Medical Hub

# Goals

Throughout this project, I intend to build an application that is capable of primarily storing data regarding a user’s daily experience of symptoms and possible triggers. The application will be divided into several layers including the database, a dedicated API, and a website for inserting and viewing data. Extensions to this project will include a mobile application accessing the API and automatic data analysis for viewers to better understand the data.

# Motivation

The primary motivation stemmed from personal accounts from friends requesting an easy-to-use method for tracking triggers for rosacea (and similar conditions) as there were many potential triggers and identifying them would prevent as many flair ups thus increasing quality of life. Another motivation is the knowledge and experience such a project will provide. I do intend to use technologies that I am not entirely familiar with and profit from the experience that building this application, but I will not be doing so at the expense of the project. Each technology has been chosen due to its applicability in this project and many are industry standards.

# Structure

The route through this project begins with research into the goal and what technologies are best suited for this project. Next, I’ll discuss the features this application should include and ending with the order in which these technologies will be implemented.

# Database Design

The initial consideration in the database design process is the selection of the appropriate database type. The two primary options in widespread use are relational and non-relational databases. While a comprehensive exploration of their differences falls outside the scope of this project, in brief, relational databases employ tables and relationships to manage highly structured data, whereas non-relational databases utilize key-value pairs and handle data with less rigidity in its structure.

Initially, my inclination leaned towards a relational database due to my familiarity with it. However, upon further reflection, I've concluded that this might not be the most suitable choice as each piece of data in this project exhibits a high degree of variability in both form and function. Consequently, using a relational database would necessitate numerous requests to retrieve data from multiple tables. This stands in contrast to a non-relational database, where all data can be stored under a single key, streamlining data retrieval and management.

Subsequently, I embarked on an exploration of various NoSQL databases, ultimately settling on MongoDB. This choice was influenced by MongoDB's status as the reigning NoSQL database at the time of this writing, accompanied by its wealth of available learning resources and comprehensive support for NoSQL database implementation. Given our utilization of a document-oriented database, the design does not entail excessive complexity. Our database will consist of two primary collections: one for users and another for entries. Each entry within the collection will encompass a range of attributes, including trigger values, symptom descriptions, emotional states, and essential metadata such as entry dates.

# Stack – Technologies

Now that the database has been chosen, the rest of the technologies can be designed to work cohesively. The chosen stack after consideration is MongoDB, Node JS, Express, React. I have chosen these technologies for several reasons. Firstly, the choice of using JavaScript came from the fact that MongoDB returns JSON like files which JavaScript has integrated functions for handling. JavaScript can also be used for each technology which means I can write the whole application in a single language.