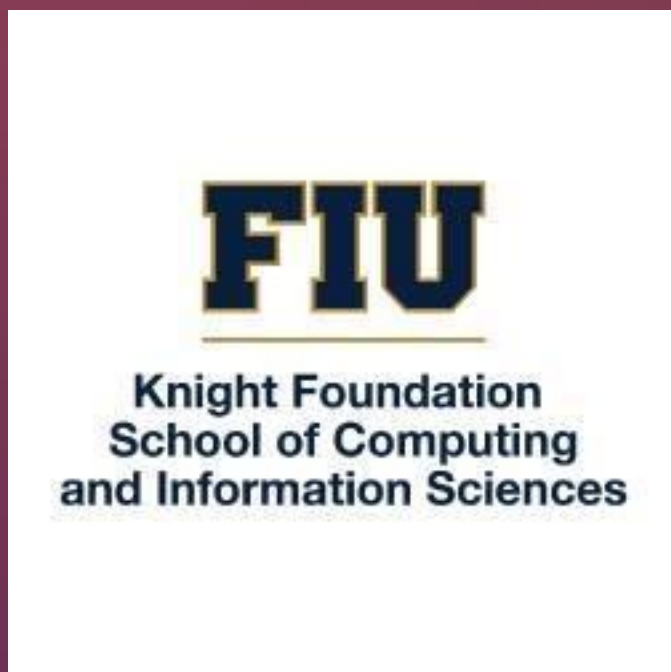




MyMedCab – Medication Management System

Students: Josue Quinonez, Cesar Calzadilla, Daniel (Niel) Escobar, Francine Portobanco, Kendrick Belizaire
Mentor: S. Masoud Sadjadi
Instructor/Faculty: S. Masoud Sadjadi, Florida International University



PROBLEM

Many patients struggle to access clear, accurate information about medications assigned by their doctors. Traditional communication methods, verbal explanations, paper instructions, or outdated patient portals often lead to confusion, missed doses, improper usage, or a lack of awareness of side effects. This gap becomes even more severe for elderly users, multilingual populations, and patients managing multiple prescriptions. There is a need for a modern, accessible platform that provides medication details quickly, reliably, and in a user-friendly format.

CURRENT SYSTEM

The current approach relies heavily on traditional methods such as written instructions, pharmaceutical labels, or even patient portals that are often too difficult to navigate or lack personalized information. These systems are barely optimized and provide limited support for users managing multiple prescriptions. As a result, patients are left without a clear, consistent way to track their medications. The absence of an intuitive, up-to-date system creates gaps in understanding and weakens communication between patients and healthcare providers.

REQUIREMENTS

Functionalities

- Patient log-in: Name, DOB, Patient ID
- Physician log-in: Name, Doctor ID, Specialty
- Patients may view prescribed medications
- Physicians are allowed to add, modify, or remove prescriptions
- Supports a pharmacy locator in the form of a map
- Data retrieval through AWS RDS / PostgreSQL
- Views control access based on entity.

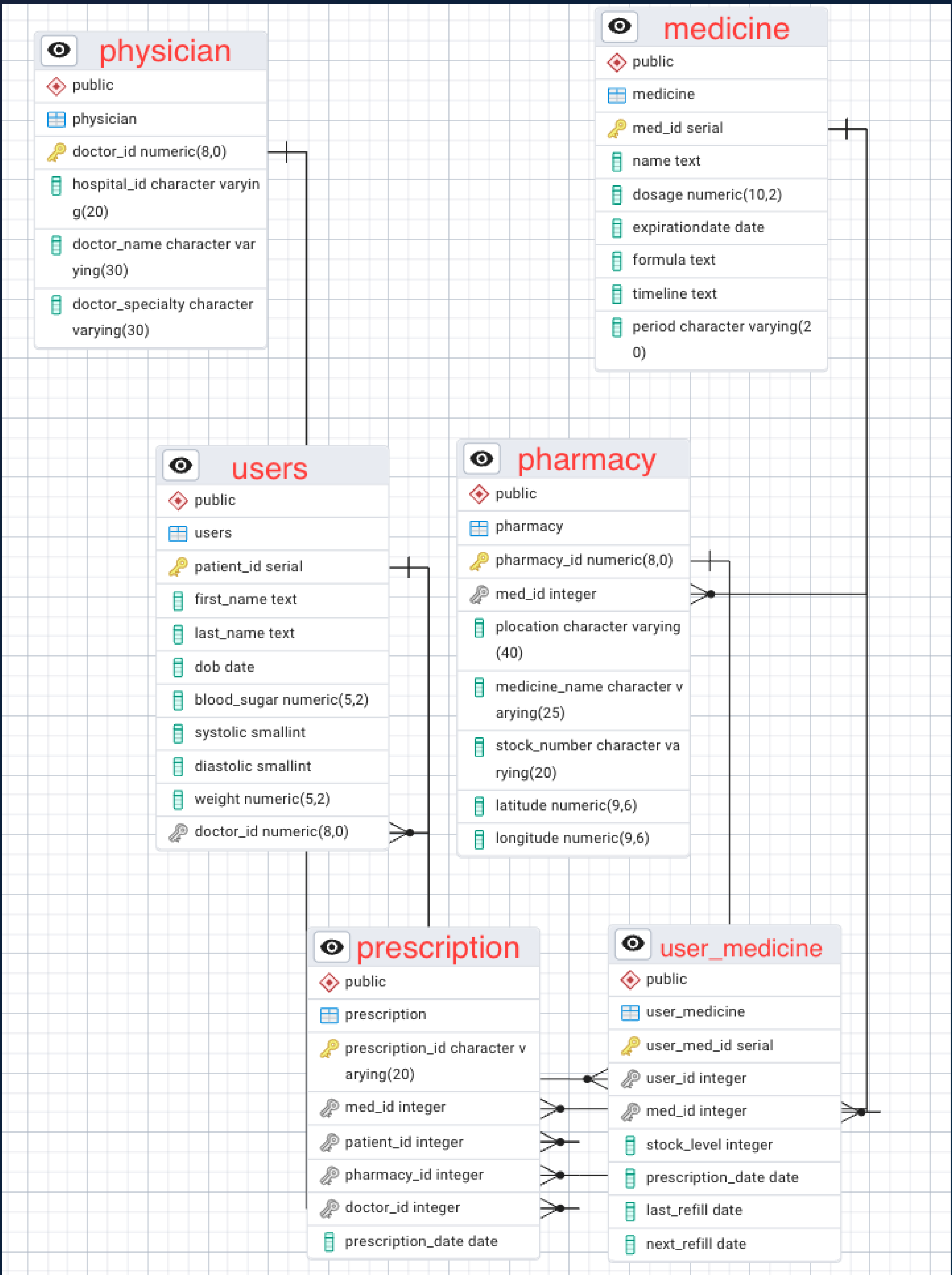
Non-Functionalities

- Secure SQL permissions and view access based on roles
- Responsive front-end through React
- Thorough AWS RDS uptime and accessibility.
- Friendly-to-use and consistent user interface

SYSTEM DESIGN

- Front-end: React + TypeScript + TailwindCSS to render UI
- Back-end: Node.js + Express for RESTful API endpoints
- Database: PostgreSQL running on AWS RDS
- Views: patient_view, physician_view, user_medication_view, physician_medicine_view
- Pharmacy Locator: Map that is based in Miami with 5 pharmacy markers
- Front-end + back-end communication through HTTPS
- Back-end queries the database with pool connections.

OBJECT DESIGN



SUMMARY

The platform provides patients with a clear, organized view of medications assigned by their doctors. It integrates a unique login system, real-time data access, and a structured system architecture to improve reliability and usability. The design focuses on clarity and safe everyday interaction for users managing prescriptions.

IMPLEMENTATION

Front-End

- Designed and implemented with React with route-based navigation.
- Contains patient and physician dashboards
- Prescription Builder
- Separate containers to view medication information
- Selector-based modification and deleting via physician dashboard
- Map view with Mapbox API

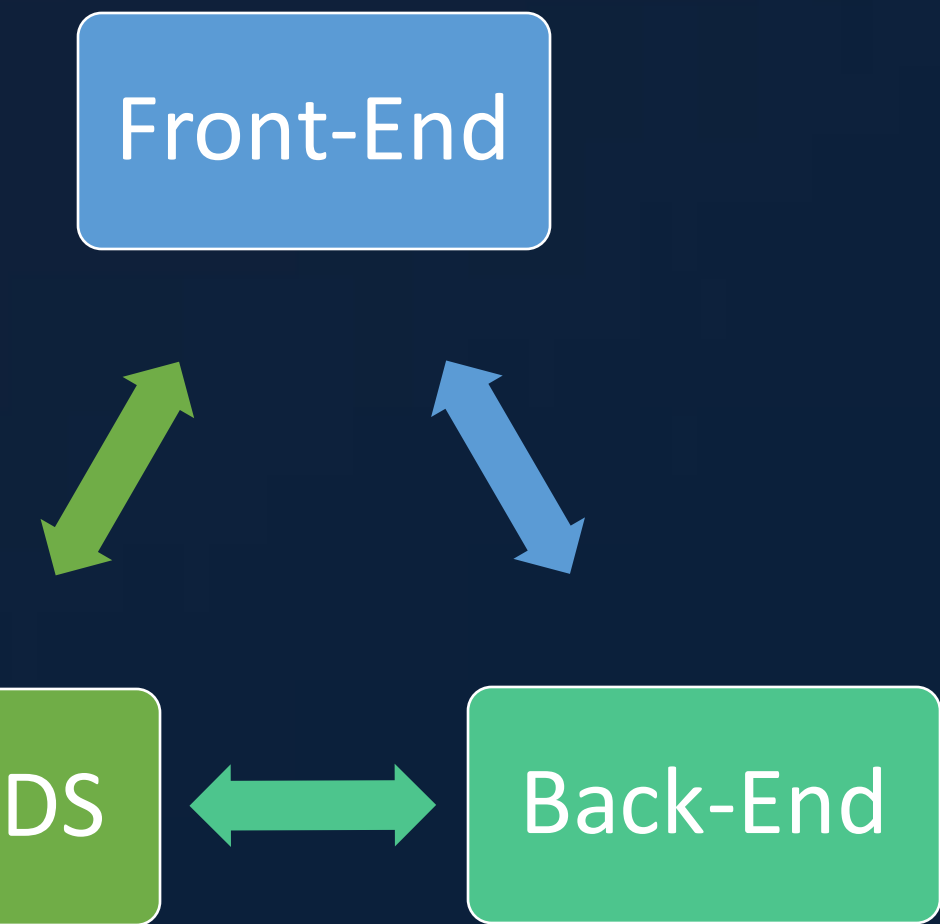
Back-End / DB

- AWS RDS + PostgreSQL instance
- Main tables: users, physicians, medicine, prescription, pharmacy, user_medicine
- Express server running on port 5001
- CRUD routes for users, prescriptions, and medicines
- Role logic for patient and physician actions
- Environment configuration using .env and connection pooling with pg

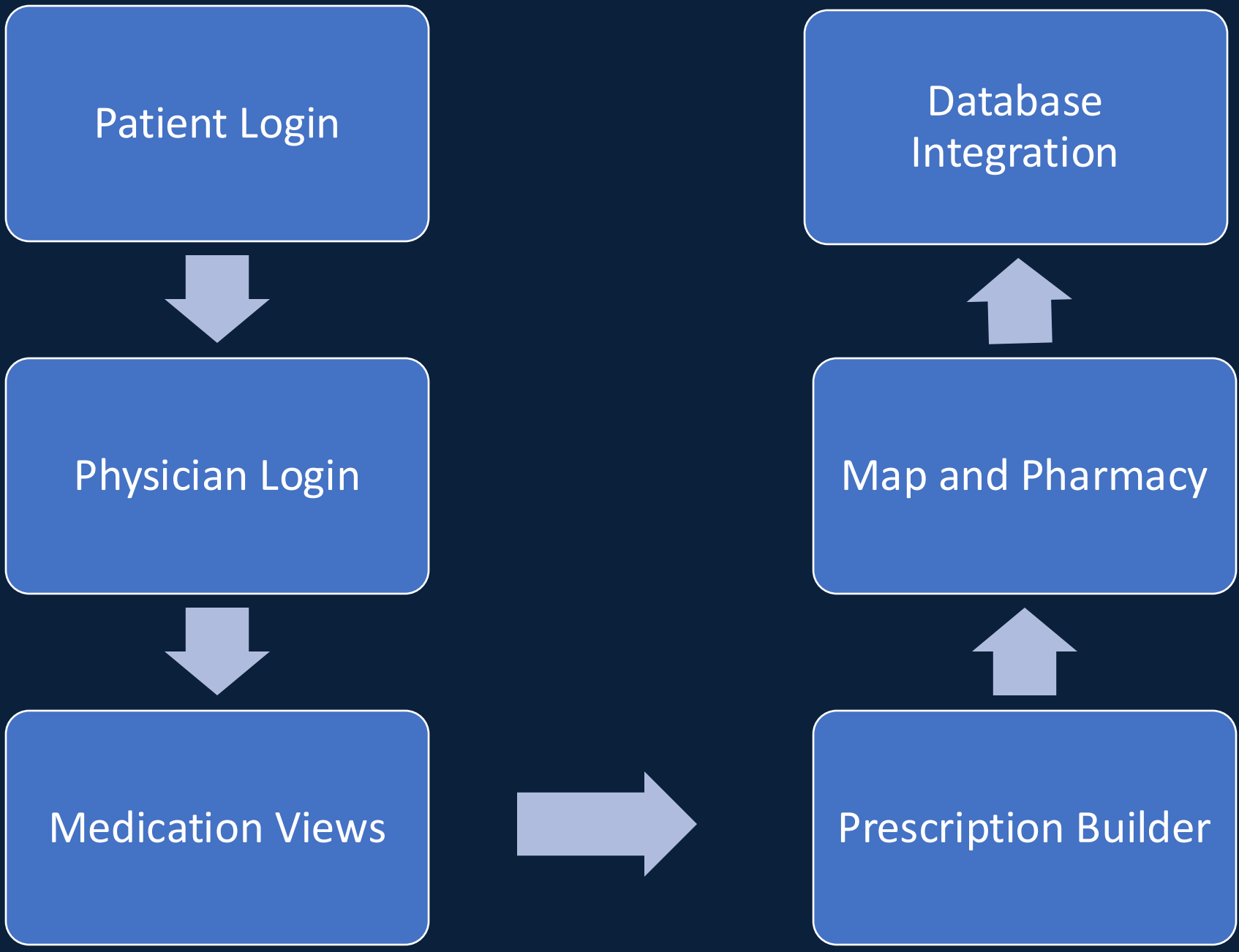


REFERENCES

- **Amazon Web Services.** *Amazon RDS for PostgreSQL – Developer Guide.* https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_PostgreSQL.html
- **React Contributors.** *React: A JavaScript Library for Building User Interfaces.* <https://react.dev/>
- **Express.js Team.** *Express Documentation.* <https://expressjs.com/>
- **PostgreSQL Global Development Group.** *PostgreSQL 17 Documentation.* <https://www.postgresql.org/docs/17/index.html>



VERIFICATION



- Patient login: we verified authentication and database lookup.
- Physician login: validated credentials and specializations.
- Medication views: ensured that any assigned medications are rendered correctly from queries
- Prescription builder: Physicians can create, update, or delete prescriptions.
- Map and Pharmacy: displayed and verified pharmacy markers and map rendering.
- DB Integration: Successful connectivity to AWS RDS

ATTENTION: Subheadings shown above is a recommended structure for a research/scientific poster. Main elements of a poster are Introduction, Research and Conclusion. Researchers can customize the section headers based on their projects' needs. REMOVE THIS TEXT ONCE YOU START WORKING ON YOUR POSTER

ACKNOWLEDGEMENT

The material presented in this poster is based upon the work supported by XXXXX (name of the project sponsor: federal, state or local government, or corporate partners, where applicable). We/I thank XXXX for his/her/their assistance and mentorship that I/we received throughout the senior design project.



Engineering
& Computing