ERStreamline

Bipin Chhetri, Nisseem Rana Magar, Raj Dhaubanjar, Roshan Acharya and Sumit Shrestha
Department of Computer Science, Texas Tech University
{bipin.chhetri | nranamag| rdhauban| roachary | sumitshr}@ttu.edu

***Abstract*—Emergency rooms (ERs) are essential to community health and depend on an effective infrastructure to deliver quick care in times of need. This proposal addresses the urgent need for a strong emergency room database management infrastructure that can operate without a hitch during medical emergencies, public health crises, and disaster relief efforts.**

***Team Roles*—Bipin Chhetri - Project Leader/FrontEnd Developer, Nisseem Rana Magar - Project Manager/Tester, Raj Dhaubanjar - Documentation Specialist/Database Developer, Roshan Acharya - Database Architect/FrontEnd Developer, Sumit Shrestha - Query Developer/BackEnd Developer**

I. PROBLEM STATEMENT

An Emergency room (ER) is a critical aspect of a community's well-being which is expected to function flawlessly with a robust infrastructure for rapid patient care in case of Emergency.

Overcrowding in ERs frequently causes delay in patient care. As a result, emergency response times become less effective, endangering patient health. It can be difficult to effectively manage patient data, records, and medical history in ERs, particularly in high-stress situations where prompt access to information is crucial while ensuring accountability under healthcare standards like the Health Insurance Portability and Accountability Act (HIPAA).

In a critical healthcare setting, our specialized "ERStreamline" database system enables real-time data access with strong security and scalability, addressing the challenges faced by emergency rooms while ensuring compliance with healthcare regulations and efficient data retrieval, surpassing traditional methods.

II. REQUIREMENTS

ERStreamline system must satisfy well-outlined requirements to support the core mission which is to provide accurate, timely, and safe healthcare to patients, especially in emergency situations and those requirements are explained below:

1) Real-time data access: The database system should provide real-time access to patient records, test results, and medical history for healthcare professionals to make quick decisions.
2) Data security and compliance: Robust security measures must be in place to protect patient information and comply with healthcare data privacy regulations (e.g., HIPAA). This includes role-based access controls, encryption, and audit trails.
3) Scalability: The system must be able to scale to accommodate a high volume of patient data, especially during emergencies and public health crises.
4) Data integrity: Ensure data accuracy and consistency to prevent errors in patient care. Implement data validation rules and constraints.
5) Disaster recovery: Establish data backup and recovery procedures to safeguard data in case of system failures or disasters.
6) User training: Provide training for all users to ensure effective use of the system according to their roles and responsibilities.

A. Technical Requirements

1) Backend: MongoDB, NodeJS, ExpressJS, Postman
2) FrontEnd: ReactJS, HTML, CSS, MaterializeUI, JavaScript Wireframes/Mockups: Balsamiq
3) Diagrams: Lucid
4) Collaborating Softwares: Github, LaTeX

III. TARGET USERS

The target use of the ERStreamline system is for healthcare professionals to streamline their work and on the other hand, patients can also utilize the system to check their test results,and medication. User roles and access privileges should be implemented to ensure data security compliance and ease of use of the system. They are explained in more detail below:

A. Healthcare professionals

This category includes doctors, nurses, and medical staff. Their needs include:

- Real-time access to patient medical records for diagnosis and treatment.
- The ability to update patient information, record test results, and prescribe medications.
- Access historical patient data to monitor progress and make informed decisions.
- Access privilege to view and modify patient records within their department or assigned patients.

B. Administrative staff

These individuals manage non-clinical aspects of the ER. Their needs include:

- Managing patient admission and discharge processes.
- Billing and insurance information management.
- Scheduling appointments and resources.

- Limited access to patient records for administrative purposes.

C. Database Administrators (DBAs)

Responsible for maintaining and securing the database system. Their needs include:

- Technical expertise in database management and security.
- Monitoring and optimizing database performance.
- Implementing data backup and recovery procedures.
- Managing user access and permissions.
- Ensuring data integrity and compliance with regulations.

D. Patients

This category includes the patients and their family members. Their needs include:

- Access to billing statements, prescribed medications.
- Access to test results and diagnosis reports.

IV. Access Control Relations

The access control relations for the tables in the database are classified below where the access control privileges are shown on the left and the tables are shown on the right. In the relations shown below (R) means read-only privileges and (R, W) means read and write privileges.

USER	ACCESS CONTROL PRIVILEGE
[Healthcare Professional]	—-(R)–>(Patient) —-(R)–>(Admission) —-(R, W)->(LabTest) —-(R, W)->(EHRVisit) —-(R, W)->(Medication)
[Administrative Staff]	—-(R, W)–>(Billing) —-(R, W)–>(Patient) —-(R, W)–>(Admission) —-(R, W)–>(Insurance) —-(R)–>(Staff)
[Database Administrator]	—-(R, W)–>(System Configuration) —-(R, W)–>(User Access Control)
[Patient]	—-(R)–>(Billing) —-(R)->(Medication) —-(R)->(LabTest) —-(R)–>(Insurance)

V. DATA

Manually inputting data for a large and diverse dataset is impractical. To address this, we will utilize synthetic data generation techniques. Specifically, we will generate realistic and randomized data by leveraging two sources: Mockaroo.com and 'Faker,' a Python library designed for programmatic synthetic data generation. To ensure comprehensive testing, we will create a minimum of 30 unique datasets for each relation. This multi-source approach will enable us to introduce significant variations in the synthetic data, closely resembling real-world usage patterns.

VI. List of Relations

Relation 1: Patients
Attributes: patient_id(PK), first_name, last_name, date_of_birth, gender, address, phone, email, blood_group

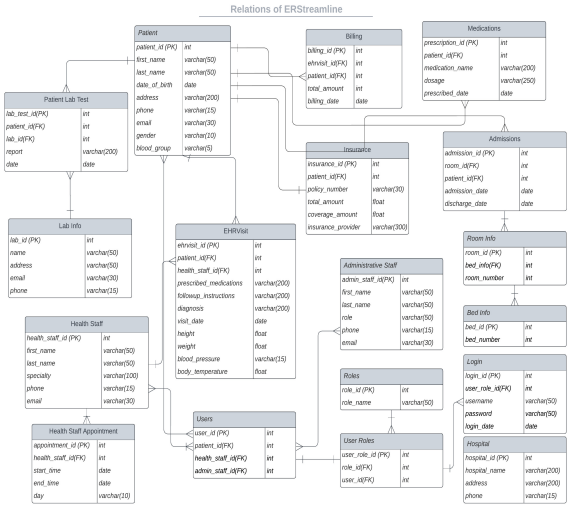

Fig. 1. Relations of ERStreamline

Relation 2: Billing
Attributes: billing_id(PK), ehrvisit_id(FK), patient_id(FK), total_amount, billing_date
Relation 3: EHRVisit
Attributes: ehrvisit_id(PK), patient_id(FK), health_staff_id(FK), prescribed_medications, followup_instructions, diagnosis, visit_date, height, weight, blood_pressure, body_temperature

VII. Interface

Our wireframe offers a visual representation of our interface design, including a login page, doctor/patient dashboard, and a patient profile form with a diagnosis report. For more details, kindly click the provided link in the figure caption.

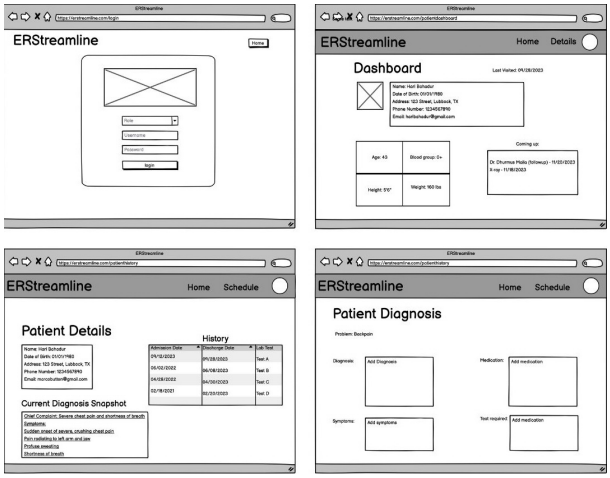

Fig. 2. ERStreamline Interfaces