**Step 3: Research**

***Project Theme: Patient Management System***

Group Members: Juela Kadriu, Klevis Kadriu

***Team Captain: Juela Kadriu***

Requirements:

Specifically, you need to

1. Know about the various platforms available to you: their options and limitations.

2. Know about the various libraries that you can use to make your work easier.

3. Actually download, install, and build sample 'Hello world' apps using the most promising

technologies. It is not enough to just read about it, you have to do it in order to learn.

4. Then, learn how to use the specific GUI-, Widget-, Database-, Physics-, 3D-, whatever-library

you choose to use for the project by building little programs.

* On our project we will be using these technologies: **Next.js**, **TypeScript**, **Twilio**, and **Tailwind CSS**

**Technology Evaluation for Patient Management System**

**Next.js**

**Pros**

Next.js is a powerful React framework that offers server-side rendering (SSR) and static site generation, which significantly enhance the performance and search engine optimization (SEO) of your application—crucial for public-facing components like doctor profiles or patient dashboards. It also includes built-in routing, image optimization, and API routes, making it ideal for building both the frontend and backend logic of your system within one framework. This can simplify tasks like appointment booking or user authentication, while also boosting overall development speed and scalability.

**Cons**

Next.js can introduce complexity for beginners who are not familiar with concepts like server-side rendering (SSR), static generation, or hybrid routing. Managing SSR and client-side rendering properly may lead to increased cognitive load and debugging difficulties. Furthermore, deploying and scaling Next.js apps with advanced features might require a deeper understanding of server environments or cloud platforms. The framework also updates frequently, which can result in breaking changes or the need for continuous learning to stay current.

**TypeScript**

**Pros**

TypeScript brings type safety and enhanced development tools to your project. By adding static types to your JavaScript code, it helps catch bugs early during development rather than at runtime. This leads to more robust and maintainable code, especially as your application grows in size and complexity. TypeScript also improves the developer experience by enabling features like autocompletion, better documentation, and easier refactoring, which collectively lead to higher productivity and fewer logical errors.

**Cons**

TypeScript, although beneficial for large-scale applications, comes with a steep learning curve for developers who are new to typed languages. It requires more verbose code compared to JavaScript, which can slow down development initially. Setting up proper type definitions, especially for third-party libraries or custom data structures, can sometimes be time-consuming and frustrating. Also, overusing types can lead to rigid code that's harder to refactor if the app's requirements change frequently.

**Twilio**

**Pros**

Twilio is an industry-leading communications platform that allows you to integrate SMS, voice, and messaging services into your system. For a Patient Management System, this means you can send appointment reminders, confirmations, or critical alerts directly to patients’ phones via SMS or WhatsApp. Twilio also offers options for integrating voice or video calls—ideal for telemedicine features. Importantly, Twilio ensures high standards of security and compliance with data protection regulations like HIPAA or GDPR, which is essential when dealing with medical information.

**Cons**

Twilio, despite being a robust and feature-rich communication platform, can become expensive as your application's messaging or calling volume increases. It uses a pay-as-you-go model, and costs can escalate if you’re sending frequent SMS reminders or using voice services heavily. Integration with Twilio APIs also demands careful setup, particularly when dealing with user authentication, secure message handling, or voice call routing. In regions with limited telecom infrastructure, message delivery may not always be consistent or timely.

**Tailwind CSS**

**Pros**

Tailwind CSS is a utility-first CSS framework that lets you design user interfaces rapidly and consistently. Instead of writing custom styles, you apply pre-built utility classes directly in your HTML or JSX, which drastically speeds up the process of building and maintaining UI components. Tailwind ensures that your app maintains a clean, modern, and responsive design with minimal effort. It also supports customization through themes and extensions, making it easy to tailor the design to match your project's branding or specific user experience goals.

**Cons**

Tailwind CSS, while great for rapid UI development, often leads to bloated HTML due to the heavy use of utility classes inline. This can make the markup harder to read and maintain, especially for developers not familiar with the framework. Unlike traditional CSS, Tailwind moves styling out of separate style sheets, which can be jarring for teams used to a more component-based or semantic styling approach. Additionally, customizing Tailwind beyond its default configurations may require time and effort to fully understand its theming system and configuration files.