KARE Documentation

Central Washington University, CS481.A01, Spring 2023

Version 2.0

Logan Halverson - Hunter Berg - Conner Kaul
Megan Phinney - Stephan Schuller - Zak Kappenman
Ryan von Fiedler - Ardit Miftaraj

Table of Contents

Introduction

Purpose

The purpose of this document is to specify the software requirements for KARE 2.0, the updated version of the KARE app for medically complex children. The scope of this SRS covers the entire KARE 2.0 software system, including the phone app, cloud database, and desktop application. The primary objective of KARE 2.0 is to provide better support for the management of daily medical needs of medically complex children, while also making it easier for healthcare providers to access and evaluate their medical history and prescribe appropriate therapies for them. The updated software will provide additional functionalities, such as better visualization, local database storage, and more applications, including sleep cycle monitoring and doctor visit tracking, all targeted for Android deployment. The project will also conduct a cost-benefit analysis of low-cost options for cloud database deployment, such as Google Cloud, and provide detailed information on deployment, upgrades, and fee structures. Finally, the existing desktop application will be improved, with enhanced visualization, basic regression tools on existing data, and exporting data in different formats.

Document Conventions

Document Conventions: This document follows standard typographical conventions for technical documents. The header font used is Georgia size 14, and the normal text font used is Georgia size 11. The priority of higher-level requirements is assumed to be inherited by detailed requirements, unless explicitly stated otherwise. Each requirement statement is to have its own priority, which will be identified as High, Medium, or Low. The priority level reflects the importance of the requirement to the overall success of the project.

In addition, the document uses the following conventions for highlighting:

- Bold: used to indicate specific terms or requirements being defined for the first time
- Italics: used to indicate a placeholder to be replaced with specific information during implementation
- Underline: used to indicate hyperlinks to external documents or resources
- Bullets: used to indicate lists of requirements or features, where each item is of equal priority.

Intended Audience and Reading Suggestions

Intended Audience and Reading Suggestions: This SRS is intended for developers, project managers, testers, and documentation writers involved in the development of KARE 2.0. It may also be of interest to marketing staff and potential users of the system.

The rest of this documentation is organized as follows:

1. Introduction: provides an overview of KARE 2.0 and its purpose, as well as document conventions and reading suggestions.
2. Overall Description: provides an overview of the system and its environment, including user characteristics, functional requirements, non-functional requirements, and constraints.
3. Specific Requirements: provides detailed requirements for the phone app, cloud database, and desktop application components, including functional and non-functional requirements.
4. Verification and Validation: describes the testing and acceptance criteria for the system, including test cases and expected results.
5. Appendices: includes supporting information such as use case diagrams, data flow diagrams, and glossary of terms.

We suggest that readers begin with the Introduction section to understand the purpose and scope of the document. Developers and project managers should focus on the Overall Description and Specific Requirements sections, as these provide a detailed understanding of the functional and non-functional requirements for each component of the system. Testers should also review the Verification and Validation section to understand the testing and acceptance criteria for the system. Marketing staff and potential users may find the Overall Description section useful for understanding the system and its capabilities.

Product Scope

Kare 2.0 is a software product designed to improve the care of medically complex children, ease the burden on parents, and prevent well-meaning parents from losing custody of their child due to an inability to provide an acceptable standard of care. The software will consist of a mobile application for Android devices, a desktop application for Windows and macOS, and a cloud-based database for storing and managing the child's medical data.

The primary objective of Kare 2.0 is to provide a comprehensive and user-friendly solution for managing the daily medical history of medically complex children at home. The mobile application will allow parents or legal guardians to record and monitor the child's vital signs, medication schedules, doctor visits, sleep cycles, and other relevant information. The desktop application will enable parents or legal guardians to download and visualize the data collected by the mobile app and perform basic statistical analysis. The cloud-based database will provide secure and reliable storage for the child's medical data, with access granted only to authorized users.

Kare 2.0 is aligned with the philanthropic goal of providing innovative solutions to improve the quality of life for individuals with complex medical needs. By using state-of-the-art technology to streamline the management of medical data, Kare 2.0 will help parents and legal guardians to provide better care for their children and reduce the risk of custody loss.

References

User Manual - Kare - *Home Monitoring Software* (Elliot, D. et al, 2023)

User Manual - Kare - *Home Monitoring Software (iOS)* (Elliot, D. et al, 2023)

Kare Desktop User Manual Version 1.5 (Schuller, S., Phinney, M., Kappenman, I., 2023)

Kare Server User Manual Version 1.0 (von Fiedler, R., 2023)

Desktop Application Test Report (Schuller, S., Phinney, M., Kappenman, I., 2023)

Problem Statement

Parents of medically complex children face significant challenges in managing their child's medical needs. One of the biggest challenges is keeping track of their child's medical records, which can be complex and extensive. Parents often struggle to maintain accurate and up-to-date records due to the lack of resources available to them, such as specialized software or training. As a result, parents may be overly burdened by the responsibility of managing their child's medical records, leading to health risks for the child and possible loss of custody.

Overall Description

Product Perspective

Kare 2.0 is a self-contained product that builds upon the previous version, Kare 1.0. Kare 1.0 was designed to assist parents or legal guardians of medically complex children in managing their care. The new version, Kare 2.0, aims to improve the user experience by integrating mobile and desktop applications into a single system.

Kare 2.0 will replace Kare 1.0 as the primary product for managing the care of medically complex children. It is a stand-alone product that does not rely on any other system or product to function. However, in the future it may interface with other healthcare systems or devices to collect and integrate data.

Product Functions

- User authentication and login
- Profile management (create, update, delete)
- Schedule management (create, update, filter, delete)
- Notification management (send and receive notifications)
- Reminder management (create, update, filter, delete)
- Tracking and logging of feeding, medication, and therapy data
- Integration with third-party devices and services
- Reporting and analytics of user data
- Admin functionality for managing user accounts and permissions

These functions can be organized into groups based on their related requirements, which will be described in more detail in Section 3. A top-level data flow diagram or object class diagram can also be provided to visually represent the relationships between these functions.

User Classes and Characteristics

The primary users of Kare 2.0 are the parents or caregivers of medically complex children, who may have limited medical knowledge or experience in caring for their child's specific medical needs. These users may have varying levels of technical expertise and may be using the product frequently. Additionally, healthcare professionals such as doctors or nurses may also use the product to review patient data and provide medical advice to the parents or caregivers. It is important for the product to be accessible and user-friendly to these primary users, while also providing the necessary features and security for healthcare professionals.

Operating Environment

Hardware Platform:

- Mobile devices (Android) with internet connectivity.
- Desktop computers (Windows) with internet connectivity.

Operating Systems and Versions:

- Android 5.0 (Lollipop) or later
- Windows 10 or later

Other Software Components or Applications:

- Database management system (e.g. MySQL) for storing data from the mobile app and transmitting it to the desktop app.
- Web server for hosting the mobile app and the API that connects it to the database.
- Visualization software for rendering data in the desktop app.

Design and Implementation Constraints

- Hardware limitations: The software should be designed to work on a wide range of devices with varying hardware specifications, including smartphones, tablets, and desktop computers.
- Operating system requirements: The mobile application should be compatible with Android operating systems, and the desktop application should be compatible with Windows.
- Compliance with regulatory policies: The software should comply with any applicable laws and regulations, including those related to data privacy, security, and HIPAA.
- Database requirements: The software should use a database system that is scalable, reliable, and secure, and that can handle large amounts of data.
- Interface with other applications: The software should be designed to integrate with other applications that are commonly used by parents of medically complex children, such as electronic medical record systems and assistive technology devices.

- Security considerations: The software should be designed with robust security features to protect sensitive data and prevent unauthorized access.
- Design and programming standards: The software should be developed according to industry-standard design and programming practices to ensure reliability, maintainability, and scalability.
- Language requirements: The software should be developed using programming languages that are well-suited to the task, and that are compatible with the chosen development platforms and tools.

User Documentation

Kare 2.0 will be accompanied by two user manuals, one for the mobile application and one for the desktop application. The purpose of these manuals is to provide users with instructions on how to use the software, including its features and functionalities.

The mobile user manual will be delivered in digital format, accessible through the mobile application. It will include step-by-step instructions on how to navigate the application, add and manage a child's medical information, and access resources and support. The manual will also provide information on how to troubleshoot common issues and contact customer support.

The desktop user manual will be delivered in digital format, accessible through the desktop application. It will include step-by-step instructions on how to view and analyze a child's medical information, customize charts and graphs, and generate reports. The manual will also provide information on how to troubleshoot common issues and contact customer support.

Both manuals will adhere to standard user documentation delivery formats and will be written in clear, concise language to ensure ease of use for all users.

Mobile Design

Mobile High Level Design

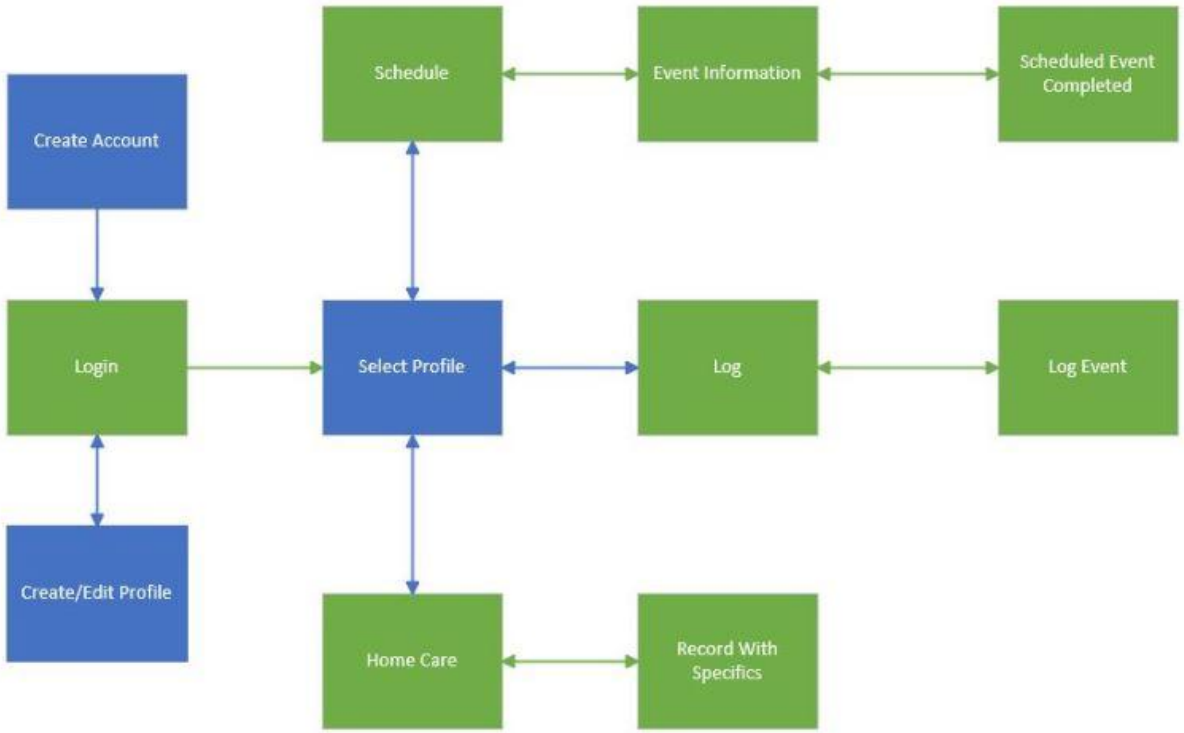

Figure 1: Mobile High Level Design

The mobile application will first prompt the user to either create an account if they do not have one or login with previously created credentials. After the user has logged in, they will be able to create a profile, edit a previously made one, or select a profile. The profile can be made for primary caretakers like parents with all privileges or a certain caretaker with limited privileges. Once the user has selected the profile, depending on privileges, they will be able to schedule a recurring event, log an event like feeding, medicine, or therapy activities, or record an event like temperature, height, weight, emesis or oxygen usage in the home care option.

Mobile Low Level Design

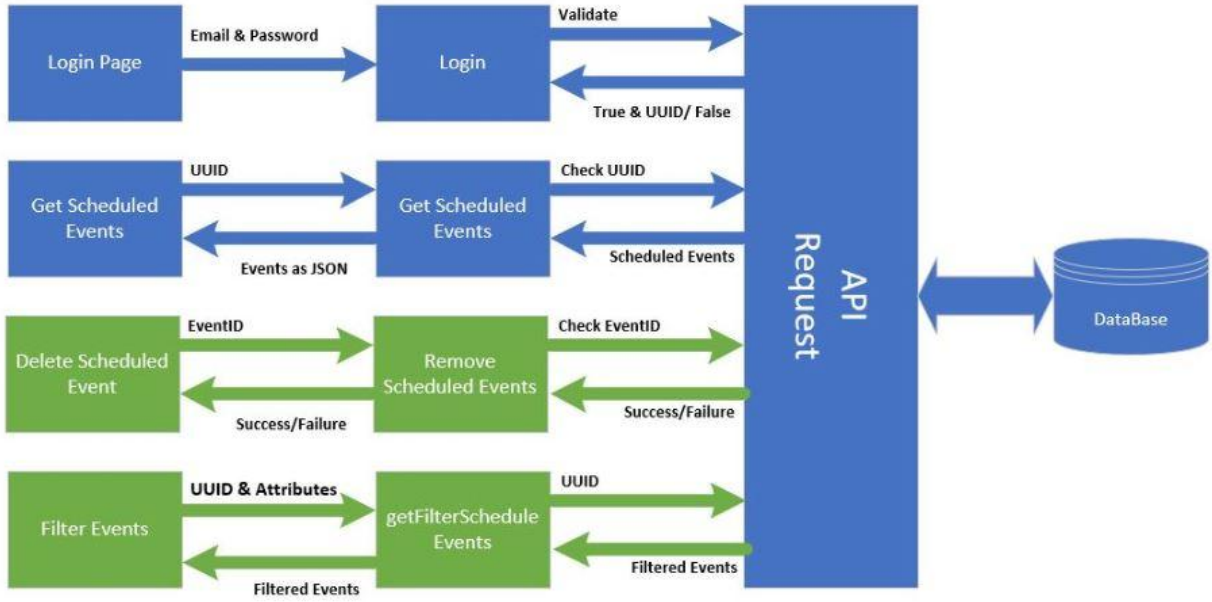

Figure 2: Mobile Low Level Design API

The mobile low-level API design includes some of the examples of the legacy API calls in blue and new API calls in green. For example a legacy API call was the login API call where the user attempts to login with their credentials and clicks the login button on the login page of the mobile app. Those credentials are then sent to the server and the server will check if they are valid. If they are, the server will send back that the validation was successful and also send the UUID tied to the user's credentials. An example of a new API call would be the Delete Scheduled Event API call where the user selects an event to be deleted. The eventID of the event to be deleted is sent to the server and the server will check if the eventID can be deleted. If an event can be deleted it is deleted from the database and a success message is passed back to the mobile application. If it was not successful then a failure message is passed back to the mobile application.

Desktop Design

Desktop High Level Design

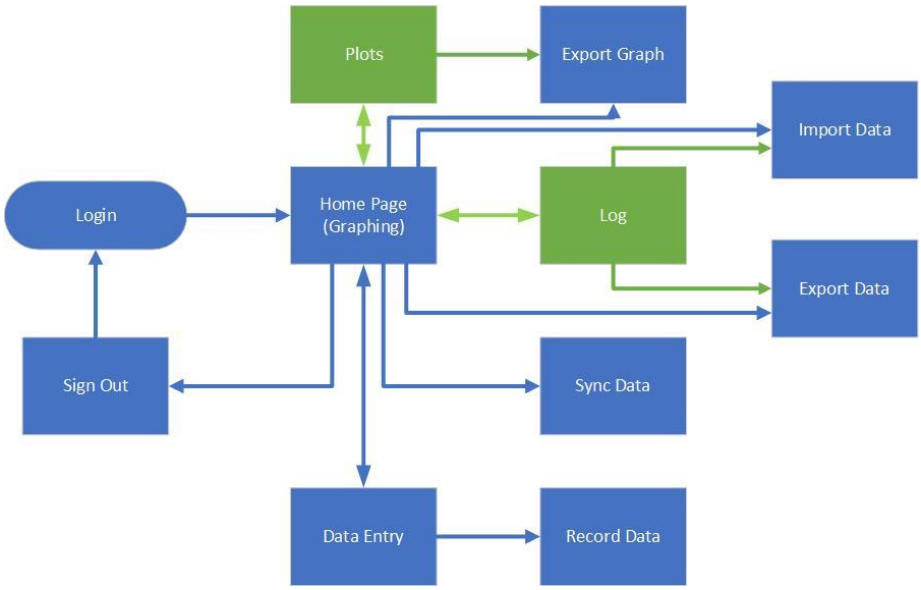

Figure 3: Desktop High Level Design

The Desktop high-level design includes legacy tools (blue) and new tools (green). Starting the Kare desktop application, the user will see the login page. On this page a user can login using the credentials of an account created on the mobile application. The user will also have to enter their specific server IP address. Upon successful validation, the user will start home_page.py creating the HomePage class which contains the "New - Graph" tool and Data control panel. From this page the user has options to import data from a file, which will allow them to use that data to graph using the data sidebar. The user may also choose to synchronize data from the database through the toolbar, pulling the data of that user's account to use in the data sidebar to graph. After creating a graph, the user may export the graph as a png image file or export the data plotted on the graph as a csv file. Clicking the record new entry button on the toolbar the data_entry_page.py will create the DataEntryPage class. On the data entry page, the user may enter data for different events that will be sent to the database. The user may also choose to use the View - Log tool which uses log_page.py to create the LogPage Class. Using this tool the user may choose to view log events line by line, filter and save them as datasets in the JSON format. The user can choose the View-Plot tool to create the StatisticsPage class . From here the user may create plots including correlation and grid line scatter plots. The user may choose to export plots from this page as well. Finally the user may also choose to logout, bringing them back to the login page.

Desktop Low Level Design

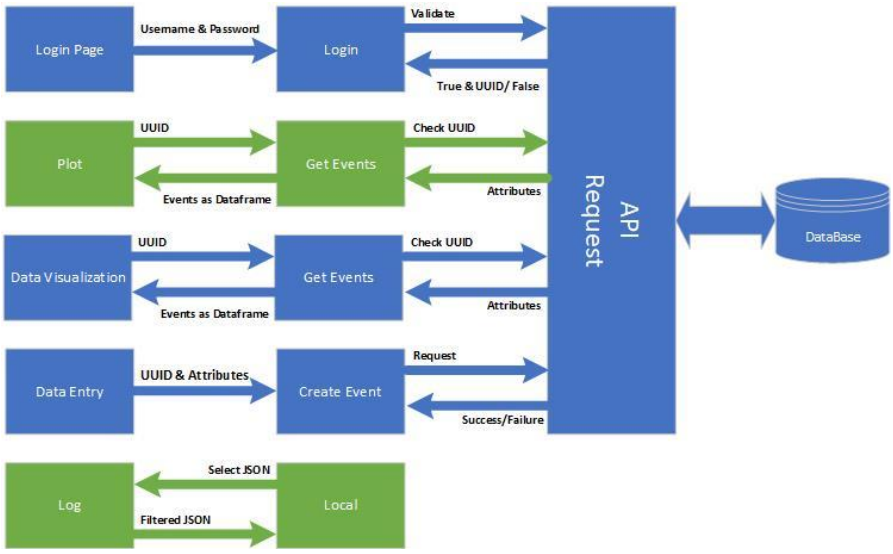

Figure 4: Desktop Low Level Design API

The Desktop low-level api design includes legacy tools (blue) and new tools (green). There are 4 main interactions in the Kare desktop application. The first is through the login page. On the login page the user enters their login information which includes an email and password and clicks the “Log in” button. This calls the “login” method in the API which takes the email and password as parameters. This API method makes a request to the database to see whether those credentials exist and are valid. If these credentials are correct, the login method returns true and returns that user's UUID. This validates the login, stores the UUID, and navigates the user to the graphing page. If the credentials are incorrect, the login method returns false and displays an invalid email/password prompt to the user. On the graphing page, when the user attempts to synchronize data, the user’s UUID is passed to the “getEvents” method in the API. This API method makes a request to the database to gather all the attribute data from that user. If successful, the “getEvents” method returns true and a .json formatted attribute data file that can be used in the graphing page to graph that data. If unsuccessful, the method returns false back to the graphing page. Third, on the statistics_ page, when the user attempts to synchronize data, the user’s UUID is passed to the “getEvents” method similar to home_page.py. Finally, there are the data entry capabilities that are done through the data entry page. When the user attempts to enter new data into the database through this page, once all necessary fields are complete and the “save” button is clicked, the user's UUID and the data that is entered into the attribute fields are formatted and passed into the “createEvent” API method. This API method makes a request to the database, adding those attributes into the database that match with the passed in UUID. If successful, the user is displayed a success message and can choose to

enter more data. If unsuccessful, the user is displayed a message indicating the failure to update the database with the newly entered data.

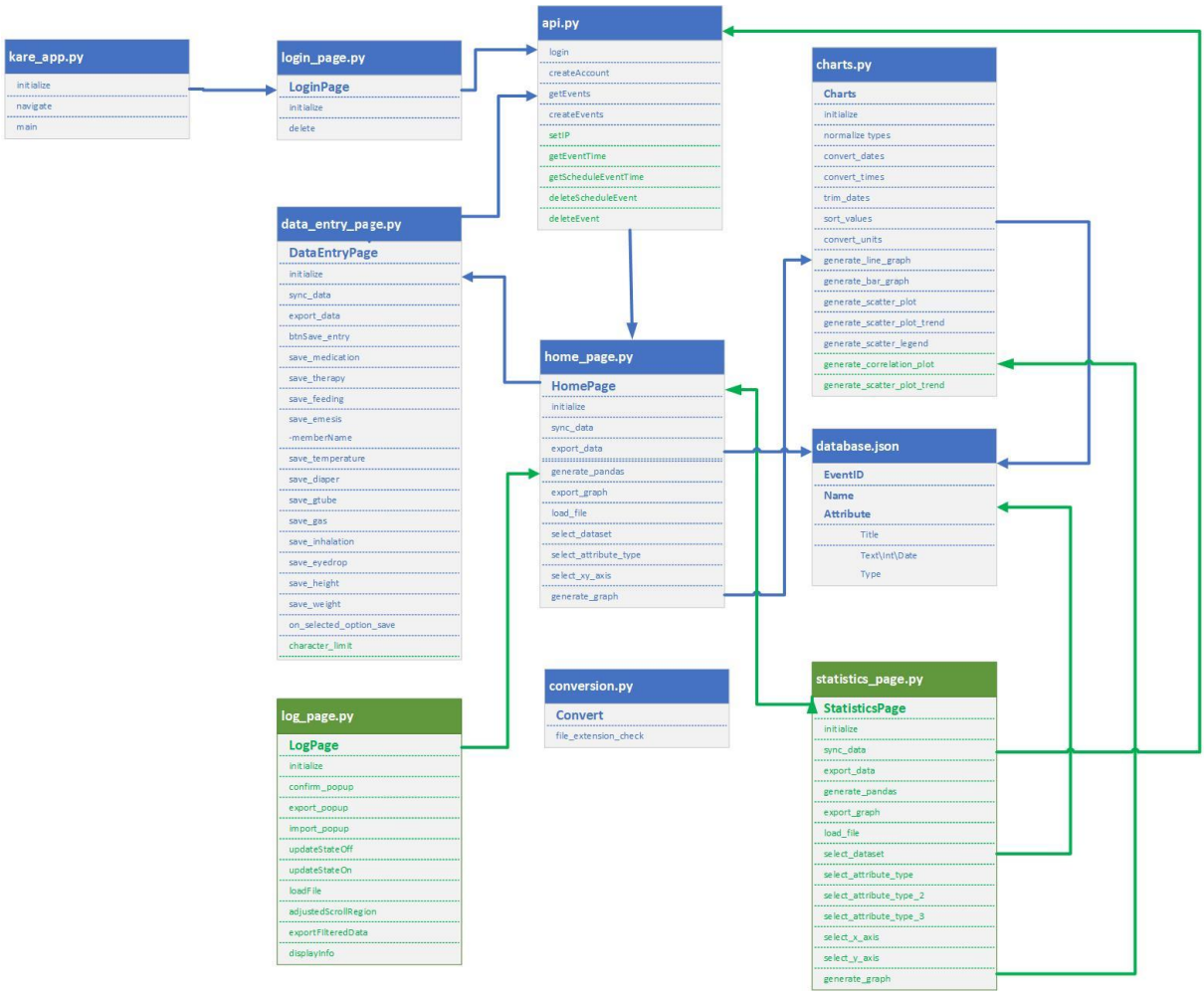

Figure 5: Desktop Low Level Design

The Desktop low-level api design includes legacy code (blue) and new code (green). The program functions by beginning with kare_app.py which contains all TKInter code for navigating pages. This automatically calls login_page.py which creates the LoginPage Class. This Class verifies entered data in the Email and Username fields, using the functions inside api.py to make a request to the MySQL database. Assuming the entered IP address is correct this will be verified and automatically create the HomePage class using home_page.py. The other pages available from this point are mutually navigable, are log_page.py which creates the LogPage Class, data_entry_page.py which creates the DataEntryPage class and statistics_page.py which creates the StatisticsPage Class. Both the HomePage and StatisticsPage class call on Charts.py to create plots.

Server Design

Server High Level Design

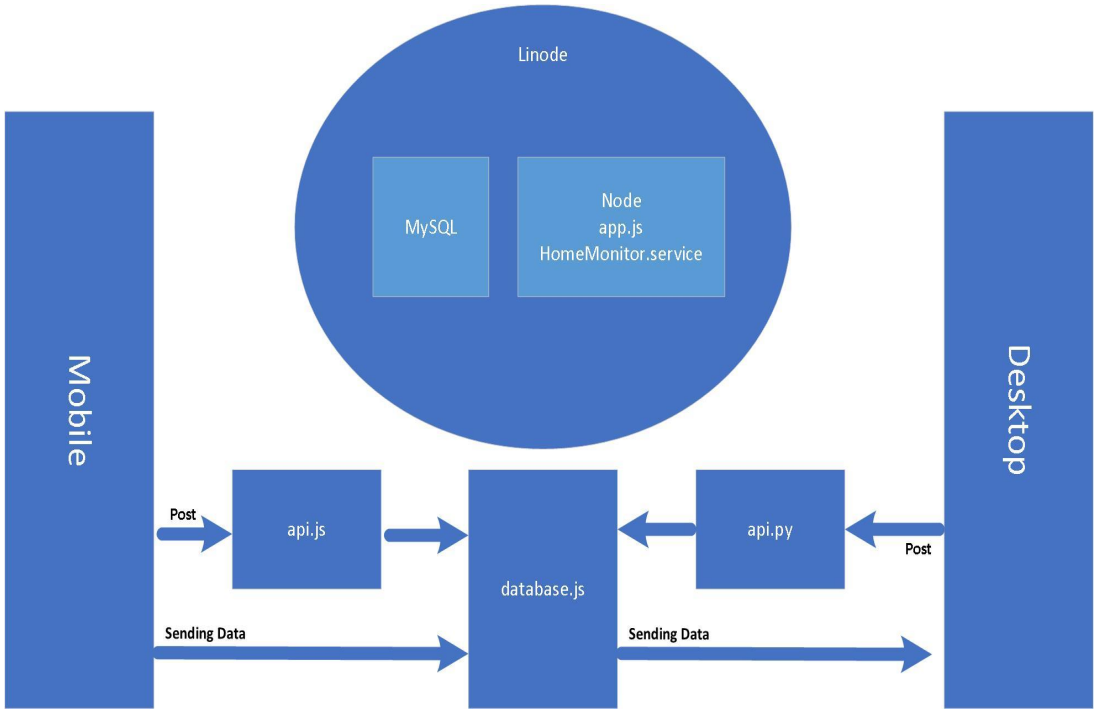

Figure 6: Server High Level Design

The server architecture is designed to facilitate the transfer of data from the mobile app to the desktop app. The server is a linode server running on Ubuntu 20.04, with a LNMP stack. It houses a SQL database and a Node codebase. The mobile and desktop issue API calls through api.js and api.py respectively. These are posted to the IP address of the server, and carry commands that are interpreted by database.js. These commands are then reinterpreted as queries to the database.

Server Low Level Design

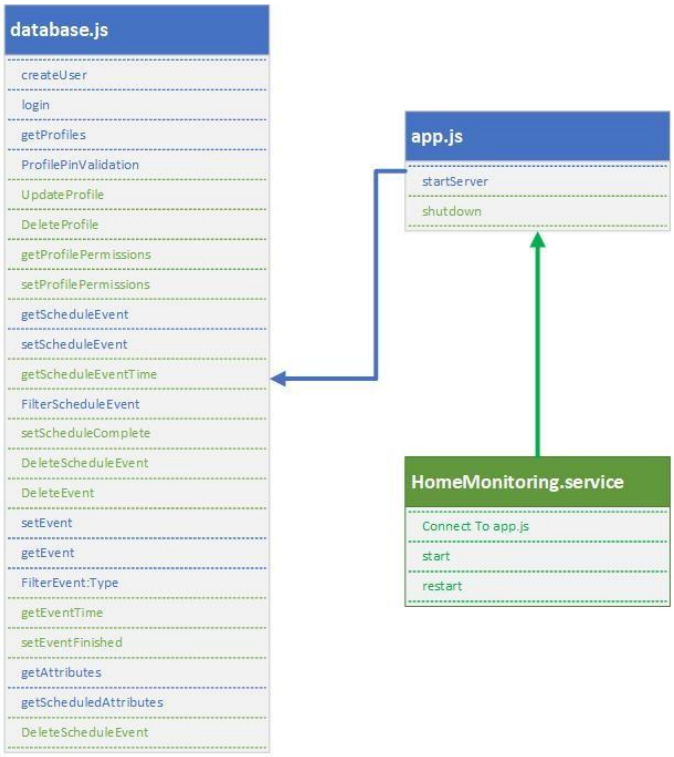

Figure 7: Server Low Level Design

Blue signifies old code. Green signifies new code. database.js is a collection of API calls that are interpreted as queries to the SQL server. It is run by app.js, which runs Node on Port 80. A shutdown process was added to allow the process to shutdown gracefully. HomeMonitoring.service is a system file that allows app.js to run at all times.

Entity Relationship Diagram

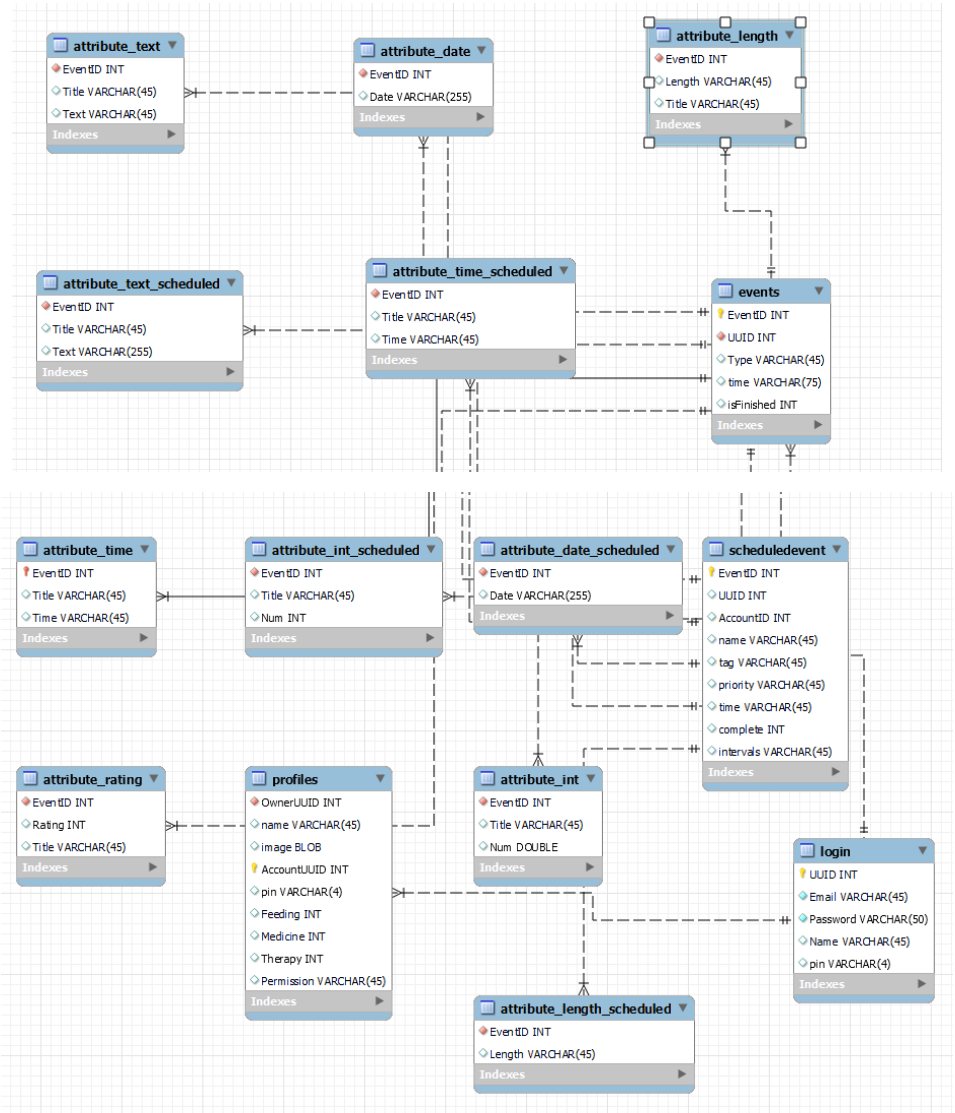

Figure 8: Server ERD Diagram

The entity relationships are structured around events and scheduled events. Each has an environment of attribute tables.

External Interface and Hardware Requirements

User Interfaces

The primary interface for the Kare 2.0 app will be a mobile application, designed for Android platforms. The app will feature a login screen for users to enter their credentials to access the app. After successful login, users will be directed to the home screen which will allow the user to access home care, the event schedule, and the event log.

A desktop app will also be available to visualize data and import and export data.

Hardware Interfaces

This application will come in two different versions. The parents/legal guardians and other caretakers will use the mobile application, while the desktop application is intended for only the primary caretaker. The following subcategories provide more details regarding the intended hardware that the Kare Application will operate on.

Mobile

The caretakers of the child will be able to utilize the software on Android mobile devices. It will not be available on iOS devices. The Android mobile device is expected to have a functioning touch screen and be able to access the internet.

Desktop

The parents/legal guardians of the child will be able to utilize the software on laptop and desktop computers. The computer is expected to be able to connect to the internet. To interact with the application, the machine is expected to have a functioning monitor, keyboard, and mouse. The computer is also required to have a minimum of 1GB of RAM to run the application and at least 100MB of free storage space to export graphs and data.

The desktop app runs on Windows 10 or later. It requires 100 MB of open storage space and 1 GB of RAM. The database requires a Debian server with 1 GB of RAM and 1 CPU.

Software Interfaces

The mobile app is built in React Native with Expo Go support. The desktop app is built using Python. The database requires a LNMP stack (Linux, Node, MySQL, PHP).

Communications Interfaces

The database server requires an SSL certificate. It is accessed using SSH.

System Features

Mobile Features

Login

Description

The software will allow users who have previously created an account to log in to the mobile application.

Use Case

The user needs to have an existing account to perform this function.

1. The user enters a valid email address into the appropriate text box.
2. The user enters the password that they used when creating the account.
3. The user taps the “Log In” button.

Possible Errors and Handling

A possible error that can occur is that the user can enter an email or password that is not saved in the database or linked to an account. If this happens, software will display an error message telling the user that the email or password is incorrect.

Priority

The priority of this requirement is high because the user will need to log in to the application to have access to the rest of its features.

Feasibility

This requirement is feasible.

Sign Up

Description

The software will allow users who have not previously created an account to sign up for the Kare Application.

Use Case

1. On the Log In page, the user taps the “Sign Up” button.
2. The user enters their first name, last name, and email address in the appropriate text boxes.

3. The user enters a password and confirms it in the appropriate text boxes.
4. The user enters a 4-digit pin that will be associated with the primary caretaker profile in the account.
5. The user submits the data to create the account.

Possible Errors and Handling

Possible errors that could occur are invalid or empty entries in the text fields. To handle this, once the sign-up form has been submitted the software checks each field to ensure that it is filled and has the appropriate formatting. If an error is found in any field, an appropriate error message will be displayed to the user.

Priority

The priority for this requirement is high because users need to be able to create an account to utilize the features in the mobile application.

Feasibility

This requirement is feasible.

Profile Selection

Description

The software will allow the user to select a profile from all existing profiles linked to the logged-in account.

Use Case

The user must be logged in to access this page.

1. The user taps on the icon of the profile that they want to continue with.
2. The user enters the passcode associated with that profile.

Possible Errors and Handling

A possible error that could occur is the user entering an incorrect passcode when they select a profile. To handle this error, the application will notify the user that the passcode is incorrect and will not let them continue further into the app if the wrong passcode is entered.

Priority

The priority of this requirement is high because different profiles have different authorizations of what data can be seen and logged for the child.

Feasibility

This requirement is feasible.

Profile Update/Create/Delete

Description

The software will allow users to update existing profiles linked to the account, create new profiles that are then linked to the account, or delete select profiles from the account.

Use Case

The user must be logged in to access this feature.

1. The user clicks on the edit icon on the top right corner of the screen.
2. To update an existing profile, the user taps a profile icon.
3. The user then can change the profile's name, passcode, and accessibilities.
4. The user then taps the update button to update the information in the selected profile.
5. To create a new profile, the user taps the profile icon containing a person figure with a plus sign next to it.
6. The user then fills out the profile name, passcode, and selects what accessibilities the profile will be granted.
7. The user then taps the create button to finish creating the new profile.
8. To delete an existing profile, the user taps a profile icon.
9. The user then taps the delete profile button to delete the profile.

Possible Errors and Handling

Possible errors could be invalid or no information entered in the fields of the edit/create profile form. To handle this, the software will check all fields once the form has been submitted and will display an appropriate error message. Another error that could occur is if someone who is not the primary caretaker attempts to update a profile to give it more permissions. To fix this, the primary caretakers passcode could be required to access the profile editing page.

Priority

The priority of this requirement is medium because having multiple profiles for different caretakers would be nice, but is not required for the app to function.

Feasibility

This requirement is feasible.

Schedule Event

Description

The software will allow users to create, edit, and view scheduled events regarding the child in chronological order.

Use Case

The user is required to be logged in to the account and be using a profile with appropriate permissions to use this feature.

1. The user taps on the schedule icon in the bottom left corner of the screen.
2. The user can scroll on the display to view all existing scheduled events for the child.
3. The user can tap on the plus icon in the bottom right corner to create a new scheduled event.
4. The user fills out the necessary information for the event.
5. To edit an existing event, the user taps on the triple-dot symbol on the top right of an event icon.
6. The user updates the necessary information.

Possible Errors and Handling

Some possible errors are that the user can leave fields blank or enter invalid information when creating or updating an event. To handle this, once the form has been submitted, the software will check that each field is filled and contains valid data. If a field is empty or invalid data is entered, the software will notify the user with an appropriate error message.

Priority

The priority of this requirement is high because keeping an updated and valid schedule is important for the child's health.

Feasibility

This requirement is feasible.

Log Event

Description

The software will allow users to log data about events that have already occurred regarding feeding, medicine, and therapy.

Use Case

The user is required to be logged in to the account and be using a profile with appropriate permissions to use this feature.

1. The user taps on the log icon in the bottom left corner of the screen.
2. The user taps on the type of event they want to log.
3. The user fills out the necessary fields for logging the event.
4. The user taps the done button on the bottom of the screen to submit the log.

Possible Errors and Handling

Some possible errors are that the user can leave fields blank or enter invalid information when logging an event. To handle this, once the form has been submitted, the software will check that each field is filled and contains valid data. If a field is empty or invalid data is entered, the software will notify the user with an appropriate error message.

Priority

The priority of this requirement is high because accurate data regarding the child's health and activities needs to be logged for the app to help caretakers and medical professionals.

Feasibility

This requirement is feasible.

Logging Home Care Data

Description

The software will allow users to log information about spontaneous activities that occur with the child such as current temperature, emesis, and inhalation/oxygen.

Use Case

The user is required to be logged in to the account and be using a profile with appropriate permissions to use this feature.

1. The user taps on the Home Care icon in the bottom middle of the screen.
2. The user taps on the type of activity they want to log.
3. The user fills out the necessary fields for logging the activity.
4. The user taps the submit button on the bottom of the screen to submit the log.

Possible Errors and Handling

Some possible errors are that the user can leave fields blank or enter invalid information when logging an event. To handle this, once the form has been submitted, the software will check that

each field is filled and contains valid data. If a field is empty or invalid data is entered, the software will notify the user with an appropriate error message.

Priority

The priority of this requirement is high because events with important data may occur with the child that are not scheduled.

Feasibility

This requirement is feasible.

Repeating Events

Description

The software must allow for users to set an interval for events to repeat. This could be as short as one hour to as long as one month. When the time has passed, the event will be marked as unfinished and able to be logged again.

Use Case

The user is creating a scheduled event. They want it to repeat.

Possible Errors and Handling

An invalid number could be provided for the amount. An invalid type of interval will not be provided, because the user selects from a predetermined list, such as hour, day, week.

Priority

The priority of this requirement is high because events need to be repeated on certain intervals.

Feasibility

This requirement is feasible.

Timer

Description

The software will allow users to time the length of an event within the app.

Use Case

The user is required to be logged in to the account and be using a profile with appropriate permissions to use this feature.

1. The user opens the log screen for an event that is either scheduled or spontaneous.
2. The user taps the start button for the event and the app saves the current time and date.
3. The user taps the stop button for the event and the app saves the current time and date.
4. The length of the event is then filled out and displayed to the user.

Possible Errors and Handling

One possible error could be that the user may forget to start or stop the timer during the event. To handle this the software can note the time and date of when the event form was opened and note the time and date when the event form is submitted. The open and close times will be overwritten if the user taps the start and stop buttons respectively.

Priority

The priority for this requirement is high.

Feasibility

This requirement is feasible.

Desktop Features

Login

Description

The software will allow users who have previously created an account to log in to the desktop application.

Use Case

The user is required to have an already existing account to use this feature.

1. The user enters a valid email address into the appropriate text box.
2. The user enters the password that they used when creating the account.
3. The user taps the “Log In” button.

Possible Errors and Handling

A possible error that can occur is that the user can enter an email or password that is not saved in the database or linked to an account. If this happens, software will display an error message telling the user that the email or password is incorrect.

Priority

The priority of this requirement is high because the user will need to log in to the application to have access to the rest of its features.

Feasibility

This requirement is feasible.

Data Visualization

Description

The software will allow users to create data visualizations (graphs) using data collected from the mobile application, or from an imported JSON file.

Use Case

The user must be logged in to access this feature.

1. In the top left of the screen, the user either clicks the from file icon to load data from a local JSON file, or the sync data icon to load data from the cloud database.
2. The user selects the dataset that they would like to use for visualization.
3. The user selects the type of data they would like to plot, along with what data points to put on the x and y axes.
4. The user selects the time frame for the data that they would like to use in the plot.
5. The user selects the type of chart they would like to generate.
6. The user clicks the generate button to create a plot based on inputted parameters.

Possible Errors and Handling

One possible error is that the user could forget to fill out a field when creating the chart. To handle this, the software does not generate a plot until all fields are filled out.

Priority

The priority of this requirement is high because the desktop application would be minimally useful to users if it could not generate charts.

Feasibility

This requirement is feasible.

Database Access

Description

The software will allow users to pull data from the cloud database to use in plotting a graph or saving as a CSV or JSON file.

Use Case

The user must be logged in to access this feature.

1. The user clicks on the sync data icon in the import section on the top left corner of the screen.
2. The application retrieves child data linked to the account from the cloud database.

Possible Errors and Handling

One possible error could be a lack of internet access. To handle this software could warn the user that the application cannot currently connect to the database and request that the user check their internet connection.

Priority

The priority of this requirement is high because the desktop application needs access to the database to retrieve information logged on the mobile application.

Feasibility

This requirement is feasible.

Importing From File

Description

The software will allow users to import data from local JSON files to be used in generating graphs.

Use Case

The user must be logged in to access this feature.

1. The user clicks on the from file icon in the import section on the top left corner of the screen.
2. The user browses their local files and selects the file that they wish to import.

Possible Errors and Handling

One possible error could be an error occurring while reading the selected file. To handle this, the software could alert the user if this occurs and request that they try to import the file again.

Priority

The priority of this requirement is high.

Feasibility

This requirement is feasible.

Adding Entries to Database

Description

The software will allow the user to log events and save them to the cloud database.

Use Case

The user must be logged in to access this feature.

1. The user clicks the record icon in the “New” section of the toolbar on the top of the screen.
2. The user selects the type of event they want to log.
3. The user fills out the necessary information for the event.
4. The user clicks the save button to submit the form and save it to the database.

Possible Errors and Handling

Some possible errors are that the user can leave fields blank or enter invalid information when logging an event. To handle this, once the form has been submitted, the software will check that each field is filled and contains valid data. If a field is empty or invalid data is entered, the software will notify the user with an appropriate error message.

Priority

The priority of this requirement is high.

Feasibility

This requirement is feasible.

Exporting Graphs/Data

Description

The software will allow users to export generated plots as PNG files and export the data included in generated plots as CSV or JSON files.

Use Case

The user must be logged in to access this feature.

1. To export a graph, the user clicks the graph icon in the “Export” section of the toolbar on the top of the screen.

2. The user then gives the graph a name when prompted and saves the PNG at the desired location on the computer.
3. To export data in a graph, the user clicks the data icon in the “Export” section of the toolbar on the top of the screen.
4. The user then gives the file a name when prompted, selects the file type, and saves the file at the desired location on the computer.

Possible Errors and Handling

One possible error is that the user could enter an invalid name for the file. To handle this, Windows’ built-in error checking feature will alert users if they enter an invalid name for the file.

Priority

The priority of this requirement is high because the primary caretakers will need to export data and graphs in order to send them to doctors and medical professionals looking over the child.

Feasibility

This requirement is feasible.

Adding More Plot Options

Description

The software will allow users to create a variety of plots using data imported from a file or from the database.

Use Case

The user must be logged in to access this feature.

1. The user follows the steps in the section on Data Visualization up to step 4 in the Use Case subsection.
2. The user can select the type of plot they want to generate, such as scatter plots, bar plots, box plots, and regression plots.

Possible Errors and Handling

One possible error is that the user could accidentally select the wrong type of plot. To handle this, the user can simply go to the drop-down menu and select a different plot.

Priority

The priority of this requirement is high because data visualization is the primary function of the desktop application.

Feasibility

This requirement is feasible

Database Clearing

Description

The desktop application must check the age of the data stored in the server database, and if that data is over one month old, it must send a request to clear the database.

Use Case

1. The user must have data in the database that was created over one month ago.
2. The user must log in to the desktop application.
3. The user must select the 'Sync Data' button on the desktop application.
4. The desktop application will check the age of the data that it received from the database.
5. Given that the data is over a month old, the user will be prompted to save the database data before deleting it.
6. The user will name and save the data, or decline and cancel the saving of data.
7. The user will be prompted to clear the database.
8. The user will select 'yes' and all database entries will be deleted.
9. If the user selects 'no' the data will remain in the database.

Possible Errors and Handling

There could be an error when attempting to delete from the database when the device that the application is running on is not connected to the internet.

Priority

High because the purpose of the desktop application is to handle and store data, it is not the database's job to store data.

Feasibility

This requirement is feasible and has been implemented.

Implementing a Log Page

Description

The software will allow the user to view the data of logged events from local JSON data files from the desktop application.

Use Case

The user must be logged in to access this feature.

1. The user clicks on the “Logs” icon from the home screen to be taken to the log page.
2. The user clicks on the “Browse Files” button under Select Dataset and selects the JSON file that they would like to use in the Log Page.
3. The user can choose to select an Event Type filter from the dropdown menu. No Filter is selected by default.
4. The user then selects a date range among the radio buttons to filter events that are within the selected time range.
5. If the “Custom” radio button is selected, then the user can manually select a start date and an end date using calendar widgets.
6. Once the user has selected the desired filters, they can click the “Load” button to load the data from the JSON file into the page with the selected filters in place.
7. To view specific event data, the user can use the scrollbar to scroll down to the list of events and click on an event button.
8. When an event button is clicked, the page will display all information associated with the event on the display to the right of the event list.

Possible Errors and Handling

Some possible errors are that the user could input invalid information in the fields while filtering the loaded data list. To handle this, the software limits user input as much as possible to only allow valid information to be entered in the filter box. This limits the amount of possible errors that can occur on the page.

Priority

The priority of this requirement is high because there was no way to view logged information on the desktop application outside of making graphs.

Feasibility

This requirement is feasible.

Comparison Graphs

Description

The user will be able to compare data of different types in the desktop application for different attributes of the data, such as time, dose, date, etc.

Use Case

1. The user will log into the desktop application.
2. The user will select ‘plots’ from the top toolbar of the application.
3. The user will select the dataset to take data points from.
4. The user will select an entry type and an attribute type related to that entry.
5. The user will select another entry type and another attribute type related to that entry.

6. The user will select a dataset to normalize the graph to.
7. The user will select the type graph to produce.
8. The user can then view a graph comparing the data and attributes.

Possible Errors and Handling

There can be an error where the type of the data or attributes that are being compared are incompatible with each other. This is handled by not producing the graph. The user can then change their selections to produce a different graph.

Priority

The priority is high since it fulfills the purpose of the desktop application.

Feasibility

This requirement is feasible and has been implemented.

Deployment

Description

The application must be prepared and functional for deploying onto user devices, for public use.

Use Case

1. The application will be packaged into a functional .exe file
2. The dependencies for the application (the database.json and the images file) will be included with the .exe.

Possible Errors and Handling

The file may not detect that the database.json file exists in the same file.

Priority

High priority, the client specifically requested this of us

Feasibility

Not feasible.

Database Features

Data Storage

Description

The software will provide a secure and efficient mechanism for storing data in a cloud database.

Use Case

1. When data is submitted to the database, it is saved in an organized fashion that can be retrieved from the desktop application later.

Possible Errors

One possible error is that some data could be lost when being sent to the database.

Priority

The priority for this requirement is high because the application requires a database to function.

Feasibility

This requirement is feasible.

Login Details

Description

The software will allow users to create and store login details in the database.

Use Case

1. When a user creates an account on the mobile application, that data will be stored in the database.
2. When a user logs into the mobile application, that information is checked against data saved in the database to verify the user.
3. When a user updates a profile on the mobile application, that information is saved in the database.
4. When a user logs into the desktop application, that information is checked against data saved in the database to verify the user.

Possible Errors

One possible error is that some data could be lost when being sent to and from the database.

Priority

The priority of this requirement is high because medical information is being stored in the database, and that data must be secure.

Feasibility

This requirement is feasible.

Event Submissions

Description

The software will allow users to submit events and store them in the database.

Use Case

1. When a user fills out a log for an event in the mobile application, that data is saved in the database.

Possible Errors

One possible error is that some data could be lost when being sent to and from the database.

Priority

The priority of this requirement is high because without it, the application would not be able to function properly.

Feasibility

This requirement is feasible.

Scheduled Events

Description

The software will allow users to schedule events and store them in the database.

Use Case

1. When the user fills out and submits a form regarding a scheduled event, that data will be sent to and stored in the database.

Possible Errors

One possible error is that some data could be lost when being sent to and from the database.

Priority

The priority of this requirement is high because a major feature of the mobile application would not be able to function without it.

Feasibility

This requirement is feasible.

User Profiles, Passcodes, and Permissions

Description

The software will allow users to store profiles, passcodes, and permissions in the database.

Use Case

1. The user creates a new profile.
2. The data for that profile is stored in the database and the passcode associated with that user is required to use the new profile.
3. The user updates an existing profile.
4. The new data for the profile is stored in the database and updated as necessary.

Possible Errors

One possible error is that some data could be lost when being sent to and from the database.

Priority

The priority for this requirement is high because certain profiles should only be able to access part of the mobile application that they are authorized to access.

Feasibility

This requirement is feasible.

Other Nonfunctional Requirements

Performance Requirements

The software application should provide an efficient mechanism for accessing data in the database to ensure optimal performance. The system should be able to handle enough data and multiple users accessing the system simultaneously without compromising its performance. The application should be designed to operate with a response time of no more than 1 second for standard requests. Additionally, the server requires a minimum internet speed of 10 MB upload and download for optimal performance. These requirements will ensure that users can access and use the system effectively without any performance issues.

Safety Requirements

The software application must ensure a secure connection between the client and server to protect data transmission. There should be an extensive test suite, as the product should have minimal bugs.

Security Requirements

The software application must comply with HIPAA regulations for handling medical information. The mobile app must have a secure login system with unique email and password to ensure only authorized individuals can access the app and view personal health information. The app must also have role-based access control to restrict access based on permissions. Users must enter a valid passcode to access their account after selecting their profile. The desktop application must have a login system for primary guardians of the child who are covered under HIPAA. The software must implement encryption to secure data stored in the database. Any security or privacy certifications required by external policies or regulations must be satisfied.

Software Quality Attributes

The product should prioritize usability, as the target audience may not be technologically savvy. The interface should be simple, uncluttered, and easy to navigate. The product should be customizable. The application should have high availability to ensure that users can access it at any time. The product should be reliable, robust, and maintainable to minimize downtime and ensure long-term use. The app should also be adaptable and flexible to accommodate future changes or updates.

Installation

Mobile

Refer to the attached mobile manual for installation instructions.

Desktop

Refer to the attached desktop manual for installation instructions.

Server

Refer to the attached server manual for installation instructions.

User Documentation

Mobile

Refer to the attached mobile manual for usage instructions.

Desktop Example 1: Copy Data for a Doctor's Appointment

See the Kare Desktop User Manual Page 37

Desktop Example 2: Review a Caregiving Event

See the Kare Desktop User Manual Page 38

Server

Refer to the attached server manual for usage instructions.

Conclusion

In conclusion, this document provides an introduction to the project that covers its purpose, as well as the scope and intended audience of the application. It addresses the functionality of the application, and software and hardware requirements to run it on a system. This document delves into the details regarding functional and nonfunctional requirements of the different parts of the project, as well as the unique requirements set by the client regarding testing.

The Kare Application is meant to improve the care of medically complex children and ease the burden on parents/legal guardians by providing a secure and user-friendly mobile and desktop application. After a medically complex child is allowed to leave the hospital, the support structure for the child and parents/legal guardians typically crumbles. Kare's goal is to help provide a structure that parents/legal guardians can lean on to provide an acceptable standard of care for their child and prevent well-meaning parents from losing custody of their child. The mobile application helps caretakers log and schedule events/activities with the child to store. The database serves as an intermediary between the mobile application and the desktop application temporarily storing data from a mobile device and sending that data to a computer running the desktop application. The desktop application can pull data from the database or from a local file on the system to generate graphs and plots over periods of time that can be used by medical professionals to help them better care for the child.

In the course of this project, we learned many things and would have changed many things. Of those, it would have been helpful to have had a project manager to oversee the entirety of the project. This could have aided in communication between sub-groups and kept everyone on the same page. It would have also been helpful to have had more time to become familiar with the code, so that we could have had more time to focus on getting a working, high quality product. We learned that, through the extensive onboarding process that came along with this project, that it is not always helpful to throw more manpower at a problem. We also learned that it takes time to get used to a project created by others, and can lead to much frustration. Overall, it was a good and interesting experience to have as a computer scientist, and we enjoyed the opportunity.