Homework 3: Knight Writers

sQuire, a Collaborative Effort

Guan, Feng (guan2264) Wade, Grant (gwade)
Jutson, Jesse (juts3869) Dennis, Scott (denn2725)
Benzing, Kevin (benz5834) Slippy, Timothy (slip5295)
Ferguson, Joseph (ferg2065) Ocker, Chris (ocke8865)

February 9, 2016

Contents

1 Introduction

This document represents the combined efforts of Team “Knight Writers”. It contains a list of functional and non-functional requirements we feel are necessary for sQuire.

2 Functional Requirements

2.1 Server Functionality:

The software shall:

1. Host project files and user accounts on a central server.
2. Lock lines that users are editing to prevent two users from editing one line.
3. Manage multi-file java projects.
4. Allow the Import and Export of source code.
5. Enable multiple users to collaborate; 32 or more clients per server.
6. Allow simultaneous editing by multiple users on any file.
7. Keep a change log of user’s edits.

2.2 Client Functionality:

The software shall:

1. Serve as an editor, compiler, and executor of java code.
2. Incorporate visual representation indicating whether code has been synced to server.
3. Compile and run of code on individual users’ systems.
4. Allow users to have multiple projects open at once.
5. Support syntax coloring.
6. Provide visual indication to see who coded what.
7. Navigate seamlessly through project files.
8. Provide a user chat environment and user activity information.

2.3 Chat Functionality:

Users shall be able to:

1. Send private messages to other users, as well as invite and accept invitations to chat.
2. Create custom channels for chat (akin to Blizzard style chat).
3. Chat with other users within project channels and individual file channels.

2.4 User Account Functionality:

The software shall:

1. Display friends' and collaborators' profiles.
2. Create standard username/password accounts on the central server, verified by email.
3. Provide user-friendly account management.
4. Allow users to add friends and see which friends are online/what projects they are working on.
5. Contain a list of publicly listed projects that users can view and request access to.

3 Non-Functional Requirements

The software shall:

1. Be tested for efficiency.
2. Be tested for usability.
3. Keep a change log.
4. Be scalable.
5. Be accessible to the general public.