Software Requirements Specification

for

CBYou

Version 1.0 approved

Prepared by Craig Mariani,Inderjit Hothi, & Andrew Battles

EGR-302-C

01/31/2019

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of CBYou is to design a mobile optimized website that will allow students of California Baptist University have a easy way to check important information that they need. CBYou will provide many features for students such as checking their schedule's, chapel services information, dining information, campus maps and even sports related content.

1.2 Document Conventions

Currently there are no fonts or highlights that are of special significance. Every requirement is at its' own priority. We do have some features that are going to be more highly prioritized than others. Such as the core features of CBYou which include student schedules, dining information and events. Other features will have medium priority and few features such as a redirect to Lancermail will be the one of the lower priority levels.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers who are currently working on the product as well as developers who are trying to grow their craft. The document is also intended for project managers and marketing staff who are tasked with advertising the project and planning the next move in product development. The rest of the SRS contains a general description of the product(see section 2), External Interface Requirements(see section 3), System Features(see section 4) and product requirements(see sections 5-6).

WILL GO BACK TO THIS LATER WILL FILL REST OF DOC FIRST

1.4 Product Scope

The product scope is to help students navigate faster and have all the information in platform rather than searching through many different areas of the compact website of CBU. Some benefits of CBYou would be to have easy accessible website for users to quickly get the information they need. The goal is to have many users using the website and to be received positively.

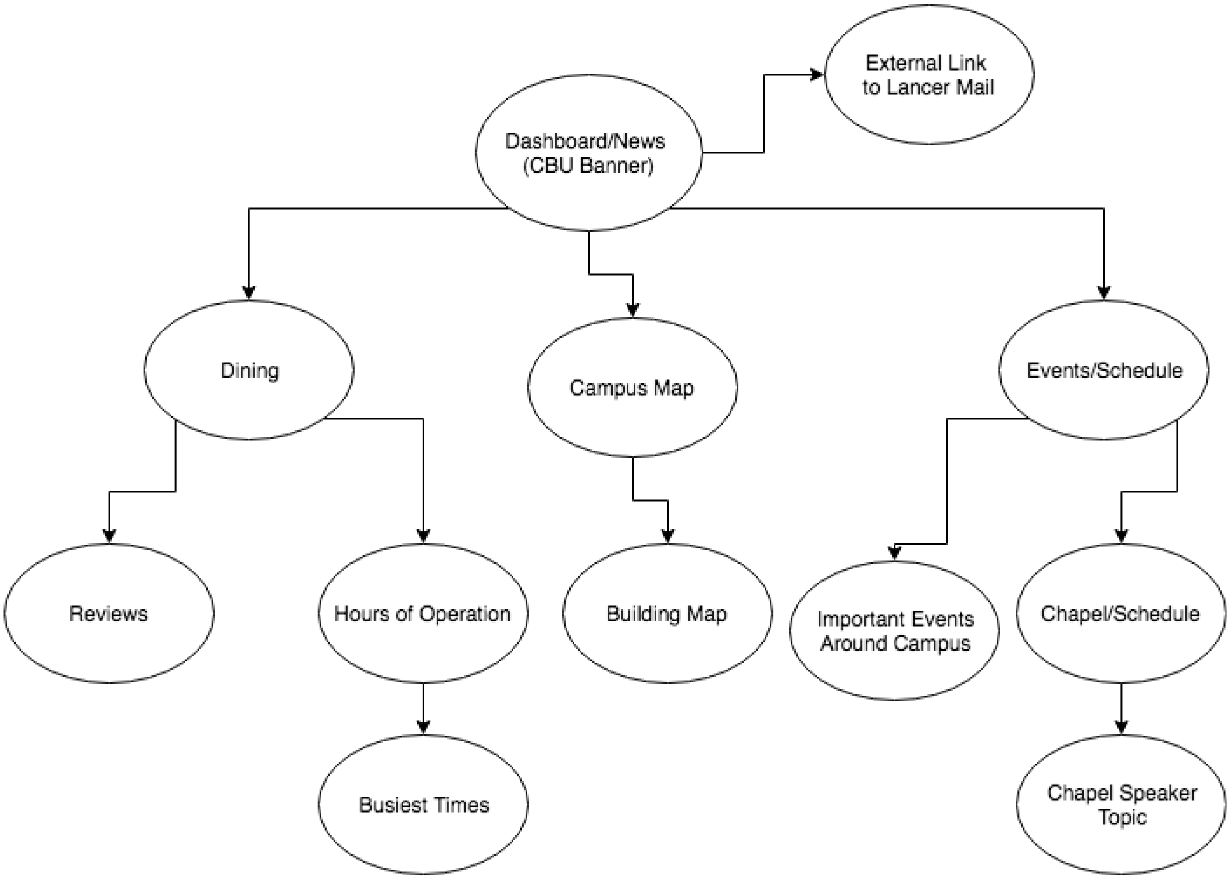
1.5 References

https://calbaptist.edu/
https://www.cbuonline.edu/
https://getbootstrap.com/
https://aws.amazon.com/

2. Overall Description

2.1 Product Perspective

The context and origin of CBYou is a follow-on member of the CBU product family. The CBYou web application is designed to aid the students when look up dining hours or schedules for classes and events by combining all of these into one convenient place for the user. The components of the larger system are insidecbu and lancermail. The requirements of this system include the ability for a user to sign in and access the site on mobile as well as desktop. When they are signed in the user is able to see the chapel/class schedule as well as events and dining.



2.2 Product Functions

- Build student schedules to output a better displaying schedule
- Chapel service weekly Schedule
- Dining: food reviews, restaurant hours, and hours that the business is busy.
- CBU sports information: scores, schedule of games.

2.3 User Classes and Characteristics

The user class for this software will mainly be the general user which in our product’s case would be a student that attends California Baptist University. The general user would typically use the website to check the different features throughout the day. Overall, the user classes would result in only the general users as it is geared towards just the students.

2.4 Operating Environment

The operating environment will be mainly directed towards just mobile phones that support browsers. However, it would still be able operate on desktop as well.

1. Mobile Devices
2. Desktop Computers

2.5 Design and Implementation Constraints

Some constraints that limit some of our product's features would be the school not giving us some permissions such as being able to log in with user's CBU credentials. Other constraints include time restrictions some feature may not be able to be included if not met the deadline. Another constraint that we had early on in the project was not having the right equipment for building a IOS application.

2.6 User Documentation

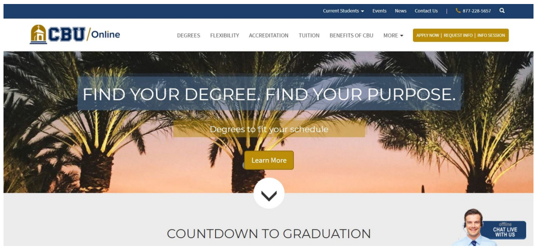
There are no user documentation as it will be a website and it will be ran like any other website that are build.

2.7 Assumptions and Dependencies

The only assumed factor that we have identified in our project would be getting the user's schedule from insidecbu and have it formatted in our desired design. We are still figuring out the method to implement for this feature.

3. External Interface Requirements

3.1 User Interfaces



The above picture is the user interface style our product will aiming towards. For our user interface of the home page, it will include a menu that lists the features we have mentioned in the SRS and it will also depict the University's main events occurring around that certain time. After the menu, the user can select whichever feature they choose to and the following style will follow throughout the website.

3.2 Hardware Interfaces

This product will be a web application and all hardware interfaces will be handled by the user's web browser Supported devices have been already stated in the SRS.Cloud 9 will be where we create the software.

3.3 Software Interfaces

The tools that we will be using will mainly be Amazon Web Services and we will be looking through bootstrap to achieve a template.Currently there is no plans for data sharing as of now.

3.4 Communications Interfaces

Our communications interface we will be trying to integrate Lancermail and Blackboard as redirects. For communication standards we will be using HTTP.

4. System Features(Insert Balsamic Mock Up)

4.1 Campus Dining Information

4.1.1 Description and Priority

The Dining webpage has a list of all dining options on CBU. Users can select specific restaurants to view it's operational hours, busiest times, and customer reviews.

The priority of this feature is medium - it should make final release but the website will not be inoperational without it. The highest priority system will be the review system as it will take the longest to implement.

4.1.2 Stimulus/Response Sequences

The user can access the dining webpage through the navmenu. The main page has a list of campus dining locations, each with a name, related picture, and their hours of operations. Users can navigate to a restaurant webpage by clicking/tapping on the name/picture of the restaurant. The restaurant's webpage has information regarding the busiest times, location on campus, menu items, & customer reviews. Users can leave a review by typing into a dialog box and submitting, if they are logged in.

4.1.3 Functional Requirements

REQ-1: User will be able to see all restaurant names & hours on main dining page.

REQ-2: User will be able to see when busiest times for a particular restaurant.

REQ-3: User will be able to find the location the location of the restaurant on a map.
REQ-4: User will be able to see a rating (1-5 scale) of the restaurant, rating is an average of all reviews.
REQ-5: User will be able to see individual comment reviews & ratings left by other users, and navigate through these reviews.
REQ-6: User will be able to submit a rating & comment review if logged in.

4.2 CBU Map

4.2.1 Description and Priority

The CBU Map is a interactable map users can use to find their way to locations on campus

4.2.2 Stimulus/Response Sequences

The user can access the map through the navbar and be able to locate buildings by clicking on the map. User can also search for buildings to find map.

4.2.3 Functional Requirements

REQ-1: User can see a list of buildings to load it
REQ-2: User can search through that list of buildings
REQ-3: User can see overall map
REQ-4: User can toggle categories of buildings (educational, dining, housing) to show or hide them.

4.3 Events & Scheduling

4.2.1 Description and Priority

Users can look at campus events and scheduling of its courses. This is high priority as it is the primary goal of CBYou.

4.2.2 Stimulus/Response Sequences

The user can navigate to the Schedule/Events through the nav menu. The user is greeted with an events schedule, or, if the user is logged in, their schedule. Users can add courses, edit courses, or remove courses from their schedule. Users can load school events over their schedule to see what is possible.

4.2.3 Functional Requirements

REQ-1: Users and guests will be able to see school events going on
REQ-2: Users will be able to add courses, edit courses, and remove courses from their schedule

REQ-3: Users will be able to see a graphic of their schedule and load school events over it (graphically)

REQ-4: Courses can be edited by changing meeting times or adding meeting times (if there are more than one classrooms/times for an individual course)

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The performance requirements for the product include being light and easy to pull up on a mobile device or desktop and being able to navigate quickly and efficiently. The User Interface is designed to be simple and easy to understand for users. It will also need to have access to real time to display events and understand the dates and times that these events take place. It will need to interact with the schedules and events that are planned on campus. Some features include displaying the dining hours, event times, class schedules, maps, and chapel times.

5.2 Safety Requirements

There is a chance that user may be hacked by an outside source, however this is unlikely because the hacker will only be able to access the class schedule of the student and possibly they're lancerMail rather than more sensitive information such as credit card numbers or social security numbers. Worst case scenario the class schedule will be changed by someone else or there will be email/review under their name that they did not send/post. These risks will be minimized thanks to the security requirements listed below.

5.3 Security Requirements

The main requirements for security will be for protecting the users information. This includes the users class schedule and lancermail login. The main user identity requirements will be the user ID (Student ID) and password, this will be identical to the user login for insideCBU and Lancer Mail. The website will be built with the security standards established by Google in July 2018, stating that websites needed an SSL or Secure Sockets Layer. On top of that the website will also be built with the same level of security and standards as the insideCBU website.

5.4 Software Quality Attributes

The product is designed to be a light and fast web application that can be pulled up on a desktop and be able to be scaled down easily on a phone when on the go. It is web based so a user with any operating system can access the site. The only things the user needs will be access to the internet either through wifi or mobile data and a device to access the user interface.

5.5 Business Rules

Students are able to login with user identification (they're student ID) once they are logged in they are allowed to read and change personal information such as class schedule, however they are not allowed to change certain parts of the application such as chapel times or events. Students are not allowed to see anything besides public information(chapel, events, dining) once they are logged out. Admin has the ability to change the updates on events and public gatherings however most of these will be automatic.

6. Other Requirements

All requirements are covered in the SRS as of 01/30/2019.

7. System Architecture

–High level overview of the overall system

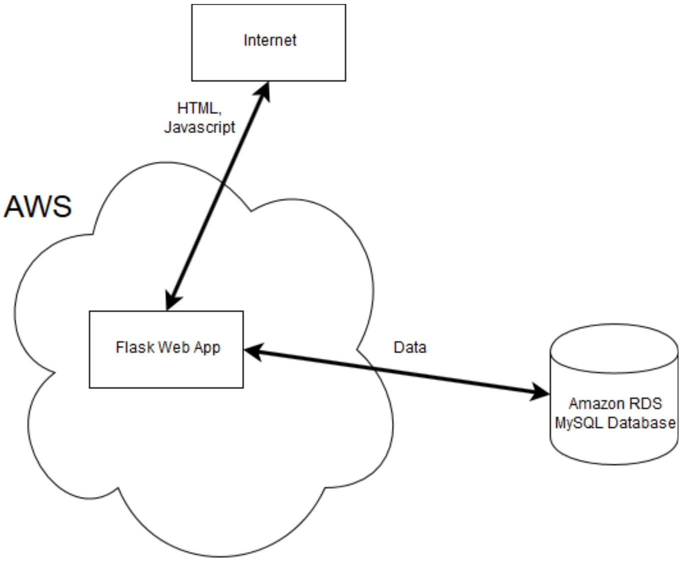
•The purpose of the design document is to give a high level overview of how the system works

–At a high level, show all the interacting components in your system

–How are they related? Data passing, library call, user action, etc.

–Show all services used

•Database, cache, 3rd party service, etc.



The web app will be using the Django framework and deployed on AWS. It will interact with a MySQL database hosted on AWS RDS.

8. Key Functionality

–Drill down to each component and explain how it will work

•Then take each component and explain how it works

–What is going on in each component

–What data is expect, how the data is processed, stored, used

–What is the output and where does it go

We will use a database, as far as what type of services we have not decided but currently leaning towards MySQL database systems(RDS, relational database). For the front end of the web app we will use HTML, CSS and Javascript with the framework, BootStrap. This is what the user will see when accessing For the back end we will use Python and Flask for the back end web framework.