DOCUMENTATION

1. Introduction

1.1 Project Title

Interactive, Data-Driven Mindmap UI named **Mindmaker.**

1.2 Project Objective

The objective of this project is to design and implement an **interactive mindmap visualization system** that represents hierarchical data in a clear, intuitive, and scalable manner. The application emphasizes **data-driven rendering**, **rich user interaction**, and **clean frontend architecture**, closely resembling the behavior and experience of a reference mindmap UI.

The project demonstrates the ability to:

- Build complex interactive user interfaces
- Visualize hierarchical data dynamically
- Maintain clean separation between data, logic, and presentation
- Handle real-world interaction challenges in frontend development

---

2. Problem Statement

The task is to implement a **seamless interactive Mindmap UI** that:

- Visualizes hierarchical relationships (parent → child)
- Is generated entirely from structured data (JSON)
- Supports user interaction such as:
    - Hovering
    - Clicking
    - Expanding and collapsing nodes
    - Editing node information
    - Navigating and exploring large structures
- Displays contextual information both inline and in a sidebar

The system must be **data-driven**, meaning that **any change in the data file should automatically reflect in the UI** without requiring changes to the rendering logic.

---

3. Technologies Used

3.1 Frontend Technologies

React (Vite)

React is used as the core framework for building the user interface. Its component-based architecture enables modular design, reusability, and efficient updates through its declarative rendering model.
Vite is used as the build tool for fast development and optimized builds.

JavaScript (ES6+)

Modern JavaScript features such as arrow functions, destructuring, and modules are used to keep the code concise and readable.

SVG (Scalable Vector Graphics)

SVG is used to render nodes and edges of the mindmap. SVG allows precise control over positioning, scaling, and interaction, making it ideal for graph-based visualizations.

Tailwind CSS

Tailwind CSS is used for styling the UI. It enables rapid development using utility classes, supports dark/light themes, and ensures consistent spacing and typography across the application.

---

4. Libraries Used and Rationale

4.1 React Context API

The Context API is used for centralized state management. It stores:

- The mindmap tree structure
- Selected node information
- Expanded/collapsed state
- Zoom and viewport controls
- Inline edits made through the UI

This avoids excessive prop drilling and keeps application logic centralized and predictable.

4.2 html2canvas

The html2canvas library is used to export the mindmap as a PNG image.
This library was chosen because it reliably captures **SVG content with embedded HTML (foreignObject)**, which many SVG export libraries fail to handle correctly.

---

5. Overall Architecture and Design Approach

5.1 Architectural Overview

The project follows a **layered frontend architecture**:

Data Layer → JSON (mindmap.json)

Logic Layer → Context (MindMapContext, ThemeContext)

Utility Layer → Layout, Fit-to-View, Export helpers

Presentation Layer→ React Components

Each layer has a **single responsibility**, ensuring maintainability and scalability.

5.2 Folder Structure

```
src/
 ├── components/
 │   ├── MindMap.jsx       // SVG renderer
 │   ├── MindNode.jsx      // Individual node UI
 │   ├── Sidebar.jsx       // Details + editing panel
 │   └── Toolbar.jsx       // Controls (expand, zoom, reset, export)
 │
 ├── context/
 │   ├── MindMapContext.jsx // Core application state
 │   └── ThemeContext.jsx  // Dark / light theme handling
 │
 ├── data/
 │   └── mindmap.json      // Hierarchical data source
```

```
│
├── utils/
│   ├── layout.js        // Node positioning logic
│   ├── fitView.js       // Centering & scaling logic
│   ├── exportPNG.js     // PNG export helper
│   └── exportJSON.js    // JSON export helper
```

---

6. Data-Driven Rendering Flow

6.1 Data Source

The entire mindmap is defined in a structured JSON file (mindmap.json).
Each node contains:

- A unique identifier
- A title
- A summary
- Optional metadata
- A list of child nodes

6.2 Application Initialization

1. The JSON file is imported into the application.
2. The data is **deep-cloned** to avoid mutating the original source.
3. The root node is initialized as expanded while child nodes are collapsed by default.

6.3 Layout Calculation

A layout utility function computes:

- X and Y positions for each node
- Depth level for visual styling
- Edge connections between parent and child nodes

This logic is isolated from UI components, making it reusable and testable.

6.4 Rendering Pipeline

1. The layout output is converted into a graph structure.
2. MindMap.jsx renders:
    - SVG edges
    - SVG nodes
3. MindNode.jsx renders each node with:
    - Dynamic size based on content
    - Color based on depth
    - Click and hover handlers

6.5 Reactive Updates

Whenever:

- A node is expanded or collapsed
- A node is edited
- The zoom or viewport changes

...the state in context updates and React automatically re-renders the affected components.

---

7. Functional Requirements Implementation

7.1 Mindmap Visualization

- Graph structure is rendered dynamically
- Parent → child relationships are visually clear
- Layout remains readable for varying depths
- SVG scaling ensures responsiveness

7.2 Interactive Features

Click Interaction

- Clicking a node selects it
- Clicking the same node again collapses or expands its children
- Expand All / Collapse All buttons are provided

Hover Interaction

- Hovering over a node shows a tooltip with a short summary
- Tooltips do not affect layout or interaction

Fit to View / Reset

- Reset fits the current graph into view
- Reset does not alter expanded/collapsed state
- Zoom level resets smoothly

7.3 Sidebar and Inline Editing

- Selecting a node displays its full summary in the sidebar
- Sidebar shows node title, summary, and metadata
- An **Edit** button allows inline editing
- Controlled inputs are used
- Saving updates application state only (no backend)
- Cancel discards changes

---

8. Data Display Strategy

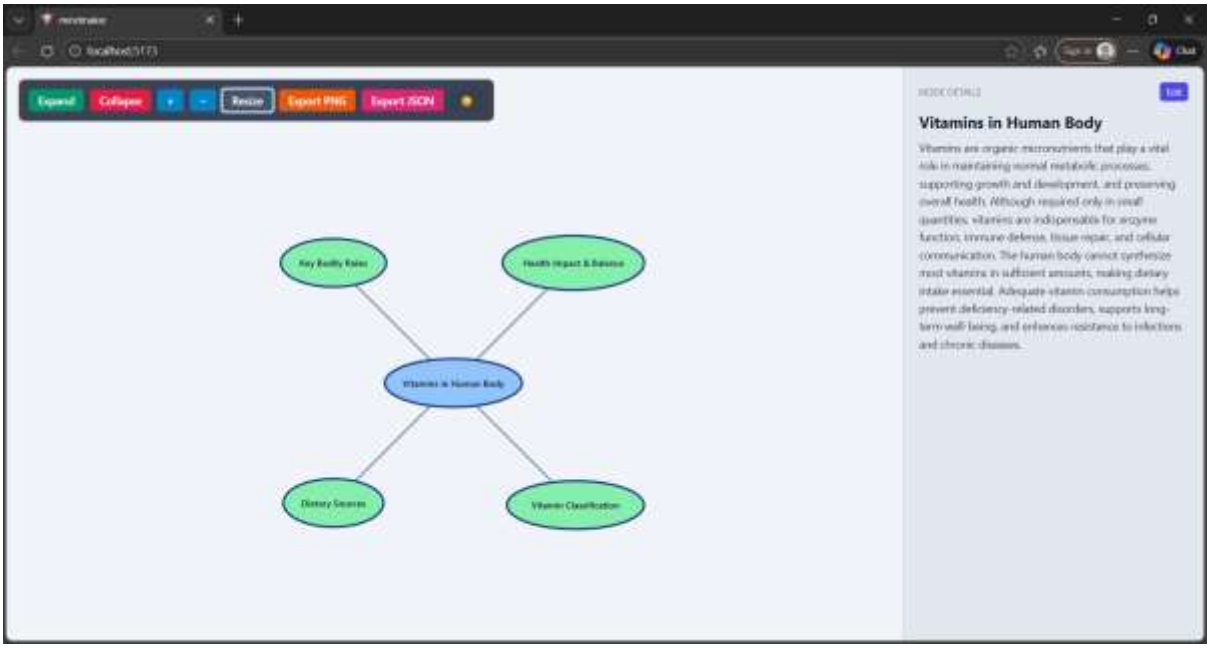
Location	**Data Shown**
Hover Tooltip	Short summary
Sidebar	Full summary, title, editable fields

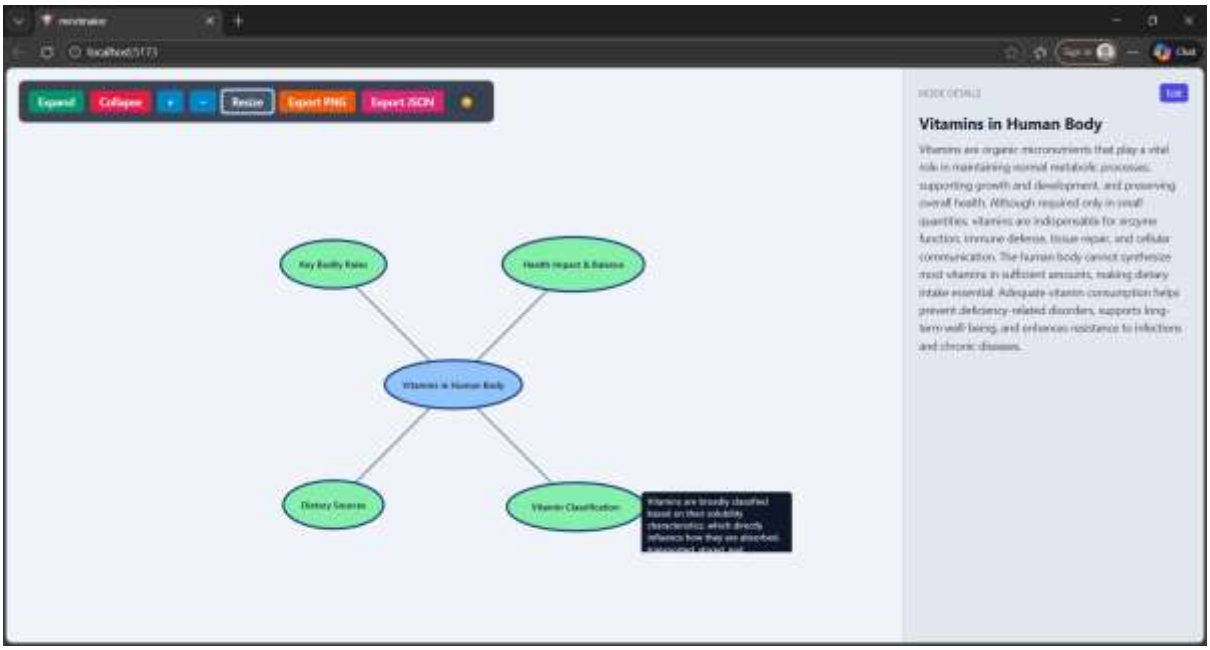
This separation avoids clutter while ensuring accessibility of information.

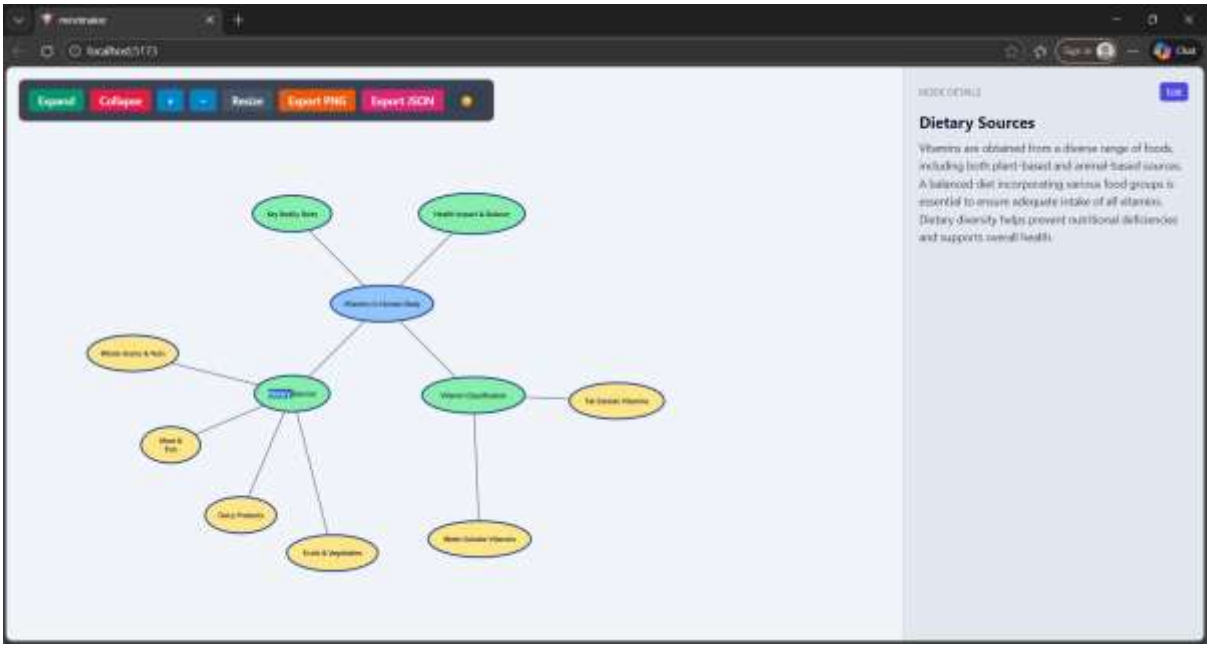
---

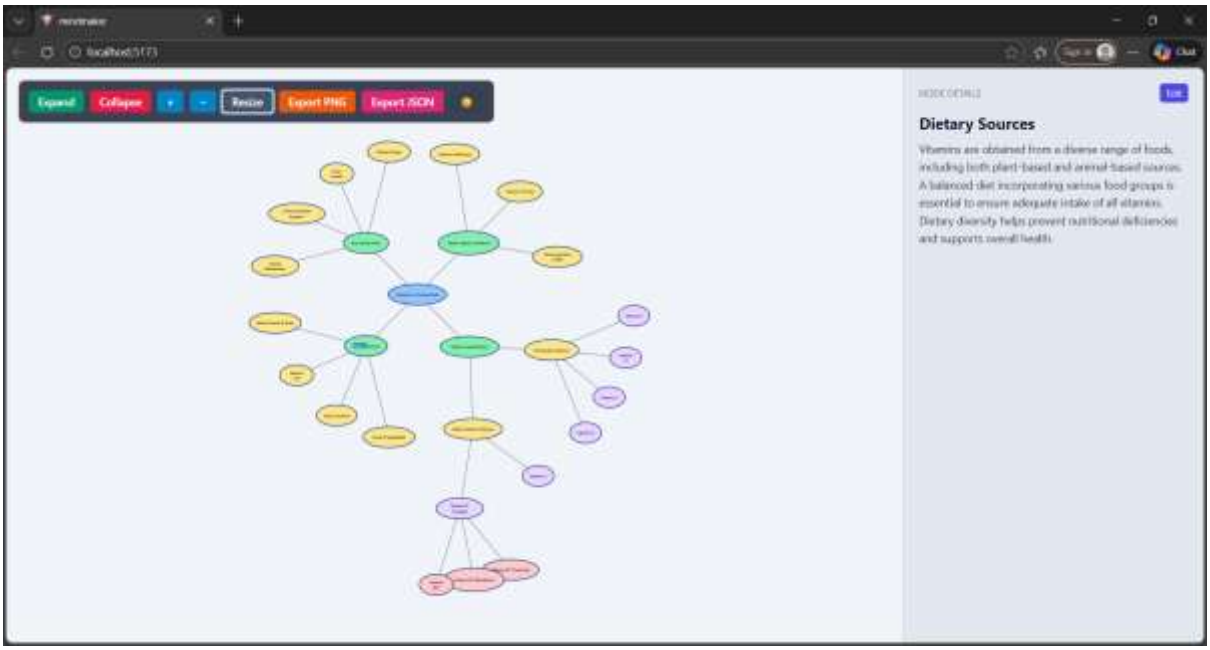
9. Screenshots (Mandatory)

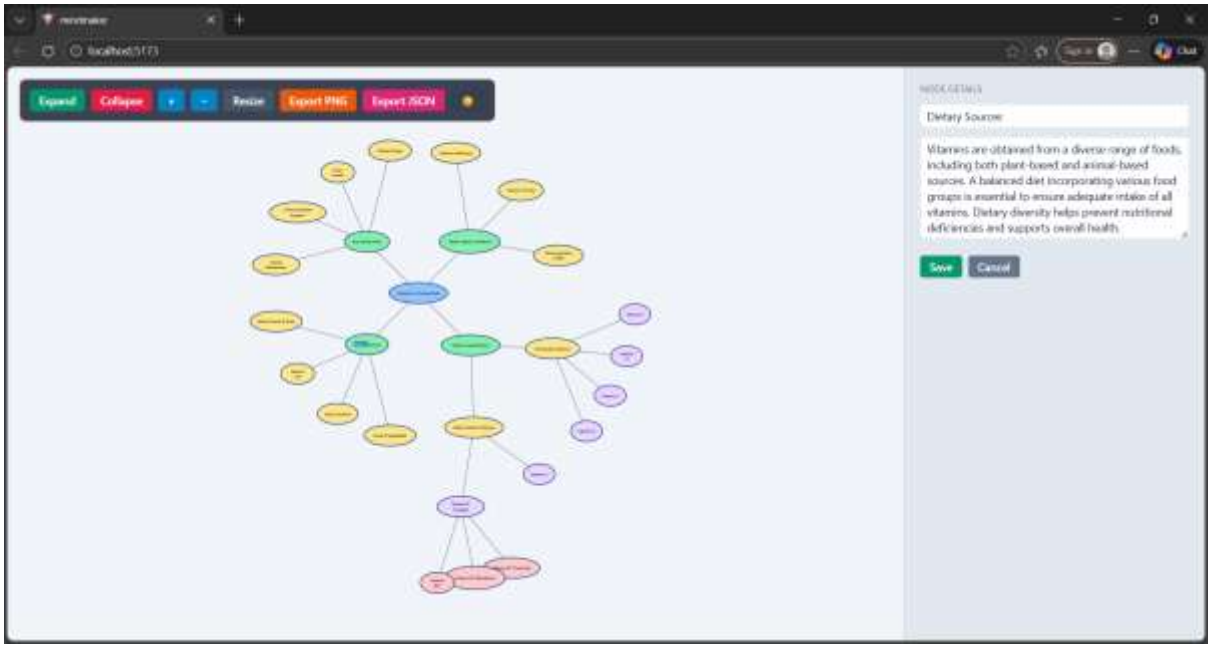

Expand
Collapse
Resize
Export PNG
Export JSON
Vitamins in Human Body
Vitamins in Human Body

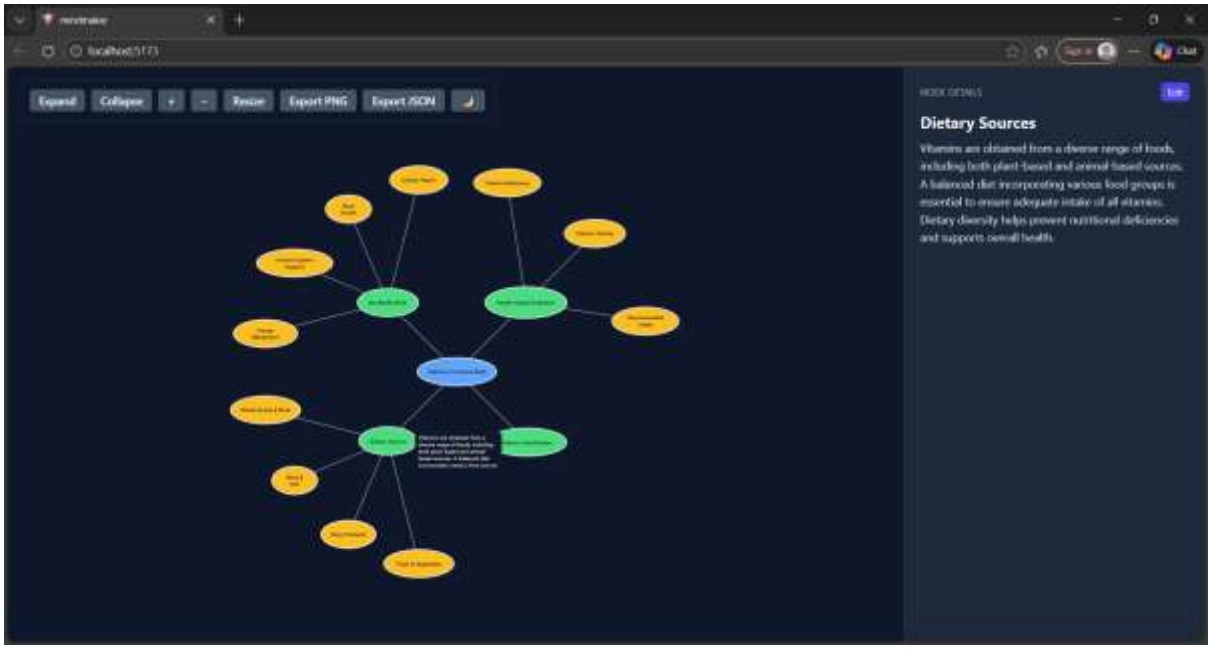

Expand
Collapse
Resize
Export PNG
Export JSON
Vitamins in Human Body
Vitamins in Human Body
Vitamins are organic micronutrients that play a vital role in maintaining normal metabolic processes, supporting growth and development, and preserving overall health. Although required only in small quantities, vitamins are indispensable for enzyme function, immune defense, tissue repair, and cellular communication. The human body cannot synthesize most vitamins in sufficient amounts, making dietary intake essential. Adequate vitamin consumption helps prevent deficiency-related disorders, supports long-term well-being, and enhances resistance to infections and chronic diseases.


Expand
Collapse
Resize
Export PNG
Export JSON
Vitamins in Human Body
Vitamins in Human Body
Vitamins are organic micronutrients that play a vital role in maintaining normal metabolic processes, supporting growth and development, and preserving overall health. Although required only in small quantities, vitamins are indispensable for enzyme function, immune defense, tissue repair, and cellular communication. The human body cannot synthesize most vitamins in sufficient amounts, making dietary intake essential. Adequate vitamin consumption helps prevent deficiency-related disorders, supports long-term well-being, and enhances resistance to infections and chronic diseases.


Expand
Collapse
Resize
Export PNG
Export JSON
Dietary Sources
Vitamins are obtained from a diverse range of foods, including both plant-based and animal-based sources. A balanced diet incorporating various food groups is essential to ensure adequate intake of all vitamins. Dietary diversity helps prevent nutritional deficiencies and supports overall health.


Expand
Collapse
Resize
Export PNG
Export JSON
Dietary Sources
Vitamins are obtained from a diverse range of foods, including both plant-based and animal-based sources. A balanced diet incorporating various food groups is essential to ensure adequate intake of all vitamins. Dietary diversity helps prevent nutritional deficiencies and supports overall health.





---

10. Demo Video

https://drive.google.com/file/d/14JeRRvAvi0K6HAtAH15dZpQMy6GmAUd6/view?usp=sharing

Video demonstrates:

- Initial rendering
- Node hover and selection
- Expand/collapse behavior

- Sidebar editing
- Reset and export functionality

---

11. Evaluation Criteria Mapping

Correctness

All required features are implemented and function as expected.

Data-Driven Design

The entire visualization is generated from structured JSON data.

UI / UX Quality

The interface is intuitive, responsive, and easy to explore.

Code Quality

The codebase is modular, readable, and maintainable.

Problem-Solving Approach

The project demonstrates thoughtful handling of SVG rendering, interaction state, and export limitations.

---

12. Assumptions and Intentional Decisions

- No backend persistence was implemented by design.
- Inline editing is limited to state updates.
- Hover highlighting was kept minimal to avoid visual overload.
- Focus was on correctness and clarity rather than pixel-perfect replication.

---

13. Conclusion

This project successfully implements a **robust, interactive, and data-driven Mindmap UI** that meets all functional requirements. The architecture is clean, extensible, and well-suited for future enhancements such as persistence, collaboration, or advanced analytics.
