CS673 Software Engineering

Lab 3 - Refactoring and TDD

## Description:

The purpose of this lab is to make sure that all students can apply the basic techniques of refactoring and TDD in their software projects.

## Instructions:

1. Identify one or two bad smells in any code in your project. Then apply appropriate refactoring techniques to get rid of those smells.
2. Choose a simple/small user story or functionality in your project as an example, to describe how the TDD/BDD flow can be used to develop the test code and the production code. It can be anything already implemented or anything you would like to implement. You will need to choose the proper automated testing framework for your project (e.g., JUnit, unittest, Selenium, Cucumber)
3. Please submit a document on Blackboard in either PDF or DOC format and include the following sections:
   1. Refactoring
      1. The original code with bad smells before refactoring. You can copy the code directly into the report or take screenshots.
      2. A description of the smells and the refactoring techniques you used to get rid of them.
      3. The modified code after refactoring.
   2. Testing
      1. Both the source code and the test code of your chosen example.
      2. A description about how the TDD/BDD can be used in this example. Any testing tools and commands used should be mentioned in your report. You can add some screenshots to show the test failed first without the proper implementation in the production code and then pass after the implementation is done in the production code.
4. Bad smell refactoring
   1. First issue: security vulnerability with plaintext API Key (covered, again for security precautions). In this case, there was an API key hard-coded into several files. To ensure that no bad actors misused our API access, I simply moved the key to a .env file included in gitignore and had the program access the key from the locally stored file. This means that it will not only no longer be hardcoded, but it won’t even be in the repository to begin with, removing this major vulnerability.

Text

Description automatically generated

Text

Description automatically generated

* 1. Second issue: ???

1. Testing