System Design for Software Engineering

Team 14, Reach
Aamina Hussain
David Morontini
Anika Peer
Deep Raj
Alan Scott

January 17, 2024

1 Revision History

Date	Version	Notes
Date 1	1.0	Notes
Date 2	1.1	Notes

2 Reference Material

SRS Documentation
HA Documentation
MG Documentation

2.1 Abbreviations and Acronyms

symbol	description
SRS	Software Requirements Specification
HA	Hazards Analysis
MG	Module Guide

Contents

List of Tables

List of Figures

3 Introduction

[Include references to your other documentation —SS]

4 Purpose

[Purpose of your design documentation —SS]
[Point to your other design documents —SS]

5 Scope

[Include a figure that show the System Context (showing the boundary between your system and the environment around it.) —SS]

6 Project Overview

6.1 Normal Behaviour

A normal user scenario for REACH would be:

1. User opens the website REACH.
2. User logs in to their account.
3. User inputs their personal information into a profile and saves it.
4. User selects the previously saved profile and inputs data about their condition.
5. Reach displays a list of applicable clinical trials sorted by distance.
6. User selects a trial and chooses to contact the research coordinator for the trial.
7. Reach creates an email template for the user using the provided information.
8. User leaves the website.

Other scenarios are documented in the SRS Documentation

6.2 Undesired Event Handling

The handling of undesired events is documented in the Hazards Analysis Documentation.

6.3 Component Diagram

N/A no hardware

6.4 Connection Between Requirements and Design

The connection between Requirements and Design are specified in the Module Guide Documentation.

7 System Variables

N/A

8 User Interfaces

[Design of user interface for software and hardware. Attach an appendix if needed. Drawings, Sketches, Figma —SS]

9 Design of Hardware

N/A

10 Design of Electrical Components

N/A

11 Design of Communication Protocols

12 Timeline

[Schedule of tasks and who is responsible —SS]

A Interface

[Include additional information related to the appearance of, and interaction with, the user interface —SS] The following images are mockups of the user interface for REACH. The mockups were created using Figma and are subject to change.

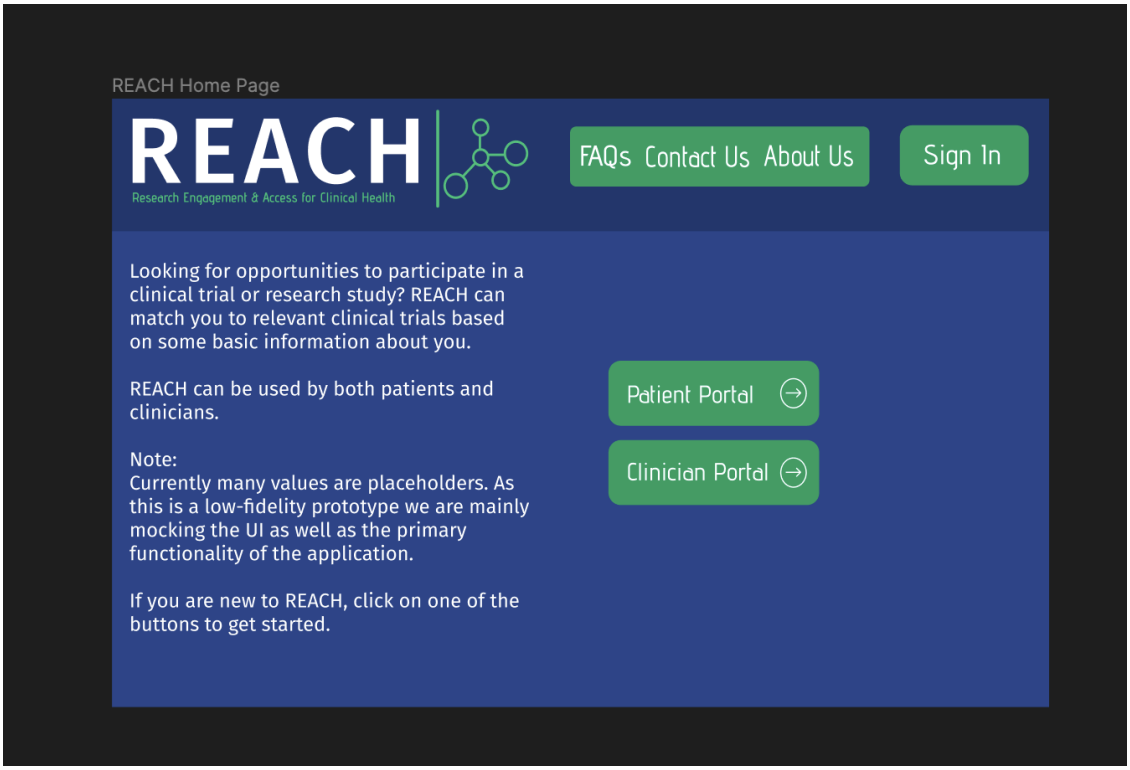

Figure 1: REACH home page

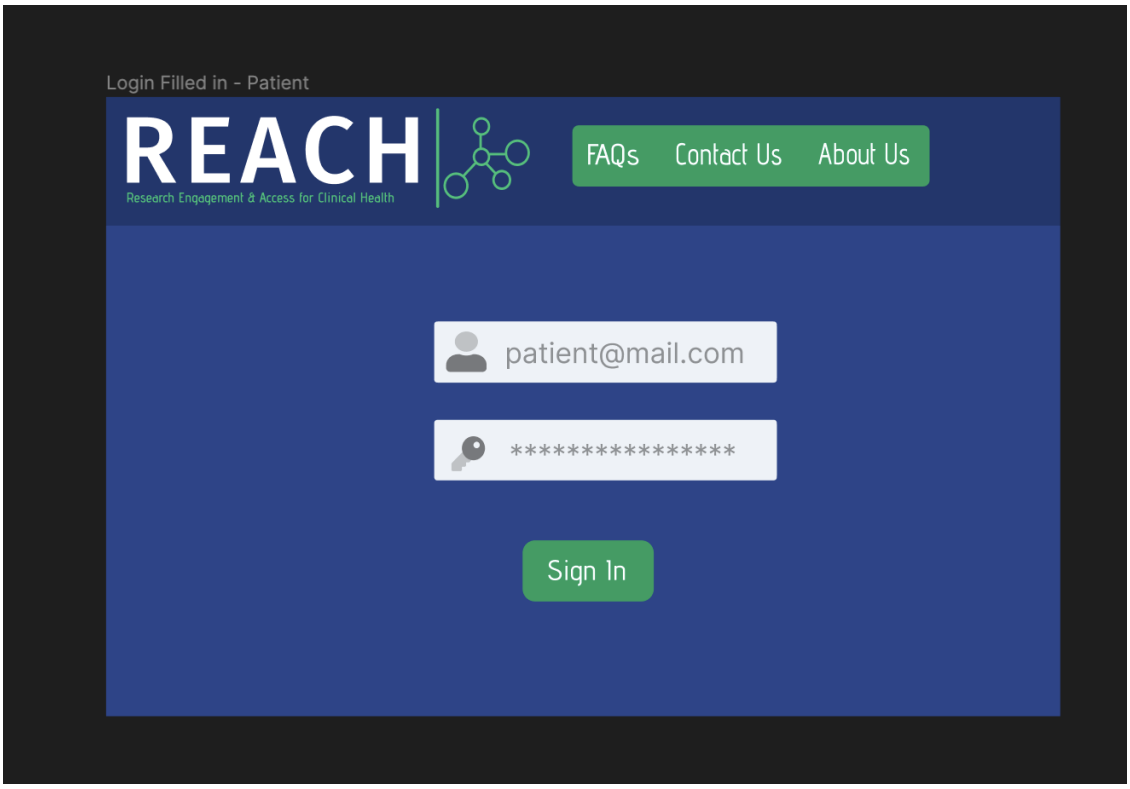

Figure 2: REACH login page

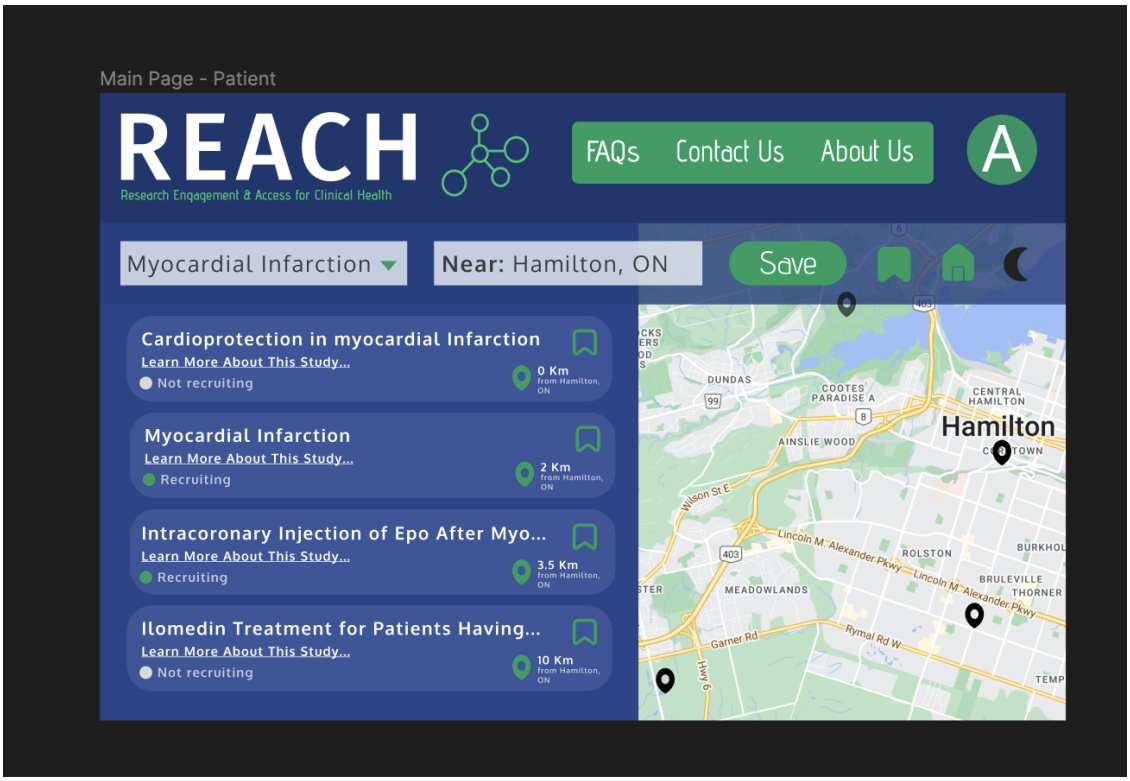

Figure 3: REACH main page with query results

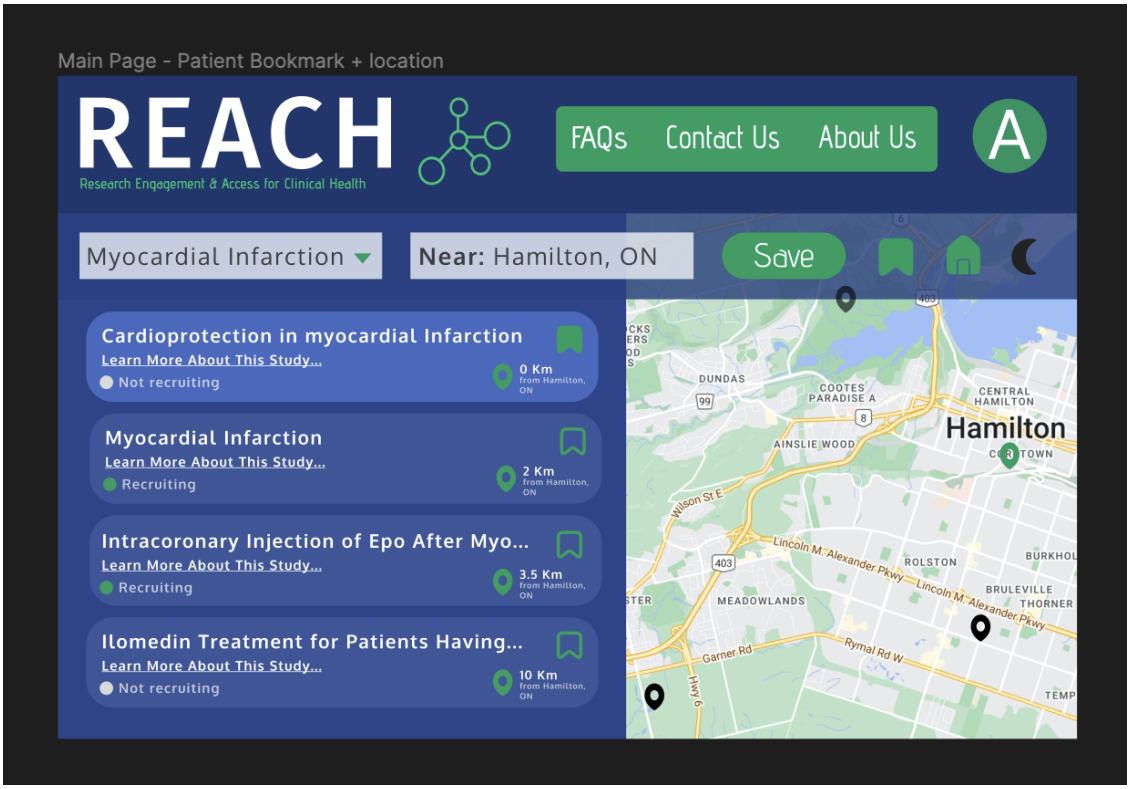

Figure 4: REACH main page with bookmark functionality

To see the mockups with their respective interactions visit the Figma Project.

B Mechanical Hardware

N/A

C Electrical Components

N/A

D Communication Protocols

E Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Problem Analysis and Design. Please answer the following questions:

1. What are the limitations of your solution? Put another way, given unlimited resources, what could you do to make the project better? (LO_ProbSolutions)

   There are a few improvements that can be made given more (or unlimited) resources. Naturally, the resources that would be most valuable are time and money, so the improvements are primarily focused around limitations due to a lack of one (or both) of these resources.

   First, while time wouldn't be an issue, due to the lack of financial resources, we are restricted in the number of notifications that can be sent to our users. As a result, we made the design decision to only include an email notification system, and ignore other types of notification systems, such as sending notifications via text message. With more money, the number of notifications we can send per unit time would increase, and we would then be able to explore these other types of notfications. Another limitation due to financial resources is the number of cloud resources that can be used. In order to keep costs low, the number of resources (such as cpus, storage, etc..) must be kept to a minimum. As a result, a limitation is posed on the system with respect to how many users can be using the system at one time. if we had more (or unlimited) money, we could greatly increase these resources, which would allow our system to support many more users at once.

   One design limitation due to the amount of time we have to implement the system, is the decision to use one clinical trial repository as opposed to several repositories. With more time, we would likely be able to design the system in a way that is suited towards using multiple repositories, however, since this would greatly increase the complexity of the application, we decided to stick with the most popular repository, clinicaltrials.gov. In the case of multiple repositories, more modules would be needed, and more methods would be required in these modules (whereas now, we only require 2 simple trial fetching/filtering modules).

2. Give a brief overview of other design solutions you considered. What are the benefits and tradeoffs of those other designs compared with the chosen design? From all the potential options, why did you select documented design? (LO_Explores)

   There were several design solutions that we considered for both the frontend and backend components of the application. Beginning with the backend modules/components, one alternative design solution was for the trial interface that would be used to interact with the clinicaltrials.gov website. Essentially, instead of having a separate module for the trial fetcher and filterer, these would be combined into one large module, which would do all the work related to trial interactions. The pros of this design (for this part of the application) is that there would be only one interface/abstraction that would need to be implemented, and it would be a bit simpler to implement. The main con is

that it would not be very maintainable/extendable, since the module would get messy very quickly. Additionally, we decided to split the modules since each module should only be hiding one secret, and in the first case, it would technically be hiding two (how the trials are fetched, and how they are filtered).

Another design solution we considered was for the frontend visualization/interface. Specifically, the interface that would allow users to view the different trials that they are eligible for. Instead of having the trials displayed vertically, with the map to the right of the trials, the design would be to have trials displayed vertically and horizontally across the page. When clicking on a trial for more information, you would be given the location of the trial. The pros of this design is that it is easier to implement, and simplifies the page as a whole, which might make the user a bit more relaxed when using the system. The cons are that the user is unable to easily see where the trial is located when searching through the trials, which is one of the most important pieces of information the user needs. As a result of this importance, we felt that the map should be included, so the user can efficiently search through the trials, using both the description and location on the map.

One final design solution that we considered was for the clinician/patient relationship. The design was that a user who logged in as a clinician would be able to save multiple patient profiles, and create searches based on these profiles, whereas a regular user (i.e., not a clinician) would only be able to create one profile. In comparison to the current design, there are not any pros/advantages to using this design, however there are a couple disadvantages. First, the relationship between the user data module and patient info module becomes more complex in the case that regular users can only create one profile. Additionally, (and most importantly), it is highly likely that a regular user might want to create multiple profiles, whether it be for themselves, or for their family/friends. For these two reasons, we decided it would be best to design the system in a way that supports both regular users, and clinicians to be able to create and save multiple profiles.