VERIFICATION & VALIDATION REPORT

CONTENTS

CODE REVIEW

STRATEGY:

We chose to divide the code amongst ourselves, focusing on areas we hadn't written ourselves.
Having reviewed the code individually, we then met online to discuss each other's points and record our debrief meeting.
Things we looked for in particular include the points discussed in class, and the following guideline that we used to help summarize our thoughts:

> **As a developer, comment on how difficult would it be to understand the code written in terms of style, commenting, naming convention and general structure. It is important to note down if code is being re-used when it can be encapsulated in classes or functions. Review designs learned in lecture and search for design flaws in our system, where it could be drastically improved by changing the design or using design patterns learned in class instead of trying to re-invent the wheel.**

SUMMARIES:

Shadman on the Create Question, Assessment and Course Pages

From an initial skim of the GUI forms I noticed that the java file names for UI is rather confusing, for example a file like HHFormFrame.java which can very easily be solved by removing the use of prefix HH, and refactoring the name to be more precise. Afterwards, I found a major issue that can be improved. Consider the UI file HHCreateAssessmentFrame.java where we retrieve several text fields, apply some backend logic, and input it into our database. Viewing this file from line 117 is very difficult for a developer to start working on even with the small amount of code that it does have, this means that as front end code gets bigger this code will become unscalable. In lecture we talked about encapsulating this backend logic and separating it from gui so as to keep it scalable.

Lina on the Database code:

Having assessed the DBConnection.java file, the most important thing that stood out that it was in fact a manager class that was mentioned in lecture as a definite flag, especially in terms of scalability. In improving this, I once again refer to the lecture as the professor mentioned creating an interface for querying, as an example, and diverging that into the different things you're querying - the students, the courses and so on. In addition to this, there were language constructs that weren't being followed -

namely, some method names were written in the pothole style as opposed to Java's conventional camel case. Documentation for complex methods is concise and to the point, especially those with numerous parameters. Other than the overwhelming nature of a manager class, the code could have been structured in 'chunks' as pertaining to the different sides of the application for readability.

James on the Login and Backend Classes:

I reviewed code pertaining to the backend classes that the gui was using, as well as the login class for the gui. I had no issues to discuss about HHLogin.java. With regards to the backend classes, there are a few issues I brought to the attention of my group. In TextAnswer.java, around line 15, I was unable to follow the reasoning and understand how the isCorrect() method was checking that answers to questions are correct. I recommended that either comments explaining the implementation be added, or that the implementation be re-implemented in a way that is simpler and easier to understand. In Assessment.java I noticed that there excessive/unused variables, like isOpt, and I felt that the current method of differentiating question types was inefficient- for example, there is a variable call isMult and if true the question was treated as a multiple choice. I discussed this with my group and we felt that this issue would be addressed if we had time, because it doesn't negatively affect the function of the application. I also noted that the function getAid() was unclear and needed comments to increase readability. The same variable is seen in Courses.java and assessment.java, cID but it is written as cid in Courses.java and cID in Assessment.java. I said we should agree on one format for the variable name and be consistent with it. Finally, in the classes related to questions I noted that the abstract parent class, QuestionAbstract.java, there were no common variable and no common methods were implemented. I recommended that the variables and methods that any question type would need be implemented or present in this class to reduce the amount of copying and pasting in subsequent question type classes, which would be children of this abstract class.

Weiqiang on the View Course, Question and Assessment Pages:

The first major issue I have noticed from reviewing the view courses is that there are no documentations for lines of code that require code from other classes, given that someone who has never seen the code prior or haven't seen it for a long time, reading through those lines of code can get very disorienting going back and forth different files. There's also the issue of bad naming convention, should all be view courses/question/assessments, as of right now it's view courses, saved question/assessments. For opening new forms and closing the current one there are a lot of duplicate code this is bad design from what we learned in class, this needs to be

changed or else in the future where there will be forms with more buttons than now, there can be a ton of code that are the same. View saved questions form has a bunch of information such as the title of the question, the number of points it gives and the answer, but none of them have any labels, this needs to be changed for better user experience.

VALIDATION TESTING

<u>USER STORY 01:</u> As Adam Smith (a professor), I want to login as a professor, so I can view courses relevant to me.

T01: User login
- **i. Login with username "user" and password "pass"**

Outcome: Successful login, redirected to view courses page.

T02: Wrong username login
- **i. Login with any other username (eg. username: "name", password: "pass")**

Outcome: Error message for wrong username/password

T03: Wrong password login
- **i. Login with a different password (eg. username: "user", password: "password")**

Outcome: Error message for wrong username/password

<u>USER STORY 02:</u> As Adam Smith (a professor) I would like to create new courses so I can start organizing my assessments.

Upon selecting the create course button from the view courses page, run the following tests on the form that is opened:

T01: Create course CSCC01
- **i. Fill in the fields with required information**
  Term: Fall 2017, Code: CSCC01, Name: Intro to Software Engineering
- **ii. Click the create button**

Outcome: Success message upon course creation
Takes you to the view courses page

T02: Create course with empty fields
- **i. Create without filling out anything**

Outcome: Empty fields error message
Leaves form unchanged upon "OK" click of message box

T03: One blank field
- **i. Create course with term field blank**

Outcome: Empty fields error message
Leaves form unchanged upon "OK" click

T04: Cancel creation of course

i. **Fill in few fields [Term: Fall 2017, Code: CSCC01]**
ii. **Click back button**

Outcome: Takes you back to the view courses page

<u>USER STORY 03:</u> As Adam Smith (a professor), I would like to create an online assessment, so that I can start organizing my questions.

Upon selecting the create assessment button from the view assessments page, run the following tests on the form that is opened:

T01: Create a proper assessment
i. **Fill in the fields as follows:**
Title: "A2", Name: "Creating Personas", Points: 20
Neither checkbox ticked.
ii. **Create the assessment by clicking the create button**

Outcome: successful creation message
Takes you to the view assessments page

T02: Empty mandatory fields
i. **Fill in the fields except Name as before**
ii. **Click create button**

Outcome: Mandatory fields empty error message
The form is unchanged when you get back to it

T03: Zero points assessment
i. **Fill in the fields as before, but with points = 0**
ii. **Click create button**

Outcome: successful creation message
Takes you to the view assessments page

T04: Leave assessment form halfway
i. **Fill in form halfway, as in test 02**
ii. **Click back button**

Outcome: Takes you back to the view assessments page

<u>USER STORY 04:</u> As Adam Smith (a professor), I want to be able to view the assessments I created.

T01: Upon clicking view assessments on a newly created course, see no assessments
Outcome: Nothing to view.

T02: Upon creating an assessment [T01 or T03 in U03]
Outcome 0: New assessment is on the list

i. Click the assessment name on the list

Outcome: View the assessment information on the side panel.

T03: Upon not creating an assessment [T04 in U03]

Outcome: The uncreated assessment is not on the list of assessments.

T04: Upon opening assessments for old course for CSCC01 in Fall 2016

Outcome 0: All assessments created for that course must be displayed in the list

i. Click each assessment

Outcome: Details of each assessment displayed on the panel

<u>USER STORY 05:</u> As Adam Smith (a professor), I want to make simple questions [in my assessments].

T01: Click on the "add question" button from the view assessment page to get to the create question page and fill out the page with name, question, answer, and marks, then click "submit" button.

Outcome: Confirmation message displaying question name and question text pops up.

T02: Create a question without filling in the name field.

Outcome: Message saying one or more fields are empty.

T03: Create a question without filling in the question text field.

Outcome: Message saying one or more fields are empty.

T04: Create a question without filling in the answer field.

Outcome: Message saying one or more fields are empty.

T05: After making a question, go back to view assessment page and see the newly created question in the list of questions.

Outcome: New question(s) present in list of questions for an assessment.

<u>USER STORY 06:</u> As Adam Smith (a professor), I want to add and remove students to courses I am teaching, so that I can control who is in my classes [in view courses].

T01: Click on view students button in view course page before adding any students.

Outcome: Empty list is seen.

T02: Add students from the view student page by clicking the add students button and entering the path for a .cvs file that has student name and password, separated by a semicolon.

Outcome: A list of students from the .cvs file can be seen in the view students page.

T03: Add students from the view student page by clicking the add students button and entering the path for a .cvs file that has student name and password in an incorrect format.

Outcome: A message pops up stating that students were unsuccessfully added because the format of the file is incorrect.

T04: Add students from the view student page by clicking the add students button and entering an incorrect path for a .cvs file that has student name and password.

Outcome: A message pops up stating that students were unsuccessfully added because the file path is incorrect or does not lead to a file with student in it.

<u>USER STORY 07:</u> As Slacky MacSlackerson (a student), I want to have an account with a login and a password, so that I can view questions for courses as professors make them visible.

T01: From login screen click register button and create a student account using name and password.

Outcome: Message confirming successful account creation and showing student name pops up.

T02: From login screen enter valid student name and password to log in.

Outcome: User gets redirected to the Handy Homework Homepage screen.

T03: From login screen enter invalid/unregistered student name and password to log in.

Outcome: Message stating that given info was not correct pops up.

T04: From login screen enter valid student name and incorrect password to log in.

Outcome: Message stating that the password was not correct pops up.

T05: After logging in and before being added to a course, click the view courses button to go to the view courses page.

Outcome: An empty list of the courses student user is added to is displayed in the view courses page.

T06: After logging in and after being added to a course, click the view courses button to go to the view courses page.

Outcome: A list of the courses student user is added to is displayed in the view courses page