**LUGANODES SOFTWARE DEVELOPMENT ASSIGNMENT**

**FARHAN ANSARI – 20BKT0077**

**Task – 2**

Your task is to develop a Total Stake Counter application that allows users to track the total stake for three different blockchain networks: Cardano, Polkadot, and Kusama. The application should track the total amount staked with the Luganodes validator for each chain. The user interface (UI) should have options to add and remove chains from the tracking list.

**Intuition**

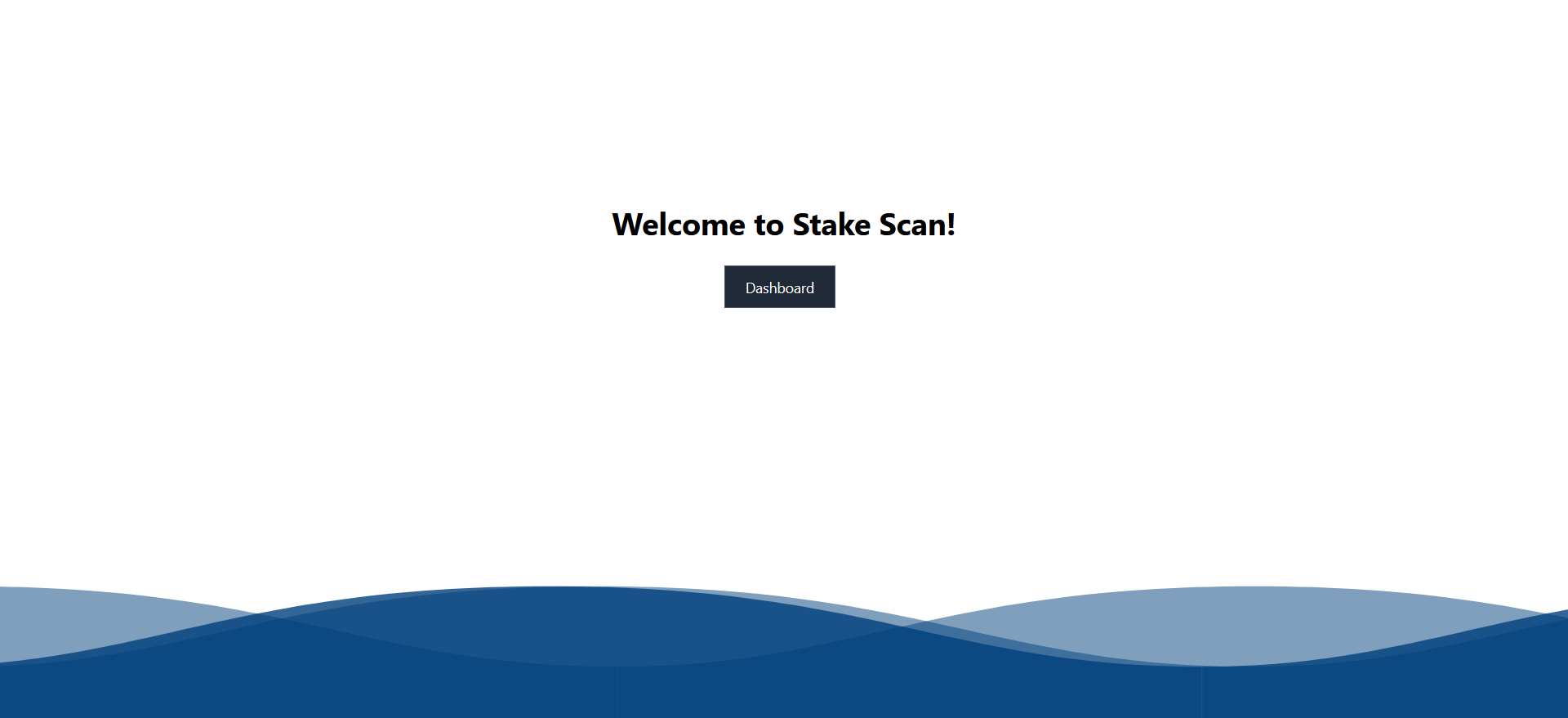
To build this application, you will need to leverage various technologies and steps:

1. **Blockchain Networks and Staking:** Understand the concept of blockchain networks and staking. In the context of this application, staking refers to holding and "locking up" a certain amount of cryptocurrency to support the operations of a blockchain network.
2. **API Integration:** The primary data source for your application is the Blockdaemon Staking Reporting API. This API allows you to retrieve staking-related data, such as the total amount staked, validator details, and more. You'll need to make API requests to gather this data for the specified blockchain networks (Cardano, Polkadot, Kusama).
3. **Technology Stack:**
   * **React.js:** This JavaScript library will be used to build the user interface (UI) of your application. It enables you to create dynamic and responsive components.
   * **Next.js:** A framework built on top of React.js that offers server-side rendering, routing, and other features. It helps with building optimized and SEO-friendly web applications.
   * **Node.js:** A runtime environment that allows you to execute JavaScript code on the server side.
   * **Express.js:** A web application framework for Node.js that simplifies the process of building robust APIs and handling HTTP requests.
   * **Docker:** A platform that allows developers to create, deploy, and run applications in containers, ensuring consistency across different environments.
4. **Application Flow:**
   * When a user accesses the application, they will see a UI built using React.js and Next.js.
   * The UI will provide options to add or remove blockchain networks from the tracking list (Cardano, Polkadot, Kusama).
   * Once the user selects a blockchain network, the frontend will make requests to the backend (built with Node.js and Express.js).
   * The backend will handle the API requests to the Blockdaemon Staking Reporting API, retrieve staking data, and calculate the total stake for the Luganodes validator on the selected blockchain network.
   * The backend will send this data back to the frontend, where it will be displayed to the user.
5. **API Requests and Data Parsing:**
   * Need to implement API request mechanisms in your Node.js backend using libraries like axios.
   * Retrieve data for each selected blockchain network (Cardano, Polkadot, Kusama) and parse the JSON responses to extract the relevant staking information.
6. **User Interface:**
   * Using React.js and Next.js, design and create a user-friendly interface that displays the total staked amount for each selected blockchain network.
   * Implement options to add or remove blockchain networks from the tracking list.
   * Use CSS or styling libraries to make the UI visually appealing and responsive.
7. **Docker Containerization:**
   * Dockerized the application to ensure that it runs consistently across different environments. This involves creating a Dockerfile that specifies the application's dependencies and configurations.
8. **Deployment:**
   * Deploy your application to a hosting platform of your choice (e.g., Heroku, AWS, DigitalOcean) to make it accessible to users over the internet.
9. **Error Handling and User Experience:**
   * Implement error handling mechanisms to gracefully handle scenarios where API requests fail or data retrieval encounters issues.
   * Provide meaningful feedback to users in case of errors and ensure a smooth user experience.
10. **Testing and Optimization:**
    * Test your application thoroughly to identify and fix any bugs or issues.
    * Optimize the performance of your application by minimizing API requests, optimizing frontend rendering, and improving backend response times.

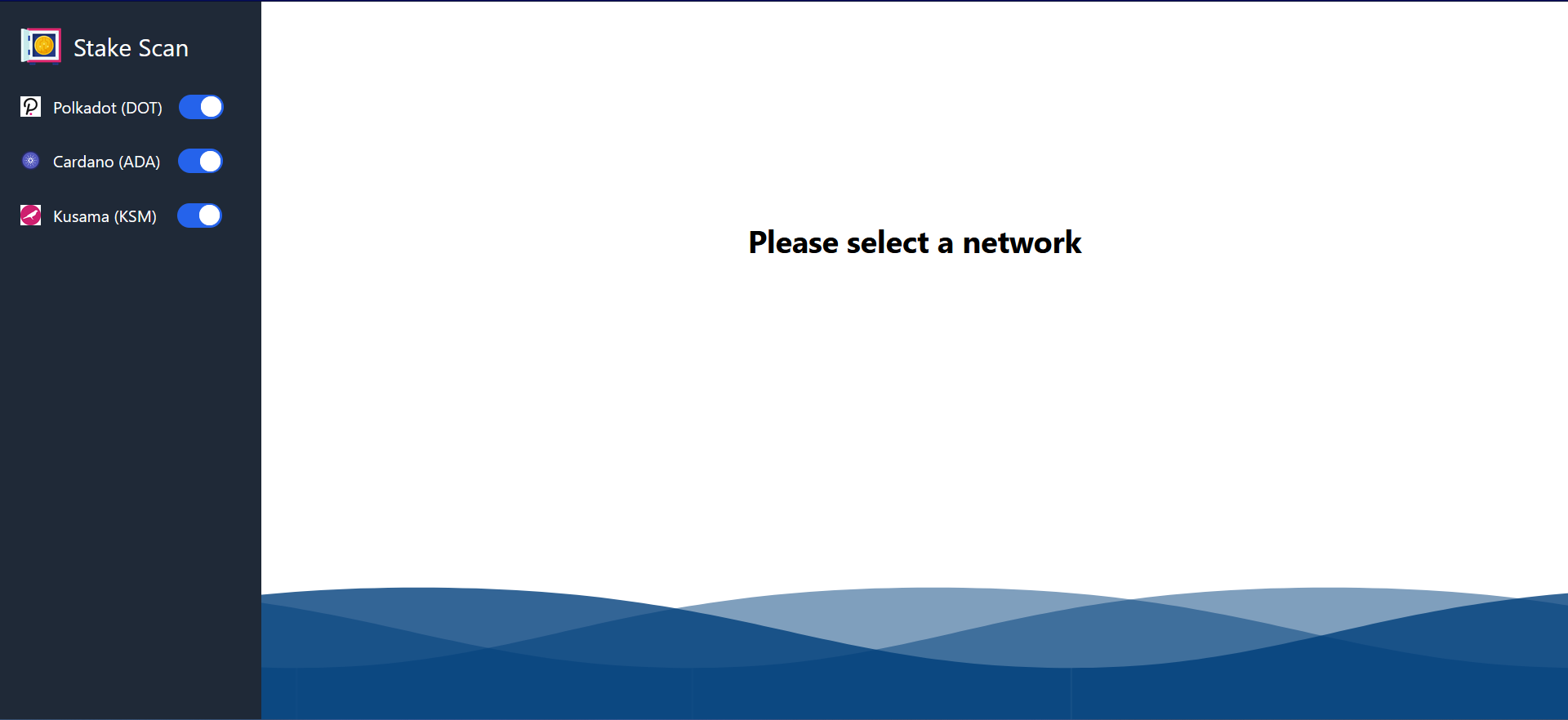
By combining these steps and technologies, you can create a comprehensive Total Stake Counter application that allows users to track staking data for different blockchain networks, with a user-friendly interface and reliable backend infrastructure.

**Sample Screenshots**

Home page



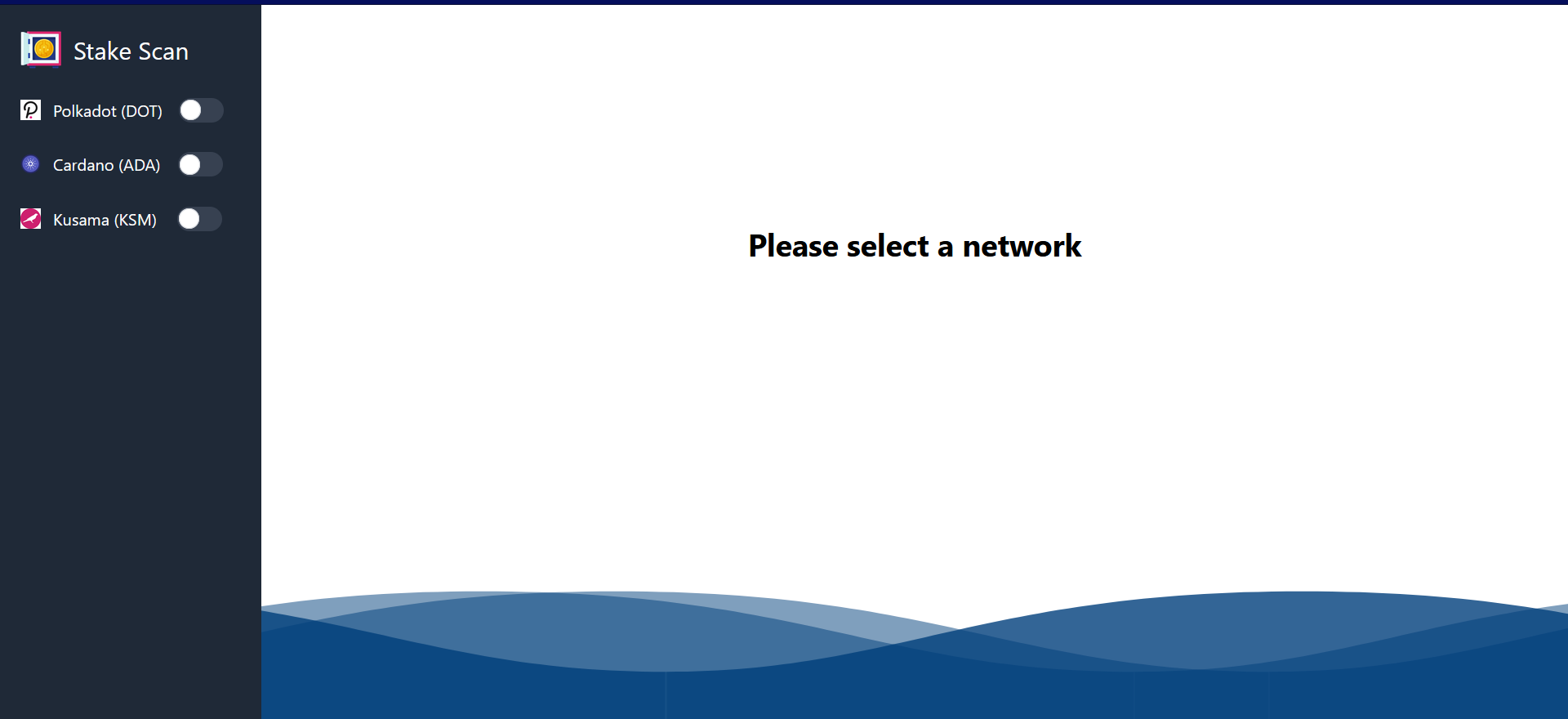
Dashboard



Polkadot staking information. An address is entered, if it is a validator node, their stake/bond information will be displayed.



Toggle to decide what networks to track



**Instructions to run the application**

* Clone the repository using ‘git clone <https://github.com/farhan-1902/Stake-scan.git>’
* Run ‘npm install’
* Run ‘npm run dev’ to start the application
* For running the docker container: ‘docker build -t nextjs-app .’ will build the docker image.
* Next, run ‘docker-compose up’ to start the application on docker (will run on localhost:3000).

The necessary docker files have been included in the repository.