Refactoring Document

EECS2311, ITR3, Group 6

Refactoring 1

Previously, our database code was scattered in many different classes and methods. We fixed this in ITR3 by consolidating all of the database code in the respective database classes, all within the DB package. As a result, many (if not all) of the Long Method and Dead Code code smells were eliminated.

Refactoring 2

We rewrote the Progress Bar implementation (ProgressGUI) by rewriting the one long method into multiple smaller ones, which was reported as a major design issue (Long Method) in the QA stage (see #BR002). By refactoring this class, it eliminated a bug found in the QA stage (see #BR001) where the behaviour of the progress bar was not consistent.

Refactoring 3

One of the earliest elements of our code was the Login class, and we finally got around to cleaning it up for ITR3. Many design issues and code smells were found in the QA stage (see #BR006, #BR007, #BR008, #BR009, #BR010). Although the implementation worked without many major issues, it was difficult to work with as a result of the bad design in the original implementation (a lot of technical debt). As a result of this, we rewrote the class to be a lot easier to work with, which was especially important when we connected Login to our remote database (as opposed to the original stub database).

Problem Report Number: #BR002
Reported by: Mohammad
Date reported: Mar 26, 2023
Program (or component) name: Progress Bar
Release number: v0.2.1
Version (build) identifier: e560210
Configuration(s): Windows Desktop, Java 8
Report type: Design Issue
Can reproduce: Unknown (Design Issue)
Severity: Low
Priority: Medium
Problem summary: There are many code smells in the ProgessGUI class.
Keywords: Long Method, Magic Numbers, Dead Code, Inconsistent Formatting, Missing Comments
Problem description and how to reproduce it:
Long Method: The actionPerformed() method of the updateProgressButton ActionListener is quite long and contains multiple responsibilities. It can be refactored into smaller, more specialised methods.
Magic Numbers: The values 0 and 100 used in the JProgressBar constructor and setValue() method should be replaced with named constants.
Dead Code: The if condition in the actionPerformed() method is not necessary as the same condition is already checked in the if block before it.
Inconsistent Formatting: The indentation of the actionPerformed() method is inconsistent. It should be indented consistently.
Missing Comments: There are many areas of the code where comments would improve the readability and make it easier to understand, like for each of the constants.
Suggested fix:
Long Method: Refactor into smaller, more specialised methods.
Magic Numbers: Replace with named constants.
Dead Code: Remove redundant if condition inside of actionPerformed().
Inconsistent Formatting: Fix indentation.
Missing Comments: Add more comments to the ProgressGUI class.
Status: Closed
Resolution: See comments
Resolved by: Mohammad

Comments
Refactored the long actionPerformed() method into a smaller, more specialised method named updateProgress().

Replaced the magic numbers 0 and 100 with named constants PROGRESS_MIN and PROGRESS_MAX.
Removed the redundant if condition inside actionPerformed().
Fixed the indentation to make it consistent.
Added comments to improve code readability and explain the purpose of constants and methods.

Problem report number: #BR001
Reported by: Mohammad
Date reported: Mar 26, 2023
Program (or component) name: Progress Bar
Release number: v0.2.1
Version (build) identifier: e560210
Configuration(s): Windows Desktop, Java 8
Report type: Coding
Can reproduce: Sometimes
Severity: Medium
Priority: Medium
Problem summary: After completing the lesson, the progress bar does not always update as expected.
Keywords: Progress Bar, Lesson, New User
Problem description and how to reproduce it: After completing the lesson, the progress bar does not always update as expected. To reproduce this issue, create a new account, and run through the test cases given above. The badges do not update until the application is restarted.
Suggested fix: Instead of pulling data from the database directly from the ProgressGUI class, utilise the Database class (more robust implementation).
Status: Closed
Resolution: See comments
Resolved by: Mohammad

Comments
Bug fixed by refactoring the code and moving the update progress functionality to a separate method (updateProgress), see #BR002 for more info.

Problem Report Number #BR010
- Reported by Rajendra
- Date reported 27-03-2023
- Program (or component) name Login Page

Release number: v0.2.1
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: High.
• Problem summary: The loginframe object is always null
• Key words : Incomplete code, inefficient
• Problem description and how to reproduce it :The JTextField objects for the username and password input fields are not set up with appropriate initial sizes.
• Suggested fixTo fix this add inputUser.setPreferredSize(new Dimension(200, 30)); and inputPassword.setPreferredSize(new Dimension(200, 30)); after the text fields are initialised.
• Status: Closed
• Resolution: See comments
• Resolved by Rajendra

Comments
The objects are appropriately set up for the initial sizes and there have been modifications made to optimise the code.
add inputUser.setPreferredSize(new Dimension(200, 30)); and inputPassword.setPreferredSize(new Dimension(200, 30)); are added after the initial text fields are instantiated.

Problem report number #BR009
• Reported by Rajendra
• Date reported 27-03-2023
• Program (or component) name Login Page
Release number: v0.2.1
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: High.
• Problem summary: The loginframe object is always null
• Key words : Incomplete code, inefficient
• Problem description and how to reproduce it : The registerBut button event listener creates a new SignUpPage object but never uses it.
• Suggested fix: To fix this, assign the registerPage object to a variable and then call its setVisible() method, like SignUpPage registerPage = new SignUpPage(accountList); registerPage.setVisible(true);.

• Status: Closed
• Resolution: See comments
• Resolved by Rajendra

Comments
The register Page is fixed and does not open a new signup page every time a new user tries to register an account.
It now checks for the signup visibility and if found false then only opens up a new signup page

Problem Report Number #BR008
• Reported by Rajendra
• Date reported 27-03-2023
• Program (or component) name Login Page
Release number: v0.2.1
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: High.
• Problem summary: The loginframe object is always null
• Key words : Incomplete code, inefficient
• Problem description and how to reproduce it :The SQL query used to retrieve the account details is constructed using string concatenation, which makes the code vulnerable to SQL injection attacks
• Suggested fix To fix this, use a PreparedStatement with placeholders instead, like String query = "SELECT * FROM account WHERE username = ? AND password = ?"; PreparedStatement statement = connection.prepareStatement(query); statement.setString(1, username); statement.setString(2, password); ResultSet resultSet = statement.executeQuery();.
• Status: Closed
• Resolution: See comments
• Resolved by Rajendra

Comments
That SQL query was removed since it was improper and did not provide code security.

Problem Report Number #BR007
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: High.
• Problem summary: Database connections is inappropriate
• Key words : Incomplete code, inefficient
• Problem description and how to reproduce it :The getConnection() method call to connect to the database is not wrapped in a try-catch block, so if the connection fails, the program will crash with an unhandled exception.
• Suggested fix To fix this, add a try-catch block around the getConnection() call, like try { Connection connection = DriverManager.getConnection(url, sqlUsername, sqlPassword); ... } catch (SQLException e) { e.printStackTrace(); }.
• Status: Tester fills this in. Open
• Resolution: See comments
• Resolved by Rajendra

Comment
The change has been implemented

Problem Report Number #BR007
• Reported by Rajendra
• Date reported 27-03-2023
• Program (or component) name Login Page
Release number: v0.2.1
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: High.
• Problem summary: Issue Minor
• Key words : Incomplete code, inefficient
• Problem description and how to reproduce it :The signInBut and registerBut buttons are not given any text to display.
• Suggested fix: To fix this, add some text to the setText() method calls, like signInBut.setText("Sign In"); and registerBut.setText("Register");.
• Status: Closed
• Resolution: See comments
• Resolved by Rajendra

Comments
The change has been made, the setText() method calls, like signInBut.setText("Sign In") are used as per required.

Problem Report Number #BR006
• Reported by Rajendra
• Date reported 27-03-2023
• Program (or component) name Login Page
Release number: v0.2.1
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: High.
• Problem summary: The loginframe object is always null
• Key words : Incomplete code, inefficient
• Problem description and how to reproduce it :The Login constructor creates a new JFrame object named loginFrame, but it never assigns it to the class-level loginFrame variable. This means that the loginFrame variable is always null, and any methods that try to access it will throw a NullPointerException
• Suggested fix: To fix this, change the line final JFrame loginFrame = new JFrame("Login Page"); to loginFrame = new JFrame("Login Page");.
• Status: Closed
• Resolution: See comments
• Resolved by Rajendra

Comments
The new frames were instantiated with valid input arguments.

Problem Report Number #BR007
Reported by: Karanpreet Raja
Date reported: Mar 26, 2023
Program (or component) name: Syntax Testing
Release number: v0.2.1
Version (build) identifier: e560210
Configuration(s): Ubuntu 22.04 LTS, Java 8
Report type: Design Issue
Can reproduce: Yes (Design Issue)
Severity: Medium
Priority: High
Problem summary: Many code smells in the Syntax Testing big user story implementation
Keywords: Primitive Obsession, Uncommunicative Names, Lazy Class, Data Clumps, Uncommunicative Names, Magic Numbers, Duplicate Code

Problem description and how to reproduce it:
SelectionButton.java File:
Primitive Obsession: "Boolean isSelected" primitive can be encapsulated together in an object with its other components
Inconsistent Names: We have variable names "isSelected" and "isSelected()" which can cause confusion
Uncommunicative Names: "SelectionButton" should be a different name considering it doesn't really explain what the class actually is first hand.
Question.java File:
Lazy Class: It is clear when observing the class that it is meant to act as the OOP backed of the Syntax Testing, however, it is not actually implemented and thus servers no purpose until it is added to SelectionButton
Data Clumps: Too many getters and setters makes the code harder to understand, it should be organise in a more better way such as using a class
Uncommunicative Names: We have "getNoAns()" and "noAns" which is quite hard to understand what the method and variable does, perhaps change it to "getNumberAns()" for instance.
SelectionButton.java and MultipleChoiceButton.java File:
Oddball Solution and Inappropriate Intimacy: MultipleChoiceButton Extends from the SelectionButton class which makes understanding the two classes hard
Magic Numbers: The color values should be customisable for the buttons and not just set to a default color when the button is selected or not selected
Missing Comments: Almost no comments exist for this class, and thus the functionality of the class is quite confusing.
Suggested fix:
Long Method: Refactor into many smaller, and more specialised methods.
Magic Numbers: Replace with easily changeable constants.
Lazy Class: Remove unused class or integrate it properly with the rest of the user story (recommended).
Missing Comments: Add comments to describe the functionality of the classes that are missing them.
Data Clumps: Remove unnecessary setters and getters that are never used
Uncommunicative Names: Either change names of functions and attributes to better describe the functionality or add a comment explaining the name during initialisation
Inappropriate Intimacy: Rewrite some code instead of using inheritance if it makes the class's functionality clearer and the majority of the code is not similar.
Primitive Obsession: encapsulate with the object instead of using primitives
Status: Closed
Resolution: See comments
Resolved by: Mohammad

Comments
To fix the reported code smells, I created a new "Answer" class that contains answer-related methods. I also renamed the noAns variable and its getter method to be more communicative.

Problem Report Number #BR003
• Reported by Abdirazak Yusuf
• Date reported 27-03-2023
• Program: SyntaxDoc.java
Release number: v0.2.1
Version (build) identifier: e560210
CSS Navigation Bug:
There is a bug in the code as it does not display the CSS document on the left panel. This issue might be due to an incorrect address/URL.
Status: Closed
Resolution: See comments
Resolved by: Mohammad

Comments
This was fixed by parsing markdown and rendering to HTML instead of grabbing HTML and attempting to render that directly (see commit ID 035de96).

Problem Report Number #BR003
• Reported by Abdirazak Yusuf
• Date reported 27-03-2023
• Program: SyntaxDoc.java
Release number: v0.2.1
Version (build) identifier: e560210
Magic Numbers:
I have found hardcoded values such as 700, 450, 200, and 10. For instance, we have setMinimumSize(new java.awt.Dimension(700, 450));. The significance of these values is unclear. The designer should have used descriptive names.
Long Method:
The SyntaxDoc(String language) constructor is extremely long and performs multiple tasks, including creating UI components and setting up action listeners.
Exposed Internals:
The instance variables LANGUAGES and LANGUAGE_URLS are static and public, allowing other classes to access them. Changing them to private would prevent unauthorized access.
Data Clumps:

Instance variables LANGUAGES and LANGUAGE_URLS are similar and could cause confusion or issues. I suggest using a data structure like a HashMap or creating a custom class.
Lack of Proper Error Handling:
No exception handling is present for the loadPage method.
Incomplete Code:
The second component of the user story is not implemented, indicating incomplete code.
Status: Closed
Resolution: See comments
Resolved by: Mohammad

Comments
Changes implemented: Replacing magic numbers with variables (WIDTH, HEIGHT, HOME_URL), shuffling around from SyntaxDoc to loadPage, changing instance variables LANGUAGES and LANGUAGES_URLS to private, wrapping loadPage in a try catch to allow for error handling, and implementing the second component of the user story.

Problem Report Number #BR005
Reported by: Karanpreet Raja
Date reported: Mar 26, 2023
Program (or component) name: Syntax Testing
Release number: v0.2.1
Version (build) identifier: e560210
Configuration(s): Ubuntu 22.04 LTS, Java 8
Report type: Design Issue
Can reproduce: Yes (Design Issue)
Severity: Medium
Priority: Medium
Problem summary: Recurring Questions
Keywords:
Problem description and how to reproduce it:
Sign in with a test account (user: test, pass: test). Launch session. Navigate through the questions and there is a good chance that you will be prompted with the same question multiple times. If this does not occur the first time you run a session, repeat the process.
Suggested fix: Utilise the Question class which is already built and modify it such that it pre selects all the questions that will be prompted to the user in a session. When constructing the "Array" of questions, make sure when appending a question, make sure that the question does not already exist within the array.

Status: Closed
Resolution: See comments
Resolved by: Mohammad

Comments
This was fixed by changing the MySQL database table to grab questions in order to prevent duplicate questions from being served to the user.

Problem Report Number #BR059
Program (or Component) Name: BadgeGUI.java
Release Number: v0.2.1
Problem Summary; Code Smells in BadgeGUI class
Large Class (Bloater)
The entire class is 1 large constructor, that is all
Limitation of java swing and potentially fixable using other libraries (Not viable for this project)
Long Method (Bloater)
Elements in the action listener should be moved outside of the constructor in order to promote reusability when the addition of other languages will be factored into the method, Potentially replaceable by an extraction method with custom exception throws in order to streamline the process rather than hard code
Long Parameter List (Bloater)
Badges and BadgeGUI can be made into separate classes which greatly decreases the size of the code in the badge GUI class as well as many useless parameters which are just reused
Primitive Obsession (Bloater)
Use of constants for holding the badge default sizes
Comments (Dispensable)
The comments or rather a lack thereof make the program difficult to read and understand without an existing understanding of the java swing library.
Lazy Class (Dispensable)
This class is built to support future functionality but in its current state it simply must be redone almost entirely in order to provide functionality outside its current limitations
Status: Closed
Resolution: See comments
Resolved by: Mohammad
Comments
This issue was fixed by rewriting BadgeGUI to fix these code smells (see commit ID 278b234).

Problem Report Number #BR059
Program (or Component) Name: BadgeGUI.java
Release Number: v0.2.1
Bug Severity: Trivial
Bug Priority: Minor
Problem Summary: All badges do not update fully when updating badges
Steps to Reproduce: Create a new account, progress in badges, and update badges in the dashboard
Status: Closed
Resolution: See comments
Resolved by: Mohammad

Comments
By refactoring the BadgeGUI class, the badge implementation is now much more consistent (See #59).

Problem Report Number #BR031
• Reported by Rajendra
• Date reported 27-03-2023
• Program (or component) name SignUp Page
Release number: v0.2.1
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: KaranPreet Raja.
• Problem summary: User can still register even if it's record is present in the Database
• Key words : Inconsistent with database
• Problem description and how to reproduce it : The button listener in the signup class would display an error message but still allow the user to register, to fix this remove the SignUp button if the user already exists in the database.
• Suggested fix To fix this remove the SignUp button if the user already exists in the database
• Status: Closed
• Resolution: See comments
• Resolved by Rajendra

Comments
Added a check to see if the user record is already present in the database.

Problem Report Number #BR055
• Reported by Rajendra
• Date reported 27-03-2023
• Program (or component) name SignUp Page
Release number: v0.2.1
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: KaranPreet Raja.
• Problem summary: There is no input validation on the username and password fields
• Key words : Inconsistent with database, potential security issues, design vulnerable to breaks
• Problem description and how to reproduce it : The is no restriction on the length of username and password thus it might not fit in the database if limits exceed at the same time allow the user to register with the max limit of username and password.
• Suggested fix To fix this remove the make sure that the maximum input values are mentioned which are cross checked with database limits, display an error message if the limits exceed
• Status: Closed
• Resolution: See comments
• Resolved by Mohammad

Comments
When rewriting the SignUp and Login classes, input validation was added to ensure that a valid username and password is inputted.

Problem Report Number #BR055
• Reported by Rajendra
• Date reported 27-03-2023
• Program (or component) name SignUp Page
Release number: v0.2.1
Version (build) identifier: e560210
• Report type: coding
• Can reproduce: Sometimes
• Severity: Medium.
• Priority: KaranPreet Raja.
• Problem summary: An empty string is inserted into Database in the classes field
• Key words : Inconsistent with database, inefficient

• Problem description and how to reproduce it : In the "Sign Up" button event listener, if the user selects no courses, an empty string is inserted into the "classes" field of the database
• Suggested fix To fix this either require the user to select at least one course or insert a default value instead of an empty string
• Status: Closed
• Resolution: See comments
• Resolved by Mohammad

Comments
A default course was added to the database to ensure that the user always has a course on their account (the Python course is selected by default).