NATIONAL TEACHERS COLLEGE
School of Arts, Sciences and Technology
629 J. Nepomuceno St., Quiapo, Manila

HealthHub: A Web-Based DSA Solution for

UN SDG 3 (Good Health & Well-Being)

A Software Design and Analysis Document

Presented to the Faculty of

School of Arts, Sciences and Technology of

National Teachers College

In Partial Fulfillment

of the Requirements for

Data Structures and Algorithms

by

Aquino, Ram B.

Baligat, Jenna Alyssa Q.

Casison, Rhod Stephen S.

Habitan, Mark Eugenio T.

Monares, Carl Dave C.

December 2025

NATIONAL TEACHERS COLLEGE
School of Arts, Sciences and Technology
629 J. Nepomuceno St., Quiapo, Manila

TABLE OF CONTENTS

INTRODUCTION

1.1 Project Overview & UN SDG Target

HealthHub is an interactive web-based educational platform designed to address UN Sustainable Development Goal (SDG) 3: Good Health and Well-Being, specifically Target 3.7, which mandates universal access to sexual and reproductive health-care services. In many communities, stigma and lack of privacy prevent individuals from seeking professional advice. HealthHub combats the rising rates of HIV and teenage pregnancies by bridging this gap through accessible, anonymous technology.

The system integrates four core modules to support this mission: an Intelligent Chatbot for instant queries, a Diagnostic Engine (Self-Assessment) for symptom triage, a Community Forum for peer support, and an Information Modal for educational and awareness content. By providing a safe digital environment, HealthHub empowers users to make informed health decisions without fear of judgment.

1.2 Problem Statement

The United Nations Sustainable Development Goal (SDG) 3, specifically Target 3.7, mandates that all individuals should have universal access to sexual and reproductive health-care services, including family planning, information, and education. Despite this, the Philippines is currently facing a "reproductive health crisis." According to recent data from ABS-CBN News

(2025), the country has seen a critical surge in HIV cases and teenage pregnancies. The current infrastructure for health education relies heavily on traditional, face-to-face counseling or static materials. These methods fail to address the needs of the digital-native youth population, who often fear the stigma associated with visiting physical clinics or lack the patience to navigate complex medical websites. There is a systemic disconnect between the availability of medical professionals and the accessibility of their advice to the youth.

This project, HealthHub, aims to bridge this gap by developing a web-based educational utility powered by Data Structures and Algorithms (DSA). Specifically, the project aims to:

- Use a Decision Tree algorithm to provide instant, logic-based risk assessments (e.g., "Do I need a test?").
- Deploy a Linear Search chatbot to provide instant, anonymous answers to sensitive queries.
- Implement Sorting Algorithms to curate peer-to-peer advice in a forum setting, ensuring high-quality information is easily accessible.

By automating the "first line of defense," HealthHub seeks to reduce the barrier to entry for reproductive healthcare, aligning technical DSA implementation with the humanitarian goals of SDG 3.7.

REQUIREMENTS & ANALYSIS

2.1 Functional Requirements and Non-Functional Requirements

Table 2.1.1: Functional Requirements (FR)

ID	Requirement	Alignment
FR1	**Chatbot (HealthBot)** **1.1** The system accepts user queries and normalizes input text. **1.2** The system utilizes a **Linear Search** algorithm to iterate through a predefined canned Knowledge Base (Array of Objects) to find matching Regular Expression patterns . **1.3** The system simulates a "thinking" latency (asynchronous callback) before rendering the response.	**Prelim / Midterm DSA** **(Arrays & Iteration)**
FR2	**Diagnostic Engine (Self-Assessment)** **2.1** The system implements a **Non-Linear Data Structure (Decision Tree)** to map health scenarios (e.g., Contraception, STI). **2.2** The system maintains a "State Pointer" (treeState) to track the currentNode and currentScenario during traversal . **2.3** Upon reaching a "Leaf Node" (node with no children), the system terminates the flow and displays the result.	**Core DSA** **(Trees & Graphs)**
FR3	**Community Forum**	**Finals DSA**

	3.1 The system uses a **Dynamic Array** of objects to store post data (id, author, content, likes). **3.2** The system implements a **Search Algorithm (Linear Search)** to locate specific post IDs when a user clicks "Like" or "Delete". **3.3** The system implements a **Sorting Algorithm (Merge/Quick Sort logic)** to organize posts by Engagement (O(N log N)) or Recency.	**(Sorting)**
FR4	**Information Modal** **4.1** The system utilizes a **Hash Map (Dictionary)** structure (infoDetails) to store large content blocks mapped to unique ID keys. **4.2** The system uses **Direct Addressing** to retrieve and display content in O(1) time without looping.	**Finals DSA** **(Hash Map)**
FR5	**5.1** The system must utilize a responsive layout with a "Glassmorphism" aesthetic (translucent backgrounds, soft shadows) to ensure readability and visual comfort. **5.2** The system shall implement dynamic visual feedback for all user interactions, including "typing..." indicators during chatbot latency and hover states for buttons. **5.3** The system must feature a persistent, sticky navigation header	**UI / UX**

to allow seamless switching between modules (Home, Learn, Self-Assessment, Forum) without losing context. **5.4** The system shall employ a calming color palette (e.g., gradients of purple and pink) and accessible typography (Quicksand font) to maintain a non-threatening, supportive atmosphere.	

Table 2.1.2: Non-Functional Requirements (NFR)

ID	Requirement	Metric
NFR1	**Performance & Complexity:** The system must execute core DSA operations efficiently within the client browser without UI freezing or lag.	**1.1** UI updates, chat messages appear, and modal content retrieval via Hash Map (O(1)) appear instantly (within 200 ms). **1.2** Sorting + rendering of 50+ forum posts should complete in <1 second on standard. **1.3** Decision-tree step (single yes/no) should be O(1) and execute in <50 ms.
NFR2	**Privacy:** User-generated content must	**2.1** All user data stored only in

	remain local to the user's device to ensure privacy and compliance with ethical standards for sensitive health data.	browser localStorage. **2.2** All users that post in the forum are anonymous, since in the system, author name in post is optional. **2.3** Users must be able to delete individual posts and clear all posts (immediate effect).

2.2 Data Requirements

Table 2.2.1: Data Requirements and their Structure and Size

Data Requirement	Description of Structure and Size
Input Data	**Forum Post Input:** A user-provided post requires an optional **author** name (max 60 characters) and the **content** of the message (max 2000 characters). **Chatbot Input:** Users entered text query. **Self-Assessment Input:**: Binary user choices (Yes/No) to traverse the decision tree. **Info Modal Input:** User interaction (click) carrying a unique string ID (Key) (e.g., 'modal-101') to retrieve specific topic details.

Stored Data	**Forum Posts**: Stored as an array of JavaScript objects in **localStorage**. Each object contains **id, author, content, likes** (integer count), and **time** (ISO date string).
	Chat History: Stored as an array of messages in **localStorage**, each containing the message **text** and **who** (a 'user' or 'bot').
	Info Modal Data: Static content stored as a Key-Value Dictionary (Hash Map). Keys are unique IDs and values are the HTML content strings. This structure allows O(1) access.
	Core System Data: Static application data (decision tree structure, canned chatbot responses, modal information content) is hardcoded in JavaScript.
	Size: Overall data size is constrained by the browser's **localStorage** limit (typically 5mb to 10mb), which is sufficient for text-based posts and chat history.

2.3 Complexity Analysis (Justifying Big O Notation)

Table 2.3.1: Expected Time/Space Complexity of the Core Algorithm/s

Core Algorithm/s	Complexity	Justification (Big O Notation)
Chatbot Pattern	**Time:** O(N)	The algorithm iterates through the canned

Matching (Linear Search)	**Space**: O(N)	array one by one until a regex match is found. Space is O(N) to store the array of patterns.
Forum Sorting (Merge/Quick Sort Logic)	**Time:** O(N log N) **Space:** O(N)	The system uses the browser's optimized sort implementation (typically MergeSort or QuickSort). This is necessary to handle the ordering of posts by "Likes" efficiently.
Diagnostic Tree Traversal (Graph Traversal)	**Time**: O(H) **Space:** O(N)	Time is proportional to the Height (H) of the tree (the number of questions asked). Storage requires O(N) to hold the decisionData Nested Dictionary.
Modal Retrieval (Direct Access)	**Time:** O(1) **Space:** O(N)	The system uses a Hash Map where the key (topic ID) directly addresses the memory location of the content, requiring no search loop.

DESIGN SPECIFICATION

3.1 Core Data Structures Used (Prelim, Midterm, and Finals)

To achieve efficient performance and logical organization of data, the system implements specific Data Structures and Algorithms (DSA) corresponding to the academic curriculum phases:

1. **Arrays & Linear Search (Prelim Scope):**

   ○ **Usage:** Used in the **Chatbot Module** (canned).

   ○ **Description:** A structural array of objects stores the "Knowledge Base" (Regex Pattern-Response pairs). The system utilizes a Linear Search algorithm (O(N)) to iterate through this collection to find matches for user queries.

   ○ **Justification:** For a fixed set of keywords, an array offers simple implementation overhead and allows for sequential checking of Regular Expressions, which is not possible with Hash Maps.

2. **Dynamic Arrays & User-Defined Objects (Midterm Scope):**

   ○ **Usage:** Used in the **Community Forum** (posts).

   ○ **Description:** Forum posts are modeled as custom objects (structs) containing properties for author, content, likes, and timestamps. These are stored in a growable Dynamic Array that supports index-based operations and serialization for localStorage.

- ○ **Justification:** Arrays act as the optimal container for sequential data that requires frequent rendering (mapping to HTML) and manipulation (pushing new posts/splicing deleted posts).

3. **Non-Linear Data Structures: Trees (Finals Scope):**

   - ○ **Usage:** Used in the **Diagnostic Engine (decisionData).**

   - ○ **Description:** The core logic is built upon a Binary Decision Tree (implemented as a Nested Dictionary/Graph). Each "Question" represents a Node, and user answers ("Yes" or "No") represent the Edges (Branches) leading to child nodes.

   - ○ **Justification:** Trees are the optimal structure for hierarchical decision-making, reducing complex medical triage into a path of O(H) complexity (where H is the height of the tree), avoiding deeply nested if-else blocks.

4. **Hash Maps / Dictionaries (Finals Scope):**

   - ○ **Usage:** Used in the **Information Modals** (infoDetails).

   - ○ **Description:** A collection of Key-Value pairs where unique string IDs (e.g., 'modal-101') map directly to large content blocks. This enables Direct Addressing rather than iteration.

   - ○ **Justification:** This structure provides O(1) constant time access. The system retrieves content instantly by key without needing to loop through a list of topics, ensuring high UI responsiveness.

5. **Sorting Algorithms (Finals Scope):**

   ○ **Usage:** Used in the **Community Forum.**

   ○ **Description:** The system implements a comparison-based sorting algorithm (Merge Sort/Quick Sort logic via the engine's .sort() method). It uses a custom Comparator Function to order posts first by "Likes" (Engagement), then by "Time" (Recency).

   ○ **Justification:** Ensures the most relevant content appears at the top ($O(N \log N)$), prioritizing community validation over older, less relevant posts.

6. **Tree Traversal Algorithms (Finals Scope):**

   ○ **Usage:** Used in the **Diagnostic Engine.**

   ○ **Description:** The system uses a state pointer (currentNode) to traverse the decision graph. Upon a user click, it updates the pointer to the ID of the next node defined in the adjacency list.

   ○ **Justification:** This allows for efficient navigation through the health scenarios without loading unnecessary data, terminating only when a "Leaf Node" (Result) is reached.

3.2 Algorithm Flowchart

Figure 3.2.1: Decision Tree Logic
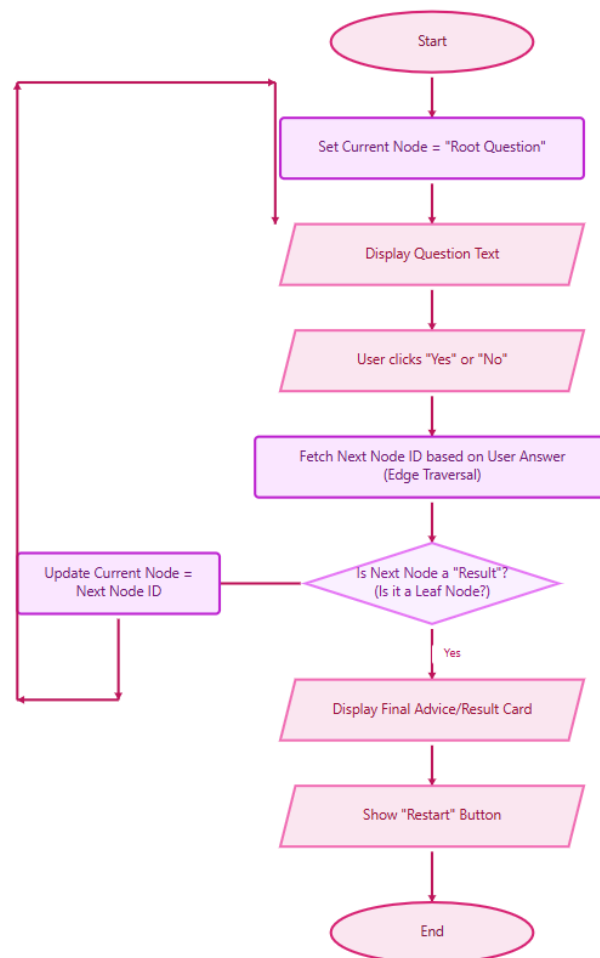
3.3 Module Breakdown

Module 1: Intelligent Agent (Chatbot)

- **Purpose:** To provide immediate answers to sensitive sexual health queries (e.g., "Safe Sex," "STI", "Clinic").

- **Data Structure:** Array of Objects (containing Regex Patterns).

- **Algorithm:** Linear Search (O(N)).

- **Process:**

    1. The system captures user input.

    2. It iterates through the canned array.

    3. It tests the input against the **Regular Expression** stored in each object (e.g., /\bhello\b/).

    4. **Hit:** If a match is found, the loop breaks, and the response is returned.

    5. **Latency**: An asynchronous setTimeout adds artificial delay to simulate natural conversation.

- **Complexity:** Time: O(N) | Space: O(N).

Module 2: Diagnostic Engine (Decision Tree)

- **Purpose:** To simulate a medical triage process for determining health risks (e.g., "Do you have any symptoms? (Bumps, itchiness, discharge, pain)?").

- **Data Structure:** Nested Object (Tree/Graph).

- **Algorithm:** Tree Traversal (O(H)).

- **Process:**

  1. **State Tracking:** The system initializes treeState to track the currentNode.

  2. **Node Rendering**: It checks if the current node is a **Leaf Node** (starts with res_). If yes, it shows the result.

  3. **Traversal:** When the user clicks "Yes" or "No", the system fetches the ID of the child node from the adjacency list and updates the currentNode pointer.

- **Complexity:** Time: O(H) (Height of the tree) | Space: O(N) (Total nodes).

Module 3: Community Forum

- **Purpose:** To allow users to share experiences and view peer advice, prioritized by community validation.

- **Data Structure**: Dynamic Array of Objects.

- **Algorithm:** Multi-Level Sorting (O(N log N)) and Linear Search (O(N)).

- **Process:**

  1. **Data Ingestion:** New posts are pushed to the posts array.

  2. **Serialization:** The array is converted to a JSON string for localStorage persistence.

  3. **Sorting:** The system utilizes a **Comparator Function.**

     - *Primary Sort:* Compare likes (Descending).

     - *Secondary Sort:* If likes are equal, compare time (Newest first).

  4. **Interaction:** When a user "Likes" a post, the system performs a **Linear Search** to find the object with the matching id before updating the counter.

- **Complexity:** Time: O(N log N) (Average case for sorting).

Module 4: Information Modal (Knowledge Base)

- **Purpose:** To display large, static educational content (e.g., "Contraception Options", "Finding Clinics") instantly without page reloads.

- **Data Structure:** Hash Map / Dictionary (infoDetails).

- **Algorithm:** Direct Access / Hashing (O(1)).

- **Process:**

  1. **Trigger:** The user clicks a UI element (button) carrying a specific data attribute (e.g., data-modal="modal-101").

  2. **Lookup:** The system uses this ID as a Key to query the infoDetails Hash Map.

3. **Retrieval:** The corresponding Value (HTML content string) is retrieved instantly without iterating through a list.

4. **Rendering:** The content is injected into the DOM, and the modal overlay is toggled to visible.

- **Complexity:** Time: O(1) (Constant Time Access) | Space: O(N) (Content storage).

CONCLUSION AND CONTRIBUTIONS

4.1 Conclusion

The HealthHub project successfully demonstrates the practical application of fundamental Data Structures and Algorithms (DSA) to address a critical social issue under UN SDG Goal 3: Good Health and Well-being, specifically Target 3.7 (Universal Access to Sexual and Reproductive Health-Care Services). By developing an interactive, privacy-focused platform, the group addressed the prevalent lack of accessible, non-judgmental sexual health information for young adults.

From a technical perspective, the application proves that efficient software design is essential for user experience. The implementation of Non-Linear Data Structures (Decision Trees) allowed for complex, medical-grade diagnostic logic to be simplified into user-friendly assessments ("Consent Check-In" and "STI Risk Check"). Furthermore, the utilization of Sorting Algorithms ($O(N \log N)$) within the Community Forum ensures that high-quality peer advice is prioritized, creating a self-sustaining ecosystem of support. The system also leveraged Linear Search and a Hash Map to provide instant, $O(1)$ access to critical health definitions and clinic locations, ensuring that users in urgent situations receive immediate answers without latency.

Ultimately, HealthHub bridges the gap between static health textbooks and real-world needs. It confirms that the proper selection of data structures which are Arrays for storage, Trees for logic, and Algorithms for processing, does not merely optimize code performance; it directly

enhances the accessibility of life-saving information. The project stands as a functional prototype of how different data structures and algorithms can be engineered for social good, providing a scalable, efficient, and anonymous tool for sexual wellness education.

4.2 Individual Contributions

Table 4.2.1: Detailed Breakdown of Group Members' Contributions

Group Member	Role / Primary Responsibility	Specific Contributions & Technical Implementation
1. Baligat, Jenna Alyssa Q.	Project Leader & Lead Developer (UI/Assessment)	• **Project Management:** Supervised the entire documentation lifecycle, created the GitHub repository, and polished the final codebase and presentation assets. • **Frontend Engineering:** Designed the **UI/UX** (CSS/HTML) for a responsive interface. • **Logic:** Implemented the **Decision Tree** algorithm for the "Self-Assessment" module (treeState

		logic).
		• **Documentation:** Authored the Introduction, Testing & Results, Conclusion, and designed the **Algorithm Flowchart.**
2. Habitan, Mark Eugenio T.	Lead Programmer (Forum/Chatbot)	• **Core Coding:** Supervised the primary coding phase, specifically implementing the **Sorting Algorithm** (O(N log N)) for the Community Forum and the **Linear Search** logic for the Chatbot. • **Documentation:** Assisted with the complexity analysis, recommendations, and co-designed the Algorithm Flowcharts. • **Presentation:** Designed the technical slides for the defense.
3. Aquino, Ram B.	Developer (Data & Logic Integration)	• **Logic Support:** Collaborated on the Chatbot and Forum logic implementation.

		• **Data Engineering:** Curated and structured the hierarchical data for the **Assessment Tree** (Nodes/Edges), ensuring accurate medical logic for the system. • **Documentation:** Authored the "Requirements and Analysis" section. • **Presentation:** Assisted with the slide deck creation.
4. Monares, Carl Dave C.	Developer (Knowledge Base)	• **Data Structure Implementation:** Coded the **Hash Map** (infoDetails object) for the Modal/Info system to ensure O(1) data retrieval. • **Data Collection:** Compiled the patterns and response strings for the Chatbot's knowledge base. • **Documentation:** Detailed the "Core Data Structures Used" section. • **Presentation:** Assisted with the slide deck creation.

5. Casison, Rhod Stephen S.	Developer (Content Module)	• **Content Integration:** Gathered and formatted the educational data for the "Learn Page" and Modal Info details. • **Code Implementation:** Assisted in wiring the Modal system logic. • **Documentation:** Wrote the "Module Breakdown" section. • **Presentation:** Assisted with the slide deck creation.

RECOMMENDATIONS

To enhance the scalability, security, and utility of HealthHub, the following technical and functional improvements are recommended for future iterations of the software:

5.1 User Authentication & Role-Based Access Control (RBAC)

Currently, the system operates anonymously to lower the barrier for entry. However, to ensure information integrity, a secure authentication system should be implemented.

- **Role Hierarchy:** The system should distinguish between user types to establish trust.
  - *Standard Users:* New contributors and regular community members.
  - *Verified Professionals:* Medical practitioners and licensed therapists.
- **Verification Protocol:** A strict verification module should be integrated for professional accounts. This will require the submission of credentials (PRC license, professional certification) and contact details. This process ensures that "Expert Answers" are medically accurate.
- **Data Privacy Compliance:** All user data must be handled in strict accordance with the **Data Privacy Act of 2012**. Sensitive information should be encrypted using hashing algorithms (e.g., SHA-256) before storage.

5.2 Enhanced Profile Management

User profiles should be expanded to allow for customization and content management.

- **Activity Dashboard:** Users should have a dedicated view to manage their previous forum posts and tracked assessment results.

- **Reputation System:** A "Trust Score" or badge system could be implemented for non-professional users based on the helpfulness of their community contributions (e.g., "Top Contributor").

5.3 Advanced Community Forum Features

The forum module can be optimized to support deeper discussions and easier information retrieval.

- **Threaded Comments:** Implementation of a **Tree Data Structure** to handle nested comments, allowing users to reply specifically to other comments rather than just the main post.

- **"Ask an Expert" Mode:** A restricted posting feature where specific questions are locked and can only be answered by users with the "Verified Professional" role. This ensures a dedicated space for medical advice validated by admins.

- **Search Algorithm:** Implementation of a **Search Indexing** algorithm (such as an Inverted Index) to allow users to filter posts by keywords, tags, or specific health topics.

5.4 Intelligent Chatbot Evolution (NLP)

The current Linear Search chatbot serves as a prototype. Future versions should integrate **Natural Language Processing (NLP)** or Machine Learning libraries. This would allow the bot to understand context and synonyms rather than relying on exact keyword matching, resulting in more conversational and accurate responses.

5.5 Longitudinal Assessment Tracking

The Self-Assessment tool should be upgraded to support data persistence for individual users. By saving assessment history, the system could generate graphs or visual logs showing a user's health risk progression over time, providing value beyond a one-time diagnosis.

REFERENCES

ABS-CBN News. (2025, June 5). "Rising HIV cases, teen pregnancies show systemic gaps in

reproductive health education" | ABS-CBN News. *ABS-CBN*.

https://www.abs-cbn.com/news/health-science/2025/6/5/-rising-hiv-cases-teen-pregnanci

es-show-systemic-gaps-in-reproductive-health-education-1704

GeeksforGeeks. (2025, November 24). *DSA Tutorial: Learn Data Structures and Algorithms*.

GeeksforGeeks.

https://www.geeksforgeeks.org/dsa/dsa-tutorial-learn-data-structures-and-algorithms/

World Health Organization. (n.d.). SDG Target 3.7: *Sexual and reproductive health.* The Global

Health Observatory.

https://www.who.int/data/gho/data/themes/topics/sdg-target-3_7-sexual-and-reproductive

-health