Abbottabad University of Science & Technology

SOFTWARE REQUIREMENTS SPECIFICATION

(SRS DOCUMENT)

For

< Binary Tree Visualizer>

By

Bibi Hajra **14640**

Program **BSCS (3A)**

Supervisor

(Sir Jamal Abdul Ahad)

Table of Content

Introduction

This section provides a high-level overview of the project and sets the stage for the rest of the report.

1.1 Purpose

- Explains the primary goal of the Binary Search Tree Visualizer: to help users visualize and understand how BSTs function (e.g., node insertion, deletion, traversal, etc.).

1.2 Document Conventions

- Lists any symbols, terms, or formatting used throughout the document for clarity and consistency.

1.3 Project Scope

- **Scope Definition**: Highlights the boundaries of the project, such as visualizing BST operations and user interaction with the visualizer.
- **Core Features**: Defines key functionalities like adding/deleting nodes, tree traversal visualization, and search path highlighting.
- **Subsequent Releases**: Discusses features planned for future updates, like balancing for AVL or Red-Black trees.
- **Alignment with User and Business Goals**: Aligns project objectives with the needs of users and stakeholders, emphasizing usability and educational value.

1.4 References

https://colab.research.google.com/drive/1yKgrzvR60HPDLySeRa_hAEprxJTK_flr?usp=sharing

2. Overall Description

Describes the product's context, audience, environment, and constraints.

2.1 Product Perspective

- **Product Context**: Positions the project within its domain (e.g., a tool for learning data structures).
- **Product Origin**: Explains the motivation or background that led to creating the visualizer.

- **Product Relationship to Existing Systems**: Discusses how the visualizer interacts with or complements other systems (e.g., education platforms).
- **Product Ecosystem**: Maps the visualizer's role in a larger system, including APIs or integrations.

2.2 User Classes and Characteristics

- **Students and Educators**: Users learning about or teaching data structures who need a clear visualization of BST operations.
- **Tech Enthusiasts**: Users interested in computer science concepts and eager to explore practical implementations.
- **Favored User Class**: The primary audience for whom the visualizer is optimized (e.g., students).
- **Alignment with User Needs**: Matches the features of the visualizer to specific user requirements, like simplicity and interactivity.

2.3 Operating Environment

- **Hardware Platform**: Describes the hardware requirements, such as desktops, laptops, or tablets.
- **Operating Systems and Versions**: Lists supported platforms, e.g., Windows, macOS, Linux, or web browsers.

2.4 Design and Implementation Constraints

- **Database Technology**: Specifies storage or data handling methods if the project saves tree data.
- **Third-Party Integrations**: Mentions any libraries or APIs used for visualization (e.g., D3.js, React).
- **User Interface Design**: Describes the design philosophy for making the interface intuitive and user-friendly.

2.5 Assumptions and Dependencies

- **Assumptions**: Lists assumptions about the environment or users (e.g., users have basic knowledge of data structures).
- **Dependencies**: Details external factors, like dependencies on frameworks or runtime environments.

---

3. System Features

Focuses on the main functionality of the visualizer.

Key Features:

- **Node Addition and Deletion**: Visual representation of inserting or removing nodes in the BST.
- **Visual Insertion and Balancing**: Displays how nodes are placed in the tree and balanced if needed.
- **Tree Traversals**: Highlights in-order, pre-order, and post-order traversals with step-by-step visualization.
- **Highlight Search Paths**: Shows the path taken to find a specific node in the tree.

---

4. External Interface Requirements

Defines how users and other systems interact with the visualizer.

4.1 User Interfaces

- **Design Standards and Guidelines**: Ensures the visualizer is visually consistent and accessible.
- **Screen Layout and Resolution**: Optimized for different screen sizes and resolutions.
- **Standard Interface Elements**: Buttons, sliders, and dropdowns are used for intuitive interactions.

4.2 Software Interfaces

- Lists APIs, libraries, or other software used in the project.

4.3 Hardware Interfaces

- Defines supported devices and their specifications (e.g., compatibility with desktops and tablets).

---

5. Quality Attributes

Highlights non-functional requirements that ensure the tool's quality.

5.1 Performance

The visualizer should handle large trees without noticeable lag.

5.2 Reliability

- Ensures consistent behavior without crashes or incorrect visualizations.

5.3 Usability

- Prioritizes an intuitive interface so that even non-technical users can understand the visualizations.

5.4 Security

- Ensures the tool is safe to use, especially for web-based versions (e.g., no unauthorized access or vulnerabilities).

5.5 Maintainability

- The codebase should be modular and easy to update or expand in future releases.

---

6. Appendix B: Analysis Model

Includes any diagrams, models, or charts that explain the project in more depth. Examples:

- **Flowcharts**: Show how data moves through the system.
- **UML Diagrams**: Illustrate system architecture and class relationships.