ABHA ONBOARDING PROTOTYPE

CODE AVEANGERS

MOHAN BABU UNIVERSITY

Sainath Nagar , Tirupati

TEAM MEMBERS: ROLLNO:

GOTTI JYOTHI 23102D020205

GURRALA ANANTHALAKSHMI 23102D020182

RAMSAI KUNDHETI 23102D020179

POONDIT PRIYA 23102D020198

ABOUT MY PROJECT:

The **ABHA Onboarding Prototype** is a pioneering initiative aimed at facilitating the creation and management of the **Ayushman Bharat Health Account (ABHA)**, a crucial element of the **Ayushman Bharat Digital Mission (ABDM)**. This project empowers citizens by providing a secure and user-friendly digital platform to manage their health records efficiently.

**Key Features**

1. **User Registration**:
   * The prototype allows citizens to create their ABHA accounts using **Aadhaar-based authentication** and **mobile number verification**. Upon successful registration, a unique ABHA number and custom ABHA addresses are generated for each user.
2. **Secure Login**:
   * A robust authentication process is implemented, supporting multiple login methods, including **Aadhaar**, **ABHA number**, and **mobile OTP**. This flexibility enhances accessibility for users.
3. **Profile Management**:
   * Users have the ability to update their personal details, such as phone numbers and email addresses, ensuring their profiles remain current and secure. This feature promotes user engagement and trust in the system.
4. **Scan & Share Functionality**:
   * The prototype includes a QR code scanning feature that enables users to share their health records with hospitals and Health Information Providers (HIPs) seamlessly. This enhances the efficiency of accessing health data during medical visits, promoting better healthcare delivery.

**Technical Stack**

* **Front-End**: Developed using **Flutter** or **React Native**, ensuring cross-platform compatibility for both Android and iOS devices.
* **Back-End**: Built on **Node.js** with **Express** for handling API requests and user authentication. **MongoDB** is utilized for data storage and management.
* **Authentication**: The system employs **JWT (JSON Web Tokens)** for secure session management and user authentication, ensuring data privacy.

**Project Goals**

The primary objectives of the ABHA Onboarding Prototype include:

* Creating an intuitive onboarding process that encourages widespread adoption among citizens.
* Enhancing accessibility to healthcare services by enabling digital management of health records.
* Ensuring the security and privacy of sensitive health information in compliance with data protection regulations.

**Impact and Future Enhancements**

This project contributes significantly to the digitization of healthcare in India, making health information more accessible and manageable for citizens. By connecting patients with healthcare providers seamlessly, it aims to improve healthcare outcomes and foster a more efficient healthcare delivery system.

Future enhancements could include:

* Implementing **multi-factor authentication** for added security.
* Integrating additional healthcare services for comprehensive health management.
* Developing advanced analytics features that provide users with insights into their health records.

**Table of Contents**

1. **Introduction**
   * 1.1 Purpose
   * 1.2 Overview of Ayushman Bharat Health Account (ABHA)
   * 1.3 Project Objective
2. **System Requirements**
   * 2.1 Functional Requirements
   * 2.2 Non-Functional Requirements
   * 2.3 Technologies and Tools
3. **Application Features**
   * 3.1 User Registration (Onboarding)
   * 3.2 User Login
   * 3.3 ABHA Address Creation
   * 3.4 Profile Update
   * 3.5 Scan & Share
4. **API Integration**
   * 4.1 ABHA Creation APIs
     + 4.1.1 Aadhaar-based Registration API
     + 4.1.2 Mobile OTP Verification API
   * 4.2 Login APIs
     + 4.2.1 ABHA Number Login API
     + 4.2.2 Aadhaar Login API
   * 4.3 Profile Update APIs
   * 4.4 Scan & Share APIs
   * 4.5 Gateway Token API
5. **User Interface Design**
   * 5.1 Registration Screen
   * 5.2 Login Screen
   * 5.3 Profile Update Screen
   * 5.4 Scan & Share Screen
6. **Database Schema**
   * 6.1 User Table
   * 6.2 ABHA Address Table
   * 6.3 QR Scan History Table
7. **Security and Authentication**
   * 7.1 Mobile OTP-based Authentication
   * 7.2 OAuth and Session Management
   * 7.3 Data Privacy and Protection
8. **Deployment**
   * 8.1 Front-End: Flutter / React Native Setup
   * 8.2 Back-End: Node.js Setup
   * 8.3 API Hosting
   * 8.4 Android APK Build
9. **Testing and Validation**
   * 9.1 Unit Testing
   * 9.2 API Testing with Postman
   * 9.3 User Acceptance Testing (UAT)
10. **Deliverables**
    * 10.1 Source Code Submission
    * 10.2 APK File Submission
    * 10.3 Documentation and Project Report
11. **Evaluation Criteria**
    * 11.1 Feature 1: ABHA Account Creation (30 Points)
    * 11.2 Feature 2: Profile Update (10 Points)
    * 11.3 Feature 3: Login Feature (25 Points)
    * 11.4 Feature 4: Scan & Share (35 Points)
    * 11.5 Bonus: Cross-Platform Support (Android, iOS, Web)
12. **References**
    * 12.1 API Documentation Links
    * 12.2 Government Resources
    * 12.3 Sample APKs
    * INTRODUCTION

The prototype involves key functionalities like **user registration**, where users can create their ABHA using Aadhaar-based authentication and mobile number verification. Once registered, users will have a secure login process that supports multiple authentication mechanisms, including Aadhaar and ABHA number login. Users can also create a custom **ABHA address**, which serves as their digital identifier within the health system, and update their personal profile details like phone number and email as needed.

Additionally, the system includes a **Scan & Share** feature, where users can scan the QR code of a hospital or Health Information Provider (HIP) and securely share their ABHA profile, allowing healthcare providers to access the user’s health records. This functionality enhances the usability of the ABHA system in real-world healthcare The **Ayushman Bharat Health Account (ABHA)** is a key initiative under the **Ayushman Bharat Digital Mission (ABDM)** aimed at empowering Indian citizens with a digital health ecosystem. The ABHA enables individuals to manage their health records digitally, giving them control and easy access to their personal health information. The goal is to create a seamless, interoperable, and secure system that integrates healthcare providers, patients, and health records through a unique ABHA number and ABHA address.

In this **ABHA Onboarding Prototype**, the challenge is to build a user-friendly, secure, and efficient process for citizens to create, manage, and share their ABHA accounts. A smooth onboarding experience is crucial for widespread adoption of the ABHA system, making it accessible and usable for all.

ABSTRACT:

The **Ayushman Bharat Health Account (ABHA)** is a foundational initiative under the **Ayushman Bharat Digital Mission (ABDM)** aimed at creating a digital health ecosystem in India. It provides citizens with a unique digital identity to manage their health records securely and efficiently. However, a critical aspect of ensuring the success of ABHA is the development of a seamless, user-friendly onboarding system that allows citizens to easily register, log in, update their profiles, and share their health information with healthcare providers.

This project focuses on building an **ABHA Onboarding Prototype**, which addresses the need for a smooth and secure process for users to create and manage their ABHA accounts. The prototype incorporates key functionalities such as **user registration** through Aadhaar-based authentication and mobile OTP verification, **secure login** using ABHA number or Aadhaar, **profile management**, and a **Scan & Share** feature that enables users to share their health profiles with healthcare providers by scanning a QR code.

The goal of this system is to demonstrate how a digital health account can be managed efficiently while ensuring data privacy and security. The system also supports the creation of a custom **ABHA address**, providing users with a unique digital identity within the healthcare system. Furthermore, it integrates the **Scan & Share** functionality, allowing for easy health data sharing between patients and healthcare providers, improving the overall healthcare experience.

The prototype is designed to be **cross-platform**, supporting Android, iOS, and web, ensuring wide accessibility. By creating a digital health account that is easy to use and secure, this project contributes to the overarching goal of transforming healthcare delivery in India, making it more accessible, efficient, and patient-centric.

This abstract outlines the objectives and scope of the ABHA onboarding prototype, which aims to enhance the user experience in managing personal health information while maintaining a high standard of security and usability.

OBJECTIVES:

The primary objectives of the **ABHA Onboarding Prototype** are to create a secure, user-friendly, and accessible platform that allows Indian citizens to easily create, manage, and share their digital health accounts through the **Ayushman Bharat Health Account (ABHA)** system. The specific objectives of the project are as follows:

1. **Simplify ABHA Registration Process**:
   * Develop a streamlined registration process that allows users to create their ABHA using **Aadhaar-based authentication** and **mobile number verification**.
   * Ensure that the onboarding experience is intuitive and easy for users, minimizing friction and complexity.
2. **Enable Secure and Flexible Login**:
   * Implement a secure login system that supports **multiple authentication mechanisms** such as ABHA number, Aadhaar, and mobile OTP.
   * Provide a user-friendly login experience that ensures quick and secure access to personal health records.
3. **Support ABHA Address Creation**:
   * Enable users to create a **custom ABHA address** as part of the registration process.
   * Allow for the management of multiple ABHA addresses under a single user account, providing flexibility and personalization.
4. **Facilitate Profile Management**:
   * Build a profile update feature that allows users to securely modify their personal details, such as phone number and email address, at any time.
   * Ensure that profile updates are reflected immediately and securely within the ABHA system.
5. **Implement Scan & Share Functionality**:
   * Create a **QR code scanning feature** that allows users to easily share their ABHA profile with **Health Information Providers (HIPs)**, such as hospitals and clinics.
   * Ensure secure data sharing by generating tokens and following ABDM guidelines for profile sharing.
6. **Ensure Cross-Platform Accessibility**:
   * Design the application to be compatible with **Android, iOS, and web platforms**, ensuring that it can reach the widest possible user base.
   * Ensure a consistent and seamless user experience across all platforms.
7. **Prioritize Data Privacy and Security**:
   * Implement **strong encryption** and secure session management to protect sensitive health data during registration, login, and data sharing.
   * Adhere to **data privacy regulations** and ensure the system complies with **Ayushman Bharat Digital Mission (ABDM)** standards.
8. **Optimize for User Experience**:
   * Focus on creating a **clean, intuitive, and user-friendly interface** that ensures users of all technical proficiency levels can navigate the system easily.
   * Provide clear guidance and support during the registration and onboarding process.

By achieving these objectives, the **ABHA Onboarding Prototype** will contribute to the broader vision of enabling digital health management in India, providing citizens with an accessible, secure, and easy way to manage their personal health data.

SYSTEM REQURIMENTS:

**1. Functional Requirements**

* **User Registration (Onboarding)**:
  + The system should allow users to register using **Aadhaar-based authentication** and **mobile OTP verification**.
  + It should generate a unique **ABHA number** and **custom ABHA address** for each user.
* **User Login**:
  + The system must support login via **ABHA number**, **Aadhaar number**, or **mobile number** with OTP.
  + Secure session management should be implemented to maintain user authentication.
* **Profile Update**:
  + The system should allow users to update their **personal details** (e.g., phone number, email) through a secure interface.
* **Scan & Share**:
  + The system must enable users to **scan a hospital's QR code** and securely share their ABHA profile using the ABDM token-based system.
* **Cross-Platform Support**:
  + The application must be functional on **Android, iOS, and web** platforms.

**2. Non-Functional Requirements**

* **Security**:
  + The system must enforce **encryption** (e.g., SSL/TLS) for all data transmission and securely handle sensitive data (e.g., Aadhaar and ABHA numbers).
  + Adherence to **GDPR** and **Indian privacy laws** is essential.
* **Performance**:
  + The application should be optimized for **low-latency** interactions, particularly for the **OTP verification** and **QR scanning** processes.
* **Scalability**:
  + The back-end should be able to **handle large user volumes**, considering the potential national scale of the ABHA system.
* **Reliability**:
  + Ensure **high availability** and **fault tolerance** of both the front-end and back-end to prevent system downtime during critical processes (e.g., health data sharing).

**3. Hardware Requirements**

* **For Development**:
  + **Processor**: Intel i5 or higher
  + **RAM**: 8 GB or higher
  + **Storage**: Minimum 20 GB available space
  + **OS**: Windows 10/MacOS/Linux
* **For Mobile Testing**:
  + **Android Emulator**: Android Studio (API level 21+)
  + **iOS Simulator**: Xcode (macOS only)

**4. Software Requirements**

* **Front-End**:
  + **Flutter SDK** or **React Native** for cross-platform mobile development.
  + **Android Studio/Xcode** for emulator/simulator.
* **Back-End**:
  + **Node.js** with **Express** for API services.
  + **MongoDB** or **MySQL** for storing user and ABHA data.
  + **Postman** for API testing and validation.
* **Other Tools**:
  + **Version Control**: Git and GitHub for source code management.
  + **Postman** for API testing and documentation.

Top of Form

Source code:

Product structure:

abha-backend/

├── controllers/

│ ├── authController.js

│ ├── profileController.js

├── models/

│ ├── userModel.js

├── routes/

│ ├── authRoutes.js

│ ├── profileRoutes.js

├── server.js

├── .env (optional)

├── package.json

Install dependices:

mkdir abha-backend

cd abha-backend

npm init -y

npm install express mongoose bcryptjs jsonwebtoken cors dotenv

Bottom of Form

User registration:

For creating frontend this code by using htm and css.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>ABHA Onboarding</title>

    <link rel="stylesheet" href="login.css">

</head>

<body>

    <div class="wrapper">

        <h1>Welcome to ABHA Onboarding</h1>

        <div class="cards">

            <!-- Registration Card -->

            <div class="card">

                <h2>Register</h2>

                <form action="/register" method="POST">

                    <div class="input-group">

                        <label for="name">Full Name:</label>

                        <input type="text" id="name" name="name" placeholder="Your full name" required>

                    </div>

                    <div class="input-group">

                        <label for="aadhaar">Aadhaar Number:</label>

                        <input type="text" id="aadhaar" name="aadhaar" placeholder="Your Aadhaar number" required>

                    </div>

                    <div class="input-group">

                        <label for="mobile">Mobile Number:</label>

                        <input type="text" id="mobile" name="mobile" placeholder="Your mobile number" required>

                    </div>

                    <div class="input-group">

                        <label for="abhaAddress">ABHA Address:</label>

                        <input type="text" id="abhaAddress" name="abhaAddress" placeholder="Choose ABHA address" required>

                    </div>

                    <button class="ram-btn"><a href="https://abha.abdm.gov.in/abha/v3/">Register</a></button>

                </form>

            </div>

            <!-- Login Card -->

            <div class="card">

                <h2>Login</h2>

                <form action="/login" method="POST">

                    <div class="input-group">

                        <label for="abhaLogin">ABHA Address/Number:</label>

                        <input type="text" id="abhaLogin" name="abhaLogin" placeholder="Enter ABHA address/number" required>

                    </div>

                    <div class="input-group">

                        <label for="otp">OTP:</label>

                        <input type="text" id="otp" name="otp" placeholder="Enter OTP" required>

                    </div>

                    <button class="ram-btn"><a href="https://abha.abdm.gov.in/abha/v3/">Login</a></button>

                </form>

            </div>

        </div>

    </div>

</body>

</html>

SAVE: save this file in new.html and run in visual studio code.

Next give css for registration page:

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

}

body {

    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

    background-color: #f1f5f9;

    display: flex;

    justify-content: center;

    align-items: center;

    height: 100vh;

}

h1 {

    color: #2c3e50;

    text-align: center;

    margin-bottom: 40px;

}

.wrapper {

    max-width: 1200px;

    width: 100%;

    padding: 20px;

    display: flex;

    flex-direction: column;

    justify-content: center;

    align-items: center;

}

/\* Card Layout \*/

.cards {

    display: flex;

    justify-content: space-between;

    gap: 40px;

}

/\* Card Style \*/

.card {

    background-color: white;

    border-radius: 15px;

    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);

    padding: 30px;

    width: 100%;

    max-width: 400px;

    text-align: center;

    transition: transform 0.3s ease;

}

.card:hover {

    transform: translateY(-10px);

}

.card h2 {

    color: #34495e;

    margin-bottom: 20px;

    font-size: 24px;

}

/\* Form Inputs \*/

.input-group {

    margin-bottom: 15px;

}

label {

    display: block;

    margin-bottom: 5px;

    font-weight: 600;

    color: #555;

}

input[type="text"],

input[type="password"] {

    width: 100%;

    padding: 12px;

    border: 1px solid #ddd;

    border-radius: 8px;

    margin-bottom: 10px;

    font-size: 16px;

}

input[type="submit"] {

    width: 100%;

    padding: 12px;

    background-color: #2980b9;

    color: white;

    border: none;

    border-radius: 8px;

    cursor: pointer;

    font-size: 18px;

}

input[type="submit"]:hover {

    background-color: #1a6c98;

}

@media (max-width: 768px) {

    .cards {

        flex-direction: column;

        align-items: center;

    }

}

.ram-btn {

    color: #ddd;

    background-color: #2980b9;

    width: 100px;

    height: 30px;

}

Save : save as new.css and run in vs code. Link with new.html and run.

They can open the registration page.

Login page:

After creating user registration we can create login page by using html and css.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>ABHA Onboarding</title>

    <link rel="stylesheet" href="login.css">

</head>

<body>

    <div class="wrapper">

        <h1>Welcome to ABHA Onboarding</h1>

        <div class="cards">

            <!-- Registration Card -->

            <div class="card">

                <h2>Register</h2>

                <form action="/register" method="POST">

                    <div class="input-group">

                        <label for="name">Full Name:</label>

                        <input type="text" id="name" name="name" placeholder="Your full name" required>

                    </div>

                    <div class="input-group">

                        <label for="aadhaar">Aadhaar Number:</label>

                        <input type="text" id="aadhaar" name="aadhaar" placeholder="Your Aadhaar number" required>

                    </div>

                    <div class="input-group">

                        <label for="mobile">Mobile Number:</label>

                        <input type="text" id="mobile" name="mobile" placeholder="Your mobile number" required>

                    </div>

                    <div class="input-group">

                        <label for="abhaAddress">ABHA Address:</label>

                        <input type="text" id="abhaAddress" name="abhaAddress" placeholder="Choose ABHA address" required>

                    </div>

                    <button class="ram-btn"><a href="https://abha.abdm.gov.in/abha/v3/">Register</a></button>

                </form>

            </div>

            <!-- Login Card -->

            <div class="card">

                <h2>Login</h2>

                <form action="/login" method="POST">

                    <div class="input-group">

                        <label for="abhaLogin">ABHA Address/Number:</label>

                        <input type="text" id="abhaLogin" name="abhaLogin" placeholder="Enter ABHA address/number" required>

                    </div>

                    <div class="input-group">

                        <label for="otp">OTP:</label>

                        <input type="text" id="otp" name="otp" placeholder="Enter OTP" required>

                    </div>

                    <button class="ram-btn"><a href="https://abha.abdm.gov.in/abha/v3/">Login</a></button>

                </form>

            </div>

        </div>

    </div>

</body>

</html>

Save: save this in login.html .after save the file we create css for login page.

Login.css:

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

}

body {

    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

    background-color: #f1f5f9;

    display: flex;

    justify-content: center;

    align-items: center;

    height: 100vh;

}

h1 {

    color: #2c3e50;

    text-align: center;

    margin-bottom: 40px;

}

.wrapper {

    max-width: 1200px;

    width: 100%;

    padding: 20px;

    display: flex;

    flex-direction: column;

    justify-content: center;

    align-items: center;

}

/\* Card Layout \*/

.cards {

    display: flex;

    justify-content: space-between;

    gap: 40px;

}

/\* Card Style \*/

.card {

    background-color: white;

    border-radius: 15px;

    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);

    padding: 30px;

    width: 100%;

    max-width: 400px;

    text-align: center;

    transition: transform 0.3s ease;

}

.card:hover {

    transform: translateY(-10px);

}

.card h2 {

    color: #34495e;

    margin-bottom: 20px;

    font-size: 24px;

}

/\* Form Inputs \*/

.input-group {

    margin-bottom: 15px;

}

label {

    display: block;

    margin-bottom: 5px;

    font-weight: 600;

    color: #555;

}

input[type="text"],

input[type="password"] {

    width: 100%;

    padding: 12px;

    border: 1px solid #ddd;

    border-radius: 8px;

    margin-bottom: 10px;

    font-size: 16px;

}

input[type="submit"] {

    width: 100%;

    padding: 12px;

    background-color: #2980b9;

    color: white;

    border: none;

    border-radius: 8px;

    cursor: pointer;

    font-size: 18px;

}

input[type="submit"]:hover {

    background-color: #1a6c98;

}

@media (max-width: 768px) {

    .cards {

        flex-direction: column;

        align-items: center;

    }

}

.ram-btn {

    color: #ddd;

    background-color: #2980b9;

    width: 100px;

    height: 30px;

}

Save: save the file login.css and link with login.html. and run in visual studio.

We can have the login page.

Dashboard :

Creation for dashboard for login page helath service product aggregators:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>User Dashboard - ABHA Onboarding</title>

    <link rel="stylesheet" href="Dashbord.css">

</head>

<body>

    <div class="dashboard-container">

        <header>

            <h1>Welcome to your ABHA Dashboard</h1>

            <p>Your ABHA Address: <strong id="abha-address">user@abha</strong></p>

            <nav>

                <ul>

                    <li><a href="/profile">View Profile</a></li>

                    <li><a href="/update">Update Profile</a></li>

                    <li><a href="/logout">Logout</a></li>

                </ul>

            </nav>

        </header>

        <main>

            <h2>Profile Overview</h2>

            <div class="profile-card">

                <p><strong>Full Name:</strong> John Doe</p>

                <p><strong>Mobile Number:</strong> +91 9876543210</p>

                <p><strong>Aadhaar Number:</strong> 1234-5678-9101</p>

                <p><strong>ABHA Address:</strong> user@abha</p>

            </div>

        </main>

    </div>

</body>

</html>

Save: save the file with dashboard.html.

We create css for dashboard

/\* General Styles \*/

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

}

body {

    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

    background-color: #f5f5f5;

    display: flex;

    justify-content: center;

    align-items: center;

    height: 100vh;

}

/\* Dashboard Container \*/

.dashboard-container {

    width: 80%;

    max-width: 900px;

    background-color: white;

    padding: 20px;

    border-radius: 10px;

    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);

}

/\* Header \*/

header {

    text-align: center;

    margin-bottom: 40px;

}

header h1 {

    color: #2c3e50;

}

header p {

    font-size: 18px;

    margin-top: 10px;

}

header nav ul {

    list-style: none;

    display: flex;

    justify-content: center;

    margin-top: 20px;

}

header nav ul li {

    margin: 0 15px;

}

header nav ul li a {

    text-decoration: none;

    color: #2980b9;

    font-size: 16px;

    padding: 10px 20px;

    border: 2px solid #2980b9;

    border-radius: 5px;

    transition: background-color 0.3s;

}

header nav ul li a:hover {

    background-color: #2980b9;

    color: white;

}

/\* Main Content \*/

main {

    text-align: center;

}

h2 {

    color: #34495e;

    margin-bottom: 20px;

}

/\* Profile Card \*/

.profile-card {

    background-color: #ecf0f1;

    border-radius: 10px;

    padding: 20px;

    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

    text-align: left;

    max-width: 600px;

    margin: 0 auto;

}

.profile-card p {

    font-size: 18px;

    margin-bottom: 10px;

}

.profile-card strong {

    color: #2c3e50;

}

Save: save the file dashboard.css and link with dashboard.html and run in visual studio.

Run the code in node.js:

First install the node.js.

Then create a new folder on desktop and open that folder in visual studio code .Then create a file name in folder.

Go to terminal and open new terminal

First acces packages and express.

Give npm init

Next access packages. Then install express. Npm init -y. then run the out put in vs code node filename.

Back end code:

const express = require('express');

const mongoose = require('mongoose');

const cors = require('cors');

const authRoutes = require('./routes/authRoutes');

const profileRoutes = require('./routes/profileRoutes');

require('dotenv').config();

const app = express();

app.use(cors());

app.use(express.json());

// MongoDB Connection

mongoose.connect(process.env.MONGO\_URI || 'mongodb://localhost/abha', {

useNewUrlParser: true,

useUnifiedTopology: true,

})

.then(() => console.log('MongoDB connected'))

.catch((err) => console.log(err));

// API Routes

app.use('/api', authRoutes);

app.use('/api', profileRoutes);

// Start Server

const PORT = process.env.PORT || 5000;

app.listen(PORT, () => {

console.log(`Server is running on port ${PORT}`);

});

User registration:

const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({

aadhaarNumber: { type: String, required: true, unique: true },

mobileNumber: { type: String, required: true },

abhaNumber: { type: String, unique: true },

phone: { type: String },

email: { type: String },

});

module.exports = mongoose.model('User', userSchema);

const User = require('../models/userModel');

const jwt = require('jsonwebtoken');

// Generate a random ABHA number

const generateAbhaNumber = () => {

return `ABHA${Math.floor(100000 + Math.random() \* 900000)}`;

};

// User Registration

exports.registerUser = async (req, res) => {

const { aadhaarNumber, mobileNumber } = req.body;

try {

const abhaNumber = generateAbhaNumber();

const newUser = new User({ aadhaarNumber, mobileNumber, abhaNumber });

await newUser.save();

res.status(201).json({ success: true, abhaNumber });

} catch (error) {

res.status(500).json({ error: error.message });

}

};

// User Login

exports.loginUser = async (req, res) => {

const { abhaNumber, aadhaarNumber } = req.body;

try {

const user = await User.findOne({

$or: [{ abhaNumber }, { aadhaarNumber }],

});

if (!user) {

return res.status(400).json({ error: 'Invalid credentials' });

}

const token = jwt.sign({ id: user.\_id }, process.env.JWT\_SECRET, {

expiresIn: '1h',

});

res.status(200).json({ success: true, token });

} catch (error) {

res.status(500).json({ error: error.message });

}

};

Profile controller:

const User = require('../models/userModel');

// Update User Profile

exports.updateProfile = async (req, res) => {

const { phone, email } = req.body;

const userId = req.user.id; // User ID from JWT

try {

const updatedUser = await User.findByIdAndUpdate(

userId,

{ phone, email },

{ new: true }

);

if (!updatedUser) {

return res.status(404).json({ error: 'User not found' });

}

res.status(200).json({ success: true, user: updatedUser });

} catch (error) {

res.status(500).json({ error: error.message });

}

};

// Share Profile via QR Code

exports.shareProfile = async (req, res) => {

const { qrCodeData } = req.body;

const userId = req.user.id;

try {

// You can implement a more advanced token generation or QR code verification here

const user = await User.findById(userId);

if (!user) {

return res.status(404).json({ error: 'User not found' });

}

res.status(200).json({ success: true, profile: user });

} catch (error) {

res.status(500).json({ error: error.message });

}

};

Authentication middlewear:

const jwt = require('jsonwebtoken');

module.exports = (req, res, next) => {

const token = req.headers.authorization?.split(' ')[1];

if (!token) {

return res.status(401).json({ error: 'Access denied, no token provided' });

}

try {

const decoded = jwt.verify(token, process.env.JWT\_SECRET);

req.user = decoded;

next();

} catch (error) {

res.status(400).json({ error: 'Invalid token' });

}

};

Routes:

const express = require('express');

const { registerUser, loginUser } = require('../controllers/authController');

const router = express.Router();

// User Registration Route

router.post('/register', registerUser);

// User Login Route

router.post('/login', loginUser);

module.exports = router;

const express = require('express');

const { updateProfile, shareProfile } = require('../controllers/profileController');

const auth = require('../middleware/auth');

const router = express.Router();

// Update Profile Route (Protected)

router.put('/profile', auth, updateProfile);

// Share Profile via QR Code (Protected)

router.post('/share', auth, shareProfile);

module.exports = router;

**TESTING The API**

Use **Postman** or **cURL** to test the API endpoints:

* **Registration**: POST /api/register
  + Body: { "aadhaarNumber": "123456789012", "mobileNumber": "9876543210" }
* **Login**: POST /api/login
  + Body: { "abhaNumber": "ABHA123456", "aadhaarNumber": "123456789012" }
* **Profile Update**: PUT /api/profile (requires JWT token)
  + Body: { "phone": "9876543210", "email": "user@example.com" }
* **Share Profile**: POST /api/share (requires JWT token)
  + Body: { "qrCodeData": "qr\_data\_from\_scan" }

With this setup, you have a basic back-end system for handling ABHA onboarding and profile management. You can expand it further by adding more features or customizing it based on specific needs.

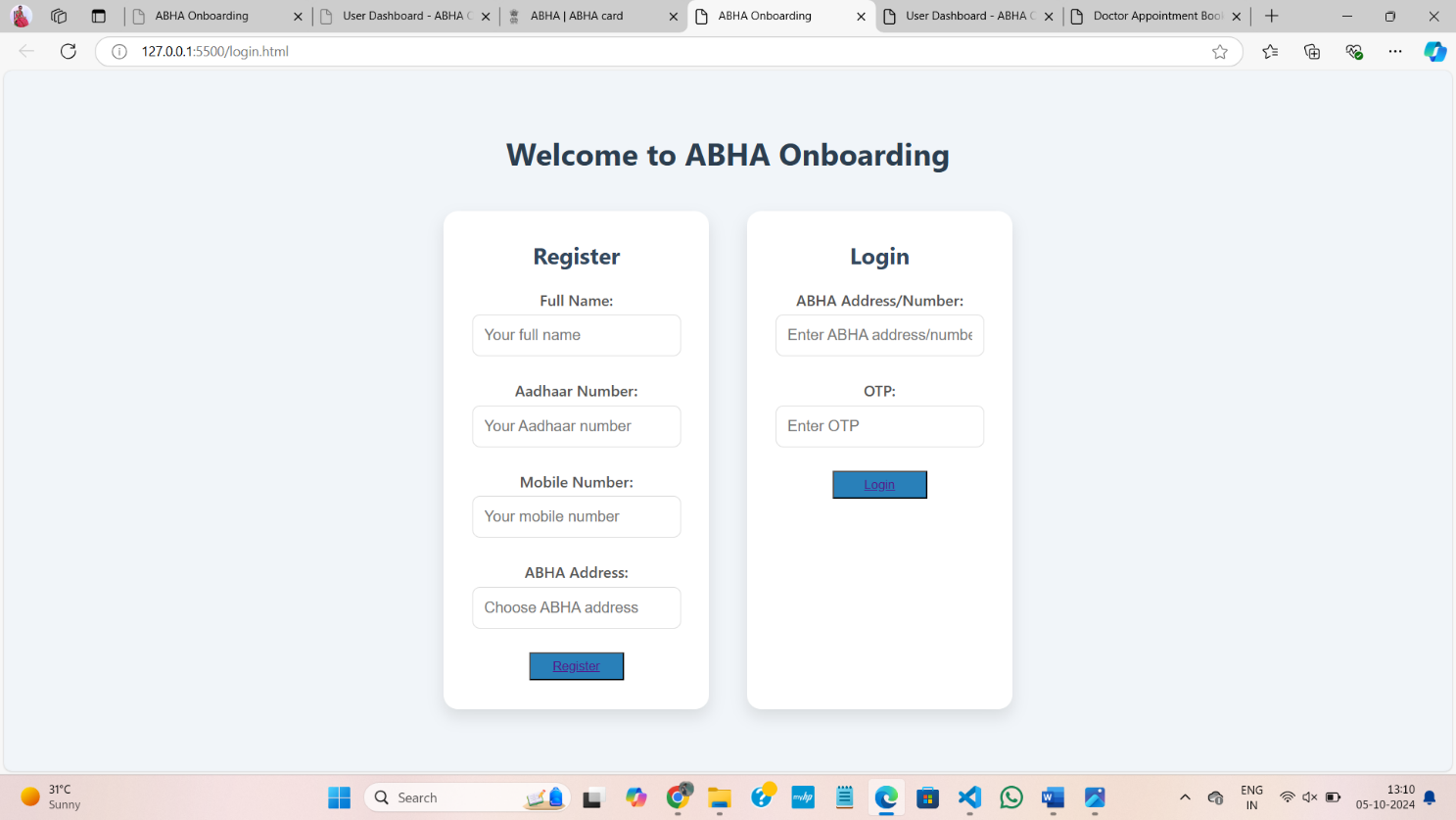
4o

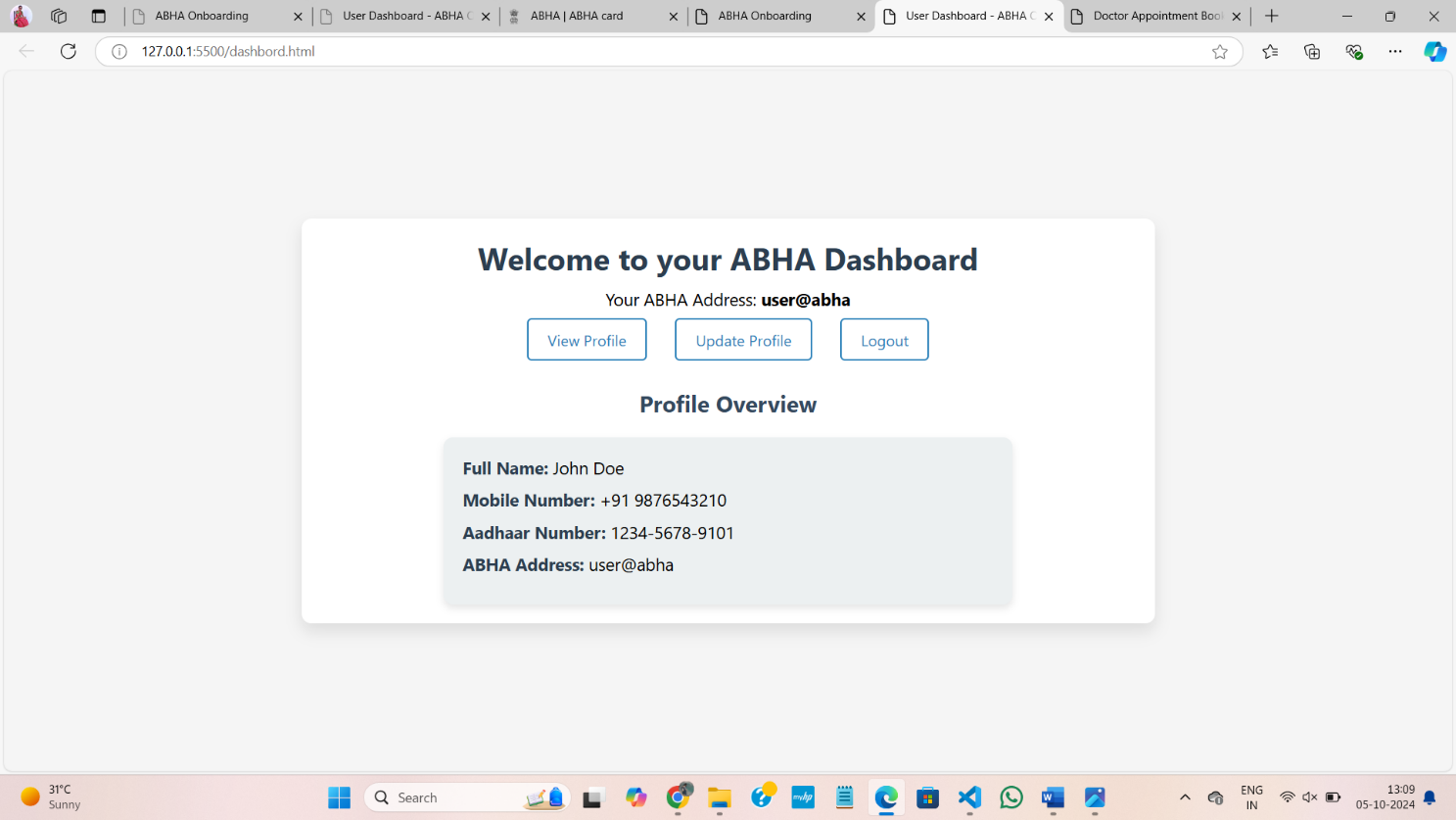
Top of Form

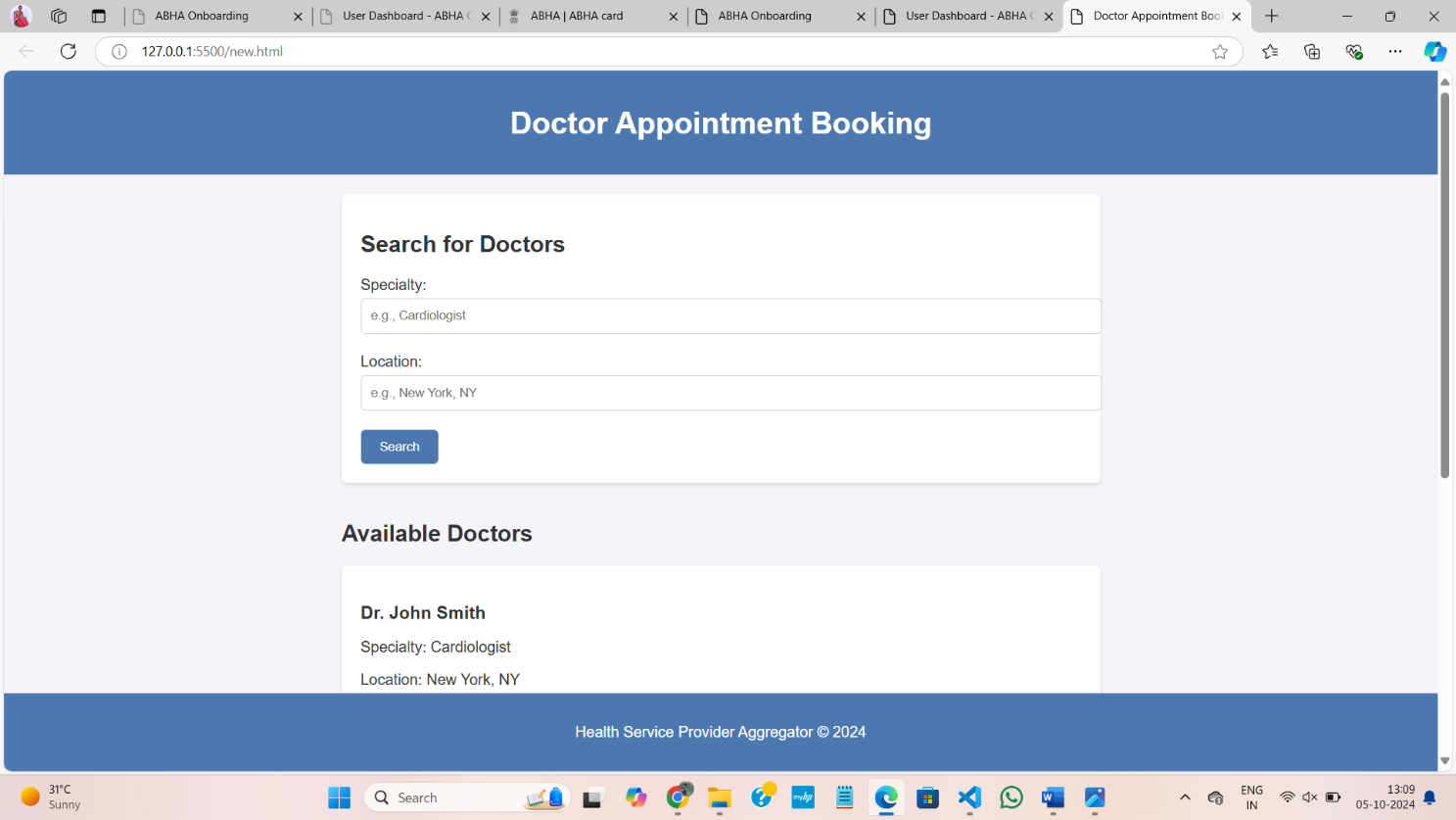
Bottom of Form

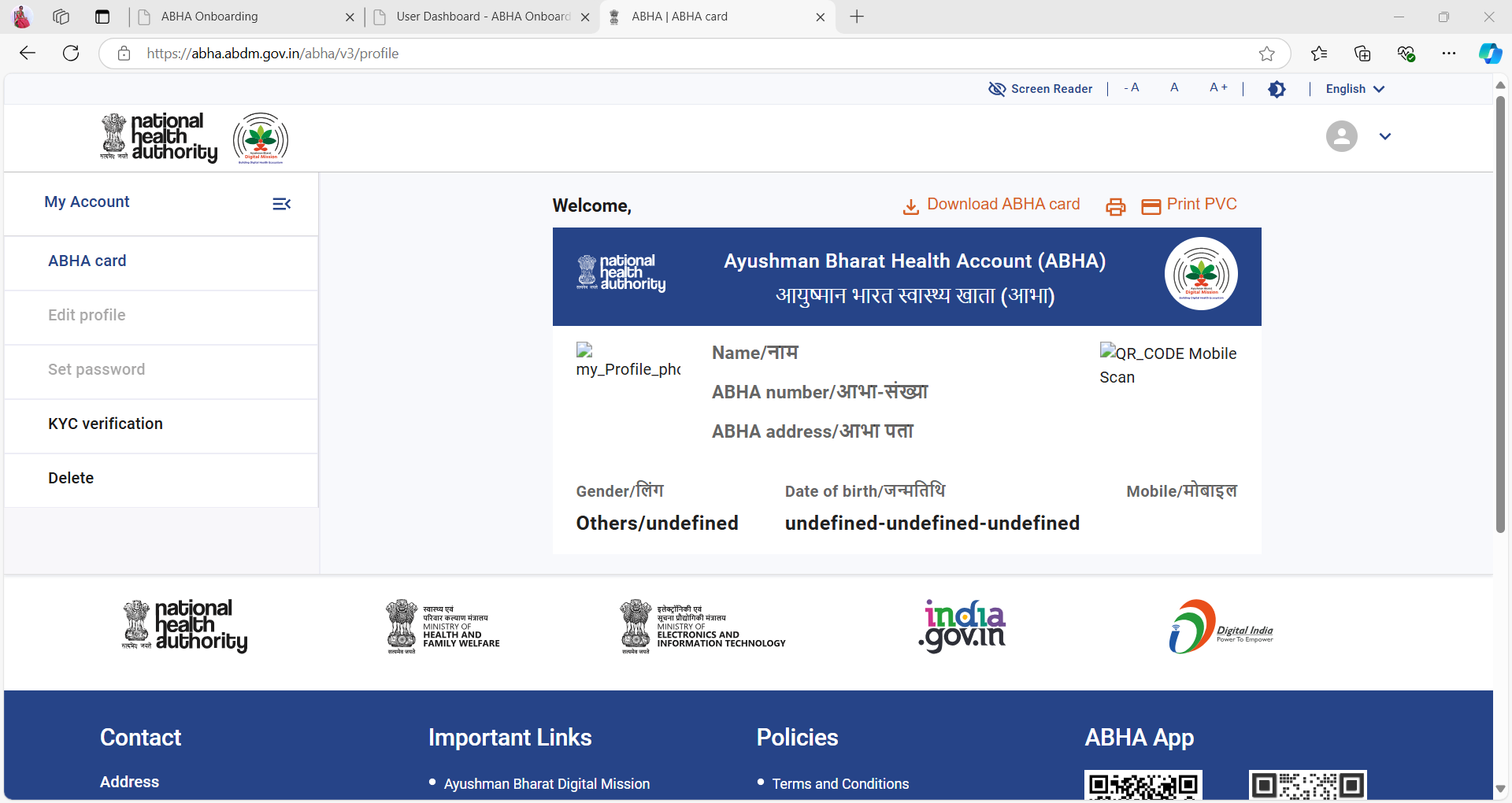
Conclusion:

The **ABHA Onboarding Prototype** serves as a secure and user-friendly platform that allows citizens to register, manage, and share their Ayushman Bharat Health Account (ABHA). It integrates essential functionalities such as **Aadhaar-based registration**, **secure login**, **profile updates**, and **QR code sharing**, significantly enhancing accessibility to healthcare services while ensuring data privacy. Built using **Flutter** for the front-end and **Node.js** for the back-end, the prototype is designed to operate seamlessly across Android, iOS, and web platforms. By employing **JWT-based authentication** and utilizing **MongoDB** for data management, the system is scalable and capable of supporting a large user base. Overall, this project exemplifies how digital solutions can empower individuals to manage their health records effectively, aligning with the goals of the **Ayushman Bharat Digital Mission (ABDM)**.

Out put view:







THANKING YOU SIR