Software Requirements Specification

UniHangouts

Date	**Changes**	**Version**
2/4/2018	**Initial Document**	**1.0**
2/7/2018	**System Requirements Inspection Changes**	**1.1**

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this document is to explain the requirements and features of the UniHangouts application.

1.2. Scope

This project, called UniHangouts, will be an application that allows students at UNL to discover and keep track of events that are happening on campus. Our application will give users the ability to create events, view a live feed of nearby events, and filter events using tags. Additionally, the application will allow the creation of groups that can promote their own events. The application will be accessible first as a web application and later as an Android and potentially an iOS mobile application.

1.3. Definitions, Acronyms, and Abbreviations

Android: Mobile operating system developed by Google
iOS: Mobile operating system used by Apple iPhones
Xamarin: Framework for building cross-platform mobile applications through Microsoft Visual Studio
Microsoft Visual Studio: Development environment designed for the creation of applications for Windows and smartphones
Microsoft SQL Enterprise: Development environment designed for creation and management of Database and Stored Procedures
Stored Procedures: Are pre-compiled database subroutines that act like a function to get data from a database. They also provide data-validation, sanitation, and access-control to the database.
REST Service: (Representational state transfer) is a web service to provide communication over the internet. This service will manipulate the database and transfer information back to clients.
ASP.NET Web Application: Server side web application framework that provides the website.

1.4. Overview

The rest of this document will cover the overall description, the specific requirements, and an appendix for the application. The overall description will describe the overall use and purpose of the application. The specific requirements will discuss the details and design structure of the application and its features. The appendix includes screenshots of what the system will look like.

2. Overall Description

2.1. Product Perspective

2.1.1. System Interfaces

2.1.1.1. The application will interface with a Microsoft SQL database for information storage.

2.1.2. User Interfaces

2.1.2.1. The user interface will be composed of using the screen and a standard mouse and keyboard to enter in info and navigate through the application.

2.1.3. Hardware interfaces

2.1.3.1. This application will have the capability to be installed and run on any device with an internet connection. Subsequent phases will expand the application to the android operating system.

2.1.4. Software interfaces

2.1.4.1. REST Service will allow clients to interface with the server which provides data from the Microsoft SQL database.

2.1.4.2. ASP.NET Web application to provide a user accessible website.

2.1.5. Communication interfaces

2.1.5.1. All communication will take place over the internet via the REST service which will require an internet connection.

2.1.6. Memory

2.1.6.1. The application will rely on web communication and will not store information locally. However, the results of the web communication will be stored in caches.

2.1.7. Operations

2.1.7.1. The main operation of the application will be displaying the feed of events to the user.

2.1.8. Site adaptation requirements

2.1.8.1. We will have to adapt the user interface when making the transition from a web application to a mobile application to accommodate for use of the touchscreen.

2.2. Product Functions

2.2.1. The function of this application is to provide a forum for students at UNL to discover events that are taking place on campus.

2.3. User Characteristics

2.3.1. The users of the application will be college students at UNL. As such, they will for the most part be young adults with a widely varying degree of experience using technology.

2.4. Constraints

2.4.1. The application will require connection to the internet to function properly, as the data for events and user data will be stored in a database.

2.4.2. Processing power will not be an issue because client apps will simply be rendering a user interface and calling web methods.

2.5. Assumptions and Dependencies

2.5.1. The application will depend on Microsoft SQL for maintaining the database, and the operating system that the application is on (either web, Android, and/or iOS depending on the current phase). It is also assumed that the user has a compatible device and up to date hardware and corresponding software.

3. Specific Requirements

3.1. External Interface Requirements

3.1.1. REST Service. Client apps will communicate with a server side service that will provide web methods such as:

- 3.1.1.1. Web methods for Accounts:
  - 3.1.1.1.1. Account_Login
  - 3.1.1.1.2. Account_Create
  - 3.1.1.1.3. Account_GetSettings
  - 3.1.1.1.4. Account_SaveSettings
- 3.1.1.2. Web methods for Events:
  - 3.1.1.2.1. Event_Query
    - 3.1.1.2.1.1. Allow clients to query events by parameters such as date from and date to, tags, or groups.
  - 3.1.1.2.2. Event_RSVP
  - 3.1.1.2.3. Event_Create
  - 3.1.1.2.4. Event_Save
- 3.1.1.3. Web methods for Groups:
  - 3.1.1.3.1. Group_Create
  - 3.1.1.3.2. Group_GetInfo
  - 3.1.1.3.3. Group_InviteAccount
  - 3.1.1.3.4. Group_RejectAccount

3.2. Functions

3.2.1. First Release

- 3.2.1.1. Tabs
  - 3.2.1.1.1. Navigation tabs will be present at the top of the page, the number of tabs corresponding to if a user is logged in or not. Tab for ‘Account’ will not be displayed if the user is not logged in.
- 3.2.1.2. Start/Account Creation
  - 3.2.1.2.1. Upon starting the application, the user will be brought to the home page with guest mode privileges. Guests are not able to select the ‘Event Creation’ button, ‘Account’ tab, or the ‘Activity Log’ tab.
  - 3.2.1.2.2. Users who do not have an account will be able to select the “create an account” button on the ‘Login’ page which will bring them to the “Account Creation” page. Account creation requires a username, password, and some location details.
  - 3.2.1.2.3. Required fields will be stated through a red asterisk.
  - 3.2.1.2.4. During Account creation, if a user enters in invalid input or is missing required input, an error message stating the problem will be displayed and the input boxes will be shaded red.
  - 3.2.1.2.5. An error message will also be displayed if a user attempts choose a username that is already taken
  - 3.2.1.2.6. Once an account is successfully created, the user will be met with a ‘success’ message and a link to bring the user to the login page.
- 3.2.1.3. Login
  - 3.2.1.3.1. Users can navigate to the ‘Login’ page and enter in their credentials if they already have an account. A mockup of our login page is given in the appendix section 4.1
  - 3.2.1.3.2. Error messages corresponding to ‘Invalid Password’ ,‘Account does not exist’, and ‘Invalid Account/Password’ will be displayed if login is not successful
  - 3.2.1.3.3. Once a user successfully logs in, They will be met with a greeting message that will display their name, The logged in user will automatically be taken to the Event feed page.
  - 3.2.1.3.4. When a user is logged in, a tab with their name will be present in the navigation bar, selecting this tab will bring the user to the ‘Account’ page
  - 3.2.1.3.5. User has the ability to exit out of the browser and login again.
- 3.2.1.4. Account
  - 3.2.1.4.1. Session Log/logout
    - 3.2.1.4.1.1. This page will display sessions in which the user has not logged out of. Each session displayed has a Logon Date and ApiKey.

3.2.1.4.1.2. The current session will be displayed with a '(current)' label and a green highlight.

3.2.1.4.1.3. User can log out of a single session by selecting the 'logout' link next to a session

3.2.1.4.1.4. User can log out of every session by clicking the 'logout Everywhere' button at the bottom.

3.2.1.4.1.5. User can restart browser, go to website, Login, and see multiple sessions corresponding to how many times they have logged on without logging out.

3.2.1.4.1.6. Once a user has logged out, they are met with a logout message before they are sent to the HomePage with Guest Privileges.

3.2.1.4.2. Account Info

3.2.1.4.2.1. Users will be able to view info about their account

3.2.1.5. Events

3.2.1.5.1. Create Event

3.2.1.5.1.1. Users must be logged on to create an event. They must provide the following information: title, start and end dates, location, and a short description of the event.

3.2.1.5.2. Delete Event

3.2.1.5.2.1. Users must be logged on and can only delete an event that they themselves have created.

3.2.1.5.3. Event Feed

3.2.1.5.3.1. Events will be shown in a scrolling layout in chronological order with newest events at the top. There will be a '+ Add Event' on the screen that will take the user to an event creation form. This button will not be accessible to user who are not logged in. A mockup of the 'Event Feed' page is displayed in the appendix section 4.2.

3.2.1.5.3.2. The event feed will also be paginated, initially loading a number of events (10 for example). A user can click on a button that is directly below the last loaded event in the feed, and the application will contact the database, adding and add the next 10 events to the feed. This will increase the initial loading speed of the feed.

3.2.1.6. API Display

3.2.1.6.1. Users can navigate to the API Display using the corresponding tab and see a visual representation of the API's functionality.

3.2.1.6.2. User can use a dropdown to select an API method. Selecting a method will populate the Route and Http Type with corresponding info.

3.2.1.6.3. When a method is selected, the inputs that the certain method can take will populate a left column. User can enter in info for each input and see how the post body will be formed in the 'Post Body' box. User can select the execute button, and see the resulting JSON from the input in the 'Resulting JSON' box.

3.2.1.6.4. User can select the 'Clear Inputs' button to clear every input box of info.

3.2.1.7. Security

3.2.1.7.1. User Password is encrypted with SHA256 before they are stored into the database.

3.2.2. Subsequent Releases

3.2.2.1. Private Events

3.2.2.1.1. Events can be created as "invite only". When such an event is created, only users who have received an accepted an invitation to the event can view it.

3.2.2.1.2. These events will not be visible to every user, but only to those that have received an invite to the events.

3.2.2.2. Mobile Version for Android

3.2.2.2.1. A dedicated Android application with the full functionality of the web application will be released on the Google Play Store for users to download.

3.2.2.3. iOS implementation

3.2.2.3.1. A dedicated iOS application with the full functionality of the web application could be released on the Apple application Store for iOS users to download.

3.2.2.4. Date/Time Filter

3.2.2.4.1. The ability for users to filter the event feed by date and time could be added. Users could exclude events that are more than a certain number of days away, and they could exclude events that do not take place during a certain period of free time.

3.2.2.5. Tags

3.2.2.5.1. Tags can be added to an event during creation. This will allow other users to filter results in their event feed to better fit what they are looking for. For example, an event with the 'sports' tag will show up when another user filters their event feed for 'sports.

3.2.2.6. Groups

3.2.2.6.1. Users would have the ability to create groups, join groups, and manage invitations to others about the group..

3.2.2.7. Mass Invites

3.2.2.7.1. Groups will be able to send invites to all members of the group simultaneously instead of having to invite them individually.

3.2.2.8. Event Images

3.2.2.8.1. Event creators could select and upload a suitable image to their event, which would then be displayed near the event title to represent the event.

3.2.2.9. Group chats/event chats

3.2.2.9.1. Chat functionality could possibly be added to allow attendees to communicate with each other prior to or even after the event.

3.2.2.10. Responsive Web App

3.2.2.10.1. The web application could be responsive to the user agent, meaning that it would adapt its layout to desktop or mobile devices.

3.2.2.11. Group Administration

3.2.2.11.1. Group proprietors could add and remove group administrators, allowing the administrators to invite new members.

3.2.2.12. Group search/request to join a group

3.2.2.12.1. User could search for pre-existing groups and send a request to a group to join. Group info would display pending requests and would allow the user to remove any pending requests.

3.2.2.13. Event Feed filter

3.2.2.13.1. Selecting a search button located on the ‘Event Feed page’ would open a search window with a search field and tag, group, or date filter options. The database would be queried based on search keywords or selected filters and would return matching events and groups to be displayed in place of the events feed.

3.2.2.14. Notifications

3.2.2.14.1. Both the web and mobile versions of the application could potentially use push notifications to inform users of upcoming events that match criteria such as a date, time, or tag.

3.2.2.15. Mapping

3.2.2.15.1. The address will be clickable and open the location in the users default map viewer such as Google Maps.

3.2.2.15.1.1. During event creation, a button could be used to open the default mapping application which would allow the event location to be selected via a pin drop, then the user can share the location back to UniHangouts.

3.2.2.16. RSVP

3.2.2.16.1. Allow users to specify if they are “interested”, “going”, “maybe”, or “present”.

3.2.2.16.2. Potentially track user location and if they are at the event, automatically switch the RSVP to “present”.

3.3. Performance Requirements

3.3.1. The system should receive and display information from the database within 2 seconds.

3.3.2. The REST service and Web Application will be running on UNL’s servers and should minimally impact the servers.

3.4. Logical Database Requirements

3.4.1. User Information

3.4.1.1. The database will store information about user accounts including, username, a secured version of their password, an email address, and in later stages will store user devices.

3.4.2. Event Information

3.4.2.1. The database will also store information about events including date, start time, end time, location, privacy setting, events tags, proprietor, and summary.

3.5. Design Constraints

3.5.1. For the near future, one design constraint involves storage for the back end portion of the application. Storage space limits may affect the number of events and users we can host, especially if we allow users to add images.

3.5.2. The cross-platform nature of the application means that the interface will need to be kept simple and adaptable to a variety of screen sizes and aspect ratios.

3.5.3. Because the project’s budget is limited, certain features such as a map view using Google’s paid services are presently out of the question.

3.6. Software System Attributes

3.6.1. Reliability

3.6.1.1. The system must preserve database integrity at all costs.

3.6.1.2. The user applications should be reliable enough to provide full functionality; no functionality-breaking bugs or crashes should be present.

3.6.2. Availability

3.6.2.1. The system should have an uptime above 95% so it is not a noticeable problem for users.

3.6.3. Security

3.6.3.1. The system will implement secure account verification and password storage practices. The application will also utilize API keys so the program has the ability to block or ignore unauthorized requests to the database.

3.6.4. Maintainability

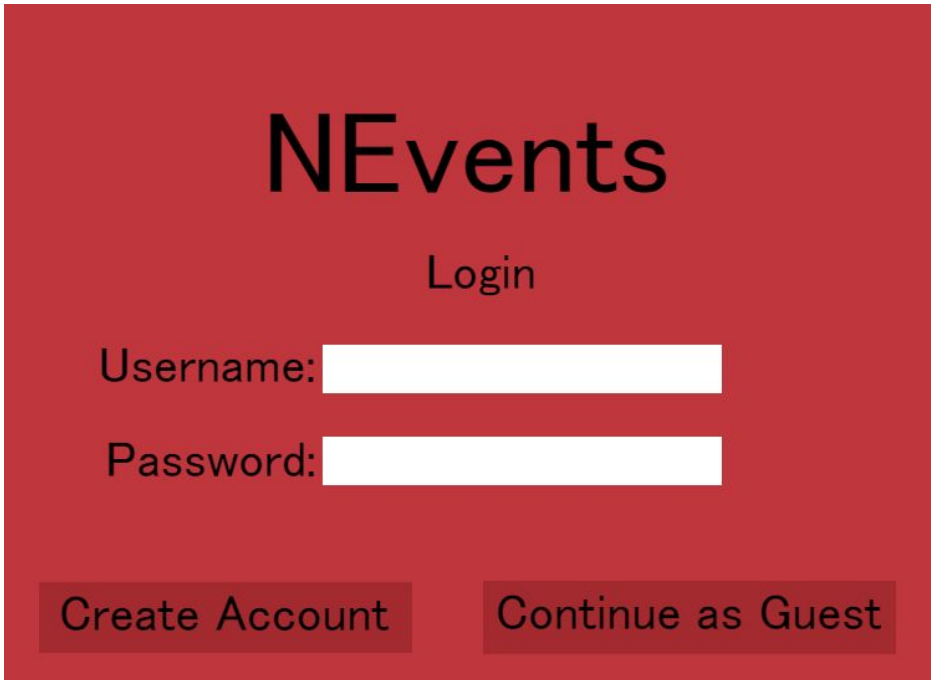
3.6.4.1. The system will be split between a backend database, and a front end web-based user interface. Updates to the front end interface will be easily done, and can be done often when a display issue is discovered, or when a large portion user feedback requests a change. Changes to the backend will be done less often, unless a performance or organization optimization can be made efficiently with minimal down time of the system.

3.6.5. Portability

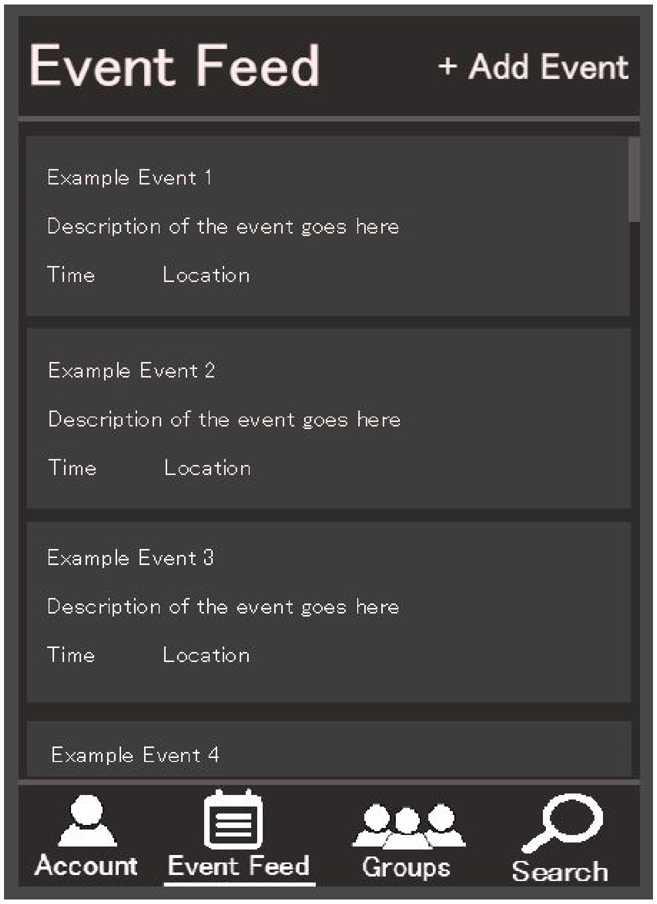
3.6.5.1. The system is designed to be a web based application, so it can be run on any device with an internet connection. In the Subsequent Release phase, the application will be ported to the Android and potentially iOS operating systems; it will be downloadable from the Google Play store and potentially the Apple application Store.

4. Appendix

4.1. Login Screen Mockup



4.2. Event Feed Mockup



Example Event 1

Description of the event goes here

Time Location

Example Event 2

Description of the event goes here

Time Location

Example Event 3

Description of the event goes here

Time Location

Example Event 4