Whoop-Txt SRS Draft Version 1.0

Software Requirements Specification Prepared By:

Anthony Balmeo, Jessica Kain, Zorayr Khalapyan, Wade Norris, Zachary Policzer, Parth Shah

Table of Contents

Table of Contents 1

Revision History 1

1. Introduction 2

1.1 Purpose 2

1.2 Intended Audience and Reading Suggestions 2

1.3 Project Scope 2

1.4 References 2

2. Overall Description 3

2.1 Product Perspective 3

2.2 Product Features 3

2.3 User Classes and Characteristics 3

2.4 Operating Environment 4

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 4

3. System Features 5

3.1 Group Creation and Management 5

3.2 Whoop Sharing 8

3.3 Whoop-Txt Conversations List 9

4. External Interface Requirements 10

4.1 User Interfaces 10

4.2 Hardware Interfaces 10

4.3 Software Interfaces 10

4.4 Communications Interfaces 11

5. Other Nonfunctional Requirements 12

5.1 Performance Requirements 12

5.2 Safety Requirements 12

5.3 Security Requirements 12

5.4 Software Quality Attributes 12

Appendix A: Glossary 13

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## 1.1 Purpose

The purpose of this document is to provide a common basis of understanding for what this project is meant to provide and to accomplish. The document also sets forth the underlying requirements and specifications for the entire project. For more general information about the project, please reference *2.1 Product Perspective*.

## 1.2 Intended Audience and Reading Suggestions

The prepared documented is intended for non-technical audience but may also be beneficial for developers, project managers, testers, quality assurance analysts, and the marketing staff who are trying to learn more about the product and its features. Although references will be made to certain technologies, this document will not cover advanced and low-level technical aspects for the project.

It is assumed that the reader as at least some familiarity with the underlying concept of mash networks, Web 2.0 technologies, and knows the general vision for the application. For those new to Whoop-Txt, it is advised to read the glossary in Appendix A before continuing with this document.

It is highly encouraged that developers read this document in its entirety. Clients and users interested in learning the functional layer of the application should concentrate on the first two chapters.

## 1.3 Document Conventions

Thanks to the freedom provided to the developers in this project, all features listed and requirements listed here are there as per suggestion of the developing team. Any feature or requirement that is requested by the client or anyone else will be specified as such.

Any diagram or figure that depicts or represents a feature or functionality will be labeled with a number that corresponds to a section in this document that elaborates more on that feature or functionality.

This document may be prone to frequent updates. As we are at the early stages, we are testing the waters with certain API's and ideas. There will likely come a time where some API's are added or discarded as, and this document will try to reflect that. Always check the revision date to determine how recent this document version is.

## 1.4 Project Scope

The project covers the entire scope of the application from database design,

## 1.5 Project References

Anthony Balmeo [anthonybalmeo@ucla.edu](mailto:anthonybalmeo@ucla.edu)

Paul Edgart [eggert@cs.ucla.edu](mailto:eggert@cs.ucla.edu)

Jessica Kain jkain@ucla.edu

Zorayr Khalapyan [zkhalapyan@ats.ucla.edu](mailto:zkhalapyan@ats.ucla.edu)

Gergana Markova [gmarkova@us.ibm.com](mailto:gmarkova@us.ibm.com)

Wade Norris [wade.norris@ucla.edu](mailto:wade.norris@ucla.edu)

Zachary Policzer [zacattackftw@gmail.com](mailto:zacattackftw@gmail.com)

Parth Shah [parthms@ucla.edu](mailto:parthms@ucla.edu)

Weiguang Si forswg@cs.ucla.edu

Other technical mentors from IBM include Chris Montalvo, Dean Ocamura, Michael Stein, Neil Sahota, and Steve Hayachi.

# 2.0 Overall Description

## 2.1 Product Perspective

Since the invention of Short Message Service (SMS) in 1984, texting became a massive commercial industry.  The simplicity of texting and its widespread support by almost all mobile devices has led it to be one of the most preferred ways of short communication. According to the latest statistics, on average there are about 200,000 texts sent every second, totaling to 6.1 trillion texts per year. Nevertheless, the underlying concept of texting remained unaffected amidst the massive changes brought by Internet. Social web connected billions of people around the world via user profiles, advanced video and text chat messaging systems, and forums, but yet left the growth of simple 180 character long text messaging system unhindered.

Understanding the vast potential of the social web and the pervasive use of texting, our team proposes to create a new way of sharing information that would combine social web with the simplicity of texting. The software will present a geo-aware mobile and desktop application that would allow creating “text-groups”, sending texts to friends in the nearby vicinity, and also sharing texts using common social websites such as Facebook and Twitter. The project will fuse multiple mobile platforms with Google+, Facebook, Twitter, Google Maps, and various other technologies and social sites to create an engaging and entertaining text-sharing environment.

Additionally the use of location opens a whole new world of options in sending texts and sharing social media. The potential for things such as viral texts or public location based tweets become possible. Non-emergency alerts for the campus with cool activities going on around campus could go out and based on interest from initial batches of recipients go viral and be spread to everyone or quickly be dumped. Imagine a world where you can read tweets from a person at a concert with you without even knowing who they are. By integrating these social platforms the possibilities are truly endless.

As a development goal, we would like to make this application available for all major mobile devices. Tools such as Mobile Web Framework developed by UCLA and PhoneGap, a native container optimized for Blackberry, iOS, and Android and other mobile devices, will greatly help to achieve this objective. Other technologies that will be utilized for the project include web services, HTML5, CSS3, JavaScript, JQuery libraries, JSON and AJAX technologies.

## 2.2 Product Features

Whoop-Txt will present a geo-aware mobile and desktop application that will support the creation of texting groups. The software will allow users to send texts to friends in the nearby vicinity, and also share texts by using common social websites such as Facebook and Twitter.

Whoop-Txt will also enable users to broadcast media anonymously to other users nearby, allowing users to build up a dialogue or commentary pertinent to the immediate area the users find themselves in.

Please see below a list of features that the current version will support. For more detailed explanation of each feature refer to features section of this document.

1. Group Creation and Management
   1. Creating a new group
   2. Inviting friends to join a group
   3. Leaving a group
2. Whoop Sharing
   1. Sharing a whoop with other groups or friends nearby
3. Whoop-Txt Conversations
   1. Display a list of most pertinent conversations

## 2.3 User Classes and Characteristics

Users of Whoop-Txt will use a smartphone or personal computer and communicate with others who either installed the application or with friends who don't own the app via Facebook.

## 2.4 Operating Environment

The client side components include the mobile platforms with webkit-based browser capabilities and Internet connection. The server side is an Apache with at least PHP 5.0 and MySQL support.

## 2.5 Design and Implementation Constraints

Design is limited by several physical restrictions including unreliable Internet connection and inconsistencies with geolocation utilities.

## 2.6 User Documentation

The only documentation provided to the user will consist of user help manuals included with the software – no other paper printed or online resource will be available to the user. Given the context of the project, it is important to design the application using modern mobile-phone ergonomic theories in order to eliminate heavy reliance on help documentation and manuals.

## Assumptions and Dependencies

Project depends on the following open APIs and utilities:

1. The Facebook Graph API
2. The Twitter API
3. The Google+ API
4. The Google Maps API

# 3.0 System Features

To provide better organization, the system features were separated into three categories: Group Creation and Management, Whoop Sharing, and Whoop-Txt Conversations Listing. Please see below more information about these features.

## 3.1 Group Creation and Management

**3.1.1 Description and Priority**

The creation and management of groups is integral to the functionality of Whoop-Txt. Groups allow for users to specify who will they can send information they are sharing to and additionally who they will receive information from. This makes group creation and management a top priority in the development of Whoop-Txt. The source of this requirement is from brainstorming and planning sessions of the entire development team.

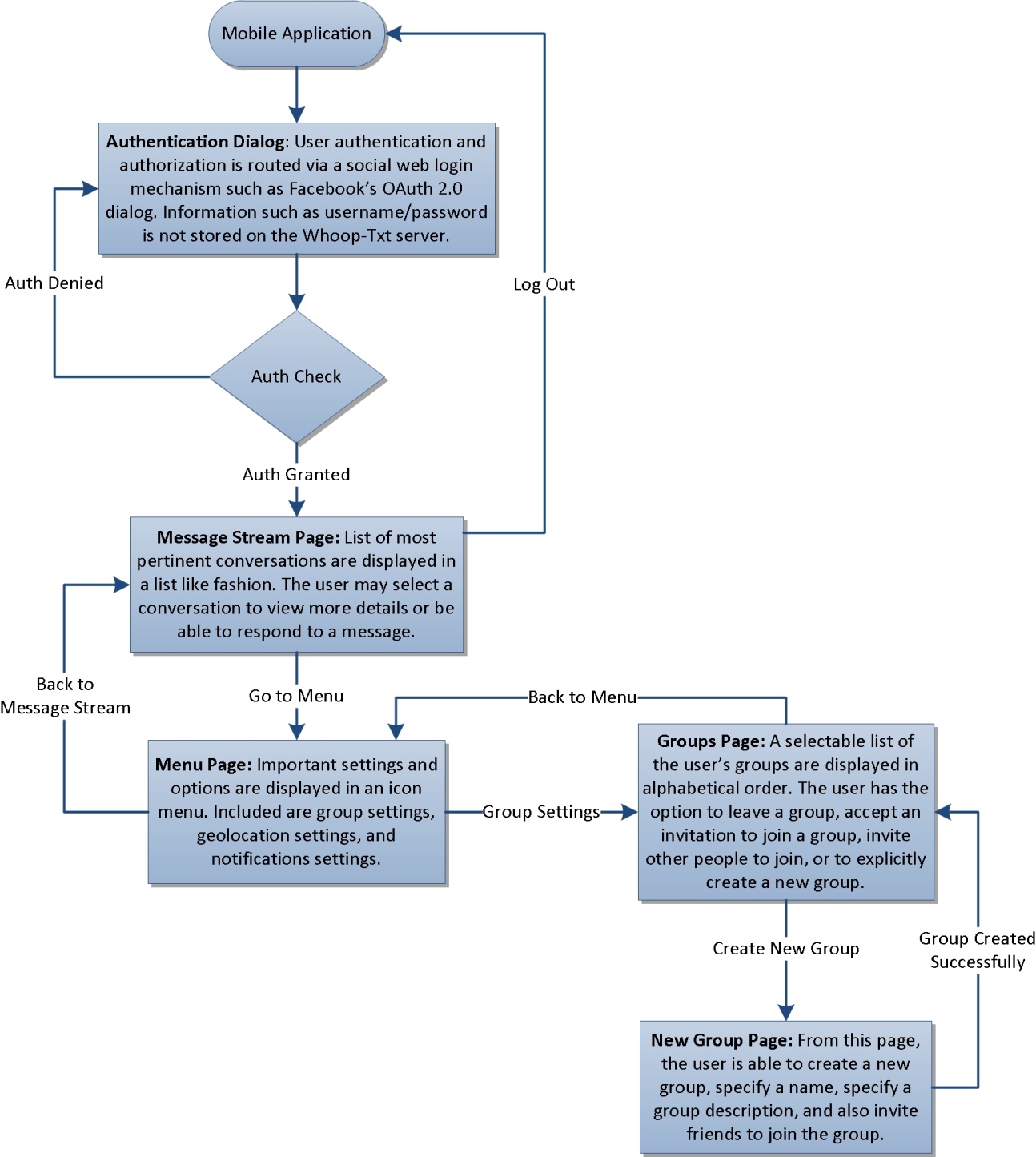
**3.1.2 Functional Requirements**

See below a list of functional requirements from the user’s perspective:

1. Create new groups for sharing information on Whoop-Txt.
2. Invite friends to become members of a group.
3. Accept or decline invitations to join a group.
4. Leave from a subscribed group.

Note that users will not be able to join groups that they were not invited to.

**3.1.3 General Workflow**



## 3.2 Whoop Sharing

**3.2.1 Description and Priority**

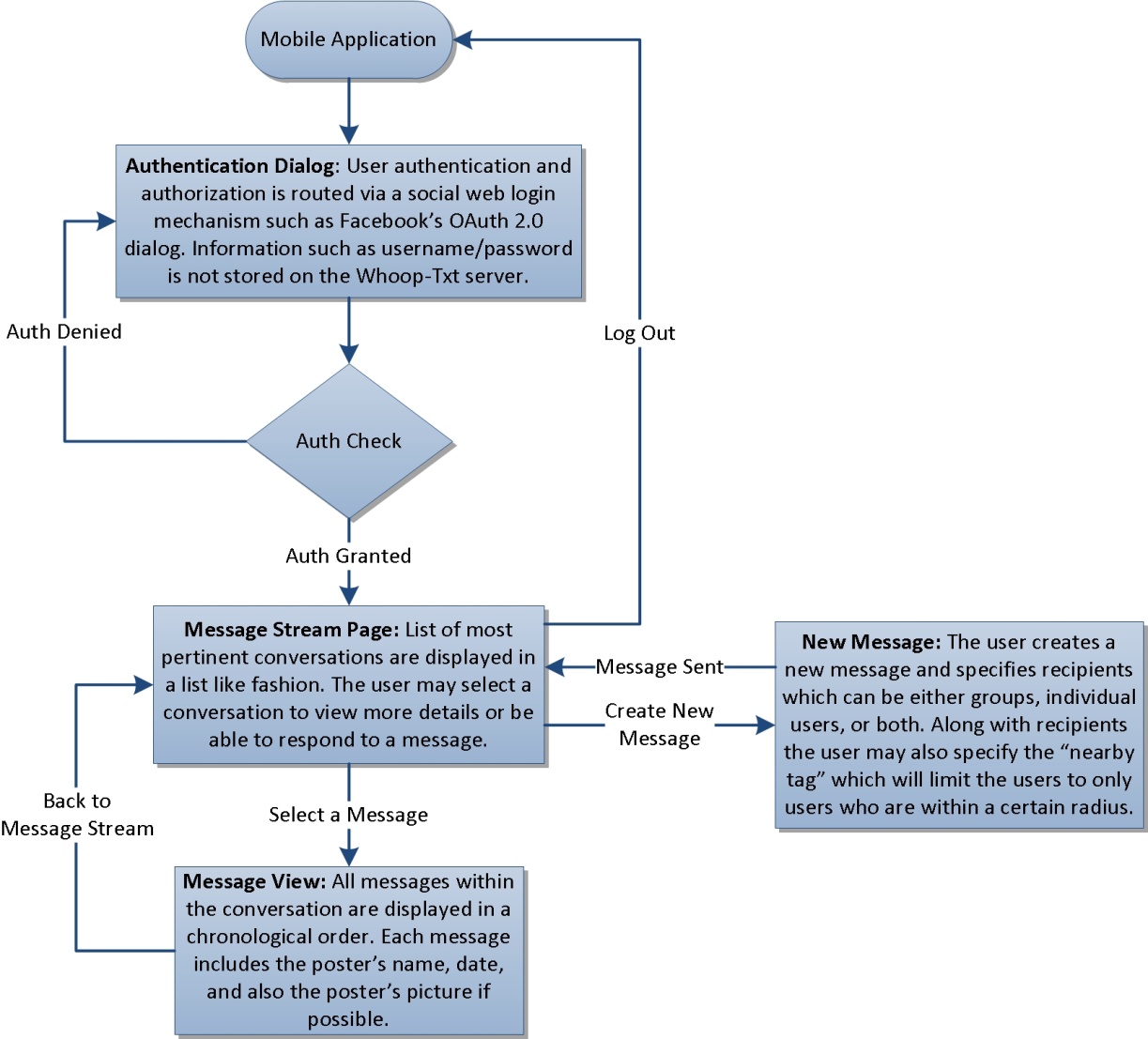
Whoop sharing is the most important part of Whoop-Txt as it is the content that users are utilizing the application in order to access. As such the implementation of Whoop sharing and related functionality shall be of the utmost importance. The source of this requirement is additionally from brainstorming and planning sessions of the entire development team.

**3.2.2 Functional Requirements**

See below a list of functional requirements from the user’s perspective:

1. Create a post and share it with explicitly selected “tags” that will modify who will be able to see or be notified of this post.
2. Select as tags groups that he or she is currently a member of in order to share a post with members of these groups.
3. The user shall be able to select “Public” as a tag to share the post with all members of Whoop-Txt.
4. The user shall be able to select “Nearby” as a tag to restrict the share only to those within a certain physical distance calculated using users’ geo-locations. This tag will override other tags such that if a users selects “Public” and “Nearby” it will share it with any Whoop-Txt members within the specified distance. If the user selects multiple groups they are members of and “Nearby” it will be shared with all of the members of the other groups that are also nearby.

**3.2.3 General Workflow**



## 3.3 Whoop-Txt Conversations Listing

**3.3.1 Description and Priority**

The users shall be able to view posts that have been shared with them by other Whoop-Txt users. This is extremely important as well because it is the main function goal of this application, delivering information to other users. Due to this it is also of the highest priority. This requirement is also from brainstorm and planning sessions of the entire development team.

**3.3.2 Functional Requirements**

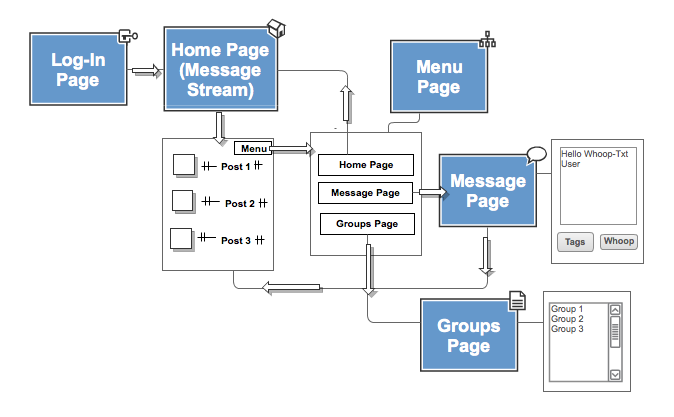
See below a list of functional requirements from the user’s perspective:

1. View a list of available posts
2. Select an individual post and view the post’s contents, group members, and the responses that have been shared by recipients of the post.
3. Share a reply to a post that will be visible by all of the recipients of the original post.
4. Re-share a post they have received with the same tagging options and interface given to a new share.

# 4.0 External Interface Requirements

## 4.1 User Interfaces

Below a general diagram describing several user interfaces. More specific descriptions are detailed in later sub-sections.



**4.1.1 Authentication Page**

Users will be able to log in to Whoop-Txt with a Facebook account.

**4.1.2 Message Stream Page**

After logging in, the user is directed to the message stream page. Viewable here is a list of their most recent message posts from other Whoop-Txt users along with their Facebook profile pictures. Included in the top screen of this page is a Menu button.

**4.1.3 Menu Page**

The Menu page lists different navigation links that redirect you to certain pages on the site.

* Home - Takes you to the Message Stream Page.
* New Message - Directs you to the Write Message Page.
* Groups - Directs you to the Groups Page.

**4.1.4 New Message Page**

The Write Message page allows you to send posts to other Whoop-Txt Users. The user will have the choice of explicitly selecting “tags” that will modify who will be able to see or be notified of this post or they can text people a certain distance away from them.

**4.1.5 Groups Page**

The Group Page lists all the groups the user is a part of, whether it is one they have created or have been added to. Users will be able to create custom groups and view members within the groups.

## 4.2 Hardware Interfaces

This software will not present any hardware interfaces.

## Software Interfaces

The constructed software will interface with MySQL, PHP, Apache, and also the management Facebook application management software.

## Communications Interfaces

The management console based on HTTP protocol running on an Apache server will be expose its communication interface via port 80 over TCP/IP connection.

# 5.0 Other Nonfunctional Requirements

## 5.1 Performance Requirements

There are no set performance requirements. The constructed software should be optimized for low quality and low reliability hardware and Internet connections – this includes connection and communication error handling and no single point of failure architecture.

## 5.2 Safety Requirements

--None--

## 5.3 Security Requirements

The application should ensure secure communication between the users and the server via SSL encryption and proper authentication mechanism.

## 5.4 Software Quality Attributes

To evaluate software quality, below quality matrices have been produced:

1. Software crash/failure resistance
2. Software correctness and error handling

Appendix I

|  |  |
| --- | --- |
| **AJAX** | Asynchronous JavaScript and XML:  With Ajax, web applications can send data to, and retrieve data from, a server asynchronously (in the background) without interfering with the display and behavior of the existing page. Ajax is not one technology, but a group of technologies. Ajax uses a combination of HTML and CSS to mark up and style information. The DOM is accessed with JavaScript to dynamically display, and to allow the user to interact with the information presented. JavaScript and the XMLHttpRequest object provide a method for exchanging data asynchronously between browser and server to avoid full page reloads. |
| **CSS** | Cascading Style Sheets:  A style sheet language used to describe the presentation semantics (the look and formatting) of a document written in a markup language. Its most common application is to style web pages written in HTML and XHTML, but the language can also be applied to any kind of XML document, including plain XML, SVG and XUL. CSS is designed primarily to enable the separation of document content (written in HTML or a similar markup language) from document presentation, including elements such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for tableless web design). |
| **Facebook User** |  |
| **HTML5** | Hypertext Markup Language:  HyperText Markup Language (HTML) is the predominant markup language for web pages. HTML elements are the basic building-blocks of webpages. The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts in languages such as JavaScript which affect the behavior of HTML webpages. |
| **JSON** | JavaScript Object Notation:  A lightweight text-based open standard designed for human-readable data interchange. It is derived from the JavaScript scripting language for representing simple data structures and associative arrays, called objects. Despite its relationship to JavaScript, it is language-independent, with parsers available for most languages. The JSON format is often used for serializing and transmitting structured data over a network connection. It is used primarily to transmit data between a server and web application, serving as an alternative to XML. |
| **MWF** | Mobile Web Framework:  The UCLA Mobile Web Framework facilitates the development of a robust, feature-rich, cross-platform mobile web presence. Focusing on mobile web standards, semantic markup and device agnosticism, the framework enables developers to create mobile-styled web applications with relative ease of use. It allows developers (including Universities) to avoid detailed device-by-device planning, implementation, upgrades and maintenance and all their associated costs, and instead makes real the promise of "develop once, use everywhere" by providing a simple abstraction layer whereby the framework itself makes device-by-device determinations freeing the developer to develop. |
| **PhoneGap** | PhoneGap is an open-source mobile development framework developed by Nitobi Software. It enables software programmers to build applications for mobile devices using JavaScript, HTML5 and CSS3, instead of often less-known languages such as Objective-C. PhoneGap currently supports development for the operating systems Apple iOS, Google Android, HP webOS, Microsoft Windows Mobile, Nokia Symbian OS and RIM BlackBerry. |
| **Whoop-Txt User** | A person who has installed the Whoop-Txt application. The person can send messages, receive messages, and can create, join, or leave a group. |
| **Group** | A collection of users within a list that share messages. Users can be added to groups and each group can have a personalized name. |
| **Nearby Group** | A feature in the application which used geolocation to find people in a certain radius from a user who is trying to send a message. All the people within this radius are part of a Nearby Group. |
| **App Invite** | A request sent to a person who does not have the Whoop-Txt application installed. This request is communicated via Facebook and posted on the requestee’s Facebook wall. |
| **Whoop-Txt (Post)** | This is the message that is sent to the user. This message can be viewed in the message feed. |