill an official means of scheduling appointments and will surpass the current error-prone process by allowing professors to easily organize their office hours and reschedule appointments.

3.2 Product Function

- Allows users a high-level view of professors' schedules including office hours, scheduled appointments and current availability
- Allows users to easily flow through the app to find a specific professor using Android's touch screen properties
- Provides professors with a visual timeline as they dynamically manipulate their schedule
- Provides users with quick access to information by developing the system for mobile devices

3.3 User Characteristic

The user must have a basic knowledge of smart phones and how to use the app store. In order to maximize the efficiency of the system, the user must have notifications enabled in their phone's settings. The phone should have touch screen capabilities and a keyboard. There are two types of user accounts: student and professor. The user must select the appropriate account type.

3.4 Constraints

The system's work space is limited by only being developed for Android mobile smart phones.

3.5 Assumptions and Dependencies

Firebase is an online database and requires an internet connection, so this system assumes the user will be connected to wifi while using the app. This system is most effective if all students use the app as it was intended, therefore all students would have to have an Android phone.

4 Specific Requirements

4.1 Functional Requirements

4.1.1 REQ1. Authentication

4.1.1.1 Introduction

AYB must allow students and professors to sign up or login to an account

4.1.1.2 Inputs

After opening the app, users will be brought an authentication page where they can either sign in to an existing account or sign up for a new account.

If signing in, the user will input information into two text fields for account verification. An email, preferably the MSU assigned email, and corresponding account password of at least six characters will be entered to sign users in.

If signing up, the user must fill in all fields to authenticate and account and get inserted into the database. Fields include: name, email, password, confirmation password, account type.

4.1.1.3 Processing

While the user is filling out a field, the input field is checking if what the user is typing is valid. An email must have an '@' and a '.' to be valid. A password must have at least 6 characters to be valid. The confirmation password must be the same as the original password. Users can only select one type of account. If all the above are true, the system is idle until a submit button is clicked. If signing in, the system checks the database to see if an account exists with the given credentials. if signing up, the system authenticates the user and adds them to the appropriate collection in the database.

4.1.1.4 Outputs

If successful in signing in to an existing account, the user will be sent to their profile page. If the credentials did not match an account, the user will get an error message. If successful in signing up a new account, the user will be sent to their profile page. User will not be able to register account if sign up fields are invalid because the sign up button will be disabled.

4.1.2 REQ2. Professor Search

4.1.2.1 Introduction

AYB must allow all users to search for a teacher

4.1.2.2 Inputs

The user will select to go to a search page. The search page will provide the user with a list of professor accounts from which they can select.

4.1.2.3 Processing

When the user selects to search for professors, the system will query all Computer Science department professors from the database

4.1.2.4 Outputs

When the user selects a professor account, it will bring the user to the requested account

4.1.3 REQ3. Professor Status

4.1.3.1 Introduction

AYB must allow all users to view a professor's schedule

4.1.3.2 Inputs

User must select a professor accout to view their schedule

4.1.3.3 Processing

Professors' schedule is stored in the database, as well as their current status and appointment timeline. When viewing a professor account, all of this information will be queried from the database.

4.1.3.4 Outputs

Everything queried from the database will be displayed on the professor's profile. This includes their name, contact info, picture and their appointment/availability information.

4.1.4 REQ4. Alerts

4.1.4.1 Introduction

AYB shall send push notifications to users phones when appointments are made or cancelled

4.1.4.2 Inputs

When the user is viewing a professor's page, they will have the option to make an appointment. A reason must be specified as to why the appointment is being made. If the professor does not want to have an appointment at the requested time, they can send a denial message.

4.1.4.3 Processing

An appointment creates a relationship between a user and a professor in the database. The uid of both the users will be stored in a Firebase appointments collection. All appointments will be considered accepted, so it is the professor's duty to deny it. Cordova will be used to create a push notification that will alert the users of their account's activity.

4.1.4.4 Outputs

A push notification will be sent to the users' phone to let them know an account has requested/denied an appointment. The appointment will be displayed in the professor's appointment timeline.

4.1.5 REQ5. Schedule Update

4.1.5.1 Introduction

AYB must update a professor's schedule when an appointment is made

4.1.5.2 Inputs

A user needs to create an appointment request, which has information pertaining to the time of the future appointment and how long it will be.

4.1.5.3 Processing

Professor's have an appointment timeline that is stored in their user document in the database. When a request is made for an appointment, it gets added to the appointment timeline field of the given professor. Other appointments will not be allowed during the requested time.

4.1.5.4 Outputs

Shows as an invalid selection in the professor's schedule. When it is time for the appointment to happen, the professor's availability status will change to "unavailable" on their profile and then back to available when the appointment ends. This status will be able to be toggled in case of an early or late ending appointment. Schedules and statuses can change dynamically for professors' convenience.

4.1.6 REQ6. Favorite

4.1.6.1 Introduction

AYB must allow users to have a list of favorite professors.

4.1.6.2 Inputs

When the user visits a professor's page, there should be a button to favorite/unfavorite the specific professor.

4.1.6.3 Processing

When a user toggles the favorite button, a method is called from the Database-Provider. Favorite professors are stored in a collection within the user's document named "Favorites". The professor's uid will be stored in the collection when being added and removed when being unfavorited.

4.1.6.4 Outputs

The button will change each time it is toggled. It will be a star if the professor is a favorite or a box displaying the star and "Add as favorite" if not a favorite. On the user's profile page, their favorite list will be displayed and change in real-time. Having a list of favorite teachers allows for quicker access to their page.

4.1.7 REQ7. Keep user signed in

4.1.7.1 Introduction

AYB must allow users to stay signed in even after exiting the system.

4.1.7.2 Inputs

The user logs into their account.

4.1.7.3 Processing

When a user logs into their account, their information is pulled from the database so the proper information can be displayed on the profile page. The system should convert that information to a JSON object and save it in the devices local storage. If the user logs out, the local storage will be cleared.

4.1.7.4 Outputs

When the user closes out of the system without logging out and then they reopen the app, they will not have to re-signin. The system will use the information from local storage to instantly bring them to their profile page.

4.1.8 REQ8. Forgot password

4.1.8.1 Introduction

AYB must allow users to reset their password.

4.1.8.2 Inputs

The user logs clicks on the "Forgot password?" prompt on the LoginSignupPage. The user will enter their email they use to log in to the app and then click "Reset."

4.1.8.3 Processing

Since Firebase authenticates users in the database, it handles reset password. The AuthProvider will call a Firebase method to reset the password, which sends an email with a link to the user's specified email.

4.1.8.4 Outputs

Opens an alert telling the user to check their email and reroutes them to the LoginSignupPage. Once the user clicks the link, they can login with their Firebase generated password and be brought to their normal profile page.

4.2 Non-functional Requirements

4.2.1 External Interface Requirements

4.2.1.1 User Interfaces

The system interface will be composed of 3 main pages: login/register, profile, professor search. The design should be simple and clean, following the basic look of an ionic app. Users will interact with the system the Android's touch screen interface and keyboard, which allows for more versatile movement.

4.2.1.2 Hardware Interfaces

The app requires phone storage to be available because it is downloaded to the phone.

4.2.1.3 Software Interfaces

Ionic uses Cordova for native builds and requires the Android phone to support API levels 16 - 25 and Android versions 4.1 - 7.1.1.

4.2.1.4 Communication Interfaces

All communication between devices will be through Firebase.

4.3 Performance Requirements

Push notifications will provide users with appointment information. Professor's will recieve a notification when a student makes a request, and student's will recieve a notification if the professor denies the request. These notifications and changes in the databse should be almost instantaneous in order to provide the users with the most pertinent information.

4.3.1 Design Constraints

The system will only be deployed on Android devices.

4.3.2 Attributes

4.3.2.1 Availability

Key functionality of the system will not be available if the user does not have an internet connection. The user would not be able to authenticate, search for a professor, make an appointment, or recieve notifications that could be important(i.e. a professor denying an appointment).

4.3.2.2 Security

User password will be encrypted by Firebase's password protection function. When a user is created, their account gets authenticated into Firebase and the password is hidden. The password is not visible to even the owners of the database, and when a user forgets their password, they must reset a new one.

4.3.2.3 Maintainability

For the best system experience, users should be active on their account and maintain a correct schedule. Users should also be signed up under the correct account type (student or teacher). If a teacher account is made and they are not verified, the user will be deleted from the database.

4.3.3 Other Requirements

4.3.3.1 Database

The database must be properly concieved and normalized. Duplicate information must directly lead to a performance boost, where a database query would be slow without it. Objects in the system should subscribe to value changes in the database in order to take advantage of the real-time characteristics of Firebase.

4.3.3.2 Operations

5 Appendices

5.1 Motivation/Scope of Project

Every successful college student has met with a professor outside of class at least once throughout their academic career. Professors post their office hours so students can see what times he or she is available throughout the week. This mobile application aims to make the process of viewing a professor's schedule or scheduling an appointment more efficient. While office hours are public, a professor's current availability is not; students may attempt to talk to a teacher only to find out that they are busy, and would have to return at a later, arbitrary time in hopes that they will be available. AreYouBusy is a time management tool that allows students to view a professor's weekly schedule and daily appointments – proving to be effective for both students and professors.