Software Requirements Specification For [GotCha – Online Team Collaboration]

[28/02/2017]
[1.0]

Prepared by:
Ahmad Hakroosh, Mohammad Dahamshi

Table of Contents

Revision History

Version	Date	Name	Description
1.0	28/02/2017	GotCha	An initial version of the project “GotCha – Online Team Collaboration”.

1 Introduction

1.1 Overview

This document defines the requirement and the scope for the web-based team collaboration tool, *GotCha*, that is being developed for *Web Development* course, instructed by *Dr. Haggai Roitman*, in the faculty for *Computer Science, Haifa University*. The purpose of this document is to represent the system requirements in a readable way so that clients and stakeholders can understand them and verify them for correctness but with enough detail that developers can design and implement a software system from them.

This document doesn't address *GotCha* issues such as schedule, cost, development methods, development phases, deliverables and testing procedures. Those are addressed in a separate project document and quality assurance test plan.

GotCha is a web-based tool for Software Team Collaboration. It provides a way for team communication using multi-user channels, or direct messages (for private messaging).

1.2 Goals and Objectives

The three main goals of *GotCha* are:

1. Provide a simple mechanism for software teams to collaborate about project they work on together.
2. Share project issues and files among complete teams to turn the development phase easier and faster.
3. Enable team members to communicate even privately.

1.3 Scope

GotCha, once a user is connected to it, will deliver his message(s) when sent by him, to the specified destination, which could be another user, or a group of users (channel). The user who is receives a message(s) will be notified by a counting indicator.

1.4 Definitions

Use case – describes a goal-oriented interaction between the system and an actor. A use case may define several variants called scenarios that result in different paths through the use case and usually different outcomes.

Scenario – one path through a use case

Actor – user or other software system that receives value from a use case.

Role – category of users that share similar characteristics.

Product – what is being described here; the software system specified in this document.

Project – activities that will lead to the production of the product described here. Project issues are described in a separate project plan.

Shall – adverb used to indicate importance; indicates the requirement is mandatory. "Must" and "will" are synonyms for "shall".

Should – adverb used to indicate importance; indicates the requirement is desired but not mandatory.

May – adverb used to indicate an option. For example, “The system may be taken offline for up to one hour every evening for maintenance.” Not used to express a requirement, but rather to specifically allow an option.

Controls – the individual elements of a user interface such as buttons, inputs and check boxes.

1.5 Document Conventions

Portions of this document that are incomplete will be marked with TBD. Each TBD item will have an owner and estimated date for resolving the issue.

1.6 Assumptions

1.6.1 User

1.6.1.1 Uniqueness

Usernames and Nicknames for each user are unique (among users and channels), means, it is not allowed for newly registering users to use already taken names.

1.6.1.2 White-Spaces

Usernames and Nicknames for each user must not contain any spaces and (or) special characters, e.g. none of the following: “@#$!%^&*()+=./’\”|`~:<>”.

1.6.2 Channel

1.6.2.1 Uniqueness

Channel names for each channel are unique (among users and channels), means, it is not allowed for newly created channels to use already taken names.

1.6.2.2 White-Spaces

Channel names must not contain any spaces and (or) special characters, e.g. none of the following: “@#$!%^&*()+=./’\”|`~:<>”.

1.6.3 Search

1.6.3.1 Channel discovery

Upon searching part of a channel name, but a complete word, the all the channels containing this word will appear in the search results (up to 100).
e.g. If “Testing-Channel” exists in the system, once entering “Testing”, “testing”, “Channel”, “channel”, “Testing Channel” or “Testing-Channel” etc. “Testing-Channel” will appear in the search results.
Upon searching part of a user’s nickname, but a complete word, channels that the searched user is subscribed to, will appear in the search results (up to 100).
e.g. if “Ahmad-Hakroosh” exists in the system (as a user), and he is subscribed to “Testing-Channel”, once entering “ahmad”, “Ahmad” etc. “Testing-Channel” will appear in the search results.

1.6.3.2 User discovery

Upon searching part of a user’s nickname, but a complete word, the user will appear in the search results. See 1.6.3.1

1.6.3.3 Channel subscription

Upon showing search results found, each channel could be subscribed to via clicking the slider switch that is relevant to it (to the right of result row).

States: ● subscribed

○ unsubscribed

2 General Design Constraints

2.1 Product Environment

GotCha will be a standalone system that runs on Tomcat web server. It has its own authentication system, relational database (Apache Derby v.10.12), search engine (Apache Lucene v.5.5.3).

2.2 User Characteristics

It’s important for developers to have a complete and accurate image of the end users. Even when the requirements of the user interface are described in detail the developer will still make many tiny design decisions during design and implementation. Knowing the general characteristics of the end users will help the developer make better decisions.

If the specific users of the system are know list them here. More likely there will be user roles or categories of uses. For each group of users list their responsibilities, characterize their knowledge of the domain and describe their characteristics including technical sophistication, background and education.

Prioritize users if necessary. This is especially important when user characteristics conflict. For example, if the system must accommodate experience users as well as novices it is important to know which should be given priority in case it's not possible to accommodate both groups.

2.3 Mandated Constraints

Ideally requirements will be specified in terms of functionality needed and developers will have free rein to design and implement a solution. In practice there are constraints on the eventual design and implementation.

Constraints may be mandated technologies. For example, the client may specify that a specific database management system, programming language, and/or operating system be used.

Constraints limit design and implementation options.

Constraints might be absolute, desirable or optional. If constraints aren't absolute the motivation for the constraint should also be given.

2.4 Potential System Evolution

2.4.1 Statistics section

Using D3 library for data driven documents to show statistics for user's leader-board, most active channels etc.

2.4.2 Attachments

Add the option for attaching files.

2.4.3 Code snippets

Beautify <code>…</code> blocks.

2.4.4 Channel deletion

Enable channel deletion

2.4.5 User deletion

Enable user deregister

3 Nonfunctional Requirements

None.

3.1 Usability Requirements

The system must be user friendly, and match every user that may want to use it. It should fit any screen size.

3.2 Operational Requirements

Not relevant.

3.3 Performance Requirements

The system should operate as single page application (SPA), means, no page loads, no refreshes, and no delays.
All server requests are performed asynchronously (AJAX).
Communication between client and server is performed in a Restful way.

3.4 Security Requirements

Not relevant.

3.5 Safety Requirements

Not relevant.

3.6 Legal Requirements

Not relevant.

3.7 Other Quality Attributes

Not relevant.

3.8 Documentation and Training

Not relevant.

3.9 External Interface

Not relevant.

3.9.1 User Interface

The user interface should be smooth, clear and responsive to different screen sizes.

3.9.2 Software Interface

Not relevant.

4 System Features

Not relevant.

4.1.1 Additional Requirements

Not relevant.