Squire: A Collaborative Software Development Tool

jank6275, mora5651, boss2849, bolt1003, gall7417, brec9824, snev7821, mars2681

February 19, 2016

Contents

Chapter 1

Introduction

1.1 Program Premise

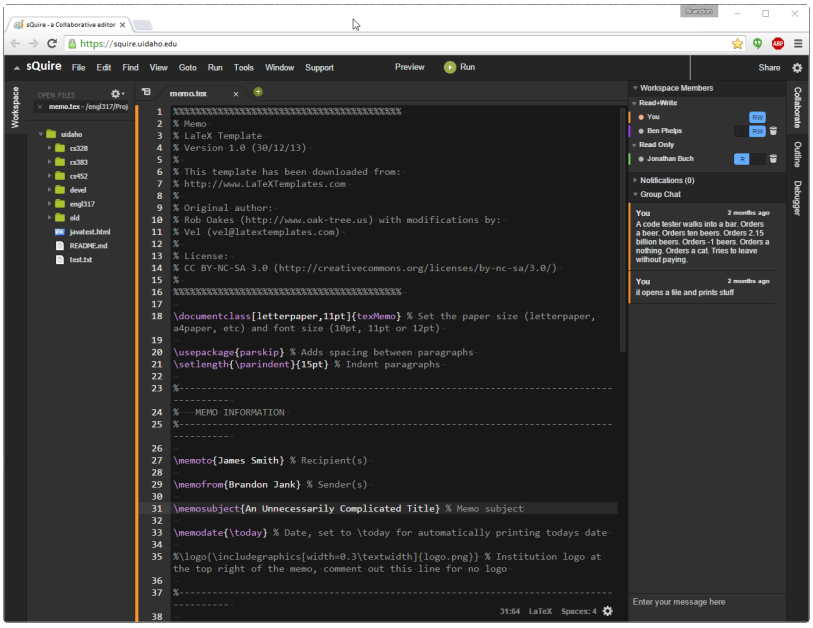


Figure 1.1: Squire is a web-based collaborative software development environment with a project development center. Squire will allow multiple users to edit files and communicate in real time. Projects can be "stubbed" out by a user and then other users can join and/or vote to support for their favorite projects. After a certain amount of support, planning, and documentation is reached for a project, the project becomes a fully fleged project and then community development can start. Think "kickstarter for code" where people pledge their help with the project and not just financial support.

Chapter 2

Class Diagrams

The "detail" subteams should show as much detail about their assigned classes' associations, attributes, and methods as is known at this time. Edits of all kinds can be made later on. Every UML diagram should have 1+ author name(s) in a small font, and be accompanied by a supporting "diagram dictionary": a section of prose text that defines the classes and user-defined associations, and clarifies any ambiguities or tricky sections of the diagram. All UML diagrams should be reviewed for completeness and accuracy by at least one other team member; add "reviewed by XXX" in a small font after the author name(s) in the diagram. Be "egoless" and take criticism seriously, fixing or adding things can help your grade. Within your team, check to make the diagrams mutually consistent and stylistically consonant. Adopt standards for fonts and such within your team.

2.1 Overview ()

Each team should assign one additional "overview" subteam to draw one or more higher-level overview class diagrams showing the aggregated big picture, with more classes and associations showing but class class detail. This could be one giant diagram, or separate diagrams for inheritance, aggregation, and user-defined associations, or provide an overview along some other structured lines.

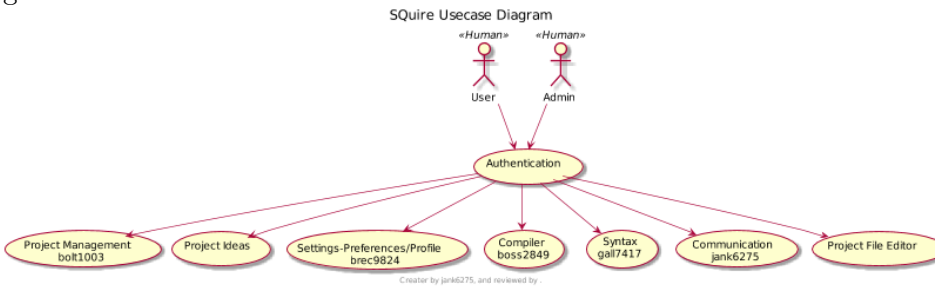


Figure 2.1: a section of prose text that defines the classes and user-defined associations, and clarifies any ambiguities or tricky sections of the diagram.

2.2 Authentication (mora5651)

2.3 Project Management (bolt1003)

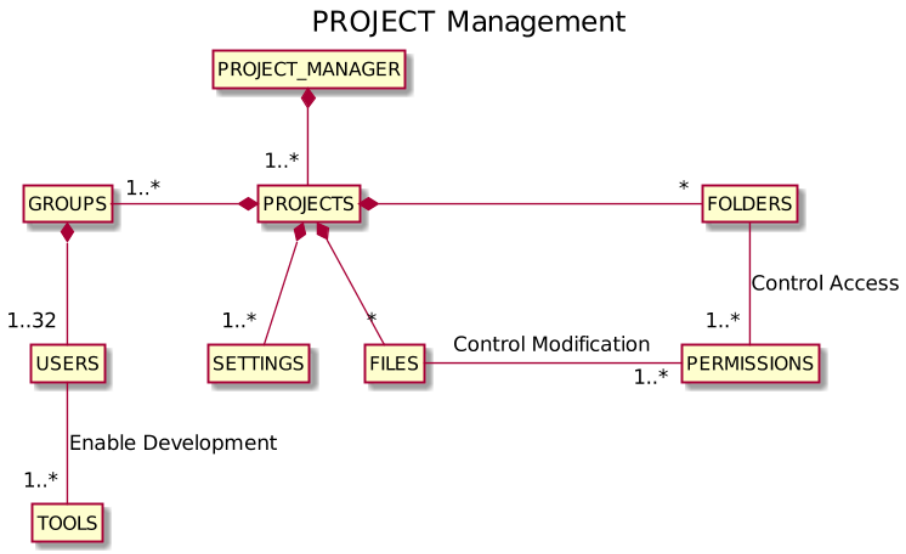


Figure 2.2: Project management allows for the group to delicate permissions and create projects

2.4 Project Ideas (mars2681)

The Project Ideas page allows the user to browse other user's projects ideas, as well as create and customize their own project ideas.

2.5 Settings - Preferences & Profile (brec9824)

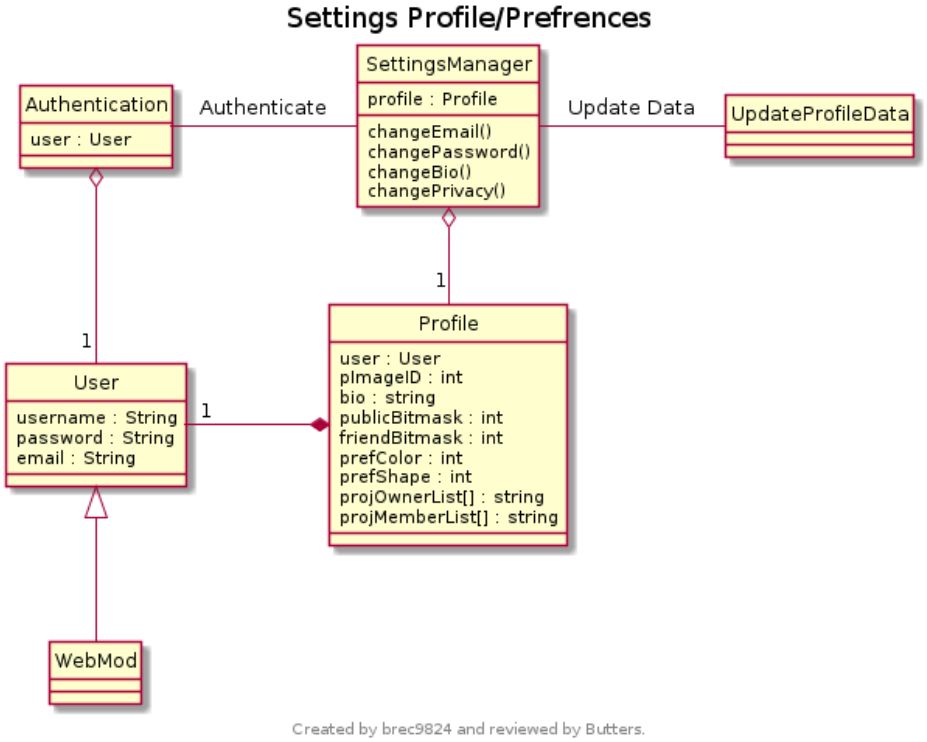
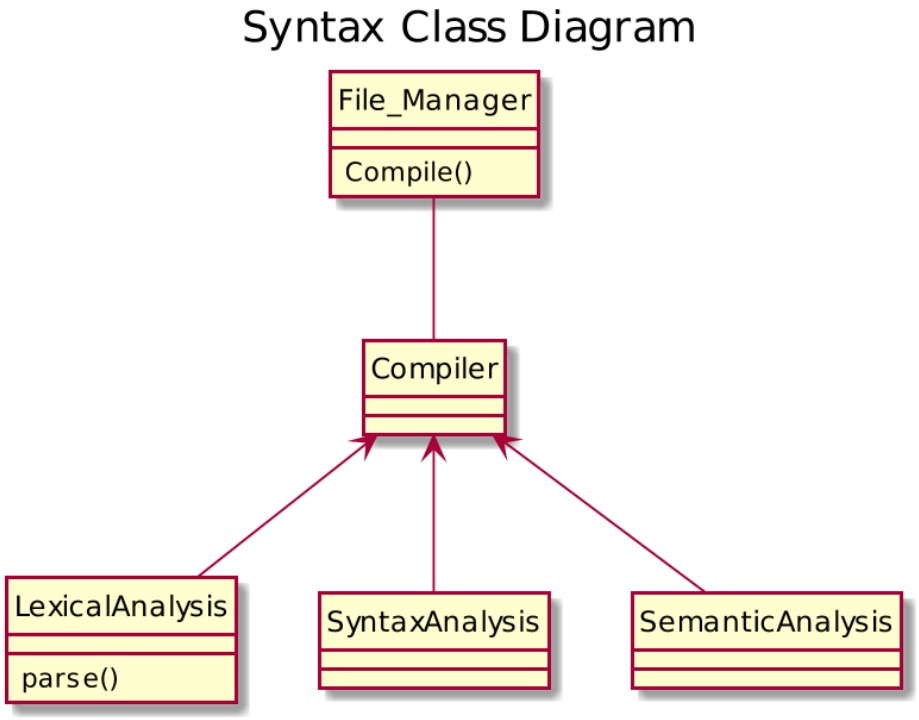


Figure 2.3: a section of prose text that defines the classes and user-defined associations, and clarifies any ambiguities or tricky sections of the diagram.

2.6 Compiler (boss2849)

2.7 Syntax (gall7417)



2.8 Communication (jank6275)

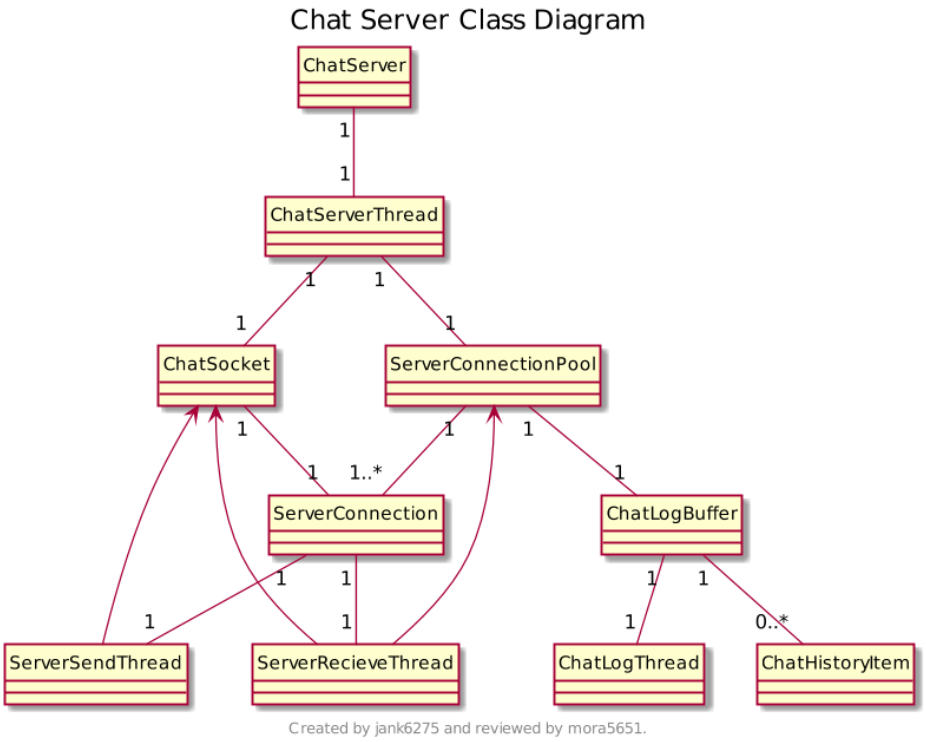


Figure 2.4: a section of prose text that defines the classes and user-defined associations, and clarifies any ambiguities or tricky sections of the diagram.

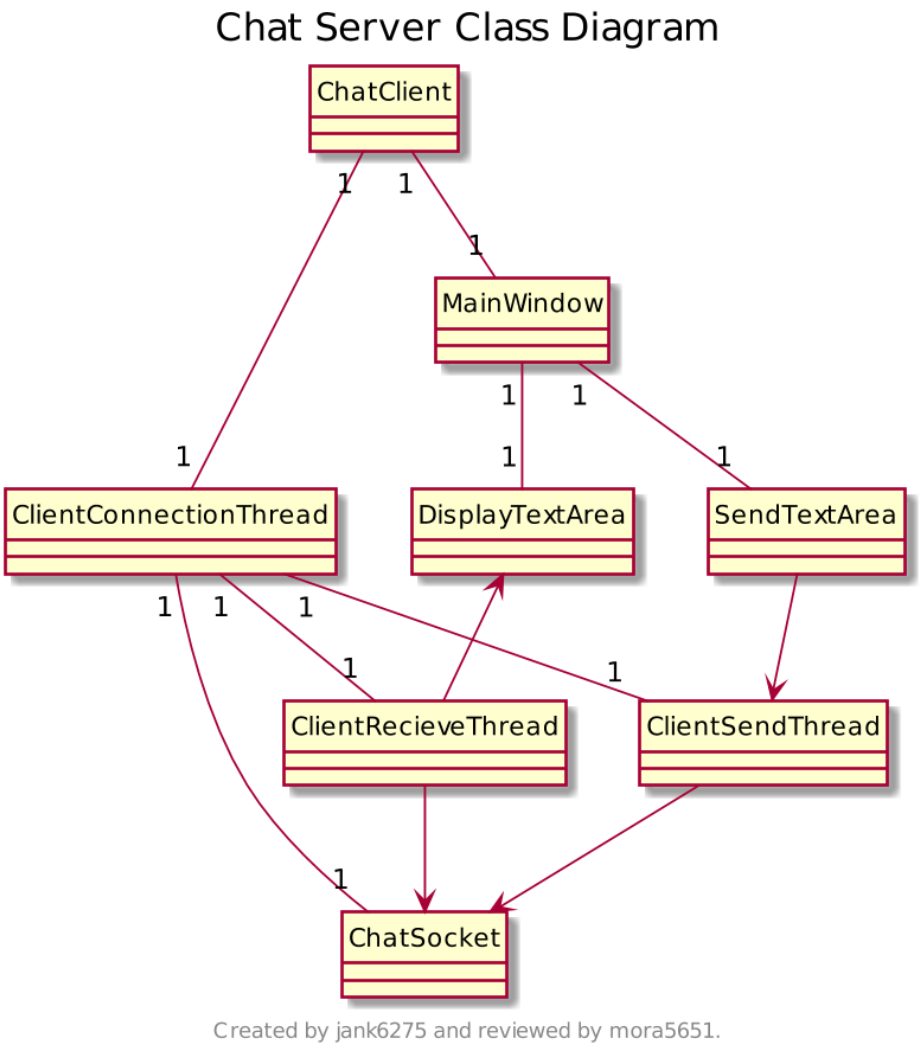


Figure 2.5: a section of prose text that defines the classes and user-defined associations, and clarifies any ambiguities or tricky sections of the diagram.

2.9 Project File Editor (snev7821)

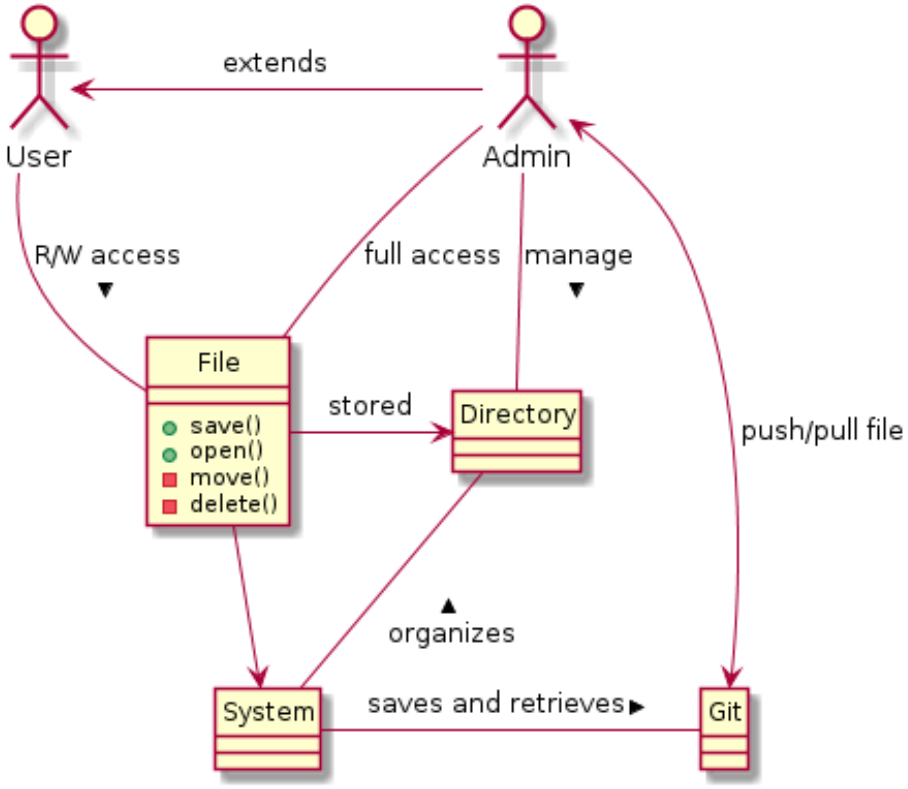


Figure 2.6: The project File Editor is simply the file system for Squire. It provides basic read/write permission for any user related to a project. Only admins of a project may move project files, delete old files, and create new files.