

Web Based Neurological Assessment

Students: Jordan Bravo (Team Lead), Nicolas Bello, Jonathan Oviedo

Mentor: Dr. Christian Poellabauer (Product Owner)

Instructor/Faculty: Dr. Masoud Sadjadi, Florida International University

PROBLEM

Patients with neurological disorders struggle to arrive to on-site practices for medical examinations and checkups with their medical provider. The examinations performed by the medical providers also require physical presence for completion.

An answer to this problem resides in the current prowess of technology and the ability to perform video calls online and perform tasks with results stored in a central database tied to that user for future viewing and examination. All in all, a software solution that aims to provide a unique and tailored experience where a user can view the necessary information they need to see and perform the tasks they are assigned by their providers. The providers can view the results of the examinations they've assigned patients along with their medical records while additionally being able to schedule appointments and participate in video calls.

The question remains in the "How?", and this project aims to answer that.

NeuroTask (placeholder name) aims to bridge the gap between the patient and provider through means of a virtual platform that allows for a seamless and unique experience for users.

SUMMARY

This is a brand-new project, meaning our team was tasked with building from the ground up and pushing a clear and concise architecture that allows for scalability and readability.

As team lead, I was tasked with choosing the technology stack and building the foundation. This included:

- Implementing NgRx into the project.
- Setting up Firebase along with Nicolas Bello.
 - Tailoring Authentication suite to Firestore documents.
 - Creating the schemas and collections for each entity.
 - Creating the Firebase Cloud Functions environment for creating server-side logic and admin functionality.
- Creating the Emulation environment for a localized Firebase instance.
- Declaring the project and folder structure with documentation.
- Writing and modifying different configuration files, implementing logging services and documenting each method.

After the groundwork was laid, I began developing the must-have essential features for any software service that interacts between multiple users:

- Registration forms for both patients and provider (Fig. 1A and 1B).
- A singular login page that integrates with Firebase Authentication (Fig 2A and 2B).
- A password reset feature, fully included with expiring tokens and email services.
- Searching for providers from different facilities to add as contacts (Fig. 3).
- A functioning contacts list with incoming and outgoing contact requests.
- A user drawer for logging out and accessing your profile settings (Fig. 3).
- Responsive design for mobile users.

Some other features that I added more as internal development stories were:

- Defining and implementing the test suite for Cloud Functions to integrate with the Emulator suite.
- Creating a script that will populate Firestore with data in bulk.
- Creating a super database service that abstracts different database operations from the other services.
- Troubleshooting and configuring the Firebase SDK to work correctly with Angular Observables and native functionality.

REQUIREMENTS

The requirements for this semester involved building a solid foundation and setting guidelines, along with designing an MVP and the data base.

The priority for the MVP for this semester included:

- A landing page with routes associated to registration and login.
- A password reset feature with tokens and validation to avoid reuse of the same link.
- A verification of provider accounts by facility admins to avoid any end user from creating a provider account with provider privileges.
- Valid authentication system with roles and different permissions.
- A dashboard akin to other patient portals seen in similar medical apps.
- A way to maintain and keep track of providers/patients you are interested in by means of a contact list, along with the ability to add providers/patients.

On the technical side of things, we required a simple architecture for future teams to pick up on and continue the work from previous semesters. This led to the use of Angular with its heavily opinionated structure, allowing teams to pick up and get running without having to worry about how the previous team worked and how they structured code.

We also opted for Firebase since it comes with everything a back-end service will mostly require while providing robust documentation and integration with multiple systems. This includes authentication, a database, and server-side handling to name a few.

IMPLEMENTATION

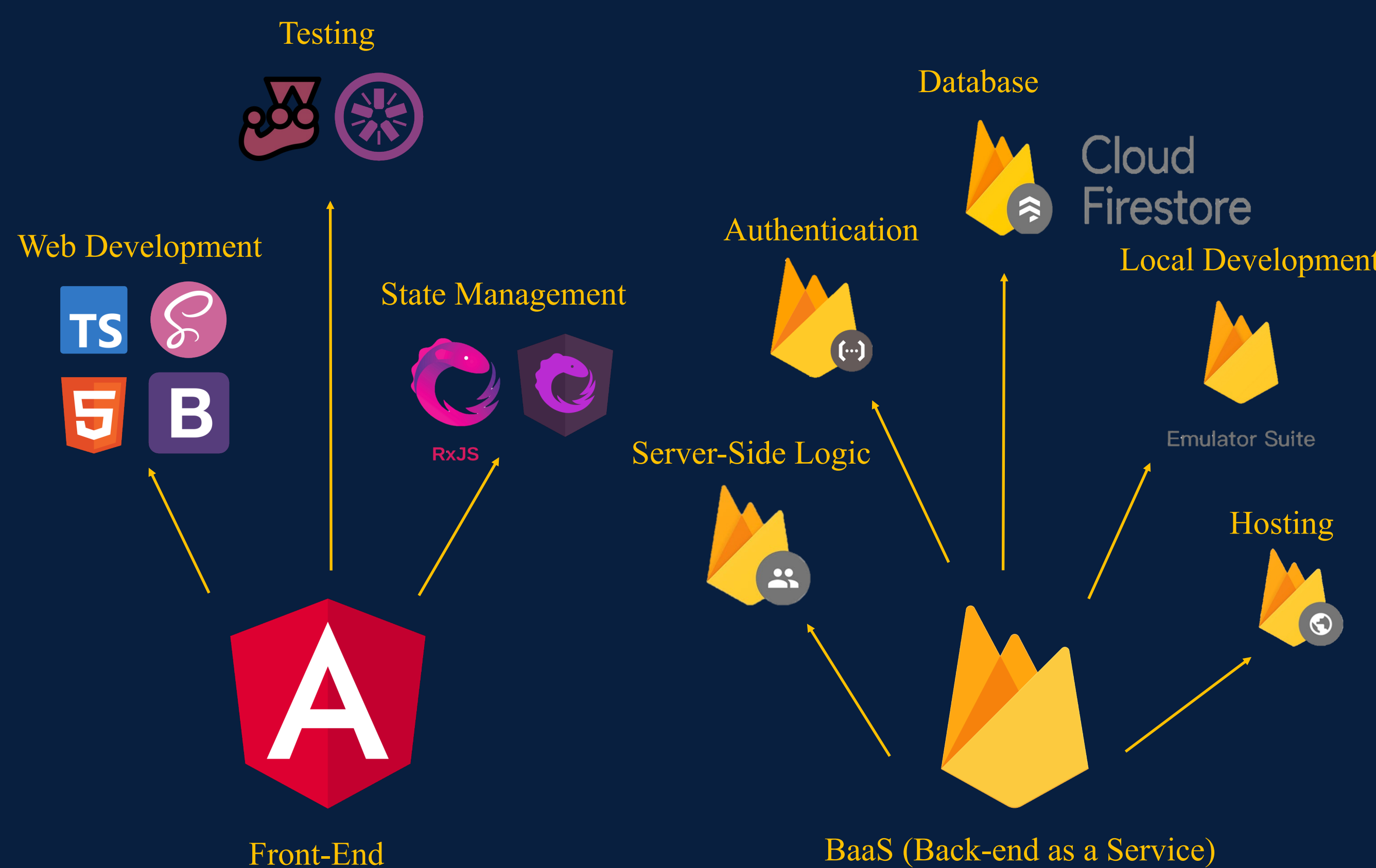


Fig 1A: Provider registration UI

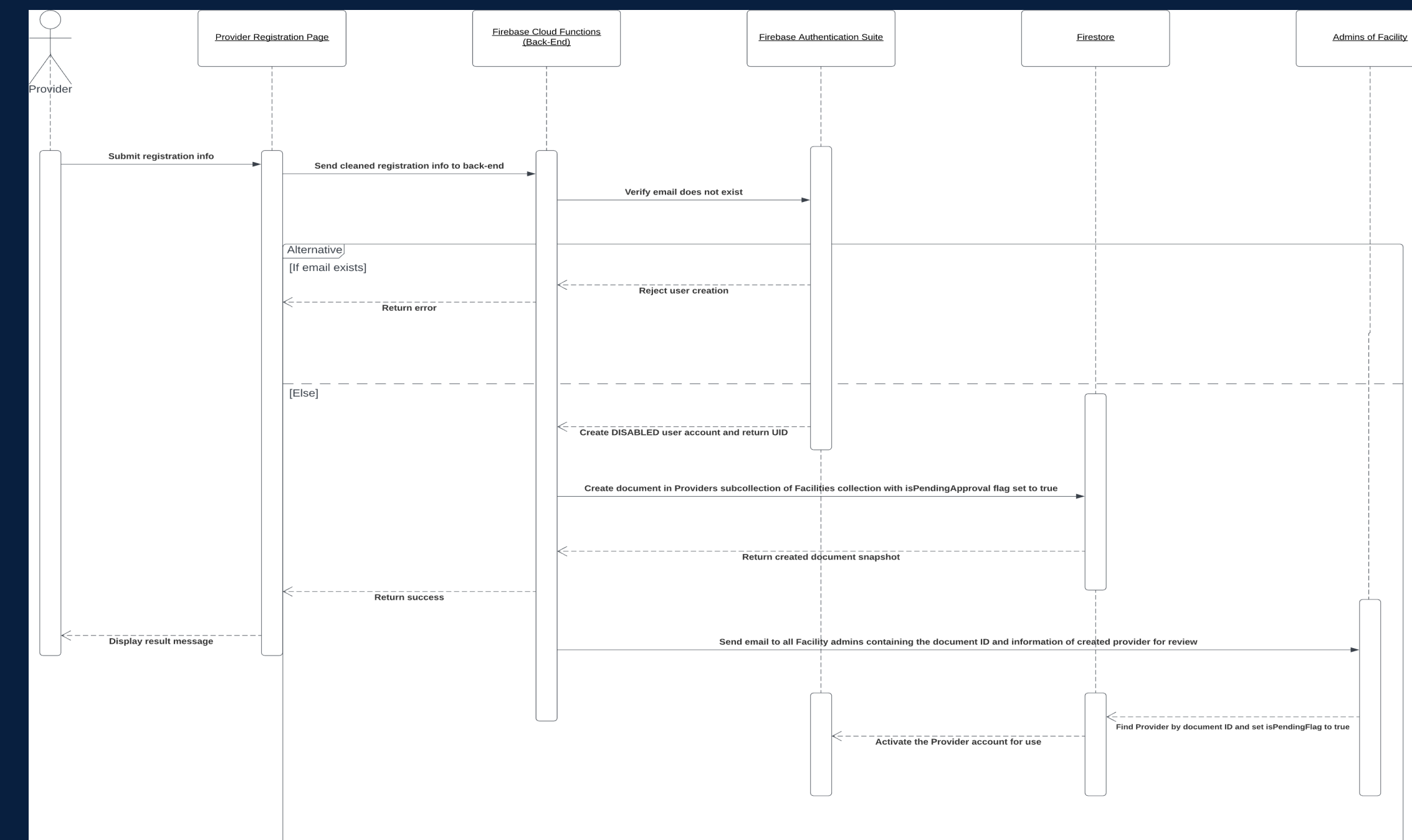


Fig 1B: Sequence diagram for provider registration

Fig 2A: Log in form UI

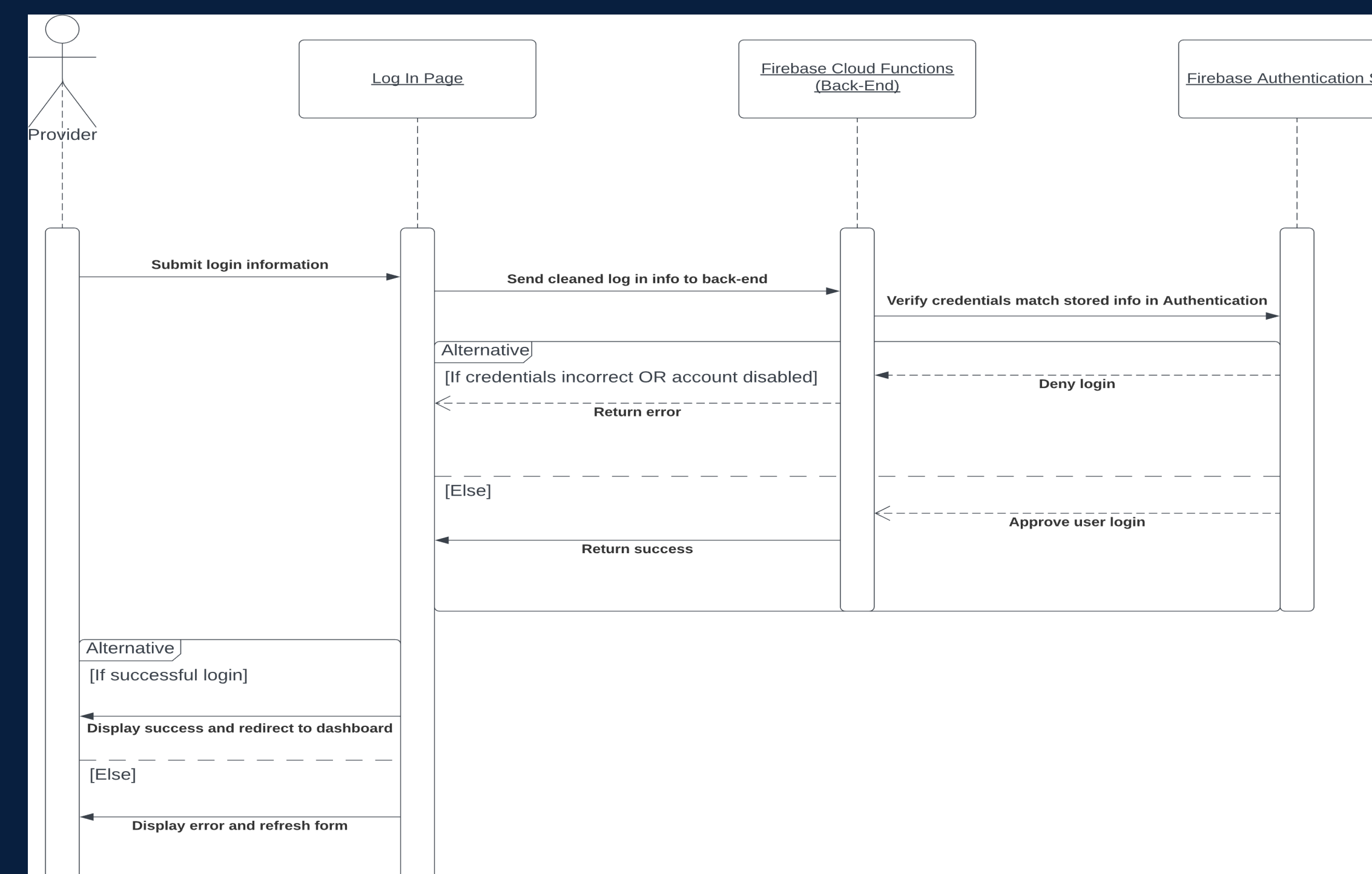


Fig 2B: Sequence diagram for provider login

Fig 3: Searching for providers by facility filter when logged in as patient. Also pictured is the side navigation bar and the user drawer on the top right.