**Software Project Specification for:**

To-Be-Named

**Version 1.0**

**February 7, 2016**

Abdul Kadir

Amandeep Singh

Winston Cheong

Phuc Tran

Darren Martin

Justin Gulutz

Senior Project

CS 04400-1

Professor Baliga

Table of Contents

[Introduction 3](#_Toc442481660)

[Goals 4](#_Toc442481661)

[Functional Requirements 4](#_Toc442481662)

[Admin 4](#_Toc442481663)

[Instructor 4](#_Toc442481664)

[Student 5](#_Toc442481665)

# Introduction

For some of the lower level programming courses, the following scenario is a common one: for an assignment, students submit their code via email to the professor, and then the professor runs each student’s code against a test script or suite of his/her own. This whole process can be rather redundant and superfluous because the professor has to download all the code per student and possibly have to build it, and then have to run it against his/her own code. A student’s code might not even compile or be valid. Those are only some of the possible scenarios that can make it very frustrating for an instructor.

What TO-BE-NAMED will attempt to do is ease the burden on the instructor for the elementary programming courses, where test scripts can be easily used for labs and projects to determine correctness. TO-BE-NAMED is a web application that will handle the test script/suite validation against the students’ code in a centralized and automated manner. Similar to homework boards, students can sign up to a particular course via unique course codes, and then registers an account. They can then submit coding assignments to a professor per assignment by uploading their files. The professor is able to run his/her test suite against all of the submitted code from the students at once, and view the results for each. It is up to the professor to construct a test suite that analyzes the code how and what they want, and what output they want to be given to them.

TO-BE-NAMED will be sold as a service. Clients will register for an account. Server space for the data layer and hosting would be provided by TO-BE-NAMED, as well as maintenance and support. Clients will have an admin control panel once registered, where they can set up views and partitions for distinct courses.

# Goals

* To make a robust application that eases the burden of repetitive and redundant actions during grading assignments
* To make an useful service to sell to customers

**Limitations**

* This application will not solve the programming assignments given to students by professors.
* This application will not provide test scripts for given programming assignments.
* The application will only work for specified programming languages not all.
* This project will be primarily for the interactions of students and teachers not the general public.

# Functional Requirements

## Admin

* 1. User must be able to register an institution
  2. User must be able to create separate partitions for courses

## Instructor

**2.1.0** User must be able to create separate assignments

**2.1.1** User must have area to specify directions and descriptions to assignments

**2.1.2** User must be able to upload a test script and be notified for validity

**2.1.3** User must have area to display test specifications and examples

**2.2.0** User can download code submitted by a student

**2.3.0** User can view code submitted by a student in the browser

**2.4.0** User can run test script against one or more submission for an assignment

**2.4.1** User can view the results of test run (based on output specified by script)

**2.5.0** User can comment on assignment submission for the student to see

**2.6.0** User can send out emails via system to send reminders or notifications

**2.7.0** User can enable/disable compilation error checking for submissions to assignments per assignment

**2.8.0** User can enable/disable multiple submissions

## Student

**3.1.0** User can register to a course via unique course code

**3.2.0** User can read and submit code for an assignment

**3.2.1** User can submit multiple files via file uploading functionality

**3.2.2** User can comment or name each of the files

**3.2.3** User can copy and paste code to submit

**3.2.4** If enabled, code submissions gets checked for compilation errors, with the results being displayed back to user

**3.2.5** User specifies what type of file (.java, .pl, .php) to use the proper compiler/interpreter

**3.3.0** User can view the test specifications

## System

**4.2.0** System must be robust

**4.2.1** System must be able to handle code passing through as valid that causes compilation errors such as infinite loops and program crashes by using timeouts and other solutions