**Online Appointment and Record System**

**Final Report for Group Project**

By Kayla Heady, Carl Lazzeri, & Kateryna Hrishina  
Live App: https://kaylaheady.github.io/webDevPortalAndScheduling/  
Repository: https://github.com/Kaylaheady/webDevPortalAndScheduling

**1. Abstract and Project Scope**

The Online Appointment and Record System is a web-based portal that streamlines appointment scheduling and medical record access for both patients and doctors. It aims to reduce administrative burden and improve data security by eliminating the need for paper-based scheduling and prescriptions. Patients can log in to book appointments, upload prescription files, and view past visit records. Doctors and admins can manage users, review appointments, and add visit notes securely.

**Target Users:** Small medical clinics and general users needing scheduling and record-keeping functionality.

**Key Features:**

* Secure login with role-based access (patient vs. doctor/admin)
* Appointment booking without time conflicts
* Prescription uploads and downloads
* Doctor’s notes tied to visit records
* Admin analytics and user management

**2. Implementation Details (Tech Stack, APIs, Tools)**

**Frontend:**

* HTML, CSS (Bootstrap 5)
* JavaScript (Vanilla)

**Application Logic:**

* JavaScript for client-side logic and API calls
* Supabase client library for data and auth interaction

**Backend/Data Layer:**

* Supabase (PostgreSQL database)
* Supabase Auth (secure login, user session)
* Supabase Storage (for file uploads)

**Database Structure:**

A diagram of a medical procedure

AI-generated content may be incorrect.

* user\_id – contains user profiles
* appointments – tracks booking details
* doctor\_data – indicates admin role
* prescriptions – stores patient documents
* doctor\_notes / visit – stores appointment notes

**2-Tier Architecture:**

* **Presentation:** HTML/Bootstrap calendar and forms
* **Application Layer:** JavaScript logic for validation, dynamic UI, and API interactions
* **Data Layer:** Supabase tables accessed via JS SDK and SQL triggers

**Tools Used:**

* Supabase Dashboard (auth, queries, triggers)
* Visual Studio Code / IntelliJ Idea
* GitHub + Git

**3. Screenshots of Functionality**

A screenshot of a computer

AI-generated content may be incorrect.Secure login with role-based access (patient vs. doctor/admin)

A screenshot of a computer

AI-generated content may be incorrect.Doctor/Admin “Dashboard” Tab with features: book appointments, upload prescriptions, view all your prescriptions, and view all your appointments.

A screenshot of a chat

AI-generated content may be incorrect.A screenshot of a medical prescription

AI-generated content may be incorrect.Doctor/Admin “Admin Tools” Tab with features: manage users, auth users, view all appointments, view all prescriptions

A screenshot of a phone

AI-generated content may be incorrect.Doctor/Admin “Analytics” Tab with features: see total users, active users, and inactive users, see user sign up over time, see appointments by hours, see feature usage comparison, and see most used feature.

A screenshot of a computer

AI-generated content may be incorrect.Regular User “Dashboard” Tab with features: book an appointment, view your prescriptions, and view your appointments.

**4. Challenges and Solutions**

**Timezone Handling:**

* **Problem:** Appointments stored in UTC using timestamptz showed wrong local time.
* **Fix:** Used toLocaleString('en-US', { timeZone: 'America/New\_York' }) in frontend and PostgreSQL AT TIME ZONE logic in analytics queries.

**Booking Conflicts:**

* **Problem:** Users could double-book time slots.
* **Fix:** Implemented conflict-check logic using Supabase queries before allowing insert.

**Secure Document Uploads:**

* **Problem:** Files needed to be tied to patients securely.
* **Fix:** Used Supabase Storage with createSignedUrl() for 60–120 sec expiring links. Prescriptions link only to uploader or matching appointment.

**Role-based Access Control:**

* **Problem:** Needed to limit admin access securely.
* **Fix:** doctor\_data table used to verify role and toggle admin UI.

**5. Division of Teamwork**

* **Kayla Heady** – Supabase setup, log in page, GitHub repo lead, UI development, styling with Bootstrap, separated Admin/User dashboards
* **Carl Lazzeri** – Logic for conflict-free scheduling, appointment system flow, database queries, admin dashboard & analytics.
* **Kateryna Hrishina** –ER diagram/database design, UI development, styling with Bootstrap, form validations, separated Admin/User interfaces, final report formatting.

All team members participated in debugging, weekly progress reviews, and presentation prep.

**GitHub Repository:**  
<https://github.com/Kaylaheady/webDevPortalAndScheduling>

**Live Web App:**  
<https://kaylaheady.github.io/webDevPortalAndScheduling>

**Submission Notes:**

* The final report (PDF) includes screenshots and system diagrams.
* Codebase contains HTML, JS, Supabase connection logic, and setup instructions.
* ER Diagram and system design docs are stored in /docs folder of the repo.