|  |
| --- |
| TeamEffort |
| Soccer Team Management System |
| Milestone 1 |
|  |
| **Luke Brisebois - lub224**  **Simon Fanner - saf725**  **Michael Fulton - mif231**  **Steven Hancock - smh875**  **Adam Mravnik - ajm207**  **Amin Shakev - ams162**  **Patrick Weckworth - paw818**  **Tom Wetzel - thw740**  **Drake Zarowny - djz587** |
| **2/2/2012** |

|  |
| --- |
|  |

Table of Contents

[1. Introduction 3](#_Toc316250593)

[1.1 System Description 3](#_Toc316250594)

[1.2 Current System Status 3](#_Toc316250595)

[2. Build and Smoke Tests 3](#_Toc316250596)

[2.1 Methods 3](#_Toc316250597)

[3. Milestones 4](#_Toc316250598)

[3.1 Mini-Milestone List 4](#_Toc316250599)

[4. Risk 5](#_Toc316250600)

[4.1 List of Risks 5](#_Toc316250601)

[4.2 Risk Report 6](#_Toc316250602)

[3. Requirements 6](#_Toc316250603)

[3.1 Incremental Deliverables 6](#_Toc316250604)

[4. Team Personnel 7](#_Toc316250605)

[4.1 Roles 7](#_Toc316250606)

[4.2 Code Reviews 7](#_Toc316250607)

[5. Testing 7](#_Toc316250608)

[5.1 Testing Results 8](#_Toc316250609)

[5.2 Defects Report 8](#_Toc316250610)

[6 Project Plan 8](#_Toc316250611)

[6.1 Plan for Milestone Objectives 8](#_Toc316250612)

[6.2 Critical Path Method Schedule 9](#_Toc316250613)

[6.3 Estimate of Software Size and Effort 9](#_Toc316250614)

[7 Activity Report 9](#_Toc316250615)

[7.1 Work Completed 9](#_Toc316250616)

[7.2 Activity Log 10](#_Toc316250617)

[8 Design Artifacts 10](#_Toc316250618)

[8.1 Modified 10](#_Toc316250619)

[8.2 New 10](#_Toc316250620)

[9 Conclusion 10](#_Toc316250621)

# 1. Introduction

## 1.1 System Description

The project chosen for the class is a Soccer Team Management System, developed by TeamLeader in CMPT 370 in 2011. The software is a web-based system which relies on a database to store and access all types of information. A variety of programming languages are utilized in the project including Java, JavaScript, MySQL, CSS, and HTML. The project was developed in NetBeans IDE and will be done so for this class as well.

## 1.2 Current System Status

There has been a variety of challenges getting the project to a running state to use for this class. The database used previously had been deleted and had to be restored. Also the SVN repository information had to be updated for use this semester. Due to these technical issues, little in the way of programming was accomplished for this Milestone. The majority of the time was spent developing a plan for the next 2 months of development.

As it stands, the current system is quite operational and functions as expected. The team is planning to add to its functionality and fix any bugs that are currently present. This will be discussed further in the project plan.

# 2. Build and Smoke Tests

In order to insure that quality code is produced and in order to catch more obvious bugs early on it is necessary for the programming team to run build and smoke tests on a frequent basis. These tests will represent the bare minimum of error checking before committing any piece of code.

## 2.1 Methods

The method the software team will use will involve continually building and running all new and modified code whenever possible. Once the building and running is verified the person responsible for the code should then do some basic testing within the GUI to test the features they have added or changed in order to insure some minimum amount of functionality.

# 3. Milestones

There will be 5 major Milestone deliverables and 5 presentations over the course of the term. Each Milestone will include some new functionality as well as its related testing. Below is a Mini-Milestone list of the proposed objectives.

## 3.1 Mini-Milestone List

Milestone 1

* Resolve technical issues with database and versioning system
* Begin detecting existing bugs, defects, and limitations of program
* Develop plan for subsequent milestones

Milestone 2

* Ability for players and managers to edit their own user profile
* Ability for users to edit their information and password
* Tool Bar/Menu for navigation
* Support for more than one league/division

Milestone 3

* Revise password authentication (more secure, terminate sessions)
* Change date selector format and input to something more user friendly

Milestone 4

* Make system more user-friendly
* Auto scheduling system
* More efficient stat tracking

Milestone 5

* Different view styles on profiles or stats (sort by teams etc)
* Player email notification
* More robust messaging system

# 4. Risk

## 4.1 List of Risks

Software Requirement Risks

* Change of requirements
* Poor definition of requirements
* Impossible requirements

Software Risks

* Project & Milestone completion dates being unrealistic
* Hardware (server issues)
* Lack of Testing
* Personal differences in design/coding techniques
* Lack of knowledge could make some features unobtainable
* Human Errors

Software Scheduling Risks

* Over-estimate time requirements
* Under-estimate time requirements
* Not managing time affectively
* Requirements changing and not being able to adequately allocate time
* Lack of skill could require additional learning to implement goals
* Tool failure, like SVN, or difficulties with NetBeans or Java Server

Software Quality Risks

* Improper or lack of design documentation
* Unrealistic scheduling leading to lack of testing and deploying bug filled application
* Lack of knowledge leading to unforeseen bugs, errors, or unexpected results
* Application’s user interface not easy to use

Team Risks

* Lack of communication
* Scheduling conflicts
* Lack of responsibility (ownership)

Software Business Risks

* No one wants the application
* Budget failure (time or financial)
* Distribution failure

## 4.2 Risk Report

In light of our possible risks, the team will take a number of actions to ensure maximum risk avoidance. The requirements will be outlined as complete as possible in the first Milestone, however, Team Effort will remain flexible in case new requirements, or requirement definitions are changed. Any requirements that are deemed to be too expensive or impossible to meet will be abandoned immediately and no more time will be allocated to those requirements. Team Effort’s leader will ensure the team stays on task and manages time affectively. Team Effort will always conservatively budget time to ensure time-cost over-runs are minimal. The team will also ensure sufficient time is allocated for testing. Through code reviews and TXL’s ‘pretty printing’ the software’s code, although written by multiple programmers, will all be uniform and conform to one single style. Team Effort will take precautions to ensure all of their data and code are backed up, database through MySQL’s dump feature, and code and documents through SVN and individual checkout’s. Since the server can easily be run locally on Glass Fish or Tomcat, there is little worry about server failure as long as the project’s source code and database information is backed up. The team’s knowledge is limited, but with nine team members with diverse backgrounds the knowledge is extensive and broad. To fix or prevent future future risks from turning into failures, Team Effort will conduct formal and informal reviews of; code, documentation, time management, team communication and coherence. With reviews the team will be able to modify its behaviour and adapt before failures occur. Through all of the actions outlined here, Team Effort will minimize risk.

# 3. Requirements

The following is a list of currently identified project requirements:

* Remove all code clones
* Thoroughly test and fix existing code
* Produce high quality code
* Commit only working changes
* Apply a standard testing procedure to all additions and changes
* Make the existing system more versatile, usable, and useful
* Ensure that all source code is similarly formatted
* Leave useful comments both in source and with any version commits

## 3.1 Incremental Deliverables

In order to complete these requirements as well, as just generally produce a high quality product, additions to the project will be delivered incrementally as outlined in the milestones section. This means that during each milestone just a small number of feature modifications and additions will be focused on. In this way the team can spend due time and attention on the task at hand and can make sure that every requirement is met where appropriate.

# 4. Team Personnel

Roles for each milestone will be alternated so that everyone has a chance to do some coding, peer review, documentation, and presentation. Leader for this Milestone was Simon Fanner. Below is a table showing how each person was assigned for this Milestone.

## 4.1 Roles

## 

For each Milestone there will be 4 coders, sometimes working in pairs of 2. There will be 2 people in charge of peer review and testing. They will work together with the coders to discuss what needs testing and report back to the coders as well as document their work. The 3 people in charge of the documentation and presentation will assemble all of the written work from the other 6 into the report and will present it on the submission date.

## 4.2 Code Reviews

Code reviews will be conducted for each Milestone by the peer review members. They will document the review and report it back to the group leader. These positions will be rotated so that each member is reviewed over the course of the term.

# 5. Testing

During Milestone 2, tests will be developed to run on the existing system, to ensure that it works (smoke tests, of sorts). We will also make (Black Box) unit tests for some future features. During each subsequent Milestone, testers will be responsible for creating and running White Box tests on code that is being produced for that Milestone. Before each Milestone, we will run all previous tests to prove that they still hold (assuming they are applicable; for instance, white-box testing for code that has been changed after the test was made may no longer be relevant).

## 5.1 Testing Results

For this Milestone, only preliminary testing was accomplished due to the variety of technical difficulties. Most of this testing constituted getting familiar with the program and code and was used to brainstorm possible system additions. For future milestones each set of tests run will be carefully documented for future reference.

## 5.2 Defects Report

In future milestones the defects report will carefully list each bug found by the testing with a short description of how it was found, what was thought to be the problem, and the way in which it was fixed if appropriate. If any bug is ignored, even temporarily, the reasons for its exclusion should be stated in this report.

# 6 Project Plan

## 6.1 Plan for Milestone Objectives

Over the course of the semester, the team will be utilizing a wide variety of tools as required by the class. These tools will help to develop a piece of software that is of high quality, free from defects, and bad code smells. The tools will also help in the workflow of the project as well as version control. Although SVN is being used for versioning, GITHUB will also be learned by all the group members as per course requirements.

Milestone 1 was basically the “planning and preparation” milestone; the team got our svn and database stuff figured out, and set out our goals for each milestone, so that there are goals to work towards.

Milestone 2 is the “getting started” milestone, where the team will create smoke tests to run against the system for each update to ensure that we don't break the system by adding to it, as well as updating the structure of the database (to make it easier for us to use for future milestones). We'll also be adding the top navigation bar during this milestone.

Milestone 3 is the date-format fixing milestone, where we will be changing all instances of a Date to a standardized format (Javascript) for internal consistency. We'll also be changing the games listing so that it lists games and stats by season, rather than by year.

Milestone 4 is the auto-complete milestone; we'll be focusing on making it so that some textboxes will auto-complete when filling out forms, as well as making it so that you can auto-generate all games for a season rather than filling each game in individually.

Milestone 5 is the wrap-up milestone: we'll create a simple announcements system (to e-mail notifications to players and managers about upcoming games and events, changes to the system, etc.), and update the password authentication algorithm (mostly to make it case-sensitive).

## 6.2 Critical Path Method Schedule

Milestone 1 was about making sure we'd be able to work on our project as easily as we expect to be able to (for instance, being able to actually access the database, which was taken down by IT but we got it back), as well as planning out the other milestones, so that all needed to be done first.

Milestone 2 will include production of smoke tests, which need to be designed sooner rather than later (projects are harder to fix if you don't realize they're broken until they're really big). We'll also be updating the database structure, and that needs to be high-priority as well (so that we don't have to deal with a changing database structure halfway through the project, and also so that there are less new things to break).

Milestone 3 involves changing the date format for the whole project, which definitely needs to be done before the auto-completion stuff for milestone 4 is started. The database changes in milestone 2 will need to be designed with the new date structure in mind.

The password authentication for Milestone 5 isn't really related to the changes we'll be making during the other milestones, but the announcement system should be the last thing we do, since it involves reporting on other parts of the system (which will be changing during milestones 2 and 3).

## 6.3 Estimate of Software Size and Effort

As the software is web-based, size is not really a large concern, however, it will be of relatively small size (<20MB). As for the scope of the project, there will be limits on what the team will implement to be able to spend more time on the tools and lessons discussed in class as opposed to solving coding issues.

# 7 Activity Report

This activity report outlines what has been accomplished by the design team over the duration of the milestone. For this first milestone this section is fairly sparse as not much besides planning has really been done yet.

## 7.1 Work Completed

The work completed so far has been getting a new Subversion repository set up with the existing project as well as new database. Unfortunately completing these two objectives, through no fault of the design team, took much more time than was originally expected and was the primary reason for which little actual code was dealt with during this milestone. In particular the team had to deal with the old database being deleted without warning and improper permissions being given to the new repository.

The other work completed during this milestone was the execution of some very light preliminary testing just to give the group an idea of what could be changed or fixed and also just to give the group an idea of the current state of the project.

With the use of TXL, the source code for the project will become uniformed. Prior to the completion of every milestone after this one, a shell script will be run on the source code that transforms every java   
file to conform to a single style. The shell script runs a TXL grammar transformation over any java file found in a given directory. With this transformation no matter how many different programmers with different coding styles the project will still have the same unified style at each milestone.

## 7.2 Activity Log

In the future the activity log will contain a printout of the SVN log, but as it just has the initial import so far it has been left out.

# 8 Design Artifacts

This section describes any changed or added system features.

## 8.1 Modified

During this milestone the only modification made was transfer the existing project to a new SVN repository and a new MySQL database. As mentioned before these relatively minor modification ended up consuming much of the time allotted for this milestone.

## 8.2 New

During this milestone no new artifacts were created.

# 9 Conclusion

The project as it stands works fairly well. It is nice to look at, reasonably easy to use, and offers some useful function. The design team has however identified many things that can be improved upon or that can be added. These feature additions and modifications have been carefully described in the milestone objectives, and a plan for completing these objectives has been carefully laid out.

Not as much programming or testing was done during the course of this milestone as had been originally desired by the group mostly due to technical issues. Despite this, a good amount of time was spent carefully planning and documenting the management of the rest of this project, and the team feels like an appropriate amount of work has been completed. What needs to be done from here on has been explicitly outlined in this report, and as long as this report is followed the team should have no major problem making the desired modifications and additions to the current system.