Design Document
PAT System for ELAC
MAY 15-04 Senior Design Group

CONTENTS:

Project Definition

The goal of the PAT- ELAC group is to create a web based Pre-Assessment Tool (PAT) that will allow students to determine their projected class placement in English and Mathematics by taking one or more pre-tests in each area. The Mathematics test will be adaptive by level, beginning with basic pre-algebra and proceeding through calculus. The English test will be two separate tests designed to test students' aptitudes in different areas of English knowledge.

Based on the result of the tests, the tool must output a placement level and a variety of resources focused on their areas of weakness to assist students in preparation for the final assessment.

For research purposes, and among other information the PAT will collect student information (ID number), overall pre-assessment quiz scores, and scores for individual items on the quiz, to be used to determine PAT question, topic, and test efficacy.

The primary stakeholder in this project is East Los Angeles College, (ELAC) with secondary stakeholders of Jefferson Community and Technical College, (JCTC) Iowa State University, (ISU) Des Moines Area Community College, (DMACC) and the Iowa Board of Regents.

The primary participants in the PAT-ELAC project are we, the 5 undergraduate software developers; Ruben Arenas, the point of contact for ELAC; and our advisor, Diane Rover.

Goals

Breaking down the definition of the PAT-ELAC project, we propose four high level criteria surrounding the construction of a web-based tool, the PAT, that will:

- Predict a community college applicant's placement in both English and Mathematics regardless of the required placement exam (ACCUPLACER, COMPASS, or ALEKS)
- Be easily calibrated to college-specific cut-scores aligned with course and remediation levels
- Recommend instructional videos and other materials specifically aligned with the applicant's deficit areas (in both English and Mathematics)
- Be accessible and repeatable by students without cost

The goals constitute a minimum viable product that will fulfill the needs of a client. Additional features will likely be added as time allows. Each criterion may be further broken down in to a number of subgoals surrounding particular challenges in algorithm design, data modeling, user experience, and other technical issues. However, we feel that these four goals provide an adequate level of abstraction without losing any definition in the overall project

Deliverables

The overall products of our development will consist of 4 primary deliverables and the required sub-criteria thereof:

- Fully Functional Website
    - Well Documented Code
    - Built in a Model/View/Controller framework
    - Consistent, Appealing, and Simple User Experience
    - Scalable
    - As per-user customizable as reasonably possible
    - Presents vital statistics in a usable analyzable form
    - Logging
    - Clear and understandable Data Models
    - Entirely (or nearly) bug free
- Project Report
    - Reflection
    - Guide to PAT use and, particularly, administration of the PAT
    - Highlights of key features
    - Avenues for expansion
    - Documentation of the design and creation process
- Project Presentation
    - Walkthrough
    - Key Challenges
    - Important outcomes
    - Addresses all of our identified goals

Additional deliverables may arise in the development of the overall product, but we consider these to be the 3 most important items for us to produce.

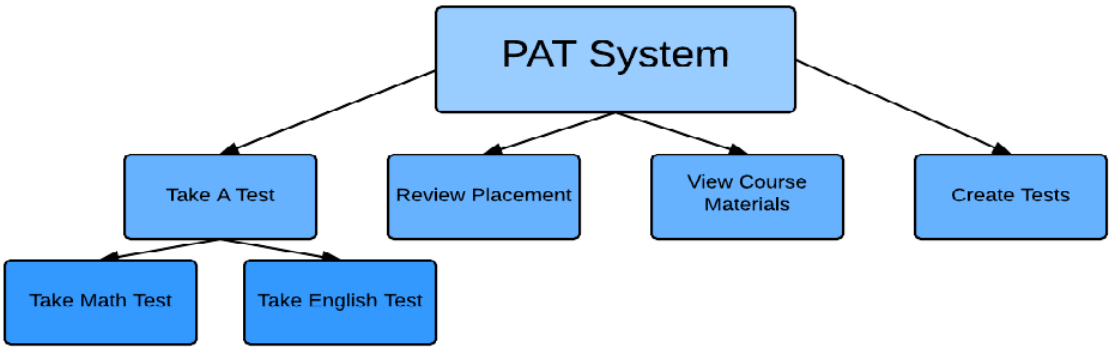
System Level Design

1. Server Side Responsibilities
    1.1. Verify users identity
    1.2. Create and store tests
    1.3. Report student placements
2. Client Side Responsibilities
    2.1. Display user options
    2.2. Display interface for tests
    2.3. Display placements
    2.4. Send information to the server on page submits

System Requirements

1. Windows Server
   1.1. Windows Server 2008 or newer
   1.2. 512 MB RAM
2. Computing Device with a Web Browser
   2.1. Chrome, Explorer, Firefox, or Safari (Including Mobile Versions)
   2.2. Internet Connection

Functional Decomposition



Feasibility Study

Much of the feasibility study for this project has already already been conducted in the process of the project proposal written by Arenas, et al. that can be accessed via this link:

https://drive.google.com/a/iastate.edu/file/d/0BwSkbYvbQthkbkJRRVdjVlB2d0dZVXlKcjN1b3FubUNJdnlV/view?usp=sharing

It can be found in the second half of the paper, which we did not attach due to its length

Technical

The most basic concept of our project is very straight forward, create a system that allows students to take tests via a web interface. As such, our technical studies led us to a fairly simple solution. In order to fully fit our criteria, we chose to use Microsoft's .NET framework which allows use of C# mixed with standard web encoding to develop web based

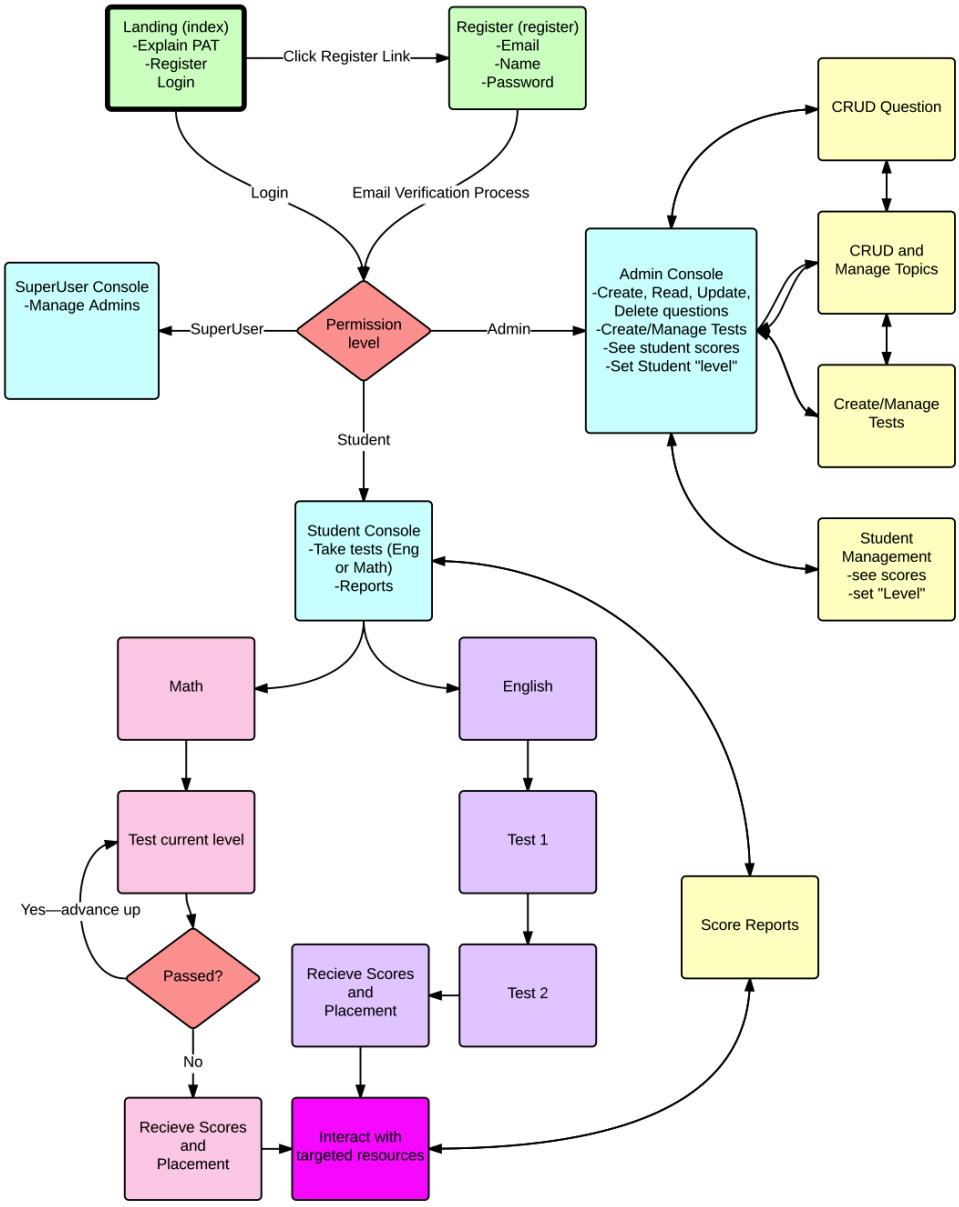
applications. Having our app installed on a Windows server will allow for the database to be stored on the same server. This approach allows us optimal speed, flexibility, portability, and ease of use for both users and developer

Pursuant to this goal, we have already gained access to a Windows server, and our computers provide a local area for debugging. we plan to update the server with the periodic versions of our development.

Investigation of the current environment

We were provided with a simple prototype that was developed in PHP by Ruben Arenas. It implemented a number of the requested features, albeit not fully. From this, as well as the provided document, we were able to design the flow for our application, as can be seen in the flowchart below. Additionally, there are a number of testing models already in existence that we can consult in the construction of our project, such as ALEKS or blackboard's internal testing system, which we hope to use for guidance and flow.

Concept Block Diagram



Detailed Page Descriptions

Landing

The web page users will will be taken to when they click on the link given to them in an email about the pre-assessment test or arrive at from clicking a link on their institution's website. There will be a brief description of the PAT (possibly a video). Users can either log in as an existing user or register a new account.

Register
New users can create an account on the Register page by providing a name, email address, and password. The new user must go through an email verification process before their account is active.

SuperUser Console
The SuperUser Console page is the home page for the administrator for the entire system. Super users will be redirected to this page when they log in. They will have the ability to view, create, edit, or delete school administrators.

Student Console
The Student Console is the home page that students will be redirected to when they log in. Students will have the ability to view previous test scores, interact with resources based off of incorrect question responses, or take a Math or English practice placement exam.

Admin Console
The Admin Console is the home page that school administrators will be redirected to when they log in. Administrators will have the ability to view, create, edit, and delete questions, topics, and tests. When administrators create questions, they indicate what topic(s) that question falls into. Administrators can create an exam by picking out specific topics to include on the exam and how many questions to include for each topic. Administrators also have the ability to view student scores and set placement level benchmarks.

Interface specifications

Models

Name	Description
Student	Defines a student using our system
Admin	Defines an administrator using our systems
Test	Represents a test/set of questions
Question	Represents a test used for quizzes
Topics	Represents a topic related to questions

Views

Name	Description
Login	Defines the login page
Register	Defines the page that allows registration
ChangePassword	
StudentAccount	Defines the student account page
AdminAccount	Defines the admin account page
Test	Defines the page for tests

Controllers

Name	Description
Home	Controls flow between accounts and test
Account	Controls login and registration of a user
Test	Controls flow of test data

Server Specifications

Our system is being developed with Microsoft's .NET framework, meaning it will be hosted on a Windows server. This server will also host a SQL database which will consist of tables that mimic our models. It will need to run Windows Server 2008 or newer.

Action	**Interface**	**Methods Used**
Store questions	Question model	createQuestion()

Store topics	Topic model	createTopic()
Send data to the client	All controllers	ReceiveData()
Receive data from client	All controllers and Models	On Page Submit

Client Specifications

Action	**Interface**	**Methods Used**
Register account	Registration web page (Click on link)	Register
Take Test	Student Profile	TakeTest
Review Placement	Placement Page (page forwarded to after test completion)	DisplayPlacement
View Course Materials	Student profile	ViewProfile
Create Test	Admin Page (login req.)	CreateTest

Software specifications

Since our final product is a web application, we have no need for hardware specifications. Our product should function properly on any major web browser (Chrome, Firefox, Internet Explorer, Safari). For our project, we are using Microsoft's ASP.NET MVC framework with Razor syntax. The ASP.NET MVC framework is an alternative ASP.NET Web Forms for creating web applications. The Razor syntax allows us to smoothly integrate C# and HTML into our project. We are using a Visual Studio as our integrated development environment .

Simulations and modeling

Our client, Ruben Arenas, designed a prototype website. His site was designed to show how a student would take a placement test and view scores. Here is a link to the site: http://www.rubenarenas.net/ELAC/

Implementation Issues and Challenges

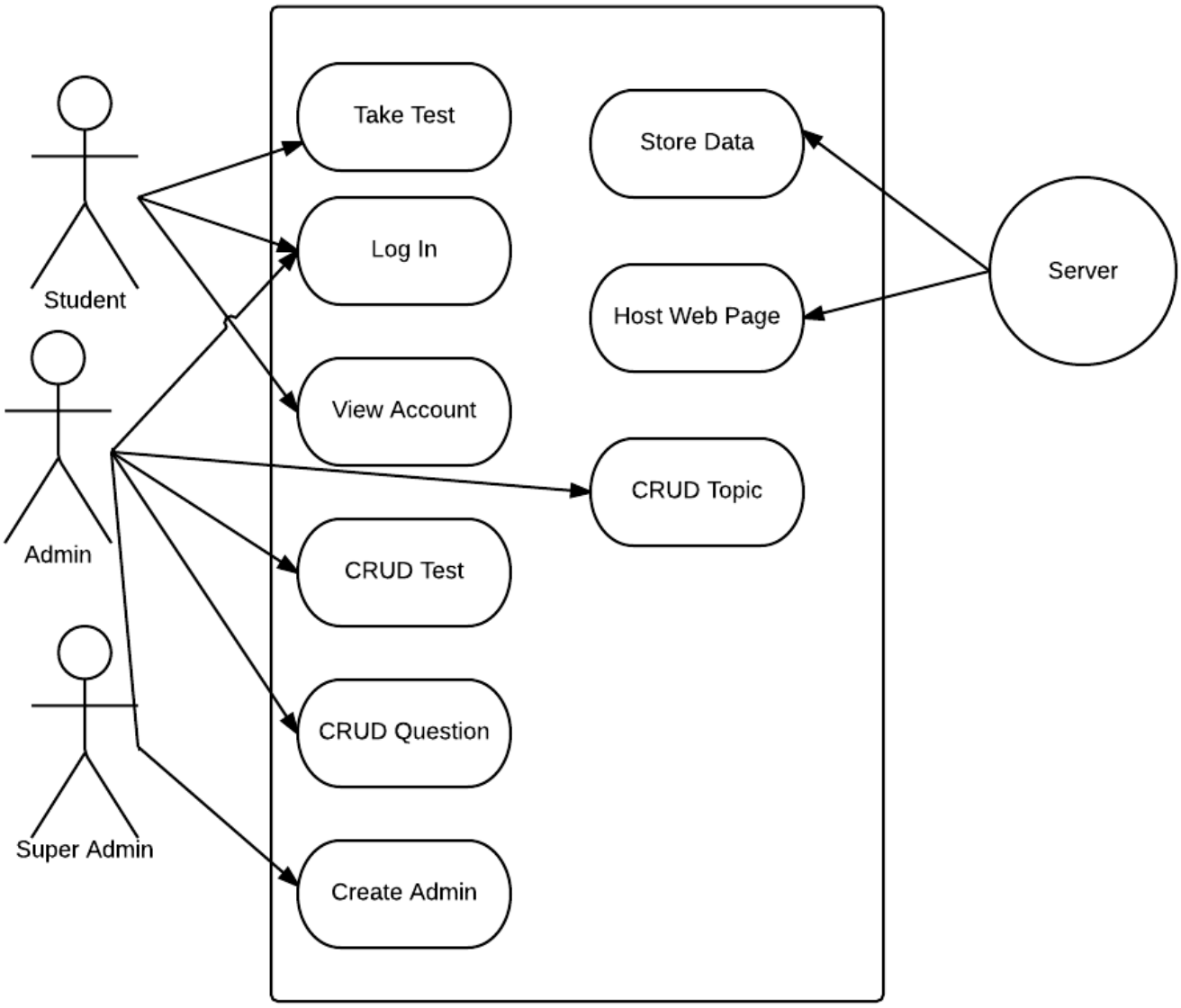
Challenge: Maintaining a constant flow throughout the entire website.
Challenge: Guaranteeing that all of our designs will match up with the client's specifications.
Challenge: Developing an accurate class placement algorithm.
Challenge: Understanding the structure of tests

Testing, procedures and specifications

We are going to focus on functional testing, opposed to unit test cases. We will develop our tests as our modules are developed. It will be a rinse and repeat process and our client will be involved with each step.

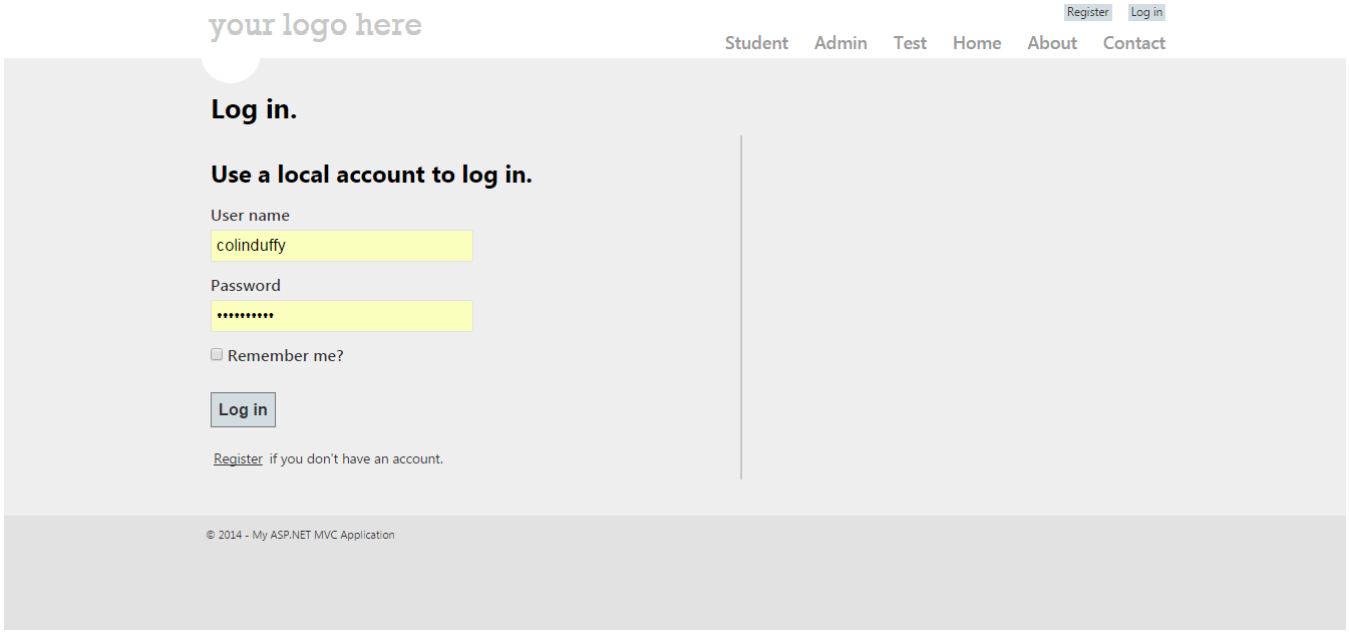
1. Create Module
2. Test Module
    2.1. Repeat Until Satisfied
3. Implement Module
4. Test Previous Modules to Ensure Nothing Was Broken

Use Case Diagrams, Use Cases, Module Diagrams

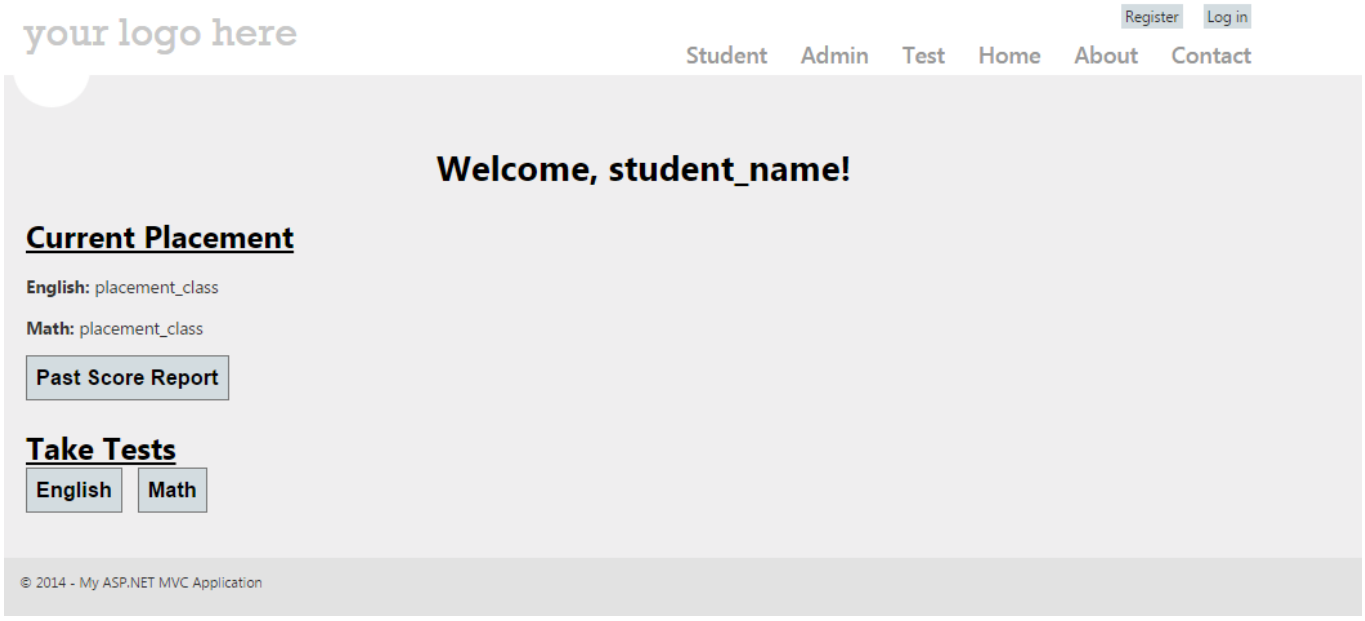

Take Test
Log In
View Account
CRUD Topic
CRUD Test
CRUD Question
Create Admin
Store Data
Host Web Page
Server
Student
Admin
Super Admin

Application Screenshots

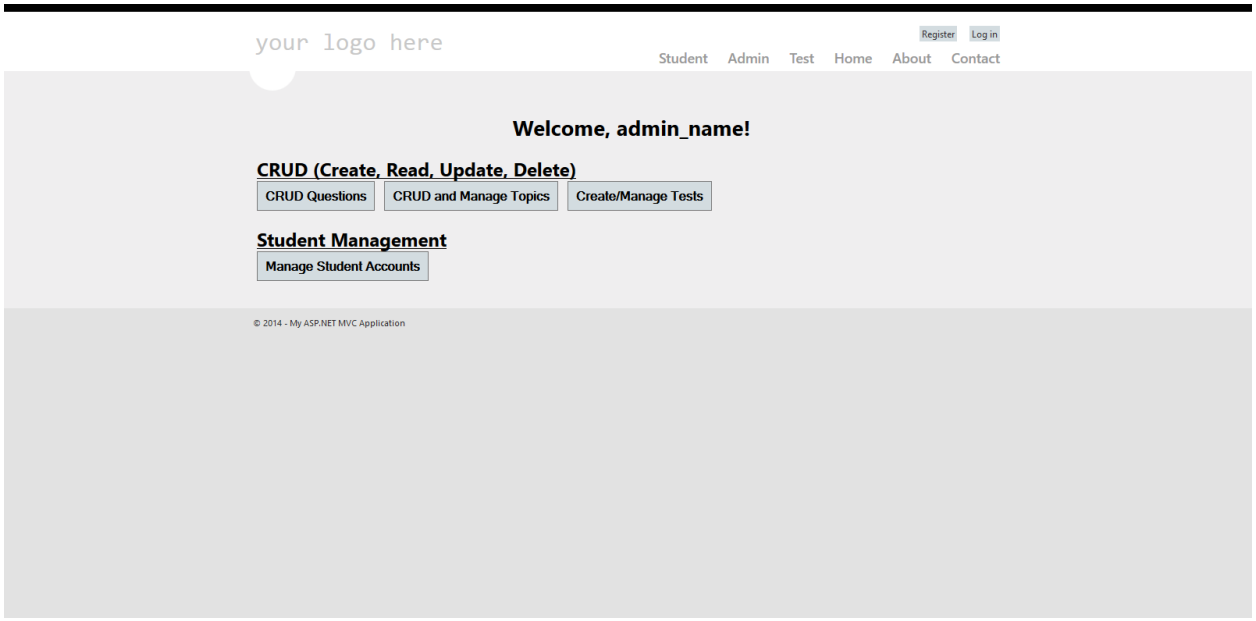
Login UI Screen



Student Account UI Screens



Admin Account UI Screen



Quiz UI Screen

//To be implemented

Create Question UI Screen

//To be implemented