Mukesh Patel School of Technology Management and Engineering

A REPORT ON

Nirog

A Hospital Management Tool

Under the supervision of

Prof. Dhananjay Joshi
AY- 2024-2025

BY

Bachelor of Technology, Computer Science
COURSE: Object Oriented Programming through JAVA

Name	Laxita Jain
Roll No.	E202
SAP ID	70552300002

Table of Contents

1) Introduction

The Nirog project is a desktop-based Java application designed for basic healthcare record management and tracking. Its primary objective is to streamline the organization and retrieval of user health information through a user-friendly graphical interface built using Java Swing. The application emphasizes simplicity, data security, and offline access, making it ideal for clinics, small healthcare centers, or even personal health tracking.

2) Requirement Analysis

In many small or rural clinics, patient records are still maintained on paper. This leads to:

- Data loss or misplacement
- Difficulty in accessing past medical records
- Time-consuming manual look-ups
- No easy way to analyze health history

The prospect clients hope to find a solution that would allow them to:

- Track patient data efficiently
- Maintain digital files for the patients
- Streamline treatment procedures
- Keep day-wise records
- Retrieve patient history of individual patients
- Find a system that is time efficient and staff friendly

By digitizing health records in a lightweight desktop app, Nirog can serve as a **low-cost alternative** to full-scale hospital management systems, especially in resource-constrained settings.

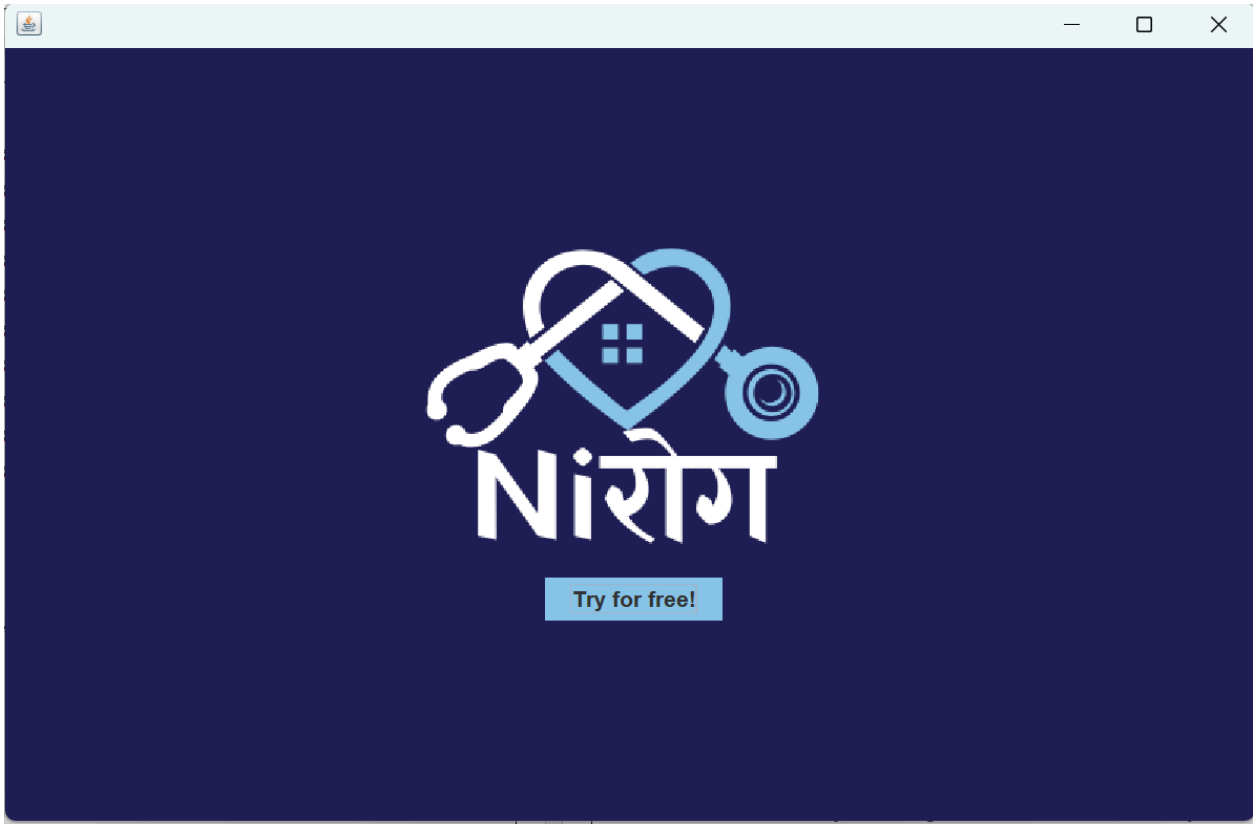
3) Development Technologies

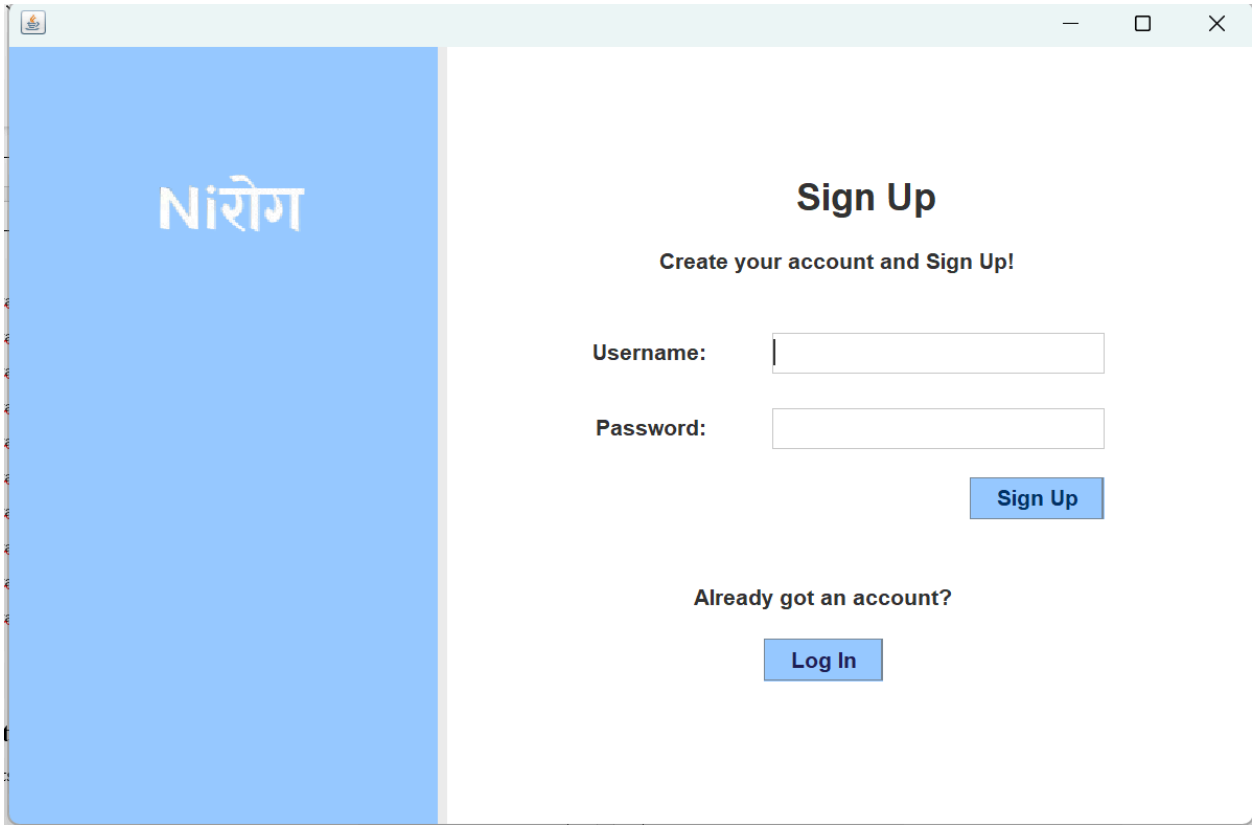
The *Nirog* application has been developed using a combination of technologies tailored for simplicity, offline operability, and ease of deployment on Windows systems.

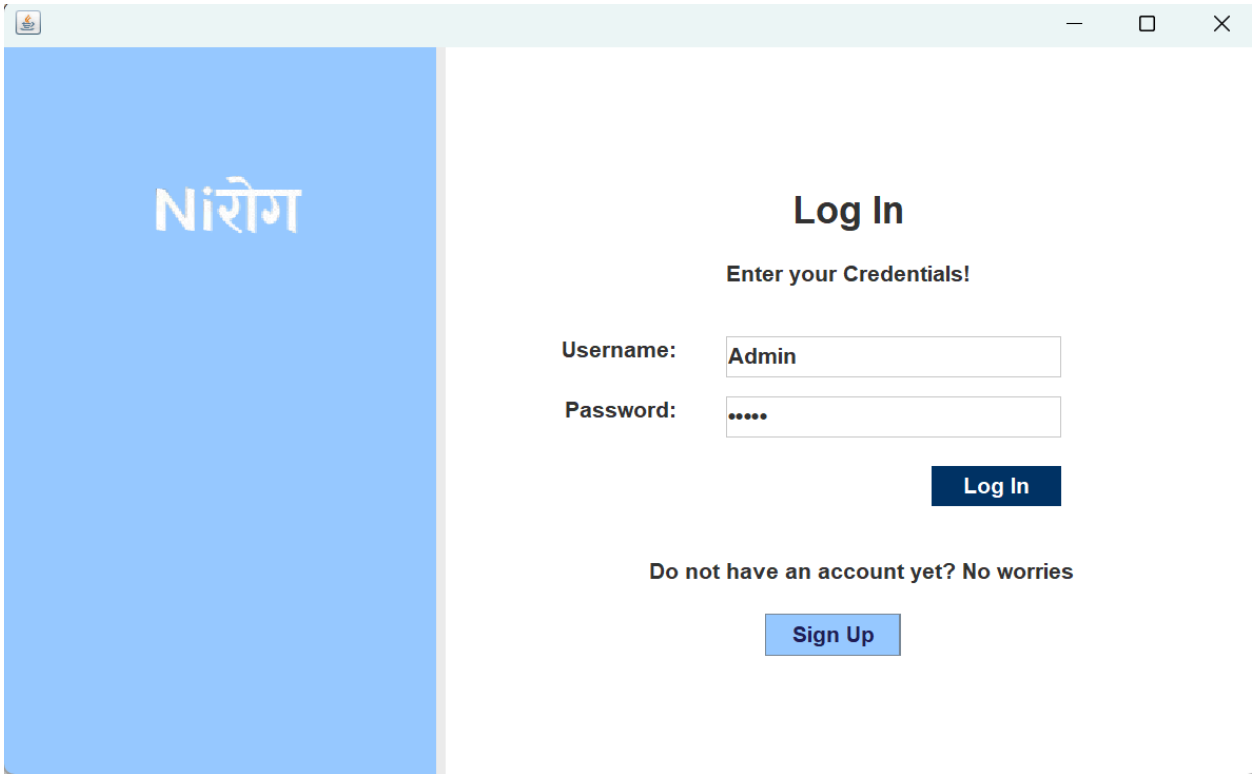
- **Frontend Interface**:
  The graphical user interface (GUI) is built using **Java Swing**, offering a responsive and intuitive experience for users to navigate healthcare records.
- **Backend Logic**:
  The core functionality is implemented using **Core Java**, ensuring platform independence and robust performance for handling user interactions and data processing.
- **Database Management**:
  For persistent storage of healthcare records, **MySQL** is used as the backend database, managed through the **XAMPP** server environment. This enables structured and scalable data storage. In simpler versions or offline modes, storage can also be achieved using text files (.txt) or embedded databases like **SQLite**.
- **Integrated Development Environment (IDE)**:
  NetBeans IDE is utilized for coding, debugging, and building the application, leveraging its seamless support for Java and Swing.
- **Build and Packaging Tools**:
  The project is packaged into an executable .jar file using NetBeans' built-in build system. To enhance user accessibility, **Launch4j** is employed to wrap the .jar into a Windows-compatible .exe file.
- **Target Platform**:
  The application is designed primarily for **Windows-based systems**, ensuring compatibility with local deployment environments commonly used in clinics and small healthcare facilities.

4) Results

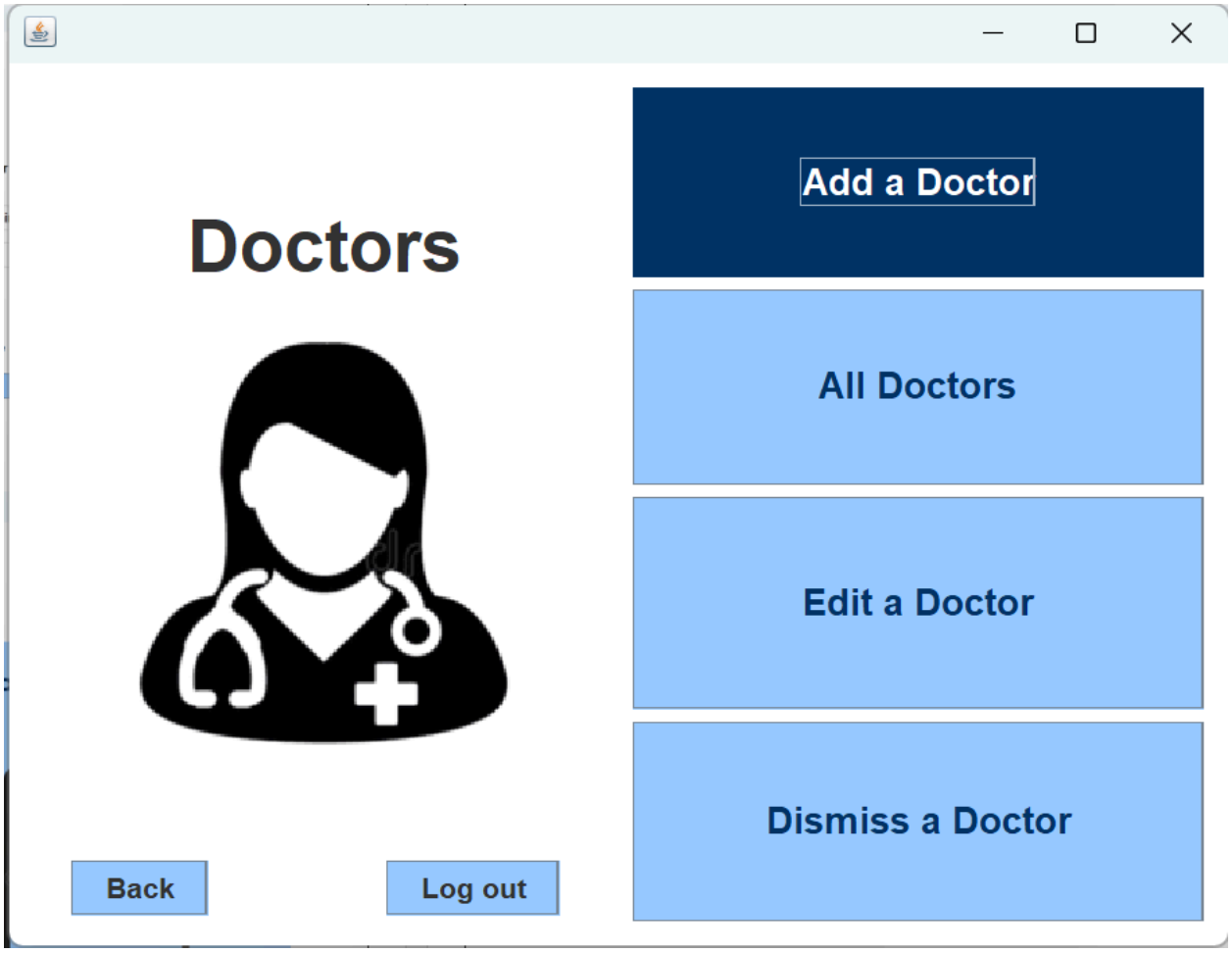
Screenshots of the running application:

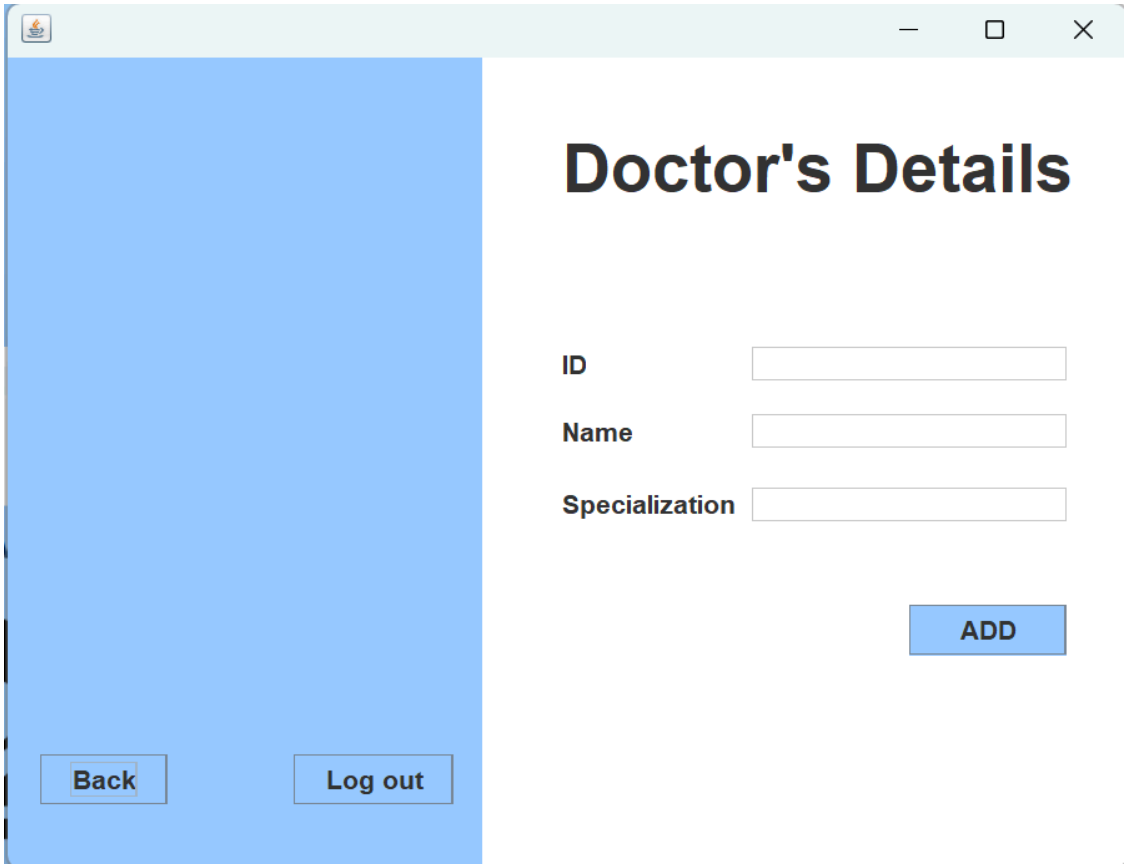

Nirोग
Try for free!

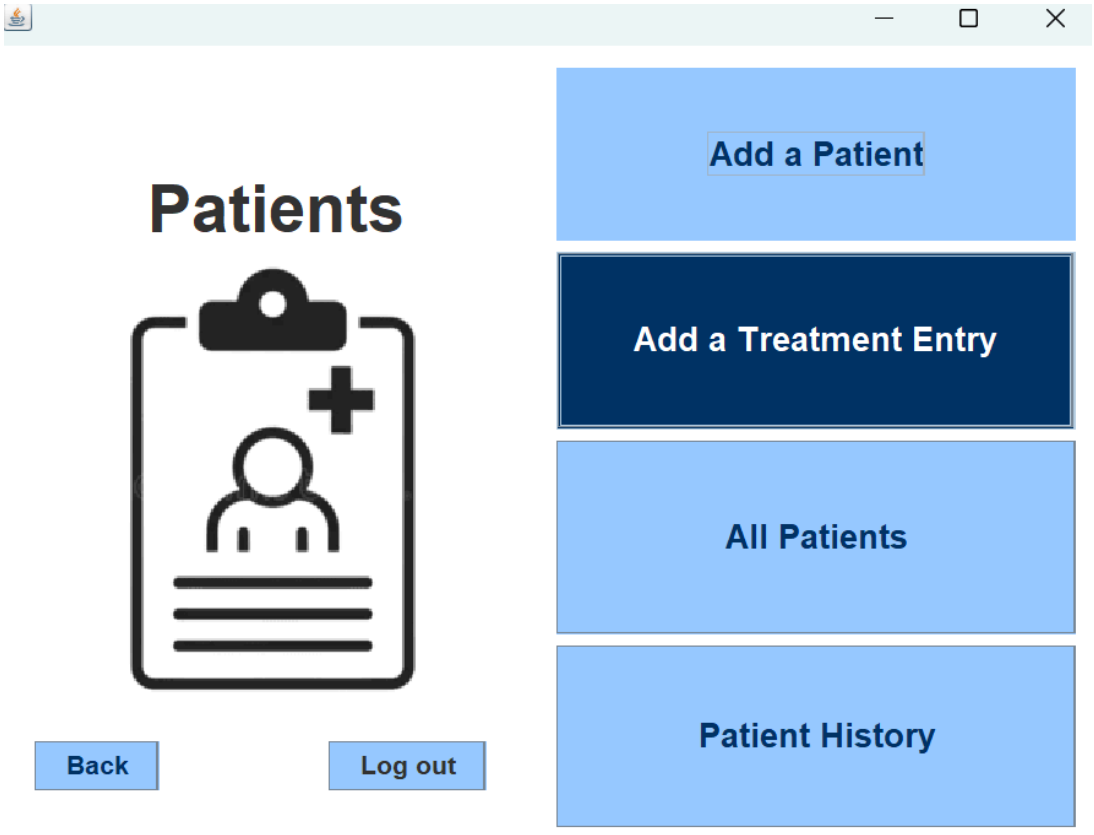

Nirोग
Sign Up
Create your account and Sign Up!
Username:
Password:
Sign Up
Already got an account?
Log In

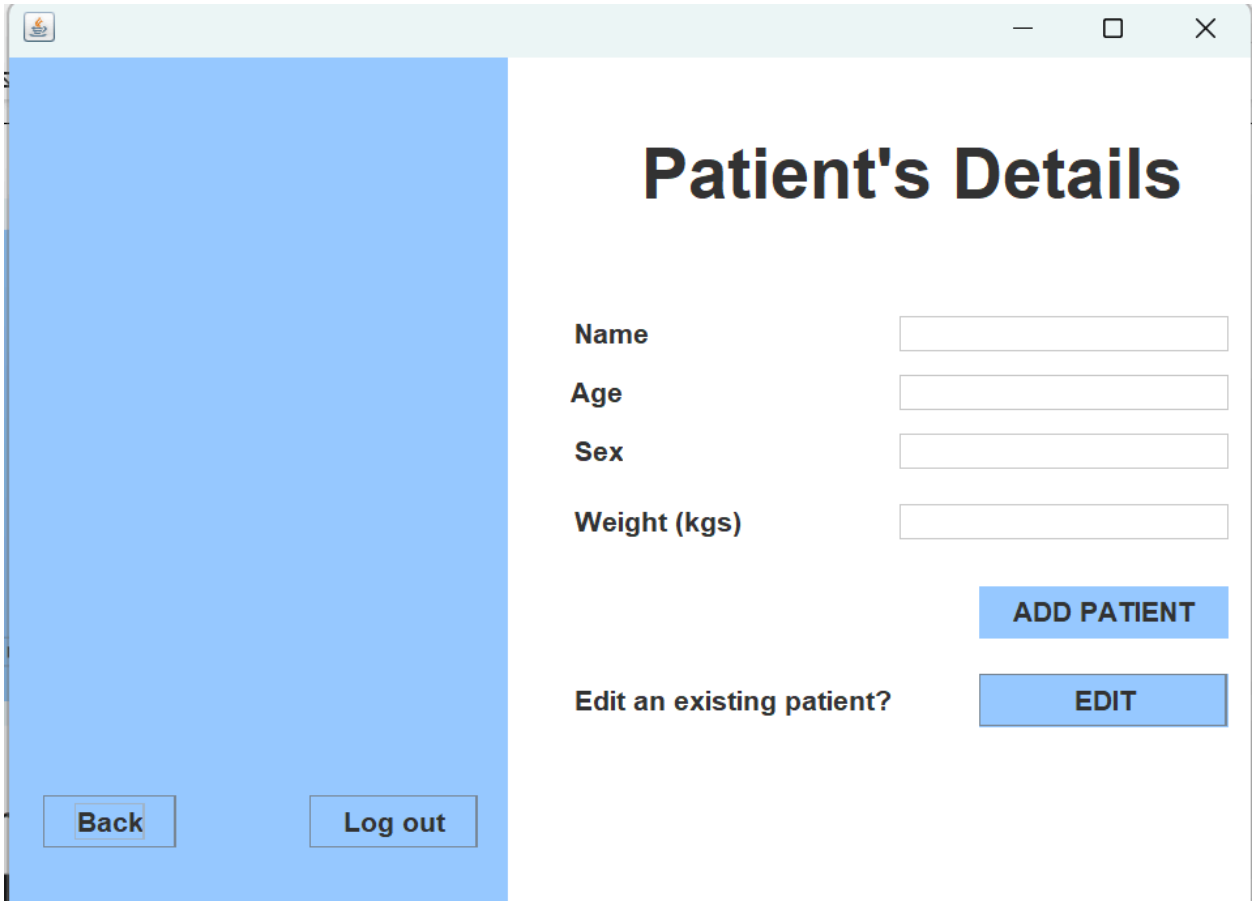

Nirोग
Log In
Enter your Credentials!
Username:
Admin
Password:
Log In
Do not have an account yet? No worries
Sign Up

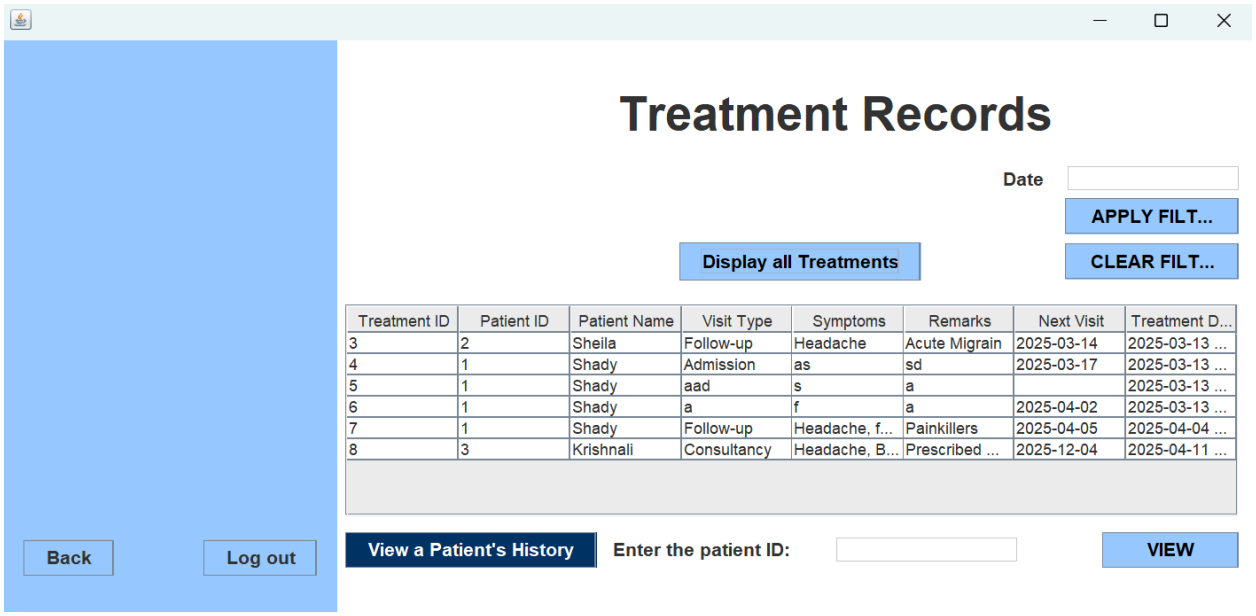

Welcome to Nirog
Log out
hospital logs made easy!
Doctor's Module
Patients' Module

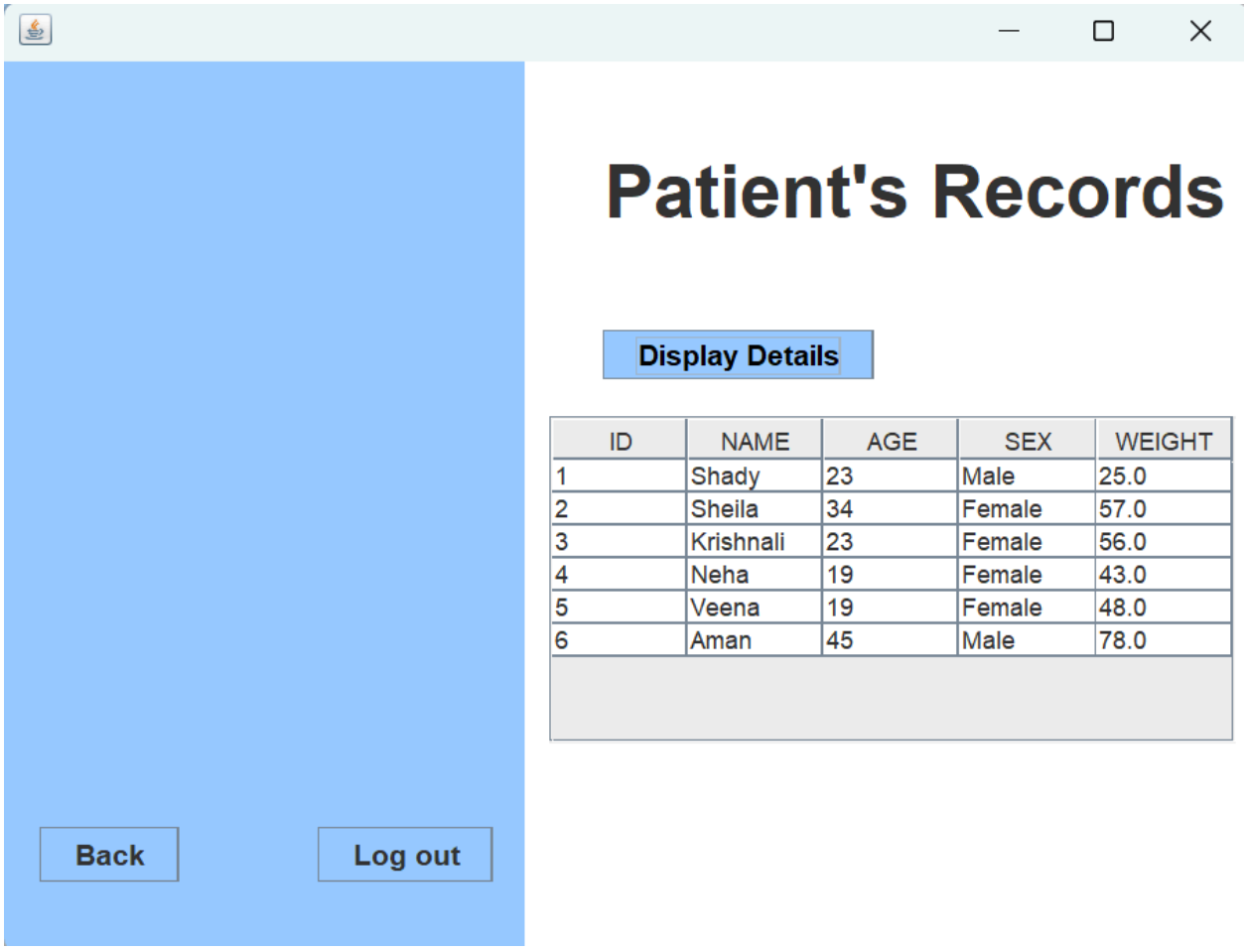

Doctors
Add a Doctor
All Doctors
Edit a Doctor
Dismiss a Doctor
Back
Log out


Doctor's Details
ID
Name
Specialization
ADD
Back
Log out


Patients
Add a Patient
Add a Treatment Entry
All Patients
Patient History
Back
Log out


Patient's Details
Name
Age
Sex
Weight (kgs)
ADD PATIENT
Edit an existing patient?
EDIT
Back
Log out


Treatment Records
Date
APPLY FILT...
Display all Treatments
CLEAR FILT...
Treatment ID | Patient ID | Patient Name | Visit Type | Symptoms | Remarks | Next Visit | Treatment D...
3 | 2 | Sheila | Follow-up | Headache | Acute Migrain | 2025-03-14 | 2025-03-13 ...
4 | 1 | Shady | Admission | as | sd | 2025-03-17 | 2025-03-13 ...
5 | 1 | Shady | aad | s | a | | 2025-03-13 ...
6 | 1 | Shady | a | f | a | 2025-04-02 | 2025-03-13 ...
7 | 1 | Shady | Follow-up | Headache, f... | Painkillers | 2025-04-05 | 2025-04-04 ...
8 | 3 | Krishnali | Consultancy | Headache, B... | Prescribed ... | 2025-12-04 | 2025-04-11 ...
Back
Log out
View a Patient's History
Enter the patient ID:
VIEW


Patient's Records
Display Details
ID | NAME | AGE | SEX | WEIGHT
1 | Shady | 23 | Male | 25.0
2 | Sheila | 34 | Female | 57.0
3 | Krishnali | 23 | Female | 56.0
4 | Neha | 19 | Female | 43.0
5 | Veena | 19 | Female | 48.0
6 | Aman | 45 | Male | 78.0
Back
Log out

5) Observations and Learnings

Project follows a modular structure with separate classes for Patient, Doctor, Appointment, etc. Uses basic Object-Oriented Programming (OOP) principles like encapsulation. Menu-driven console interface for user interaction. All data is stored in memory (not saved after program ends).No GUI or database integration in the basic version. Basic input validation and exception handling present. No role-based access or authentication system.

Through this project, we gained a better understanding of Object-Oriented Programming concepts and class-based design, which helped us structure the application more effectively. We learned how to manage user input and control the program flow using a menu-driven system. This experience made us realize the importance of planning the overall structure before writing code. We also understood the limitations of handling data only in memory and identified opportunities for future enhancements, such as adding a database and developing a graphical user interface. Overall, this project boosted our confidence in building real-world console-based applications and emphasized the importance of proper error handling and data validation.

6) Future Scope

In the future, we can enhance this Hospital Management System by integrating advanced functionalities such as automated appointment reminders, secure role-based access for different users like doctors, nurses, and administrators, and comprehensive analytics for patient trends and hospital performance. Features like cloud storage for remote access, telemedicine support, e-prescriptions, and integration

with wearable health devices can further expand the system's capabilities. These improvements will make our application more scalable, intelligent, and suitable for modern healthcare environments.

7) Conclusion

In conclusion, our Hospital Management System successfully demonstrates how core Java concepts, database integration, and a user-friendly GUI can be combined to manage essential hospital operations effectively. Through this project, we streamlined tasks such as patient record management, appointment scheduling, and staff handling. The system not only meets basic healthcare administration needs but also lays a solid foundation for more sophisticated future enhancements.