Project Requirements

1. Project Overview

- **Application Name**: Kinnect
- **Purpose**: Kinnect is designed to centralize and streamline various types of information for children of aging parents, making it significantly easier to manage healthcare details. It will serve as a real-time "hub" accessible by trusted family members, ensuring everyone stays updated on the latest scheduling and medication details for consistency.
- **Core Features**: The application will include a Medication Log, an Appointment Calendar, and a Vitals Tracking Dashboard.

2. Functional Requirements (FR)

- **User Management**
  - **FR01**: The software must enable the user to create an account.
  - **FR02**: The software must enable the user to log in to their account.
- **Medication Log**
  - **FR03**: The software must enable the user to add new medications, including name and dosage.
  - **FR04**: The software must enable the user to view a list of all added medications.
  - **FR05**: The software must enable the user to mark a medication as "taken."
  - **FR06**: The software must enable the user to delete a medication from the list.
  - **FR07** - The software must enable the user to assign a recipient's name to each medication.
  - **FR08** - The software must enable the user to edit the details of a medication that was administered.
- **Appointment Calendar**
  - **FR09**: The software must enable the user to add new appointments with details such as date, time, location, doctor's name, and purpose of the visit.
  - **FR010**: The software must enable the user to view all appointments on a collaborative calendar.
  - **FR11**: The software must enable the user to click on an appointment to view its details.
  - **FR12**: The software must enable the user to add a post-visit summary for each appointment.
- **Vitals Tracking**
  - **FR13**: The software must enable the user to enter and save readings for various vital signs.
  - **FR14**: The software must enable the user to view a chart/graph displaying vital sign trends over time.

3. Non-Functional Requirements (NFR)

- **Usability**: The application must feature a clean and straightforward interface, easily

understandable by non-technically inclined users.
- **Performance**: The application should load quickly, with all data appearing as instantaneously as possible.
- **Security**: All user data must be stored securely. User passwords must be hashed before being stored in the database.
- **Compatibility**: The application must function correctly on the latest versions of all major web browsers.
- **Technology Stack**: The project must be built using React for the frontend, a Node.js/Express server for the backend, and MongoDB/SQL for the database.

4. Domain Requirements

- **DR01**: Marking a medication as taken must create a timestamp and record the name of the family member who administered it.

5. Use Case Specifications

Managing Medications

Description: Allows users to add, delete, and update medications, as well as keep track of administration events.

Actor: Family Member

Entry Condition: The family member authenticated and navigated to the "Medication Log" from the main application screen.

Basic Path: Add New Medication

1. Application displays a Medication Log screen, showing the medications that have been added and a form to add new ones.
2. The medication name, dosage, and recipient name are added.
3. The user presses the "Add Medication" button.
4. The application validates the user input.
5. The new information is saved.
6. The medication list is updated
7. The use case returns to step 1.

Alternative Paths:

- **A01: Interacting with Existing Medication**: The application displays a list of medications with their name, dosage, recipient, administration history, as well as options to "Mark as Taken," "Edit," or "Delete."
- **A02: Edit a Medication**:
    1. The user chooses to edit a given medication.
    2. The application displays the pre-filled out edit form.
    3. The user makes modifications and clicks “Save Changes” (or Alternative Path A05)
    4. The new information is validated. (unless Exception Path E01)
    5. The application updates medication info in the database.
    6. The medication list is refreshed.
    7. The use case returns to step 1.
- **A03: Delete a Medication**:
    1. The user chooses to click the “Delete” button for a given medication.
    2. The medication is removed from the list/database.
    3. The medication list is refreshed.
    4. The use case returns to step 1.
- **A04: Mark a Medication as Taken**:
    1. The user choose to click the “Mark as Taken” button for a given medication.
    2. The application records the administration event.
    3. The application updates the administration history.
    4. The use case returns to step 1.
- **A05: Cancel an Edit**:
    1. From section A02, the user chooses to click the “Cancel” button.
    2. The form closes without changes being saved.
    3. The use case returns to step 1.

Exception Path:

- **E01: Invalid or Missing Information**:
    1. The application detects missing/invalid information during "Add" or "Edit."
    2. An error message is displayed.
    3. The user returns to the form for corrections to be made.

Managing Appointments

Description: Allows users to add new appointments to a collaborative calendar, view appointment details, and add post-visit summaries.

Actor: Family Member

Entry Condition: The user is logged into the application and has selected the “Appointment Calendar”

Basic Path: Add New Appointment

1. The application displays the Appointment Calendar with any existing appointments as well as the option to add a new appointment.
2. User clicks the "Add New Appointment" button.
3. The application presents a form to fill in appointment details.
4. The user fills the form in and presses the "Save" button. (or Alternative Path A01)
5. The application validates this information. (or Exception Path E01)
6. The application saves the new appointment.
7. The application updates the calendar display.
8. The next action is ready to be taken.

Alternative Paths:

- **A01: View and Update an Appointment**:
    1. The user clicks an existing appointment.
    2. The application displays a detailed view of that appointment.
    3. The user clicks the "Add/Edit Summary" button.
    4. The application presents an editable text field for the summary.
    5. The user enters/updates the summary and presses the "Save Summary" button.
    6. The application saves the summary.
    7. The user returns to the calendar view.

Exception Path:

- **E01: Invalid or Missing Information**:
    1. Missing information is detected.
    2. An error message is displayed by the application.
    3. The user returns to the form to add/edit information.

Tracking Vitals

Description: Allows users to enter/save vital sign readings and view trends over time on a chart.

Actor: Family Member

Entry Condition: The user logged into the application and has selected the “Vitals Tracking” dashboard.

Basic Path: Enter New Vital Reading

1. The application displays the Vitals Tracking screen with a chart of any existing data that has been entered, as well as a form to add additional data.
2. The user presses the vital type button, enters values, and presses the date/time button, and enters values.
3. The user presses the "Save Reading" button.
4. The application validates the user data. (or Exception Path E01)
5. The application saves the new vital reading.
6. The application updates the chart with the new data point.
7. The next action is ready to be taken.

Exception Path:

- **E01: Invalid Reading**:
    1. A non-numeric or invalid vital sign is detected.
    2. An error message is displayed by the application
    3. The user returns to the form to correct the invalid information.