Software Design Document

for

Community Project Tracking

CS 472

Draft 0.2

Prepared by:
Tim Grediagin
Devin Jones
Brent Mello
Blake Eggemeyer

May 4, 2012

Contents

1 Introduction

1.1 Overview

This document is intended to be used by the developer of the TODO software.

1.2 Stakeholders

The stakeholder in the design is also the client.

1.3 Definitions

1.3.0.1 Should: Requirements with this marker are desired, but not crucial, and will be a part of the final deliverable contingent on time and progress.

1.3.0.2 User: The person, or persons, who operate or interact directly with the product.

1.3.0.3 Will: Requirements with this marker are guaranteed to be in the final delivered product.

1.3.0.4 Item: Item refers to a data object. This object is either an appointment or a task.

1.3.0.5 CSV: CSV is an acronym for Comma Separated Value. This is a standard and common format of simple tabular data.

1.3.0.6 GUI: Acronym for Graphical User Interface. Used to refer to the look and feel the user experiences.

1.3.0.7 Immediately: Immediately refers to actions that will begin as soon as the user has given the input for the action to occur.

1.3.0.8 Client: Julie Engfer, the Office Manager for Festival of Fairbanks.

1.3.0.9 Administrator: A user with special permissions as specified in section 3.1.2 User Management.

1.3.0.10 CPT: The name of this application.

1.3.0.11 Worker: The person(s) responsible for the hours worked in a Project. Users are also Workers, but there may be Workers that are not Users, e.g., "boyscouts."

1.3.0.12 Project: Seen at top of timesheet, e.g., "Bicycle Path."

1.3.0.13 Program: A project which occurs annually, e.g., "Clean Team."

1.3.0.14 Activity: The specific type of work a Worker does, e.g., "Ice Chipping."

1.3.0.15 Location: The place where an activity is done, e.g., "CORE 1st - 3rd."

1.3.0.16 Tool/Equipment: The implement used to complete an Activity. Corresponds to "Equipment Used" on the original timesheet.

1.3.0.17 Comment: A remark a User may optionally provide on the timesheet.

1.3.0.18 Timesheet: The name for the web page on which the various data are entered.

1.4 References

The `1998 - IEEE Standard for Information Technology - Systems Design - Software Design Descriptions` was referenced to produce this document.

1.5 Revision tracking

0.1	Feb 16	Empty document created.
0.2	April 26	Submitted for Review.

2 Design Considerations

2.1 Programming Languages

The project will be completed using Groovy on Grails to produce a java web application.

2.2 Database

The H2 database manager is the default database manager used by Grails and according to information from third parties is sufficient to handle the task necessary.

2.3 Project Management

This project will use `Git` version control in conjunction with `GitHub` to keep track of changes. The repository can be reached at `https://github.com/blake6489/Community-Project-Tracking`.

3 Domains

3.1 User

There will be two types of user accounts: admin and employee. Both types of accounts will have unique usernames and passwords. Employee accounts will have the following permissions: fill in and submit the current time sheet, and view past timesheets submitted by that specific account. Admin accounts will have the following permissions: create/delete admin or employee accounts, view usernames and passwords of all accounts, view and edit all submitted timesheets, create and submit timesheets on behalf of volunteer group, and access to automated report generation.

3.2 Activity

3.2.0.1 constraints

3.2.0.2 name 250 characters, alphanumeric and spaces

3.2.0.3 uniqueName 250 characters, alphanumeric and spaces set to lower case

3.2.0.4 setName

3.3 Location

3.3.0.1 constraints

3.3.0.2 name 250 characters, alphanumeric and spaces

3.3.0.3 uniqueName 250 characters, alphanumeric and spaces set to lower case

3.3.0.4 setName

3.4 Project

3.4.0.1 constraints

3.4.0.2 name 250 characters, alphanumeric and spaces

3.4.0.3 hasMany

3.5 Role

3.5.0.1 constraints

3.5.0.2 mapping

3.5.0.3 authority

3.5.0.4 type

3.6 Timesheet

3.6.0.1 constraints

3.6.0.2 hasMany

3.6.0.3 date

3.6.0.4 project

3.6.0.5 template

3.6.0.6 worker

3.7 Tool

3.7.0.1 constraints

3.7.0.2 name 250 characters, alphanumeric and spaces

3.7.0.3 uniqueName 250 characters, alphanumeric and spaces set to lower case

3.7.0.4 setName

3.8 User

3.9 Worker

3.9.0.1 constraints

3.9.0.2 name

3.10 Work Record

4 Controllers

4.0.1 Activity

4.0.1.1 create Creates activity with get request. Saves existing activity with post request.

4.0.1.2 delete Calls activity's destructor.

4.0.1.3 index Redirects to list() function.

4.0.1.4 list Returns the parameters of the activity and the total count of activities.

4.0.1.5 save Stores activity in database and updates the total count of activities.

4.0.1.6 show Fetches parameters of activity with get request. Calls update() with post request.

4.0.1.7 update

Change activity's parameters.

4.0.2 Home

4.0.2.1 grails

Renders the appropriate home page for the user/admin.

4.0.2.2 home

Redirects user to either adminHome or userHome based on the account/role being used.

4.0.3 Location

4.0.3.1 create

4.0.3.2 delete

4.0.3.3 edit

4.0.3.4 index

4.0.3.5 list

4.0.3.6 save

4.0.3.7 show

4.0.3.8 update

4.0.4 Login

4.0.4.1 ajaxDenied

4.0.4.2 ajaxSuccess

4.0.4.3 authfail

4.0.4.4 denied

4.0.4.5 index

4.0.4.6 full

4.0.4.7 login

4.0.5 Logout

4.0.5.1 index

4.0.6 Project

4.0.6.1 create

4.0.6.2 delete

4.0.6.3 edit

4.0.6.4 index

4.0.6.5 list

4.0.6.6 save

4.0.6.7 show

4.0.6.8 update

4.0.7 Tool

4.0.7.1 create

4.0.7.2 delete

4.0.7.3 edit

4.0.7.4 index

4.0.7.5 list

4.0.7.6 save

4.0.7.7 show

4.0.7.8 update

4.0.8 User

4.0.8.1 create

4.0.8.2 delete

4.0.8.3 edit

4.0.8.4 index

4.0.8.5 list

4.0.8.6 resetPassword

4.0.8.7 save

4.0.8.8 show

4.0.8.9 update

5 Views

5.1 Layouts

Layouts are application-wide views not tied to any specific controller. Layouts will be in the "views/layouts" folder.

5.1.1 application

This is the application-wide layout. It's responsible for deciding what layout templates to render based on user access levels. It must also render flash messages to alleviate other views from having to do so.

5.1.2 _login

This template renders when the user hasn't logged in. It'll contain an ajax form allowing the user to login. Once the user has successfully logged in, the page will refresh.

5.1.3 _admin

This template renders when an admin is logged in. It'll contain links to parts of the application admins can access.

5.1.4 _user

This template renders when a non-admin is logged in. Not sure what it'll show.

6 Interface

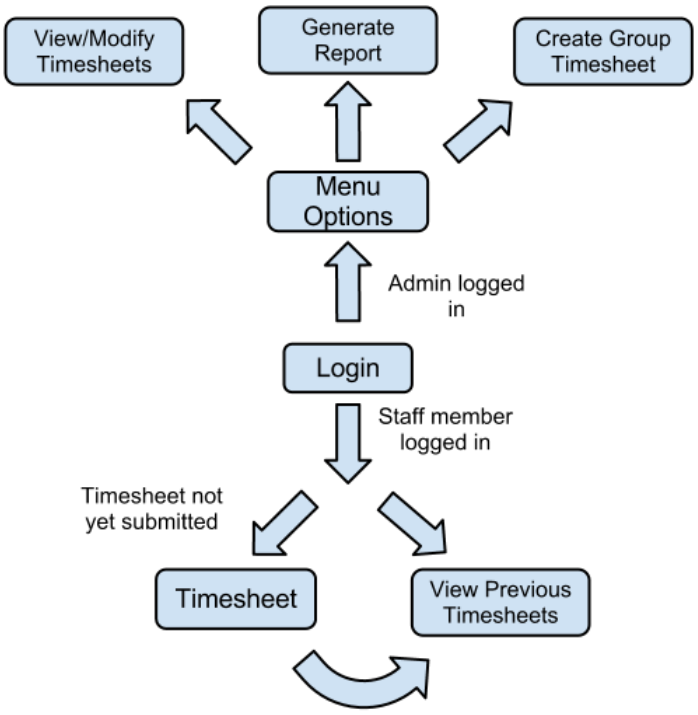

Figure 1: Diagram of user travel through site

7 Test Cases

8 Traceability matrix

	Functional						Non-functional	
	3.1.1	3.1.2	3.1.3	3.1.4	3.1.5	3.1.6	3.2.1	3.2.2
3.2								
3.3								
3.4								
3.5	✓							
3.6								
3.7								
3.8	✓							
3.9	✓							
3.10								
4.0.1								
4.0.2								
4.0.3								
4.0.4	✓							
4.0.6								
4.0.7								
4.0.8	✓							