**Course Project – Project Proposal**

Tim Mastarone

Rasmussen University

Java Programming

Instructor: Anastasia Rashtchian

Module 01

May 11, 2024



**Project Proposal - BMI Rate Calculator**

**Introduction**

The BMI Rate Calculator is a simple Visual Basic application that will be more useful and maintainable if we convert to a Java application. In the insurance industry we should be using current technology that is user friendly and can provide additional functionality in the future. Benefits that Java provides over Visual Basic are performance, scalability, community support, and ecosystem stability. The main goal of this application is to collect user information which is used in calculating a BMI score and therefore determining an insurance rate. This document will spell out exactly how we aim to fulfill these requirements with a Java solution.

**Project Overview**

Converting from Visual Basic to Java will provide countless benefits. Migrating this app from Visual Basic to Java will improve its performance, scalability, and compatibility but also position it for continued growth. Our goal is to convert the application into Java while following the requirements laid out in this document.

**Scope of Work**

The BMI Rate Calculator aims to provide an interactive user interface backed by a robust backend database system. Input values will be accepted and validated. When values are deemed as valid the user will have an option so persist them to database storage.

**Technology and Tools**

The front-end application will be written in Java. The IDE that will be used for development is IntelliJ IDEA. This is a feature rich IDE that is very popular among Java developers in the industry. We will be using version control which IntelliJ has built in functionality for. Another advantage provided by the IDE is a useful intellisense editor. The IDE can be configured with our development guidelines which will be used by the editor to provide suggestions and check the integrity of the code without compiling.

The backend persistent data storage will utilize MySQL. Like our IDE MySQL can be implemented at no cost. MySQL supports more than enough functionality to cover the scope of this application. It can also easily be scaled if this application is to grow in the future. MySQL workbench will be used to query and visualize the database for testing.

**User Interface Design**

The user interface will be designed with simplicity in mind. Our main objective is to provide a simple to use and robust interface. The interface will be intuitive and designed to offer as much built in error protection as possible. We will implement drop downs in cases where user data can easily be entered incorrectly.

**Functionality**

**Functional Requirements**

* Take user input of Name, Weight, Date of Birth, and Height.
* Calculate BMI based on the user’s input.
* Determine the insurance rate based on the BMI calculated.
* Output the result to the user.
* Persist the information in a database.

**Non-Functional Requirements**

* **Validate all entered data.**
* **Provide an intuitive user-friendly experience.**
* **Create an aesthetically pleasing user interface.**
* **Use error handling to create a robust application.**
* **The application and database system will follow best practices for security.**

**Coding Practices and Version Control**

All code will be stored in a git repository. The repository will have a ‘main’ branch that is protected so that only qualified users can merge changes into it. Issues will be entered into the issue tracking system built into the version control GUI. For any new feature or bug, it will have an issue number. We will follow a development process to create branches in version control for any feature or bug. This will have a correlated issue on our git system and once the branch is merged into the ‘main’ branch, the issue will automatically be closed. Having this version control and issue tracking system will keep a detailed history of any changes made throughout the development lifecycle.

**Data Handling and Storage**

The data from the user will always be validated before storage to ensure data integrity. We will use carefully crafted queries in a database manager class to interact with our MySQL backend. This will provide clean sending and receiving of data to the front-end while abstracting the functionality to reduce any potential problems. This structure will also allow us the ability to scale the application in the future.

**BMI and Rate Calculations**

The BMI percentage will be calculated with the simple formula “weight (lbs) / height (in2) x 703”. We will use the entries from the user and determine BMI calculation and then the insurance rate based on the table below:

|  |  |
| --- | --- |
| **User’s Calculated BMI Range** | **Insurance Payment Rate** |
| < 18.5 | Low |
| 18.5-24.9 | Low |
| 25-29.9 | High |
| **>29.9** | Highest |

**Testing**

We will implement our code with test driven development. This means for any functionality of the program we will write an automated test to accompany it. When we add new features later, we can call these original tests to make sure functionality in other areas was not affected with new changes. As the application evolves, we could also include testing reports when an automated test is completed.

**Conclusion**

We created this document to convey exactly how the application will be implemented. We also have created guidelines for the efficient creation of a stable program. Our version control system will ensure we do not lose our progress, and we have a clear history of any change that has been made. This combined with the modular design of the program will make it easily scalable to accommodate and future changes.