Lab 2 – Thought Locker Prototype Product Specification

Deshawn Bailey, Olivia Gee, Nathan Greaney, Jerome Palmer, Ty Schiffer, Ryan Wilkinson

Old Dominion University

CS411W

Professor James Brunelle

29 September 2023

Version 1
# Table of Contents

1 Introduction
   1.1 Purpose 4
   1.2 Scope 4
   1.3 Definitions, Acronyms, and Abbreviations 6
   1.4 References 10
   1.5 Overview 12

2 General Description
   2.1 Prototype Architecture Description 12
   2.2 Prototype Functional Description 13
   2.3 External Interfaces 17
      2.3.1 Hardware Interfaces 17
      2.3.2 Software Interfaces 17
      2.3.3 User Interfaces 17
      2.3.4 Communications Protocols and Interfaces 17

3 Specific Requirements 17
   3.1 Functional Requirements 17
      3.1.1 Account Registration 17
      3.1.2 User Permissions 17
      3.1.3 Image Upload 18
      3.1.4 Task Scheduling 18
      3.1.5 Event Viewer 18
      3.1.6 Item Location 19
3.1.7 Emergency Contact

3.2 Performance Requirements

3.2.1 Real-time Tracking

3.2.2 Sound Playback

3.2.3 Reminder and Notification

3.3 Assumptions and Constraints

3.4 Non-Functional Requirements

3.4.1 Security

3.4.2 Maintainability

3.4.3 Reliability

3.4.4 Usability

Appendix A: Site Map

Appendix B: Database Schema

List of Figures

Figure 1: Thought Locker Prototype Major Functional Components

List of Tables

Table 1: Thought Locker Feature Description and Prototype Implementation
1 Introduction

In a world where dementia affects one in ten Americans aged 65 and older, our mobile assistant offers a lifeline to those with mild and moderate dementia. With features such as reminders, item location, monitoring, and analytics, Thought Locker empowers patients to maintain their independence, alleviating stress for caregivers. As dementia diagnoses steadily rise, Thought Locker stands as a vital tool for enhancing the quality of life for both patients and their dedicated caregivers.

1.1 Purpose

- Thought Locker assists individuals with mild to moderate dementia in daily tasks.
- Features include reminders, item location, and contact support.
- It enables caregivers to monitor progress and mood changes.
- Compatible with Android and iOS devices, offering user-friendly authentication.
- Includes a calendar, item finders, and sensors for location tracking.
- Caregivers access motion sensors and patient activity monitoring.
- Customizable for each patient's needs, with some patient control.
- Combines hardware (mobile device) and software (JavaScript, React, Node.js) components.
- Utilizes GitHub for version control, issue tracking, and continuous integration.

1.2 Scope

- Thought Locker supports dementia patients, families, friends, and caregivers by promoting independence and reducing stress.
- The prototype demonstrates core functionalities, even though not all real-world features are included.
● Hardware and software requirements encompass Android compatibility, JavaScript, React, Node.js, MySQL, AWS, Jest, Git/GitLab, and more.

● Prototype features cover user accounts, task management, calendars, item tracking, and notifications.

● Some features use simulated data, like item location and contact info.

● Challenges include design tool familiarity, React Native complexity, GitLab usage, database integration, server setup, API interfacing, and simulated data sourcing.

[ This space left intentionally blank. ]
1.3 Definitions, Acronyms, and Abbreviations

**Amazon Web Services (AWS):** A cloud computing platform that provides a variety of services including compute, storage, databases, analytics, machine learning, networking, mobile, developer tools, security, and enterprise applications.

**Android:** An open-source mobile operating system based on the Linux kernel and developed by Google.

**Application Programming Interface (API):** A set of protocols, routines, and tools for building software applications that specify how software components should interact with each other.

**Application Server:** A software framework that provides an environment for running applications.

**Authentication:** The process of verifying the identity of a user.

**Biometric:** unique physical or behavioral characteristics of an individual to identify them.

**Bluetooth Low Energy (BLE):** a wireless communication technology used for short-range communication between devices.

**Caregiver/Family Member:** Any person that is able to provide assistance with managing a dementia patient’s symptoms. Their duties consist of transporting patients to and from their commitments, maintaining a patient’s daily routine, and reminding them to take their medications at regular intervals.

**Continuous Integration (CI):** A software development practice that involves frequently integrating code changes from multiple developers into a shared repository, verifying that the changes do not break the build and that the software continues to function correctly.
**Containerization**: A method of packaging and deploying software applications with all their dependencies into a single unit, called a container, which can run reliably and consistently across different computing environments.

**Database Server**: A computer program or software application that provides database services to other computer programs or clients.

**Docker**: A software platform that allows its users to build, test, and deploy applications in standardized executables resembling containers.

**GeoTag**: A physical tag that emits a Bluetooth signal to assist devices in determining their location.

**GitLab**: An open-source repository service that allows its users to work on a single project simultaneously.

**Google Maps Geolocation API**: a service provided by Google that allows developers to determine the location of a device using Wi-Fi or mobile network signals.

**iOS**: A mobile operating system developed by Apple Inc based on the Unix operating system.

**Issue Tracking**: The process of managing and resolving software issues, bugs, and feature requests.

**JavaScript**: Scripting programming language that creates dynamic web page content and mobile applications.

**JavaScript Object Notation (JSON)**: A lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate.

**Jest**: Test-runner for JavaScript applications that supports a JavaScript library for creating, running, and structuring tests.
**Jira**: A project management and issue tracking tool developed by Atlassian. It is widely used by software development and project management teams to plan, track, and manage their work.

**MongoDB**: Non-relational document database that provides support for non-relational querying.

**Node.js**: An open-source, cross-platform JavaScript runtime environment that enables the execution of server-side JavaScript code.

**Non-relational Database**: A type of database that doesn't rely on the traditional structure of tables, columns, and rows found in relational databases. Instead, they are designed to handle large and complex sets of unstructured, semi-structured, or structured data.

**Patient**: Any individual suffering from mild to moderate dementia. Their symptoms typically consist of occasional disorientation, difficulties with making decisions, and short-term memory loss.

**PostgreSQL**: Free and open-source database management system that supports relational (SQL) and non-relational (JSON) querying.

**Programming Language**: A formal language used to communicate instructions to a computer or other machine.

**React**: An open-source JavaScript library that is used to build user interfaces for web and mobile applications.

**Relational Database**: A type of database that stores data in a structured format, using rows and columns to represent data entities and attributes.

**Relational Database Management System (RDBMS)**: A software system that enables users to create, maintain, and manipulate relational databases.

**Repository**: A central location where digital files, usually in the form of software source code, are stored and managed.
Structured Query Language (SQL): A programming language used to manage and manipulate data in a relational database management system (RDBMS).

Testing Framework: A set of guidelines, standards, and tools that software developers use to create and run automated tests for their code.

Version Control: A system that tracks and manages changes to software code, documentation, and other files over time.

[ This space intentionally left blank. ]
1.4 References


[ This space left intentionally blank. ]
1.5 Overview

- The product specification will encompass the design of Thought Locker's architecture and provide a functional description of the prototype.
- The architecture comprises both the requisite hardware for accessing Thought Locker and the development software.
- The software employed in Thought Locker encompasses all the necessary programming languages for both the front-end and back-end components.
- The functional requirements of Thought Locker and their implementation within the prototype will be detailed.

2 General Description

The Thought Locker prototype is a mobile application consisting of a set of features to assist those with dementia and caregivers.

2.1 Prototype Architecture Description

- Two main users types: Patient and Caregiver
- Prototype application will be ran on Android/iOS platforms
- Server: The Thought Locker will be using AWS as our hosting service to host our Web Application. The front-end and back-end will be connected via React Native API.
- Database: The database the Thought Locker prototype will use will be MYSQL. This database is used to store our end users’ information, profiles, and data history.
- APIs: React Native APIs will be used to connect the ThoughtLocker database to its server. A Google Maps API will be used by the Thought Locker prototype for location generation.
2.2 Prototype Functional Description

The major functional components of the Thought Locker prototype include the following:

- **User Registration**: Secure registration through which patient and caregiver are distinguished and connected. User profile is created and connected to the database.

- **Login Authentication**: Authentication and login into application utilizing username and password setup in user registration.

- **User Profile Management**: Allows users to update profile, delete profile and edit permissions.

- **Calendar**: Application allows users to view the calendar.

- **Task Scheduling**: Users are able to display the current assigned task and check when completed. If permission is given, users are allowed to delete and modify tasks.

- **Item Locator**: Users are able to locate items assigned through the database. Items utilize bluetooth tracking and audio. Assisted through pictures.
• Contact Center: Users are able to contact pre-programmed numbers of others.
• Emergency Contacts: Through the application, users are able to one button cycle through a set of emergency contacts.
• Monitoring: Users are allowed to set up and receive monitoring alerts due to pre placed sensors.
• Notifications: Notifications are sent to caregiver
• Event Viewer: User is able to access the logs of the application. Viewing logs is primarily done by the Caregiver role.
• User Analytics: Analytics are displayed with the ability to download. Queries login, task manager, item finder, contact log and monitoring.

Table 1

<table>
<thead>
<tr>
<th>Features</th>
<th>Description</th>
<th>Prototype Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ACCOUNT MANAGEMENT</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Account Creation</td>
<td>Set up a new patient or caregiver account</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Account Deletion Verification</td>
<td>Ensure the intentional removal of a user's account, requiring confirmation steps to prevent accidental deletion</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Login / Authentication</td>
<td>Users prove their identity and gain access to the application by providing the required credentials</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>User Profile Management</td>
<td>Administration of individual user accounts within the application, allowing users to customize their settings, preferences, and personal information</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>User Location Information</td>
<td>Identifies the geographic position of a user through their device by enabling location-based services</td>
<td>Partially Implemented: Location data will be simulated</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>USER INTERFACE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task Scheduling</td>
<td>Organize specific tasks to be executed at predetermined times</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Task Completion</td>
<td>Mark a task complete after successfully executing it</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Task Deletion Verification</td>
<td>Ensure the intentional removal of a specific task, requiring confirmation steps to prevent accidental deletion</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Item Locator</td>
<td>Identifies the location of items, making it easier to find and manage them</td>
<td>Partially Implemented: Location data will be simulated</td>
</tr>
<tr>
<td>Item Registration</td>
<td>Record and document information about an item in a database, including details such as ownership, specifications, and relevant data for tracking and management</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Item Deletion Verification</td>
<td>Ensure the intentional removal of a specific item from the database, requiring confirmation steps to prevent accidental deletion</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Event Viewer</td>
<td>Provides logs and details about system, security, and application events, aiding in system troubleshooting, monitoring, and maintenance</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Calendar</td>
<td>Visual tool that organizes and displays dates, days, and events, facilitating time management and scheduling</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
<td>Implementation Status</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>Sensor Monitoring</td>
<td>Continuous observation and collection of data from sensors to track and assess changes to item movement in the environment</td>
<td>Partially Implemented: Sensor data will be simulated</td>
</tr>
<tr>
<td>Emergency Contacts</td>
<td>Individuals designated to be contacted in urgent or critical situations, typically for medical, safety, or personal reasons</td>
<td>Partially Implemented: Contact data will be simulated</td>
</tr>
<tr>
<td>Notifications</td>
<td>Alerts delivered through the application to inform users about important information, events, or updates</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td><strong>DATA MANAGEMENT</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User Analytics</td>
<td>Collecting and analyzing data about the behavior and interactions of patients with the application, aiding in decision-making and patient care</td>
<td>Partially Implemented: Analytics will be simulated</td>
</tr>
<tr>
<td>Reports</td>
<td>Structured documents that provide information, analysis, and findings on specific user defined topics</td>
<td>Fully Implemented</td>
</tr>
<tr>
<td>Item Tagging</td>
<td>Assign key metadata values to items, enabling efficient categorization and retrieval within the database</td>
<td>Partially Implemented: Geotag data will be simulated</td>
</tr>
<tr>
<td>Images</td>
<td>Upload images of tracked items, which must correspond to items in the database</td>
<td>Fully Implemented</td>
</tr>
</tbody>
</table>

[This space left intentionally blank.]
2.3 External Interfaces

2.3.1 Hardware Interfaces

- Android mobile device with Android 11 or higher
- iOS mobile device with iOS 15 or higher
- Computer running Linux, Windows, or Mac OS with a modern web browser

2.3.2 Software Interfaces

- GitLab
- MySQL
- AWS
- Android Studio

2.3.3 User interfaces

- Mobile device with internet connection

2.3.4 Communications Protocols and Interfaces

- IEEE 802.11b-ax

3 Specific Requirements

3.1 Functional Requirements

3.1.1 Account Registration

This function shall create and store account information with a username, password, email-address, date of birth, and chosen user role. The account and role is necessary to access individual features of Thought Locker based on permissions.

3.1.2 User Permissions

This functionality will determine what features are available based on the given user needs and capability.
3.1.3 Image Upload

This function will provide the capability for the user to upload an image of a selected item that requires tracking. The image must correlate to an item that has been registered in the database.

3.1.4 Task Scheduling

This is a curated list of tasks to be performed during the day by the patient. Patient and caregiver(administrator) roles must be able to add, delete, and modify tasks accordingly. Unique constraints must exist in the database so that the patient is unable to create the exact same task twice in a given period.

The following functional requirements must be met:

- Query the database to retrieve the view of task_scheduling
- Provide the capability for the administrator to filter for individual tasks.
- Provide the capability to allow the administrator to see what tasks have been marked completed.
- Provide the capability to see the number of completed tasks versus incomplete tasks in a given day.
- Provide the capability to add, delete, or modify a task.
- Allow the caregiver to view the amount of times a task has been completed

3.1.5 Event Viewer

This feature is intended to allow the caregiver to view all relevant logs per individual feature. The caregiver has the capability to filter which feature the user would like to view. Furthermore, additional filtering shall be provided in accordance with the time or date of an event.
3.1.6 Item Location

Bluetooth tracking will be utilized to indicate the location of pertinent items. An image, item description, and item name will be uploaded to the database. Provided capability to search for the necessary image by name or picture whichever is easier. The item will use a noise notification to indicate the current location to the user.

3.1.7 Emergency Contact

In the event of an emergency, a patient will have the capability to contact a specialized list of contacts in an expeditious fashion. The following functional requirements must be met:

- At least five emergency contacts must be included to utilize the feature
- Provided the capability to initiate a priority queue based on the list of contacts provided
- It will transverse the list of contacts, if no response is received it will move on to the next available contact
- Provided capability will be measured in amount of time per no response before moving on to next contact

3.2 Performance Requirements

3.2.1 Real-time Tracking

- If delay is longer than two minutes then the device will resynchronize the application with the Google Maps API to update to the best known location
- Bluetooth capability requires that the device be within 15-45ft otherwise the item cannot be located, thus will be grayed out

3.2.2 Sound Playback
● Sound will be emitted from Geotag within 3 seconds of the item being selected to be found from the item list.

● Geotag sound will repeat for 30 seconds once an item is selected to be found.
  ○ After 30 seconds, the user will be prompted to play sound again.

● Sound playback will cease once the user finds the item and deactivates it.

3.2.3 Reminder and Notification

● Notification must be sent to the patient and caregiver within 5 seconds of the item being declared “found.”

● Notification must be sent to caregiver if item is not declared “found” in one hour.

● Notification must be sent to caregiver and patient within 5 seconds of task being declared “complete” by patient or caregiver.

● Notification is sent to the caregiver after 30 minutes of a task not being completed within the allotted time slot.

● Door and window sensors will immediately notify caregiver and/or family members if opened by a patient.

3.3 Assumptions and Constraints

● Assumptions
  ○ User privacy concerns are a priority.
  ○ Compliance with data protection regulations is essential.
  ○ Internet connectivity is available.
  ○ The app must work on various devices and platforms.
  ○ Secure user authentication is required.
  ○ Data storage and transmission must be secure.
○ Regular updates are necessary.
○ Accuracy of medical data is critical.

● Constraints
○ Limited budget.
○ Project time constraints.
○ Regulatory compliance impacts design.
○ Strong security may increase complexity.
○ Balancing user experience with security.
○ Data storage limitations.
○ Resource availability (personnel and hardware).
○ Network connectivity variations.
○ Device performance differences.
○ Integration with external systems.
○ Defining data ownership and access rights.

3.4 Non-Functional Requirements

● Database
○ Relational database
○ Patient and Caregiver data
   ■ Username and password data
○ Event finder data
○ Geotag data
○ Contact center data
○ Notification data
○ Analytics Data
○ Login data

3.4.1 Security

● Password encryption
  ○ Minimum length and complexity
● Log patient and caregiver actions
  ○ Login
  ○ Calls to emergency contacts
● Establish a secure connection to the server after login
● Administrator hierarchy in database access
● Encrypted data transfer
● Database access control

3.4.2 Maintainability

● Updates sent via containerized deployment by development team responsible
  (front-end, back-end, database administrator)
● Archive and specify software development versions in updates
● Update software periodically to align with current application security protocols

3.4.3 Reliability

● Availability of application 24/7 to caregivers and patients
● All features must have ~100% success rate.
  ○ Adding and deleting events
  ○ Adding and removing reminders
● Critical functions
○ Emergency contact feature must be ~100% operational at all times

3.4.4 Usability

● Usable for those with mild to moderate dementia

○ ~90% of patients must be able to perform all essential functions of application

■ Open events

■ Call emergency contacts

■ Locate items with item locator

● Usable for caregivers

○ ~80% must be able to perform all essential functions of application

■ Add and delete events, tasks, and items

■ Create emergency contact list

■ Open and interpret analytic report results

● ~70% must be able to pass focus group test on reading analytic reports

[ This space left intentionally blank. ]
Appendix A: Site Map
Appendix B: Database Schema