Software Requirements Specification

for

Ideal Body Weight Calculator

(IBWC)

Prepared by

Group Name: 04

Akindu Randira	23ug1-0152
Isuru Balasooriya	23ug1-0162
Malindu Kanishka	23ug1-0015

Course:	ECS2301-Software Engineering and Projects
Date:	4th January 2025

Contents

1.Introduction

1.1 Document Purpose

This Software Requirements Specification (SRS) document outlines the functional and non-functional requirements for the **Ideal Body Weight Calculator** application. The application will calculate the ideal body weight (IBW) based on the Devine formula, determine whether the user's actual weight falls within a healthy range, and provide actionable recommendations for those who are underweight or overweight.

The purpose of this document is to define the scope, functionality, and expected behavior of the software, ensuring that all stakeholders, including developers, testers, and end-users, have a clear understanding of the system. This SRS focuses solely on the desktop application component of the project, detailing its interface, functionality, and constraints.

1.2 Product Scope

The **Ideal Body Weight Calculator** is a desktop application designed to help users determine their ideal body weight based on the Devine formula. Users can input their gender, height, and actual weight to calculate their IBW, assess their weight status (underweight, healthy, or overweight), and receive personalized recommendations to achieve or maintain a healthy weight. This application aims to promote health awareness by providing a simple and accurate tool for weight management.

The primary goal of the application is to empower users with insights into their body weight and guidance to improve their overall well-being. It is intended for a wide audience, including individuals focused on fitness, healthcare professionals, and those seeking to monitor their weight. The application offers benefits such as ease of use, quick calculations, and actionable feedback, making it a valuable tool for maintaining a healthy lifestyle.

1.3 Intended Audience and Document Overview

Intended Audience

This document is intended for the following readers:

1. **Client**: To evaluate the scope, purpose, and functionality of the Ideal Body Weight Calculator application and ensure it meets the requirements and objectives.
2. **Developers**: To understand the functional and non-functional requirements needed for implementing the application.

3. **Testers**: To identify key features and scenarios for validation during the testing phase.

4. **Documentation Writers**: To reference the software’s requirements when preparing user manuals or guides.

Document Overview

This Software Requirements Specification (SRS) document is organized into the following sections:

- **Section 1**: Introduction, which provides an overview of the purpose, scope, and intended audience of the document.
- **Section 2**: Overall Description, which includes the product perspective, functionality, user characteristics, assumptions, and constraints.
- **Section 3**: Specific Requirements, which details the functional and non-functional requirements, including user interfaces, system features, and performance criteria.
- **Section 4**: Appendices, which may include additional references, diagrams, or supplemental information.

Suggested Reading Sequence

1. Begin with **Section 1** (Introduction) to understand the purpose and scope of the document.
2. Proceed to **Section 2** for a high-level understanding of the product's design and functionality.
3. Read **Section 3** for detailed specifications required for development and testing.
4. Refer to **Section 4** as needed for supplementary details or clarifications.

1.4 Definitions, Acronyms and Abbreviations

Below is a list of terms, abbreviations, and acronyms used throughout this document:

- **API**: Application Programming Interface.
- **GUI**: Graphical User Interface. The visual interface through which users interact with the application.
- **IBW**: Ideal Body Weight. The weight calculated based on the Devine formula, which provides a guideline for a healthy body weight based on height and gender.
- **JDK**: Java Development Kit. A software development kit used to develop Java applications.

- **SRS**: Software Requirements Specification. A document that outlines the requirements and expectations for a software system.
- **UML**: Unified Modeling Language.

1.5 Document Conventions

Font	• Arial, with a font size 12 for body text • Arial, with a font size 18 in bold for headings • Arial, with a font size 14 in bold for sub headings
Spacing	• Single line space with 1.0 inch
Comments	• Italic
Bullets and Numbering	• Lists use bullets or numbers to organize and present information clearly.
Margins	• 1 inch margins on all pages

1.6 References and Acknowledgments

https://github.com/Abdullah-Niaz/BMI_Calculator

https://creately.com/diagram/example/inlvgo102/bmi-calculator-activity-diagram-classic

https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database

Booch, G., Rumbaugh, J., & Jacobson, I. (2005). "The Unified Modeling Language User Guide."

2 Overall Description

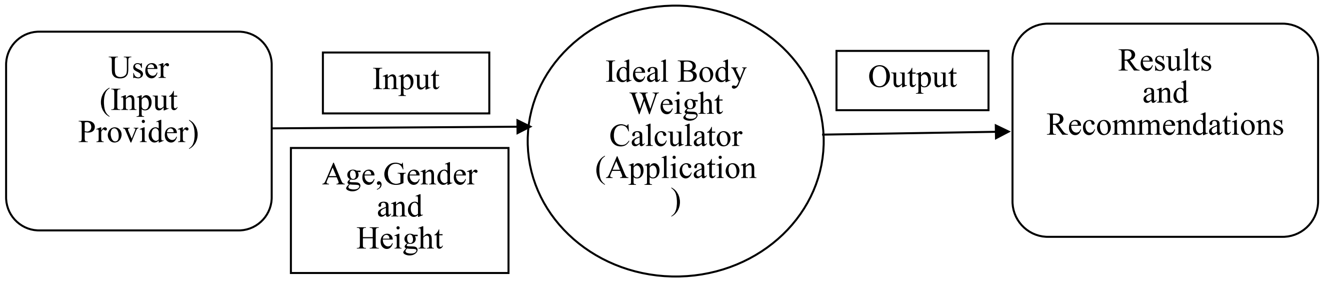
2.1 Product Overview

The **Ideal Body Weight Calculator** is a standalone desktop application that calculates the ideal body weight (IBW) for users based on the Devine formula. This application is designed to be a self-contained system, requiring no external dependencies or integration with larger systems. Users interact directly with the software via a graphical user interface (GUI) to input their height, gender, and actual weight. The application outputs the IBW, evaluates the user's weight status (underweight, healthy, or overweight), and provides actionable recommendations for achieving or maintaining a healthy weight.

The product is intended for individual users, healthcare professionals, and fitness enthusiasts who need an easy-to-use tool for weight assessment. By promoting health awareness and providing practical guidance, this system supports users in adopting healthier lifestyles.

Product Context and Interfaces

The application operates as a new, self-contained product, interacting with users through its GUI. It does not rely on external hardware or software beyond standard desktop systems. Input data (gender, height, and weight) is processed locally to calculate IBW, while all logic and recommendations are hardcoded within the application.


User (Input Provider)
Input
Age,Gender and Height
Ideal Body Weight Calculator (Application)
Output
Results and Recommendations

2.2 Product Functionality

The **Ideal Body Weight Calculator** application provides the following major functions:

- **Input Data**:
    - Allows the user to input their gender, height, and actual weight through the GUI.
- **Ideal Body Weight Calculation**:
    - Calculates the user's ideal body weight (IBW) using the **Devine formula** based on the provided height and gender.
- **Weight Status Assessment**:
    - Determines if the user is **underweight**, **within a healthy weight range**, or **overweight** by comparing their actual weight with the calculated IBW.
- **Recommendations for Weight Management**:
    - Provides actionable suggestions for users who are underweight or overweight to help them achieve a healthy weight.
- **Error Handling**:
    - Displays appropriate error messages for invalid or incomplete inputs, such as non-numeric entries or missing data.
- **User-Friendly Output**:
    - Displays the calculated IBW, weight status, and recommendations clearly and concisely on the GUI.

2.3 Design and Implementation Constraints

1. Hardware Limitations

- **Memory Requirements**: The system should operate within a typical computer's memory and storage capacity, ensuring it runs efficiently on devices with at least **4 GB of RAM** and **1 GHz processing speed**.
- **Input Devices**: The application assumes the availability of a **keyboard** and **mouse** or **touchscreen** for user interaction.
- **Display Resolution**: The application should function properly on screens with a minimum resolution of **1280x720 pixels**.

2. *Software and Tools*

- **Programming Language**: The system must be developed using **Java**, following object-oriented programming principles.
- **Development Environment**: The application will be built using **NetBeans IDE** to streamline development and debugging.
- **Libraries and Frameworks**:
  - **Java Swing**: For building the graphical user interface.
  - **Math Libraries**: To handle calculations required for the Devine formula and other numerical operations.

3. *Design Methodologies*

- **UML Modeling**: Unified Modeling Language (UML) diagrams must be used to represent the system's architecture, including class diagrams, sequence diagrams, and use case diagrams.
  - **Reference**: Booch, G., Rumbaugh, J., & Jacobson, I. (2005). "The Unified Modeling Language User Guide."

4. *Interfaces*

- The system is standalone and does not interface with external systems, databases, or applications in its initial version.
- Future extensions may include integration with mobile apps via **REST APIs** or **Bluetooth protocols**.

2.4 Assumptions and Dependencies

Assumptions

- **User Input Accuracy**: It is assumed that the user will input their height and weight accurately, as the calculations rely heavily on this data. Incorrect input could lead to inaccurate results and recommendations.
- **Height Range**: The application assumes that users are within a reasonable height range (e.g., above 152 cm) as per the Devine formula. Users with extreme heights may not get accurate results.

- **Platform Compatibility**: The application is assumed to run on Windows, macOS, and Linux platforms with Java Runtime Environment (JRE) installed.
- **User Familiarity with Basic Metrics**: Users are assumed to be familiar with basic health metrics, such as their weight and height, and understand what IBW means.

Dependencies

- **Java Development Kit (JDK)**: The project depends on the JDK for development, compiling, and running the Java-based application.
- **Operating System Compatibility**: The application must be compatible with various operating systems (Windows, macOS, and Linux), which could affect deployment and user experience.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The **User Interface (UI)** for the **Ideal Body Weight Calculator** will be designed to ensure ease of use for individuals of all technical backgrounds. The user will interact with the application through a **Graphical User Interface (GUI)**, which will include the following key elements:

1. **Input Fields**:
   - **Gender Selection**: A drop-down menu or radio buttons will allow users to select their gender (Male/Female).
   - **Height Input**: A text field where users can enter their height in centimeters.
   - **Weight Input**: A text field where users can input their current weight in kilograms.
2. **Calculate Button**:
   - A **button** labeled "Calculate" that the user will click to trigger the calculation of their Ideal Body Weight (IBW) based on the inputs provided.
3. **Results Display**:
   - After clicking the "Calculate" button, the results will be displayed in a dedicated section, showing the calculated IBW and weight status (underweight, healthy, overweight).
4. **Recommendation Display**:
   - If the user is underweight or overweight, the system will display personalized recommendations to help them achieve a healthy weight.
5. **Error Messages**:
   - The interface will show error messages (in case of invalid inputs like non-numeric values or missing information) in a clear and user-friendly manner.

User Interaction

- **Touch Screen / Mouse Input**: Users will interact with the application using a mouse or touchscreen (if available), clicking on fields to input their data and pressing the "Calculate" button to trigger the calculations.
- **Text Input and Selection**: Height and weight are manually entered as numerical values, while gender is selected from predefined options (radio buttons or a drop-down menu).
- **Output**: Upon pressing the "Calculate" button, the results and recommendations will appear as text in a designated area on the screen.

3.1.2 Hardware Interfaces

The **Ideal Body Weight Calculator** is a desktop application that requires minimal interaction with hardware components. The following describes the logical and physical characteristics of the hardware interfaces used in the system:

1. Input Devices

- **Keyboard**: Used for entering numeric data such as height (in centimeters) and weight (in kilograms) into the input fields.
- **Mouse/Touchpad**: Used for selecting options (e.g., gender), interacting with buttons (e.g., "Calculate"), and navigating the user interface.
- **Touchscreen (Optional)**: If available, users can interact with the application through a touchscreen interface for an intuitive experience.

2. Output Devices

- **Monitor/Display**:
  - Displays the graphical user interface (GUI), including input fields, calculation results, and personalized recommendations.
  - Supports resolutions of at least **1280x720 pixels** to ensure proper layout and readability.

3. Processor and Memory

- **Central Processing Unit (CPU)**:
  - The application relies on the CPU to perform all calculations, including height-to-inches conversion, IBW calculation using the Devine formula, and weight status determination.
  - Minimum requirement: **1 GHz processor**.
- **Random Access Memory (RAM)**:
  - Memory is required to load the Java application into memory and handle runtime operations.
  - Minimum requirement: **4 GB of RAM**.

3.1.3 Software Interfaces

Mobile App Integration (Optional Future Feature):

- **Communication Protocol**: The mobile app could send user data (such as height, weight, and gender) to the desktop application for calculation via a **network-based communication protocol** (e.g., REST API or Bluetooth).
- **Data Exchange**: The mobile app could act as a **frontend interface**, gathering user input and sending it to the desktop application, which would process the data and send back the results (e.g., IBW, weight status, and recommendations).
- **Command Execution**: The mobile app could trigger specific actions (such as triggering calculations or retrieving results) by sending predefined commands to the desktop application.

3.2 Functional Requirements

The **Ideal Body Weight Calculator** must perform the following functions to meet its intended behavior and fulfill the requirements outlined in Section 2.2:

1. User Input Handling

- **Gender Input**: The system shall provide an option for the user to select their gender (Male/Female) using a radio button or dropdown menu.
- **Height Input**: The system shall allow the user to input their height in centimeters via a text field.
- **Weight Input**: The system shall allow the user to input their weight in kilograms via a text field.

2. Calculation of Ideal Body Weight (IBW)

- **Ideal Weight Calculation**: The system shall calculate the user's ideal body weight (IBW) using the **Devine formula**, which is:
  - For males: IBW = 50 + 2.3 * (Height in inches - 60)
  - For females: IBW = 45.5 + 2.3 * (Height in inches - 60)
- The system must convert height from centimeters to inches when performing the calculation.
- The system shall ensure that the IBW calculation is accurate and updated whenever the user changes any input value.

3. Weight Status Assessment

- **Comparison with IBW**: The system shall compare the user's actual weight to the calculated IBW and assess their weight status:
  - **Underweight**: If the actual weight is below 90% of the IBW.
  - **Healthy Weight**: If the actual weight is within 90% to 110% of the IBW.
  - **Overweight**: If the actual weight exceeds 110% of the IBW.

4. Display Results

- **IBW Display**: The system shall display the calculated IBW after the user clicks the "Calculate" button.
- **Weight Status Display**: The system shall display the weight status (underweight, healthy weight, or overweight) based on the comparison of the actual weight with the IBW.

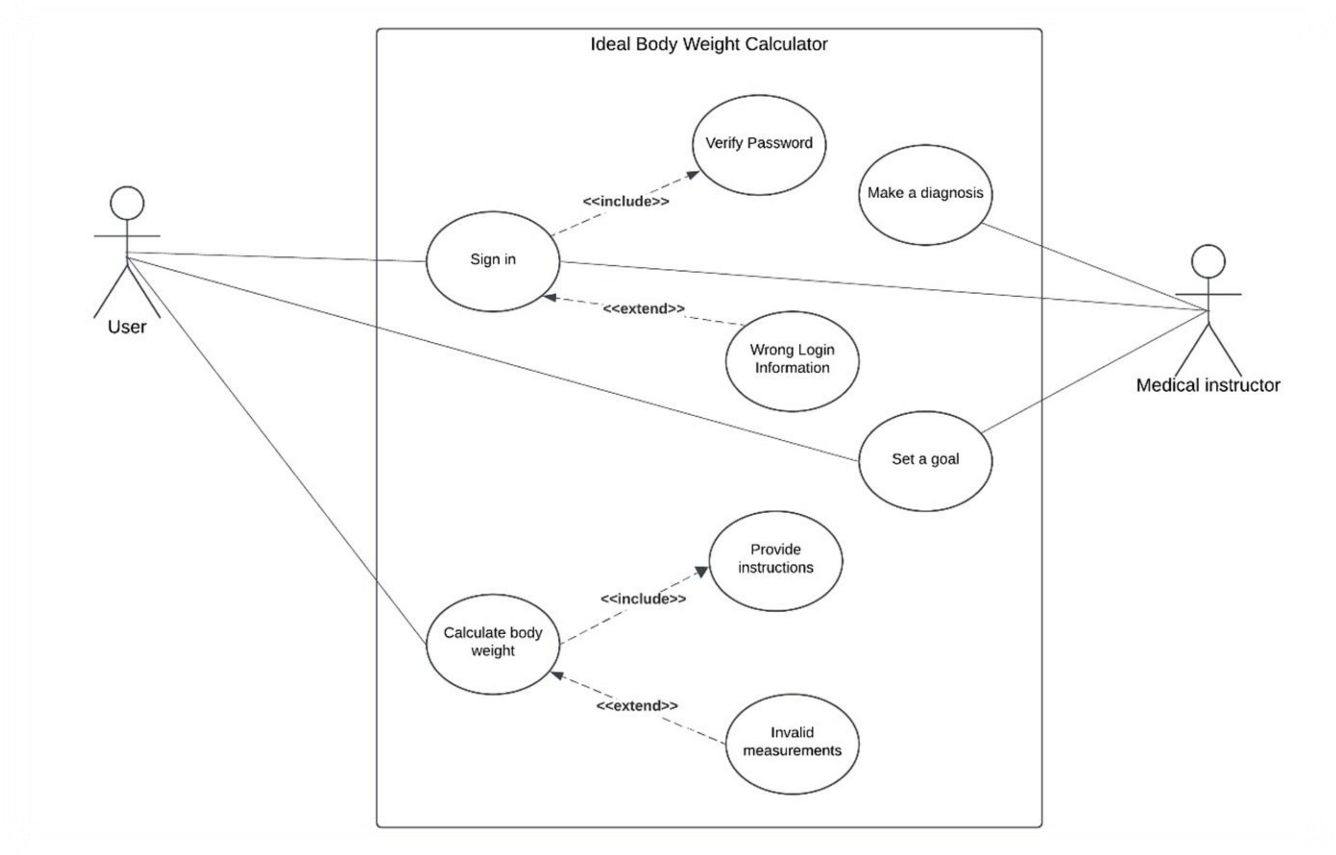
5. Personalized Recommendations

- **Underweight Recommendation**: If the user is classified as underweight, the system shall provide personalized suggestions to help them gain weight healthily. For example:

  - "You are underweight. Consider consulting a healthcare professional for personalized weight-gain strategies."

- **Overweight Recommendation**: If the user is classified as overweight, the system shall provide personalized suggestions to help them lose weight healthily. For example:
  - "You are overweight. Consider consulting a healthcare professional for personalized weight-loss strategies."
- **Healthy Weight**: If the user is within the healthy weight range, the system shall provide a message encouraging the user to maintain their current weight and lifestyle.

3.3 Use Case



Author

23ug1-0152_Akindu Randira

23ug1-0162_Isuru Balasooriya

23ug1-0015_Malindu Kanishka

Purpose

To allow users to input their height and gender into the system for IBW calculation.

Requirements Traceability

- **FR1**: Accept user input for height, weight, and gender.
- **FR2**: Validate input for completeness and correctness.

Priority

High

Preconditions

- The system must be running and accessible.
- The user must have basic knowledge of their height and weight.

Postconditions

- The system successfully receives and validates the user's input data.
- The input data is ready for processing in the IBW calculation module.

Actors

- **User**: Inputs the required personal details.

Flow of Events

1. **Basic Flow**:
    1. The user launches the application.
    2. The system displays input fields for height, weight, and gender.
    3. The user enters their data and clicks the "Submit" button.
    4. The system validates the input for accuracy and completeness.

5. If the data is valid, the system confirms successful input and proceeds to the calculation step.

2. **Alternative Flow**:
    - **AF1**: If the user leaves a required field empty, the system displays an error message and prompts for completion.
3. **Exceptions**:
    - **EX1**: If invalid characters are entered (e.g., letters in a numeric field), the system highlights the field and displays an error message.

Includes

- Validation Module: Ensures the input data is correct.

Notes/Issues

- Consider adding input hints (e.g., acceptable range of values) to guide users.
- Ensure accessibility for users with disabilities (e.g., large font, voice input options).

4 Other Non-functional Requirements

4.1 Performance Requirements

1. **Responsiveness (P1)**:
   - The system must provide the calculated Ideal Body Weight (IBW) and recommendations within **1 second** after the user presses the "Calculate" button.
   - Rationale: Ensures a smooth and fast user experience, avoiding delays in computation.
2. **Scalability (P2)**:
   - The system should support concurrent user operations if adapted for web-based or mobile versions, handling at least **100 simultaneous requests** without performance degradation.
   - Rationale: Ensures future adaptability to multi-user scenarios.
3. **Data Input Handling (P3)**:
   - The system must validate and process input data in less than **200 milliseconds**, including checks for invalid or missing entries.
   - Rationale: Prevents delays caused by invalid inputs.

4.2 Safety and Security Requirements

1. **User Data Security (S1)**:
   - If the system saves user data (e.g., personal height/weight records), it must encrypt the data using a minimum of **AES-128 encryption**.
   - Rationale: Prevent unauthorized access to sensitive user data.
2. **Secure Mobile Communication (S2)**:
   - All communication between the desktop application and the mobile app must be secured via **HTTPS** with SSL/TLS encryption.
   - Rationale: Prevent interception or manipulation of data during transmission.

3. **Input Validation (S3)**:
    - The system must prevent invalid or harmful inputs, such as excessively large or negative values, to ensure calculations remain accurate and reliable.
    - Rationale: Avoids system crashes or incorrect outputs due to input errors.
4. **Data Privacy (S4)**:
    - User data collected by the system will not be shared or used for purposes other than the primary functionality without user consent.
    - Rationale: Ensures compliance with privacy standards (e.g., GDPR).

4.3 Software Quality Attributes

1. Reliability

- **Requirement**: The system should achieve **99.9% uptime**, ensuring it is operational under normal conditions.
- **Implementation Plan**:
    - Utilize thorough exception handling mechanisms in the code.
    - Conduct unit and integration tests to identify and fix potential issues during development.

2. Usability

- **Requirement**: The interface should be user-friendly, with an intuitive layout and clear instructions, ensuring users with minimal technical expertise can operate it.
- **Implementation Plan**:
    - Conduct usability testing with target users to refine the GUI.
    - Use consistent and meaningful labels for all buttons, input fields, and error messages.

3. Maintainability

- **Requirement**: The software should allow for future updates, such as the integration of new sensors or additional calculation methods.

- **Implementation Plan**:
    - Follow a modular design approach, separating calculation logic, input validation, and the GUI into distinct components.
    - Maintain clear and comprehensive documentation of the codebase.

4. Adaptability

- **Requirement**: The system must be adaptable for various input units (metric or imperial) and future expansions, such as integrating with wearables or IoT devices.
- **Implementation Plan**:
    - Design the input and calculation modules to support unit conversion.
    - Use a standardized API framework for integrating additional features.

5. Testability

- **Requirement**: All system components must be testable individually to ensure correctness before deployment.
- **Implementation Plan**:
    - Write automated unit tests for each module (e.g., input validation, calculation, recommendations).
    - Implement logging for tracking errors and debugging.

Appendix – Group Log

All three members are equally supported and contributed effectively to the successful completion of the SRS document. Each member played a vital role in ensuring its accuracy and completeness.