Software Requirements Specification

Spartan Course Analysis & Matching

Version 2.0 - 5/16/17

San José State University Charles W. Davidson College of Engineering

Mark Casapao
Christian Timbol
Preyrna Yadav
Hiep Nguyen

Revisions

Version	Primary Authors	Description of Version	Date Completed
v1.0 draft	Mark Casapao Christian Timbol Preyrna Yadav Hiep Nguyen	An outline or foundation of the document	3/1/17
v1.0 final	Mark Casapao Christian Timbol Preyrna Yadav Hiep Nguyen	A finalized and confirmed iteration of the document	3/5/17
v2.0 final	Mark Casapao Christian Timbol Preyrna Yadav Hiep Nguyen	A refined iteration of the document. All docs are now interconnected.	5/16/17

Table of Contents

1.0 Introduction

This section will focus on the overlaying description of the product, and will provide a general outlook about the uses and applications of SCAM.

1.1 Product Overview

The purpose of this product is to provide the all enrolling students in SJSU a website dedicated to facilitating a combination of social media and student portals. Such a system will be designed to allow ease of access to student schedules while simultaneously providing an immediate option to interact with other users that have (or do not have) similar schedules. This will be done based on an account and profile system that gives users plenty of options for what they want to display or hide from other users.

Students will be able to submit their own semester schedules as well as set their desired courses for the following semester. This allows for an easy way for all users to immediately see what is going on in regards to courses with their social circles. Through the website, students will have several ways to communicate information about their classes. For example, students will be able to provide feedback about their current or taken classes. On top of that, students will also be able to directly communicate with each other via a messaging system.

Such a product will be very reliant on its user pool as well as the internet to sustain itself, and so user friendliness as well as a secure database must be maintained in order for the product to thrive.

2.0 Specific Requirements

This section focuses on the functionalities of the product and what the product shall do for the user. Anything that is unclear about functionalities in the introduction is covered.

2.1 External Interface Requirements

External interfaces will be the primary topic of this section, explaining how it works for the user, any hardware, and any software.

2.1.1 User Interfaces

Upon entering the website, the user will always be prompted to a login screen where the user can attempt to sign in, otherwise they will be allowed to register a new account on the same screen. Once the user is logged in to their account, there will be a “welcome” display and several tabs available to view. Clicking upon these tabs will display a different screen menu corresponding to its function. By default, the first screen that shall show is a profile page.

The profile page will contain information of the user (should they choose to fill it in), so that they can let themselves be identified by other students. This can be done by clicking on an edit button which shall allow the user to change the data and choose whichever information to hide.

The second page that will be available is a schedule page, which displays the student’s schedule for the current semester. It will contain a visual representation of what their schedule is like. Just like the profile page, there shall be an edit button, but this will prompt the user into another screen/window that handles searching/adding/deleting classes from the schedule. The schedule will also be interactive in the sense that if the user decides to click on any of the present classes in the visual representation, more information about that class will appear to the user. Such information will contain items such as teacher’s name, how many spots are left, waiting lists, other students in the class roster, and students based reviews of the teacher. The user will be able to provide feedback about the class through that screen as well by using an edit function.

The next page is going to be another schedule page, but specifically of the classes the user intends to take the following semester. It will not take any input from the user, as they have not taken the classes yet. However, the user can still preview information about that class from past students. This page will mostly be used for helping students see who they can take classes with next semester.

The last page will display a list of contacts that the user can interact with and a messaging inbox. This page will also provide a search bar that can bring the user to other user profiles and send them a friend request. The friend requests shall be received in the inbox and given an option to confirm or decline. The messaging inbox will allow students to interact with one another. A second way students may be found is through the class rosters.

2.1.2 Hardware Interfaces

Since the website does not have any designated hardware, it doesn't have any direct hardware interface. The device's connection will be handled by the operating system on the web server, and the database will also be handled the same way.

The website will be able to run on any hardware that supports the following web browsers: Safari, Google Chrome, or Mozilla FireFox. The website does not write information directly to the user's computer, but instead uses a database that is located on a network server. The user's computer transfers and receives data from the server using basic networking protocols. The system information is stored in the server's database which stores the data on the server's disk.

2.1.3 Software Interfaces

The website will communicate with the database in order to get information about the classes, and will communicate by reading and modifying the data found. The program will read off of the data and thus create a visual representation of it that will be easy for the user to read.

The website, built using front-end software languages such as HTML5, CSS3, and JavaScript, will be communicating with a MySQL database through the PHP5.6.3 software language. Bitnami WAMP Stack 7.1.2 software will also be used for development. This software provides PHP5.6.3, MySQL, and Apache, among other tools to support a development environment. The jQuery framework will be used for more intuitive JavaScript development. The front-end of the website (HTML/CSS/JS) will be receiving user input, such as secured log-in information, profile pictures, and class schedules. PHP will then act as the back-end of the website which communicates the front-end information with the MySQL database.

2.1.4 Communications Protocol

It is absolutely necessary that the communication between the separate components of the whole system are promised to not fail, but there is no priority in the process of how they communicate to each other, and so most of it is given to the underlying operating systems.

2.2 Software Product Features

This segment will cover the specific features of the website in a step by step detail. Every feature will be categorized into five main categories: login, profile, schedule, intended schedule, and inbox.

2.2.1 Login

This fragment covers the login feature requirements of the website in full detail.

2.2.1.1 Functional Requirement 1.1 : Registration

Upon visiting the login screen, the user must be able to acquire a prompt to create a new account if they have not registered an account into SCAM yet. It must be done at the same screen as the login page. Otherwise, a new window pops up.

<u>2.2.1.2 Functional Requirement 1.2 : Logging In</u>

Given that the user has already created an account, the user must be able to log into SCAM using the newly generated account. Upon logging in, the user must automatically be prompted into their profile page.

2.2.2 Profile

This fragment covers the profile page feature requirements of the website in full detail.

<u>2.2.2.1 Functional Requirement 2.1 : Edit Profile</u>

Once the user has signed into their profile, they must see a clickable button that allows the profile page to go into edit mode, in which the user is now allowed to fill in the blanks and provide any necessary information about them. They are also able to set a new existing profile picture, should they choose to do so.

<u>2.2.2.2 Functional Requirement 2.2 : Delete Info</u>

When the user enters edit mode on the profile page, they must be given an option to hide any reasonable information from their profile, ensuring the right to privacy. The user shall be able to delete any extraneous information of their choosing simply by clicking on a delete button.

<u>2.2.2.3 Functional Requirement 2.3 : Delete Account</u>

Every student eventually graduates, transfers, or exits college in some way. Students must be able to access an option to delete their account on SCAM, which can be accessed somewhere on the profile page. This must be confirmed through several security tasks for the user to ensure it is intended and actually the input of the right user.

2.2.2.4 Functional Requirement 2.4 : View Profile

The user must be always to view his own profile every time they go back to the profile page. It must contain the same information they had left it with from editor mode.

2.2.3 Schedule

This fragment covers the schedule page feature requirements of the website in full detail.

2.2.3.1 Functional Requirement 3.1 : View Schedule

Upon entering the schedule page, the use must be able to see a visual representation of their schedule. It shall show the necessary information without too much cluttering.

2.2.3.2 Functional Requirement 3.2 : Edit Schedule

Given that the user already has an account, the user can go to the schedule page and enter edit mode to add/drop/modify their class schedules. The schedule will then be saved into their profile and available for everyone to see.

2.2.3.4 Functional Requirement 3.3 : Feedback

The user shall also be allowed to contribute his own feedback about any relevant information on the class such as teacher reviews. This will all be done through a commenting type system that comes with the schedules.

2.2.4 Inbox

This fragment covers the inbox page feature requirements of the website in full detail.

2.2.4.1 Functional Requirement 4.1 : View Inbox

Users must be able to access their inbox page from the menu. The inbox will then show a preview of all messages, much like in the format of an email, and stay updated in live time.

2.2.4.2 Functional Requirement 4.2 : Send/Delete Messages

Every user shall be able to create new messages and send them to other users in order to communicate. At the same time, an option to delete their own, as well as other old messages must be present to prevent clutter and accidental mail. A confirmation must be prompted to the user every time they want to send or delete a message. This is to prevent any unintended actions on top of deletion of mails.

2.2.4.3 Functional Requirement 4.3 : Search Users

There shall be a search bar to allow users to find other users in order to send them a friend request and or send them a message. This will make it easier for the user to find someone.

2.2.4.4 Functional Requirement 4.4 : Add User to Friends

Users shall be able to find other users via search bar and add them as a part of their friends list. A friends list will also be present in the inbox page for the sake of immediate access and less effort finding people. Another way should be that students can view their class rosters from a class in their schedule, and the user can send friend requests to anyone in that roster as well.

2.2.4.5 Functional Requirement 4.5 : View User Profile/Schedule

Users shall be able to utilize their contacts and view their profiles as well as their courses. Users shall also be able to at least view the profile of someone they’re sending a friend request, to be able to confirm it's the right person.

2.2.4.6 Functional Requirement 4.6 : View Notifications

Users shall receive notifications of any type of information and feedback that update in their social network for SCAM. Anytime someone relays anything about their schedule it should appear on their notifications list.

2.3 Software System Attributes

This section discusses software system attributes which are used to evaluate the performance of the overall system. The specific attributes that will be observed are reliability, availability, security, portability, and performance.

2.3.1 Reliability

The website must be reliable when receiving and reflecting the user's inputted data. This means that there should be limited to no failure when a user registers, logs in, inputs class data, views class data, and etc. Since there are many features that rely on relational data (e.g. user's friends, class schedules, etc.), a well-defined MySQL database is essential to ensure reliability. The protection of user data must also be reliable as well. In other words, user inputted data shall be kept unless intentionally deleted by the user.

2.3.2 Availability

The website must suffer from no crashes and be available to the user at any time. This means that the website must continually run and be connected to the database even after many user inputs are received.

2.3.3 Security

The website shall protect user's log-in information through encrypting account data. The communication between the system and the database must also be secured as well. General security of the website shall also prevent non-users from being able to log-in to the website. If many log-in failures are detected from a specific user, the system administrator shall be notified. The MySQL database shall also have an encrypted user and password to protect the user's data. Users' account names shall also be secured where if a username is taken, it shall not be allowed to be registered. All passwords for logging-in would need one uppercase, one number, and contain at least eight characters, further securing the website.

2.3.4 Maintainability

After the website is up and running, it will require to be maintained. For example, fixing bugs we and/or the users find. This will require lots of testing, and we should be able to fix any errors. The code should be written in a way where it's easy to fix errors and implement new functions. This process is never-ending because there will always be something to fix, and if not just fix, something to improve upon. Based on user requirements, we may decide to change some parts of the layout for visuals, and/or some back end.

2.3.5 Portability

The website should be usable on current versions of Mozilla FireFox, Safari, and Google Chrome. The Operating System, whether it be Windows, MacOS, or Linux, shall not affect the portability either.

2.3.6 Performance

The website should have no lag and should not freeze while using its features. Sending and receiving textual data should be instantaneous. Since pictures are larger in size, sending and receiving that data may take longer than the textual data. Multiple user inputs should be allowed to perform on the website.

2.4 Database Requirements

The database required is a MySQL database and to communicate the database's data with the website, PHP software language will be used. The way a MySQL database is structured is it contains defined tables that store defined data of many different types (strings, integers, etc.).

The website will have the following defined MySQL tables to store data: users, personal information, classes, class information, photos, and user photos. The MySQL database and website are still being developed, so these defines tables may change. However, it's important to require a defined table for all users and not a table for each user. Having a MySQL table for all users is more manageable than having a table for each user.

Within each of these defined MySQL tables, there are rows that correspond to variable IDs for user input on the website. So, a requirement for this database is to accurately match these variable IDs being used on the website to ensure successful data storage. Since users are allowed to have profile pictures, the MySQL database must be capable of storing and retrieving user pictures through the defined "photos" and "user photos" MySQL tables.

Furthermore, the MySQL database will take advantage of its ability to store multiple data types. So, each row of each table will have a carefully selected data type to save space. For example, a user's first name and last name will be stored as variable characters (varchar) in the defined "users" table, but the user's gender will be stored as an ENUM. ENUMS are string objects with a value chosen from a list of permitted values. These permitted values will be short, such as M or F for gender, which preserves space in the database.

3.0 Additional Material

The following section contains a traceability matrix that will stay updated through each document and phase of the project, as well as a glossary that will clarify any terminology that may be unclear throughout the SRS.

3.1 Traceability Matrix

Functional Requirements	Design Specifications	Test Cases	Implementation
2.2.1.1 Registration - User must be able to register SCAM account.	1.1 Users register via login page	3.1.1 Registration is tested via BBT - working	1.1 Executed as the default screen
2.2.1.2 Logging In - User must be able to log in using the newly generated account.	1.2 User is capable of logging in by entering fields via LoginPage	3.1.2 Login is tested via BBT and unit tests - working	1.2 Executed as the default screen
2.2.2.1 Edit Profile - Fill in blanks with necessary information.	2.1 ProfilePage provides an option to edit profiles	3.2.1 Profile is tested via BBT and unit tests - working	2.1 Executed at the Profile Page
2.2.2.2 Delete Info - Privacy options	2.2 ProfilePage's options provides a feature to hide all information necessary	3.2.2 Hiding is tested via BBT and unit tests - working	2.2 Executed at the Profile Page
2.2.2.3 Delete Account - Option to delete account	2.3 Users can delete accounts via ProfilePage	3.2.3 Deletion is tested via unit tests - working	2.3 Executed at the Profile Page
2.2.2.4 View Profile - User must be able to view their own profile.	2.4 Users are able to see their profile via ProfilePage	3.2.4 Viewing is tested via BBT - working	2.4 Executed at the Profile Page
2.2.3.1 View Schedule - Visual schedule	3.1 A schedule is displayed through SchedulePage	3.3.1 Viewing is tested via BBT - working	3.1 Executed in the Wall Page
2.2.3.2 Edit Schedule - Add/drop/modify class	3.2 SchedulePage provides options to edit class schedules	3.3.2 Editing is tested via unit tests and BBT - working	3.2 Executed in the Wall Page
2.2.3.3 Feedback -	3.3 Users are able to	3.3.3 Feedback is	3.3 Executed in the

User must be able to contribute information.	rate and review classes	tested via unit tests - working	Wall Page
2.2.4.1 View Inbox - access inbox page from the menu	4.1 Inbox is provided in SocialPage	3.4.1 Inbox is tested via unit tests and BBT - working	4.1 Executed using the menu in the corner of the screen
2.2.4.2 Send/Delete Messages - send their own messages, delete old messages.	4.2 Messages are sent back and forth between inboxes	3.4.2 Msg is tested via unit tests - working	4.2 Executed using the menu in the corner of the screen
2.2.4.3 Search Users - find users via search bar.	4.3 A search bar is provided within the SocialPage	3.4.3 Searching is tested via BBT - working	4.3 Executed using the menu in the corner of the screen
2.2.4.4 Add User to Friends - add user to friends	4.4 Every user has a friends list paired with him/her	3.4.4 Adding is tested via BBT and unit tests - working	4.4 Executed using the menu in the corner of the screen
2.2.4.5 View User Profile/Schedule -view contacts' profile and schedules	4.5 Users can view the profiles of other users they search up in the search bar	3.4.5 Viewing is tested via BBT - working	4.5 Executed using the menu in the corner of the screen
2.2.4.6 View Notifications - view any recent events around the user's network	4.6 Users can view their notifications via the notifications list	3.4.6 Notifications were tested via BBT - working	4.6 Executed using the menu in the corner of the screen

3.2 Glossary

Term	**Definition**
HTML	Hypertext Markup Language, a standardized system for tagging text files to achieve font, color, graphic, and hyperlink effects on World Wide Web pages.
CSS	Cascading Style Sheets, a style sheet language used for describing the presentation of a document written in a markup language.
JS	JavaScript, an object-oriented computer programming language commonly used to create interactive effects within web browsers.

MySQL	My Structured Query Language, an open source relational database management system.
PHP	Hypertext Preprocessor, a script language and interpreter that is freely available and used primarily on Linux Web servers.
Apache	The world's most used web server software.
jQuery	A JavaScript library that allows web developers to add extra functionality to their websites.
ENUM	A data type consisting of a set of named values called elements, members, enumeral, or enumerators of the type.
Varchar	Variable Character Field, a set of character data of indeterminate length.
User / student	A person who uses or operates something, especially a computer or other machine.
Student portal	An online gateway where students can log into a school website to access important program information.
Social media	Websites and applications that enable users to create content, share content, and participate in social networking.
Login	An act of logging in to a computer, database, or system.
Profile	A short article giving a description of a person or organization.
Schedule	A student's plan for carrying out a set of meetings with classes they are enrolled in.
Intended Schedule	A student's planned schedule for the following semester, preceding their current one.
Inbox	An electronic folder in which messages received by an individual are held.

Definitions were found via: Google Search. Google. 30 May 2013. Web. 5 March 2017.