Software Design Document

InPress - Group 3

Abdul Hadi Muhammad (0951862)
Liu Wenbo (0970709)
Justin Kan (0843763)

Table of Contents

Change History

The following table shows the change history for this software design document.

Version	Date	Author	Comments
—	—	AHM, LW, JK	Original content.
0	Jan 13/2014	AHM, LW, JK	Initial check in
1	April 15, 14	AHM, LW, JK	- Modifying the ER Model to reflect current design - Formatting issues - Error Handling Changes - Requirements Matchup - UI Changes

Table 1: Change History

1. INTRODUCTION

The following high-level design conforms to the IEEE Recommended Practice for Software Design Description (IEEE Standard 1016-1998).

1.1 Purpose

This document provides a high level description of the design and implementation of InPress. It has been written for software engineers, software architects, and technical program managers.

1.2 Scope

InPress is a web-based application, which allows students in universities and colleges to respond real time to questions posted by their instructor. InPress confines to three main design principles - Separation of Concern (separating the software into distinct sections), Non-Repeatable Code (reducing repetition of information of all kinds) and Simplicity (keeping the code simple and eliminating unnecessary complexities)

1.3 Overview

Topics covered in this document include the following:
- System Overview - Brief overview of the product;
- System Architecture - How the product is partitioned;
- Data Design - How data is stored, processed, and obtained;
- Component Design - Summary of algorithms used in each partition;
- Human Interface Design - Frontend design;
- Requirements Matrix - Linking requirements to design

2. SYSTEM OVERVIEW

InPress can be subdivided to two perspectives - the Student, and the Instructor. Instructors are able to add/remove courses, enroll students to their respective courses, create/remove assessments, and view/analyze results. Students are able to view at a particular assessment they are enrolled in. All users interact with the system with user-friendly graphical interface. The frontend of InPress is made fully in HTML and Bootstrap (http://getbootstrap.com/). Backend of InPress is created in PostGrepSQL (http://www.postgresql.org/). Communication between the frontend and backend are done via the web framework Django (https://www.djangoproject.com/).

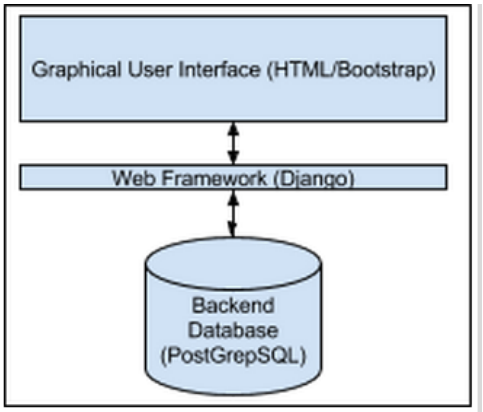

Figure 1: Basic System Overview

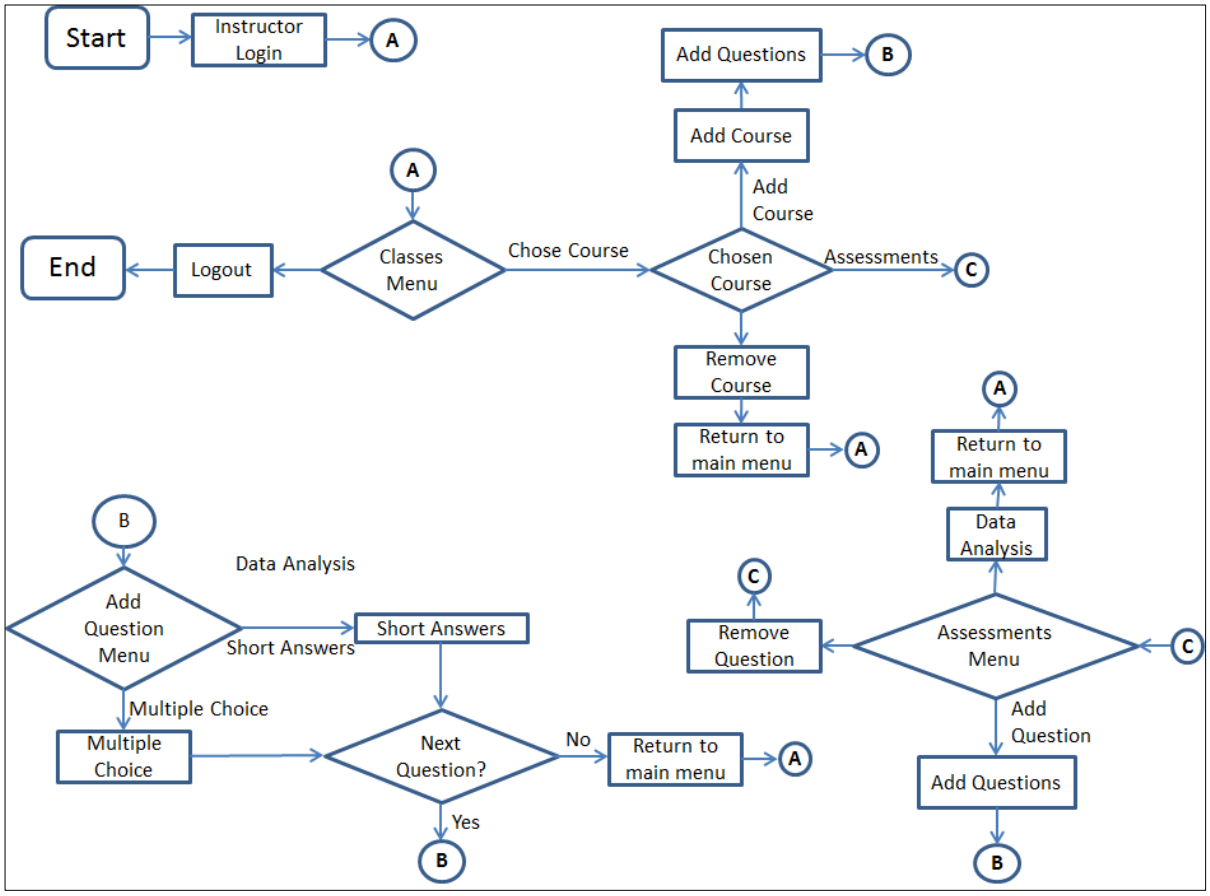

Figure 2: Instructor System Diagram

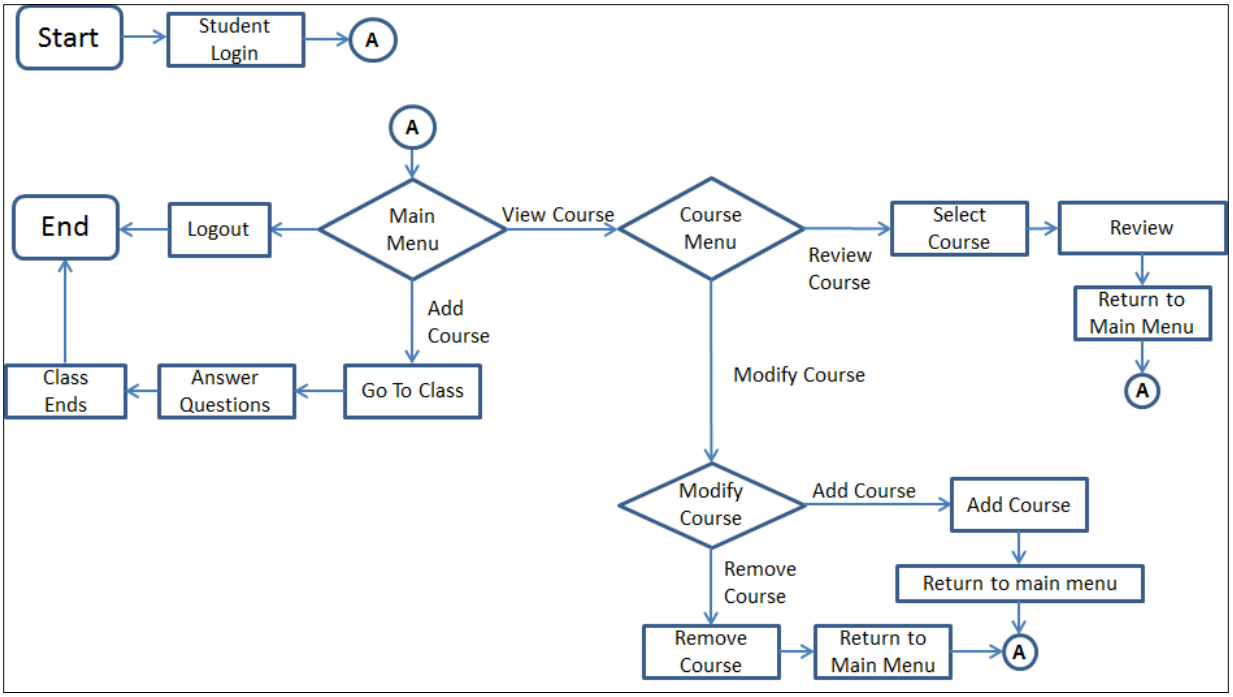

Figure 3: Student System Diagram

3. SYSTEM ARCHITECTURE

3.1 Architectural Design

Our product has two major sublevels, the instructor level and student level. We can further divide the instructor level into the various actions that the instructor has to their disposal such as add/remove classes, add/remove assessments, add/remove questions, and view class results. The student level can be further divided into three areas - view available courses that he/she is enrolled in, view assessments available, complete an available assessment.

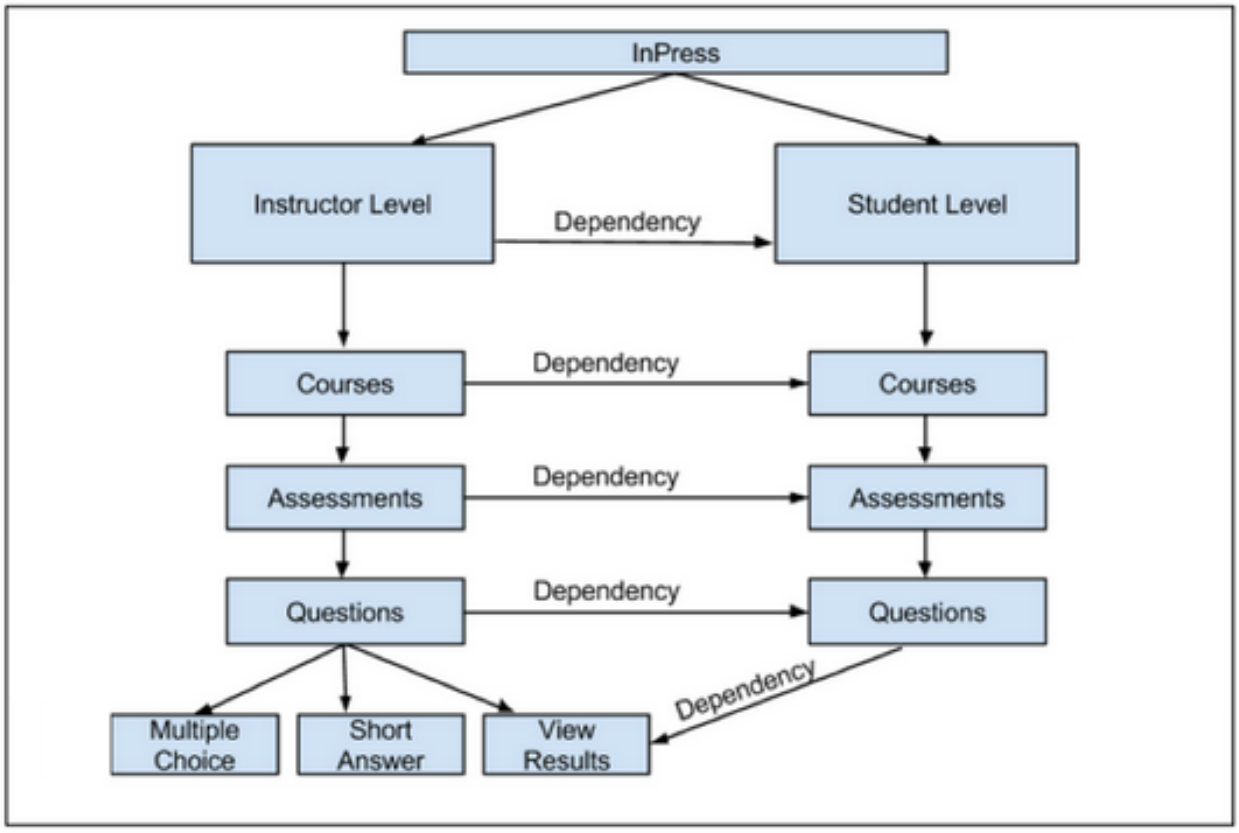

Figure 3: Decomposition of InPress

3.2 Decomposition Description

The division of the instructor and student level is regulated with a login screen. Instructors will be required to login to InPress with their unique username and password. Students will also require a login, but their login has to be created when the instructor creates the course.

At the instructor level, there are a total of three subsystems. The first of these subsystems are “Courses”. Instructors are able to add and remove classes. While creating a course, the instructor is asked basic questions of the course being offered such as the course name, course code, and a file (which is uploaded) listing all students needing enrolment in the course. Upon submitting this information, InPress creates the course, and login ids for all students enrolled in the course.

Once the course is created via the “Courses” subsystem, the instructor may choose to interact with the “Assessment” subsystem, which allows the instructor to add or remove assessment for a specific course. For an assessment to be created, there must be at least one course created. “Assessment” is a subsystem of “Courses”. Only the name and effective date of the assessment is required for the instructor to create an assessment.

Once an assessment has been created via the “Assessment” subsystem, the instructor may choose to populate the assessment with questions or answers. For questions to be generated, there must be at least one assessment created. “Questions” is a subsystem of “Assessment”. Instructors have a choice of populating the assessment with Multiple Choice or Short Answer type questions. InPress allows a maximum of five answers for a multiple-choice question.

Once the instructor creates a course, students are able to access the courses they are enrolled in. Once the instructor creates the assessment, and populates the assessment with questions/answers, the student may login and complete the assessment. Note that if the instructor has no assessment active for a course that a student is enrolled in, then the student will not be able to see anything as well.

Instructors are able to see the results of their students by accessing the assessment. Mathematical analysis such as class average, median, and graphs are generated automatically in the Results subsection.

3.3 Error Handling

Errors are handled gracefully in InPress. Information of the errors can be found on the trace files that are populated on the server. There are three types of errors we track in our system – Server errors, Database errors, and software errors. Server errors are reported in the /var/log/apache2/ directory. By using Django, we have setup the settings file to automatically send all InPress developers with an email of the stack trace of the error. This includes the description of the error, a complete python traceback, and detail HTTP request that caused the error. Database errors and Software errors are outputted using the runserver command in Django. A clear explanation of the error is outputted, and its output can be pushed to a local file.

4. DATA DESIGN

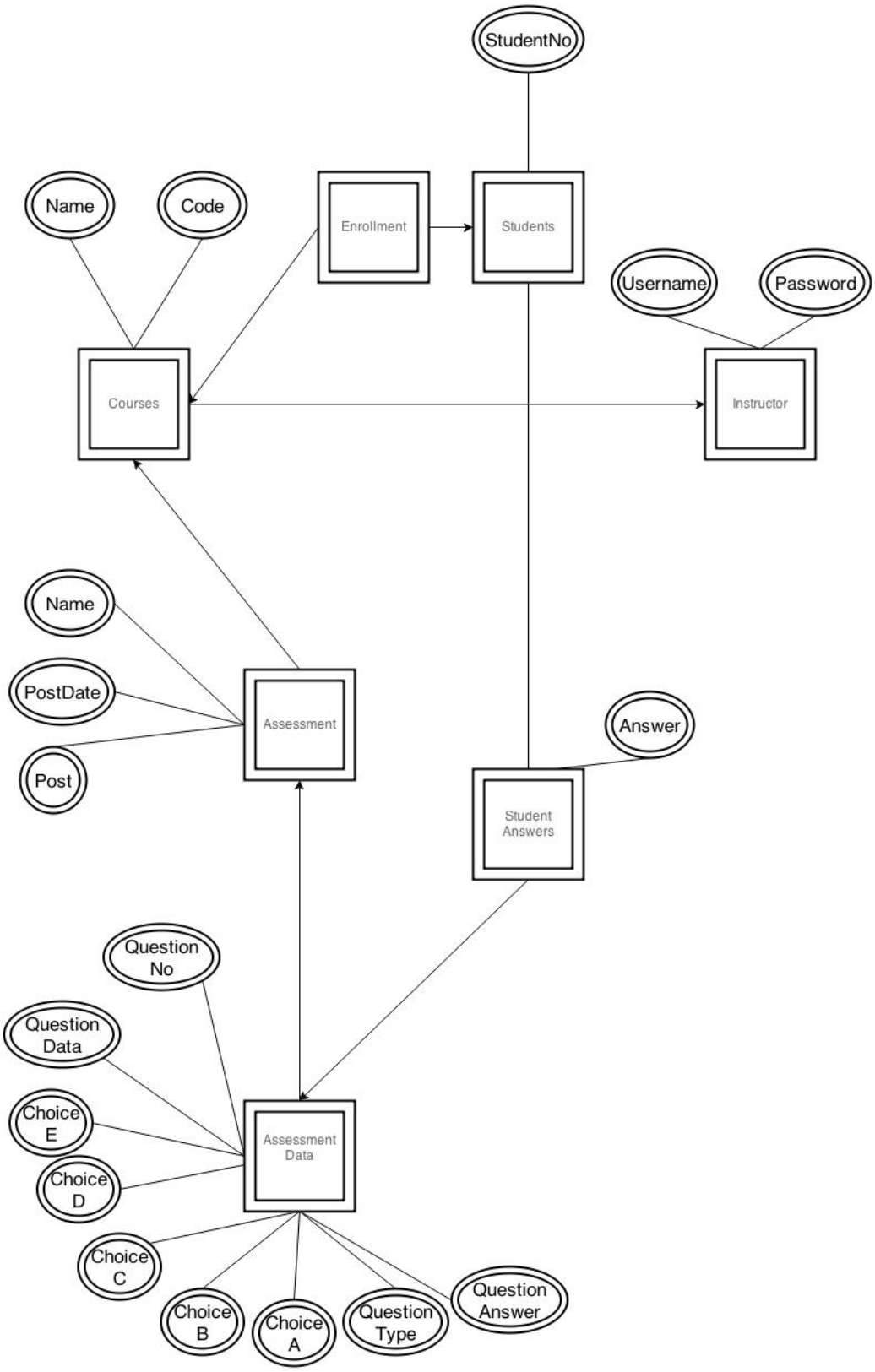

Figure 4 Database Schema Design

4.1 Data Description

All data that is inputted by the student or the instructor is stored into our internal database. This includes users profile data, course data, assessment data, and data for each question. Django allows us to transform data received by the user into data recognizable by the database easily. There are a total of five tables in our relational database – Students, Instructor, Enrollment, Courses, Assessment, StudentAnswers, AssessmentData.

The students table contains student numbers of all students enrolled in a course. Using the enrollment table, a look up of which courses each student is enrolled in can be found. All instructors are listed in the instructors table, and contain the username and password of that instructor. The courses table contains information of each course listed in InPress.

The assessment table contains basic information of all assessments available for each course such as assessment name and effective date. There is a foreign key to the course that contains the assessment. The AssessmentData table contains questions and answers of all questions on the system. There is a foreign key to the assessment that contains the questions, thus finding out all the questions for a specific assessment can be found easily.

Another table called “StudentAnswers” tracks student Answers. Foreign keys to the student and AssessmentData tables are used for easy access and retrieval purposes.

5. COMPONENT DESIGN

InPress contains a key number of algorithms. Login algorithms allow us to ensure that the person logging in has the correct username and password. This security protocol ensures that no intruder can access others’ personal information. Algorithms are also used in the analysis of student data (answers to the questions they submit). If the question is a multiple-choice answer, then a direct comparison is done to make sure the answer is correct. Short Answer question require a direct match of keywords to ensure if the answer is correct or not. Further analysis, such as number of active students and answers distribution made from the class data is also created algorithmically.

6. HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

InPress is a web based software, and thus works on all devices that are connected to the Internet. These devices include but are not limited to iPhones, iPads, Android mobile devices, tablets, and laptops. In order to make the software compatible with these various devices, we used Bootstrap, which provided us with a compatible API. Simplicity is embedded in our website. Instructors are able to create assessment in seconds, and students are able to answer questions created by their instructor seamlessly.

6.2 Screen Images

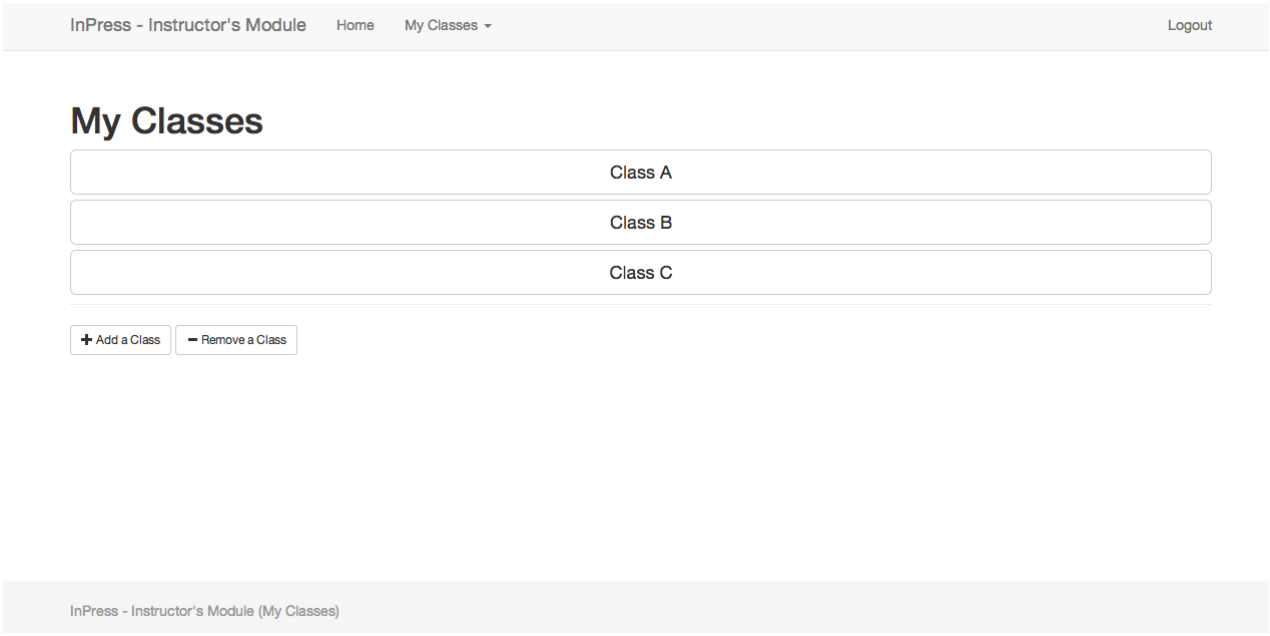

Figure 5: Instructor Module-My Classes

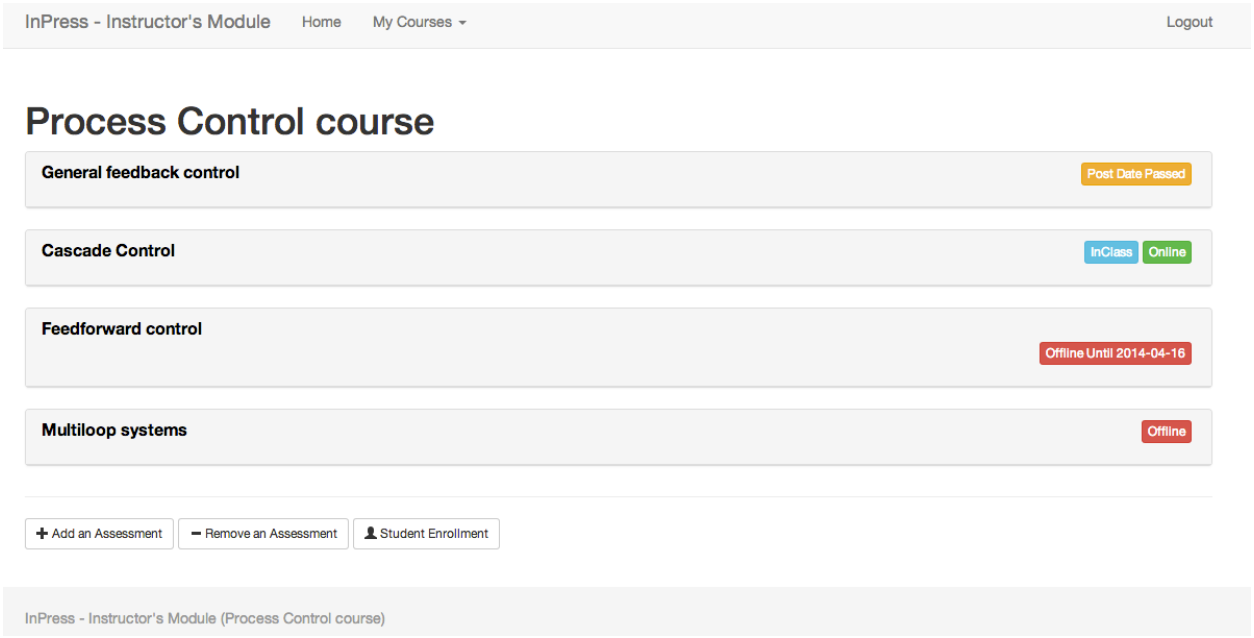

Figure 6 Instructor Module-Class View

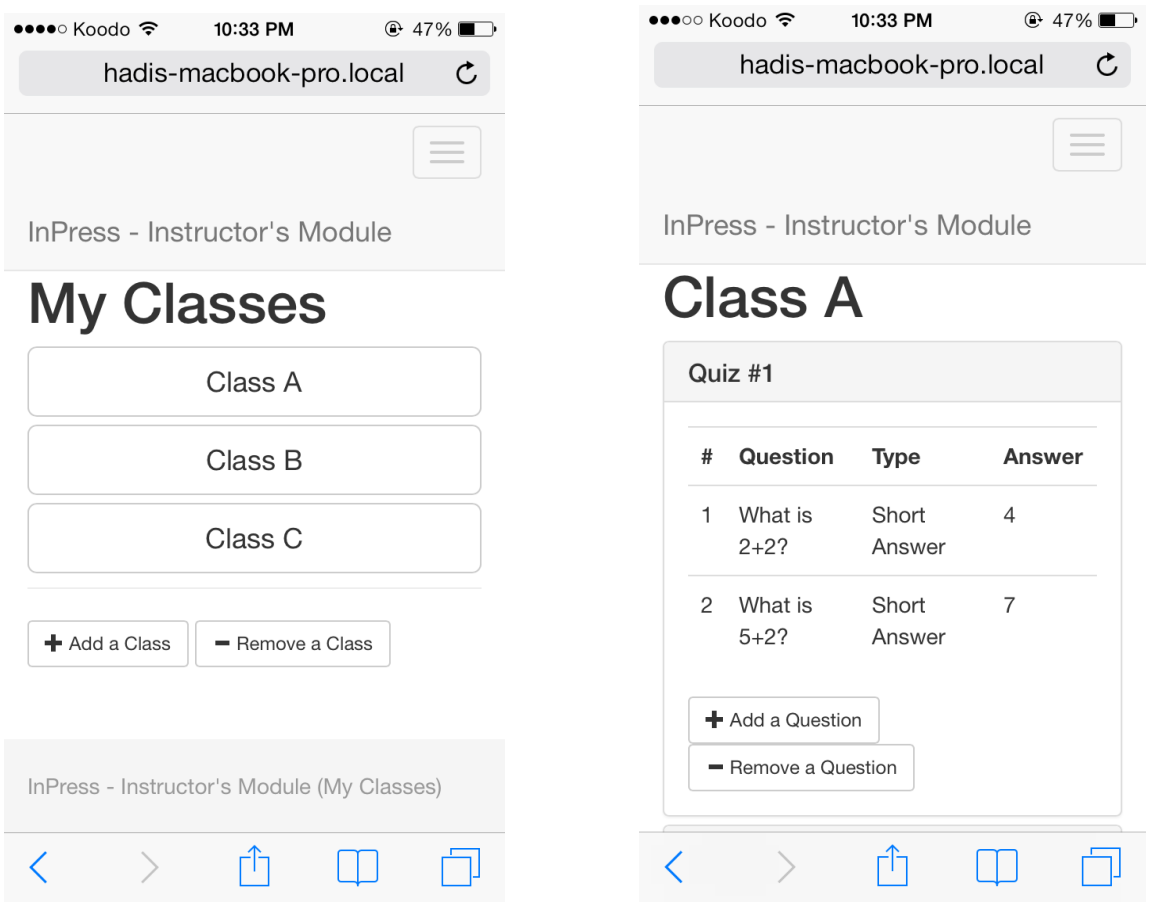

Figure 7 Instructor Module- My Classes and Class View on iPhone 5

6.3 Screen Objects and Actions

All webpages in our website all have the same web theme. This continuity in our webpages allows users to know where common elements across webpages are, and allows them to get to where they want on the website with greater speed. Furthermore this design principle, we made the number of icons limited on each webpage. Only useful icons are listed on each webpage, and thus cluttering of design is avoided.

7. REQUIREMENTS MATRIX

Requirements	Functional Design
R1: Create an Instructor Account	Created by System Administrator
R2: Login as an Instructor	Instructor Login Menu
R3: Login as a Student	Student Login Menu
R4: Create a course	Create a Course UI (Course Component)
R5: Delete a Course	Delete a Course UI (Course Component)
R6: Add an Assessment	Add an Assessment UI (Assessment Component)
R7: Remove an Assessment	Remove an Assessment UI (Assessment Component)
R8: Add questions to an assessment	Add a Question UI (Questions Component)
R9: Post/Unpost an Assessment	Post and Unpost Assessment UI (Assessment Component)
R10: Effective Date for Assessment	Change Effective Date Assessment UI (Assessment Component)
R11: Data Analysis	Analysis UI (View Results Component)
R12: Data Analysis for Students (Answer Key)	Analysis UI (View Results Component)
R13: Add and Remove Students	Add and Remove Students (Course Component)
R14: LaTex in Questions/Solutions	Add a Question UI (Questions Component)
R15: Answer Questions	Answer Questions UI (Questions Component)

Table 1: Requirements Matrix

8. REVISION 0

Revision 0 will contain the following critical components:

- (Instructor/Student) Able to login as an instructor and student
- (Instructor) Able to create and remove courses, assessments
- (Instructor) Able to create multiple choice questions, and short answer questions
- (Student) Able to answer questions
- (Instructor) Able to view answers and see basic statistics of the results

9. ANTICIPATED CHANGES

Below are some of the anticipated changes we may see in the future:

- Unable to fully secure the website from hacking
- May be required to restrict the number of the users online on the system concurrently due to performance latency issues

10. TESTING SCHEDULE (Tentative)

Date	Summary
Wednesday January 15, 2014 - Wednesday February 5, 2014	Unit testing for various features during development phase. (Components (Courses, Assessments, Questions) must be complete)
Mid January – Early February	Revision 0 completed
Wednesday March 5, 2014	Final Product GA Build Deadline
Wednesday March 5, 2014 - Monday March 24, 2014	Final manual testing of test cases. Automated testing framework should also be setup, and all test cases should be automated. Only critical defect fixes will be allowed to be delivered during this period.

Table 2: Testing Schedule