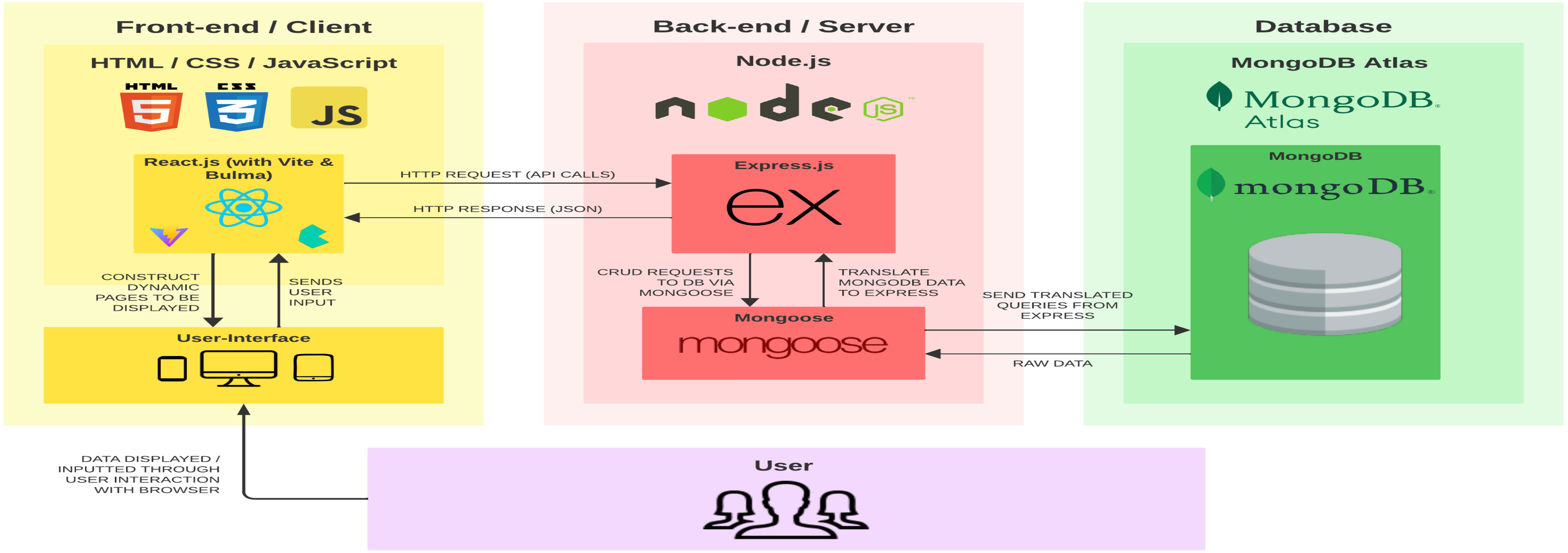
DocWait: T3A2

DEANDRE SUGANDHI, JOHN FABER RODRIGUEZ, SHAHIL PRASAD

What is DocWait?

- In the dynamic landscape of healthcare, general practitioners (GPs) operating on a walk-in basis encounter the challenge of managing patient queues efficiently, which often results in prolonged wait times and negatively affects patient satisfaction. Recognising this gap, our project introduces a sophisticated patient queue monitoring app designed to revolutionise the walk-in clinic experience for both patients and healthcare providers.
- DocWait is a web application that aims to mitigate the challenges associated with walk-in clinic visits by offering real-time updates on queue status and estimated wait times. This innovative solution not only enhances patient satisfaction by providing clarity and convenience but also empowers clinic staff with effective tools to manage patient flow, thereby optimising operational efficiency and demonstrating a commitment to excellence in patient care.

Application Architecture Diagram



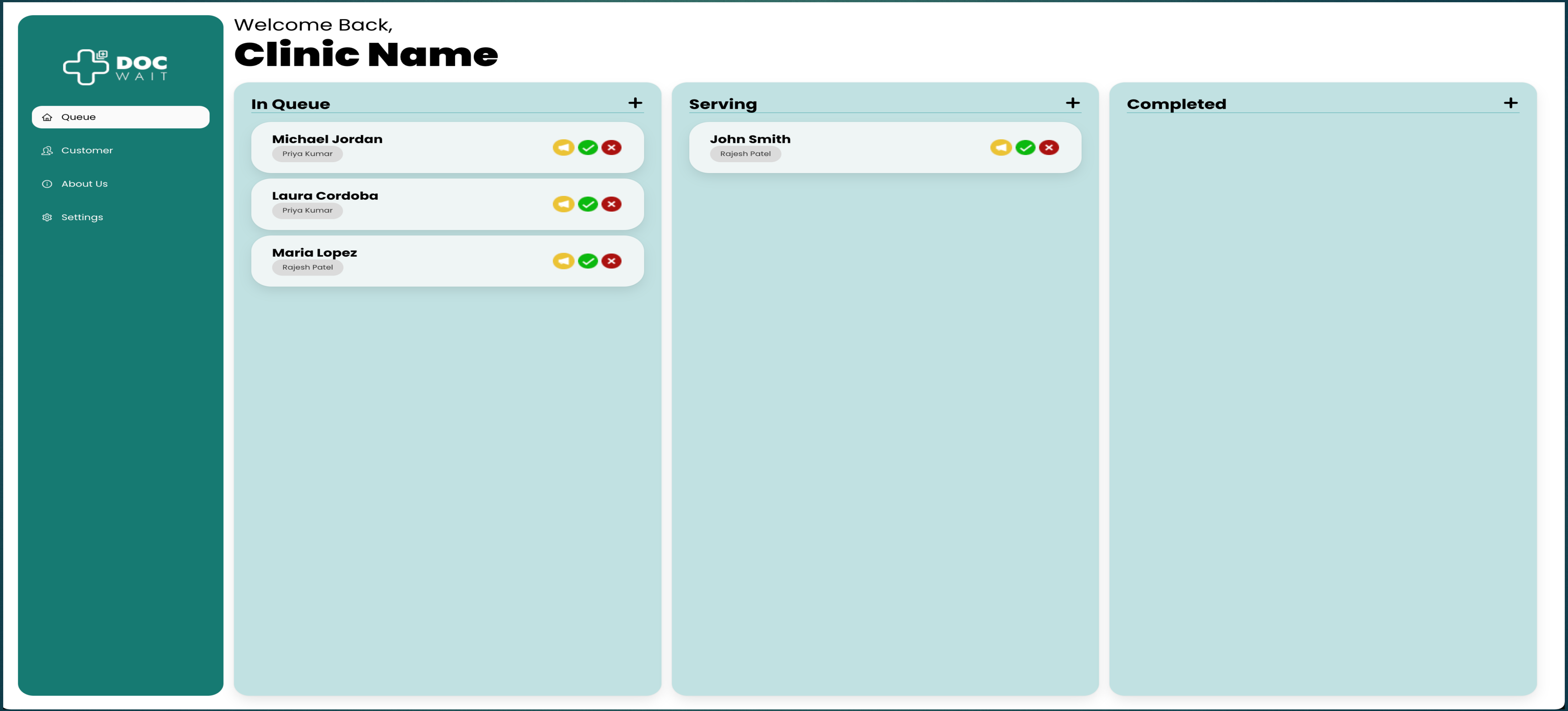
Tech Stack used

- MongoDB is a scalable, NoSQL database designed for modern applications. It stores data in flexible, JSON-like documents, allowing for varied data structures and quick adaptation to changes. It's ideal for projects requiring efficient management of large volumes of data.
- Express.js is a lightweight Node.js framework for building web and mobile applications. It simplifies the development of server-side applications with its robust routing and middleware capabilities, making it perfect for creating RESTful APIs that interact with React frontends.
- React is a JavaScript library for building dynamic user interfaces. Its component-based approach facilitates the development of reusable UI elements, enhancing the application's maintainability and performance, especially for interactive, single-page applications.
- Node.js is a JavaScript runtime that enables server-side scripting with JavaScript. It's known for its event-driven, non-blocking I/O model, which makes it suitable for building scalable and efficient network applications, including web servers and real-time applications.

Additional tools and libraries used

- Mongoose is an Object Data Modelling (ODM) library for MongoDB and Node.js, used to manage relationships between data and facilitate the connection between the server-side of an app (in this case, Express.js) and MongoDB. It translates data represented by MongoDB into JavaScript objects that Express.js can work with, and vice versa.
- Vite is a platform-agnostic build tool used as a development environment to streamline the development process of web apps. In this case, its use complements React.js. Some key features include a built-in development server supporting hot module replacement (HMR) allowing changes in the code to be reflected on the browser in real-time, use of native ES modules to compile code spontaneously eliminating the need for a traditional bundler, and a build command that bundles the app code with Rollup for performance improvements.
- Jest is a JavaScript testing framework, and Supertest is a testing library for testing HTTP requests and APIs in Node.js applications. These two are used together to generate automated tests for both the front-end and back-end functionalities of the app.
- Netlify is used to deploy the app’s front-end, while Render is used to deploy the app’s back-end. MongoDB Atlas is the cloud database service used to host the app’s MongoDB database.
- Draw.io & Figma is used to create wireframes, diagrams, and other planning resources.

Admin Side Webpage Example



DOC WAIT

- Queue
- Customer
- About Us
- Settings

Welcome Back,

Clinic Name

In Queue +

- **Michael Jordan** — Priya Kumar
- **Laura Cordoba** — Priya Kumar
- **Maria Lopez** — Rajesh Patel

Serving +

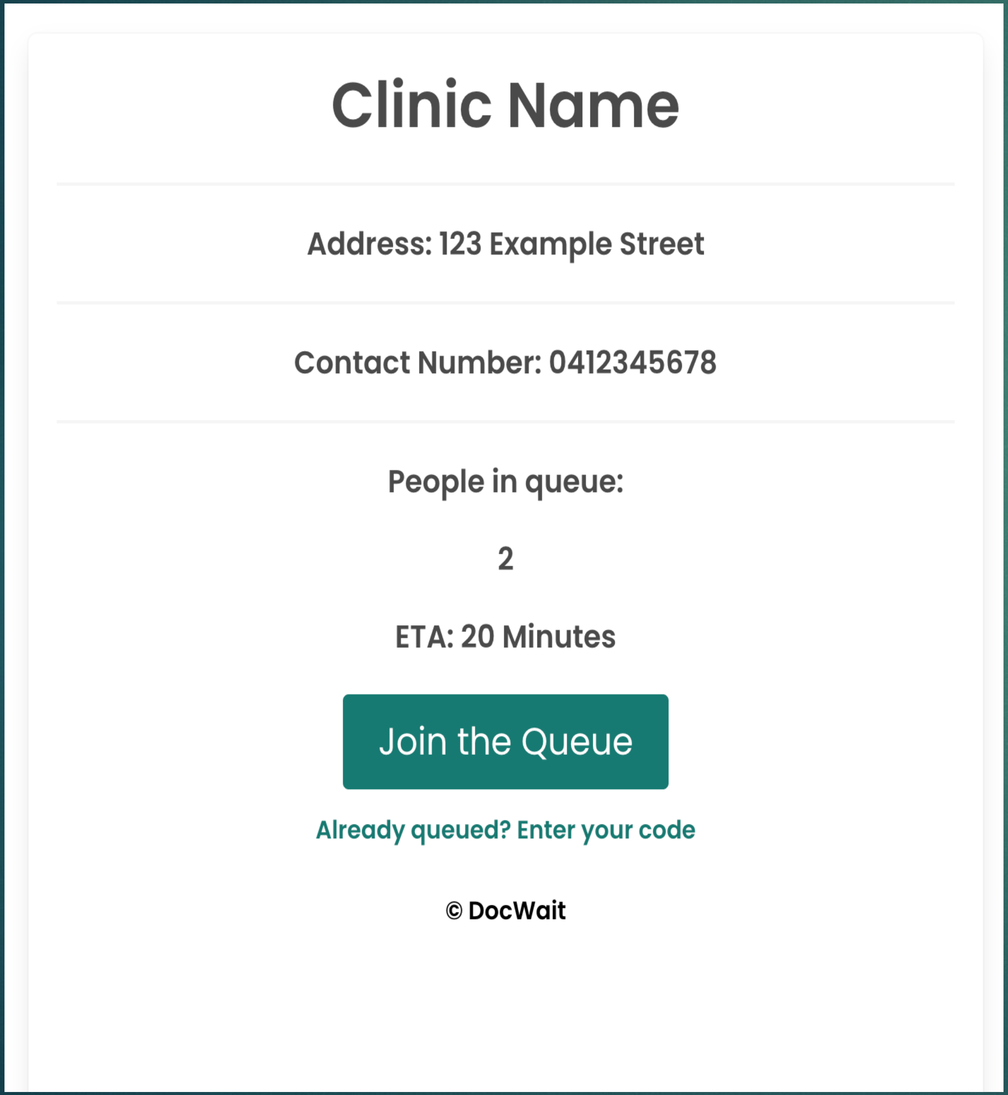
- **John Smith** — Rajesh Patel

Completed +

Key Features Admin Side

- Efficient Queue Management: A user-friendly dashboard allows clinic staff to manage the patient queue with ease, facilitating the addition, removal, and reordering of patients as required to accommodate real-time needs.
- Ease of Clinic Management: Allows admins to easily manage the practitioners availability and make changes within seconds to clinic name, address, opening hours etc.

Patient Side Webpage Example



Clinic Name

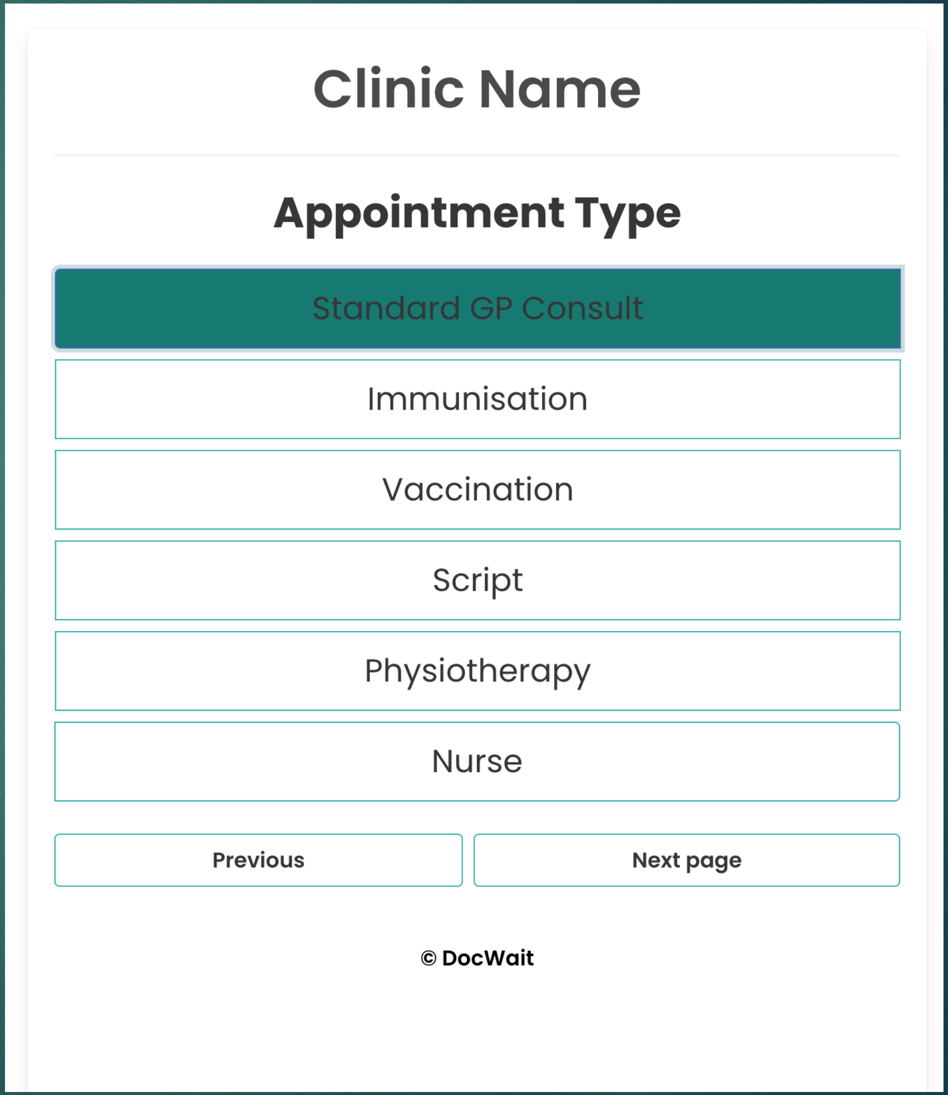
Address: 123 Example Street

Contact Number: 0412345678

People in queue:

2

ETA: 20 Minutes



Clinic Name

Appointment Type

Key Features Admin Side

- Real-Time Queue Visibility: Patients can view their position in the queue and receive accurate estimates of their wait times, enabling them to manage their schedules more effectively.
- Remote Queue Registration : This feature enables patients to join the queue remotely, further reducing wait times and streamlining clinic visits.

Issues Overcome

- Some of the issues we faced started out with working as a team using git branches. We were able to share with each other our processes and figure out correct processes to be able to work in separate branches and then merge then back correctly.
- Another issue we encountered was having styling issues as we had two people working on different front-end components. This was resolved by using adding additional class names and styling them accordingly.

Lessons learnt

- We found using a CSS framework such as Bulma was highly useful, although a framework with more flexibility to customize further would have assisted in styling process.
- In another project we would have had more pushes to main whilst working on branches so we would be able to see the working product in its stages whilst using version control if needed.