LVX
VERITAS
VIRTVS

Northeastern
University

A project report on

Cancer Statistic Explorer

Course code: CS5200

Course name: Database Management System

Under the Guidance of

Professor Kathleen Durant

Submitted in partial fulfilment for the award of the degree of Masters

by

Andrew Panzone

Ishan Aggarwal

Maalolan Bharaniraj

Fall Semester 2023

README Section

Introduction

This README provides detailed instructions for setting up and running the CancerStatsExplorer project on your computer. The CancerStatsExplorer is a comprehensive tool designed to manage and analyze cancer-related statistics. This guide will cover the prerequisites, installation steps, and how to run the application.

Overview

CancerStatsExplorer is a Java-based application integrated with a MySQL database. It is designed to facilitate the management and analysis of cancer-related statistics, providing a user-friendly interface and robust data handling capabilities.

Prerequisites

Before setting up the project, ensure the following software and tools are installed:

1. Java Development Kit (JDK):
   Purpose: The JDK is essential for compiling and running Java applications.
   Version: JDK 11 or higher.
   Download: Java JDK Download.
   Installation Directory:
   Windows: Typically C:\Program Files\Java\jdk-11.
   Linux: Usually /usr/lib/jvm/java-11.
   Configuration: Set the JAVA_HOME environment variable to the JDK installation path and update the PATH variable to include the JDK bin directory.

2. MySQL Server:
   Purpose: Serves as the backend database for storing and managing data.
   Version: MySQL 8.0 or higher.
   Download: MySQL Server Download.
   Installation Directory:
   Windows: Default is C:\Program Files\MySQL\MySQL Server 8.0.
   Linux: Typically /usr/local/mysql.
   Configuration: Ensure MySQL service is running and accessible. Set up a

user with necessary privileges for database creation and manipulation.

3. Maven (for Java project dependency management):
Purpose: Maven is used for project management, build automation, and dependency management.
Download: Maven Download.
Installation Guide: Maven Installation.
Configuration: Add Maven bin directory to the PATH environment variable.

Project Setup Instructions

Backend Setup (Database)

1. **Database Initialization**:
   - Start MySQL Server using your preferred method (e.g., MySQL Workbench, command line, or service manager).
   - Using a MySQL client or command line, execute the SQL script located at **Backend/cancerDB.sql**. This script creates the **cancerstatisticdb** database and sets up the required tables.
   - Verify the database and tables creation by connecting to MySQL and checking for **cancerstatisticdb**.
2. **Data Import** :
   - If CSV files are provided in **Backend/db csv**, import them into the corresponding tables in the database.
   - Use MySQL import functionality or tools like phpMyAdmin for data import.

Frontend Setup (Application)

1. **Building the Application**:
   - Open a terminal or command prompt.
   - Navigate to the **Frontend** directory of the project: **cd path/to/Frontend**.
   - Run **mvn clean install** to build the project. Maven will download dependencies, compile the source code, and generate an executable JAR file.

- Verify the build success by checking for a **BUILD SUCCESS** message in the terminal and the presence of the **target** directory containing the JAR file.

Running the Application

- After a successful build, the executable JAR file is located in **Frontend/target**.
- Run the application using the command **java -jar target/CancerStatsExplorer.jar** (adjust the jar file name based on the Maven build output).
- Upon launching, the application should connect to the MySQL database and be ready for user interactions.

Libraries and Dependencies

The project's Java dependencies are managed by Maven and are specified in the **pom.xml** file located in the **Frontend** directory. Key libraries typically include:

- **JDBC** for MySQL connectivity.
- **Swing** for building the Graphical User Interface (GUI).
- Additional utility libraries as required.

Troubleshooting and Verification

- **Database Connection Issues**: Ensure MySQL is running and the **cancerstatisticdb** is correctly set up. Check the database connection parameters in the Java application, typically found in a configuration file or within the source code.
- **Build Failures**: Verify that JDK and Maven are correctly installed and configured. Check for any error messages in the Maven build output for specific issues.

Technical Specifications

Overview

The CancerStatsExplorer is an advanced software application designed to manage, analyze, and visualize cancer-related statistical data. This section provides a detailed overview of the technical specifications of the project, encompassing the software architecture, technologies used, and specifics of the implementation.

Software Architecture

Multi-tier Architecture:

- The project follows a multi-tier architecture, dividing the application into separate layers: frontend (client-side), backend (server-side), and database.
- This separation enhances maintainability, scalability, and security.

Client-Server Model:

- The frontend and backend communicate over a network, following the client-server model, ensuring a responsive and interactive user experience.

Technologies Used

Frontend:

- Framework/Libraries: The frontend is developed using standard web development technologies and Java-based frameworks.
- Responsiveness: The application is designed to be responsive, ensuring compatibility across various devices and screen sizes.

Backend:

- Programming Language: Java, known for its portability, efficiency, and robustness, is the primary language used.
- Framework: Java-based frameworks like Spring Boot or Java EE might be used for efficient backend management.
- APIs: RESTful APIs for handling client-server communication, ensuring a stateless, scalable, and flexible interaction model.

Database:

- Database Management System: MySQL is used for data storage, offering reliability and efficiency in handling structured data.
- Schema: A relational schema, optimized for complex queries and data integrity.

Build Tool:

- Maven: Used for dependency management and build automation, simplifying the build process and ensuring consistency across development environments.

Development Environment

Integrated Development Environment (IDE):

- Recommended IDEs include IntelliJ IDEA or Eclipse, both offering extensive support for Java development, debugging tools, and integration with Maven and version control systems.

Version Control:

- Git, hosted on platforms like GitHub or Bitbucket, for source code management, allowing for collaborative development and version tracking.

Application Specifications

User Interface:

- Intuitive and user-friendly, designed for ease of navigation and clarity in data presentation.
- Interactive elements like charts, graphs, and tables for data visualization.

Data Processing:

- Capabilities for importing, exporting, and processing cancer-related data.
- Features for filtering, sorting, and searching data to aid in analysis.

Security:

- Implementation of security best practices to protect sensitive data.
- Authentication and authorization mechanisms for user access control.

Performance:

- Optimized for performance, ensuring quick response times even with large datasets.

- Use of caching, efficient algorithms, and database optimizations.

Scalability and Maintainability

Scalability:

- Designed to handle increased load and data volume effectively.
- Scalable architecture, allowing for the addition of new features and modules.

Maintainability:

- Code structured for readability and maintainability.
- Comprehensive documentation and adherence to coding standards.

Compatibility

Cross-Platform Compatibility:

- The application is compatible with multiple operating systems, including Windows, macOS, and Linux, thanks to the use of Java and web technologies.

Browser Support:

- The frontend is compatible with major browsers like Chrome, Firefox, Safari, and Edge.

Deployment

Deployment Environment:

- The application can be deployed on various environments, including local servers, cloud platforms, or dedicated hosting services.
- Containerization (e.g., Docker) can be used for easy deployment and scaling.

Backend Specifications

Database System

- **Type**: Relational Database Management System (RDBMS).
- **Technology**: MySQL.
- **Recommended Version**: 8.0 or higher, known for its performance and security features.

Database Schema

Structure:

- Consists of multiple interconnected tables designed to store cancer statistics, user data, and other relevant information.
- Tables such as demographic group, state, users, user_action, activitylog, etc., suggest a multifaceted approach to data management.
- Normalization: The database design likely follows normalization principles, potentially up to the third normal form, to reduce data redundancy and improve integrity.

Data Integrity and Relationships:

- Use of primary and foreign keys to define relationships between tables.
- Inclusion of integrity constraints (e.g., **NOT NULL**, **UNIQUE**) to enforce data quality.
-

Data Import and Management

CSV Integration:

- Indicated by the **db csv** directory, the system seems to support importing data from CSV files, facilitating easy data entry and updates.

Frontend Specifications

Application Development

- **Language**: Java.
- **Required JDK Version**: JDK 11 or higher for compatibility and performance.
- **Source Code Management**: Presumably organized following Maven's standard directory layout.

Build and Dependency Management

Maven:

- Utilized for dependency resolution, build automation, and project lifecycle management.
- The **pom.xml** file in the Frontend directory contains configurations for Maven, including project dependencies, plugins, and other build settings.

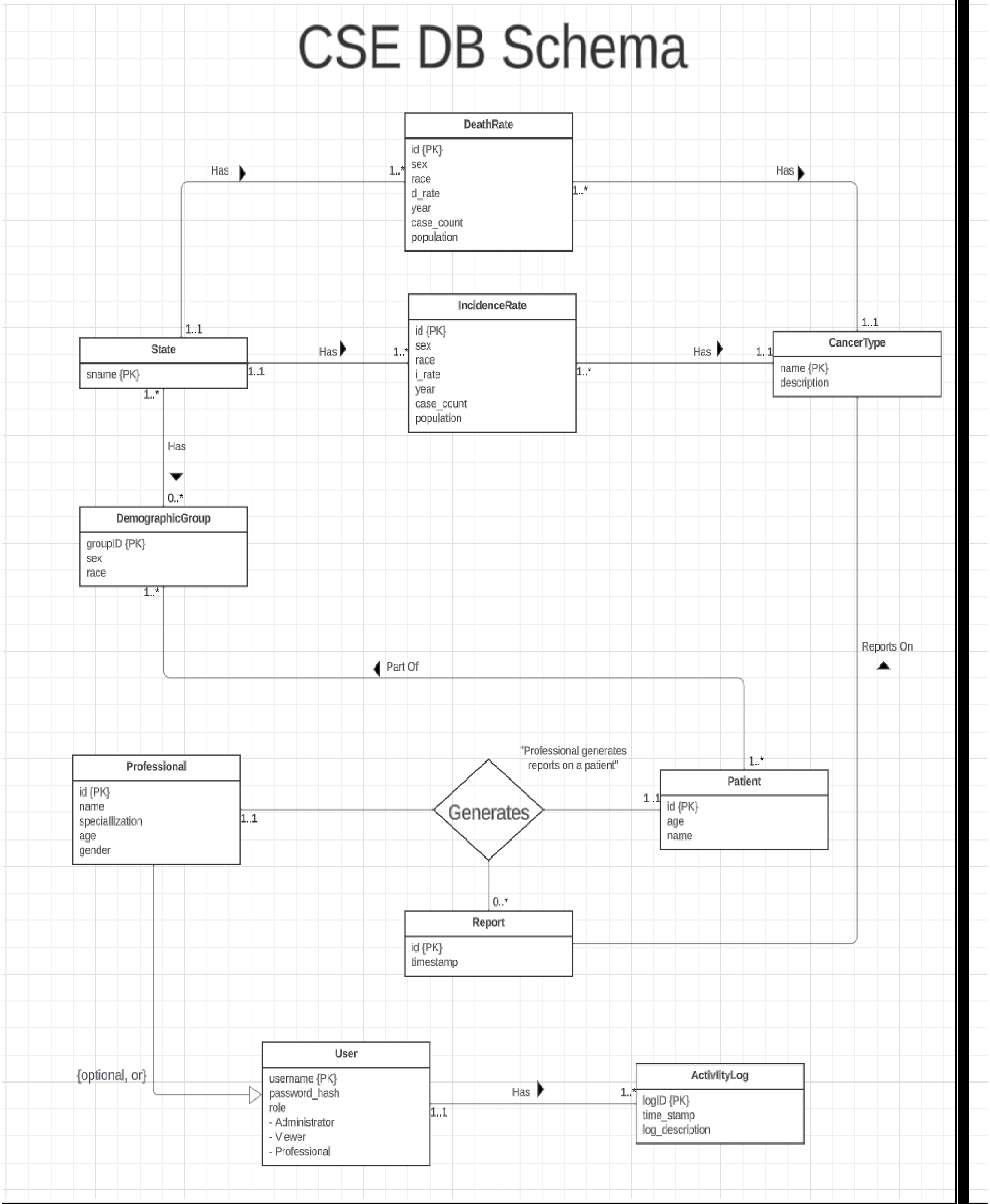
Key Libraries and Frameworks

- **JDBC (Java Database Connectivity)**: Facilitates interaction between the Java application and the MySQL database, handling operations like querying and updating the database.
- **Java Swing**: Employed for creating the graphical user interface (GUI), indicating the application has a desktop-based interface.
- **Additional Libraries**: The project may use additional Java libraries for various functionalities, which can be identified from the **pom.xml** file.

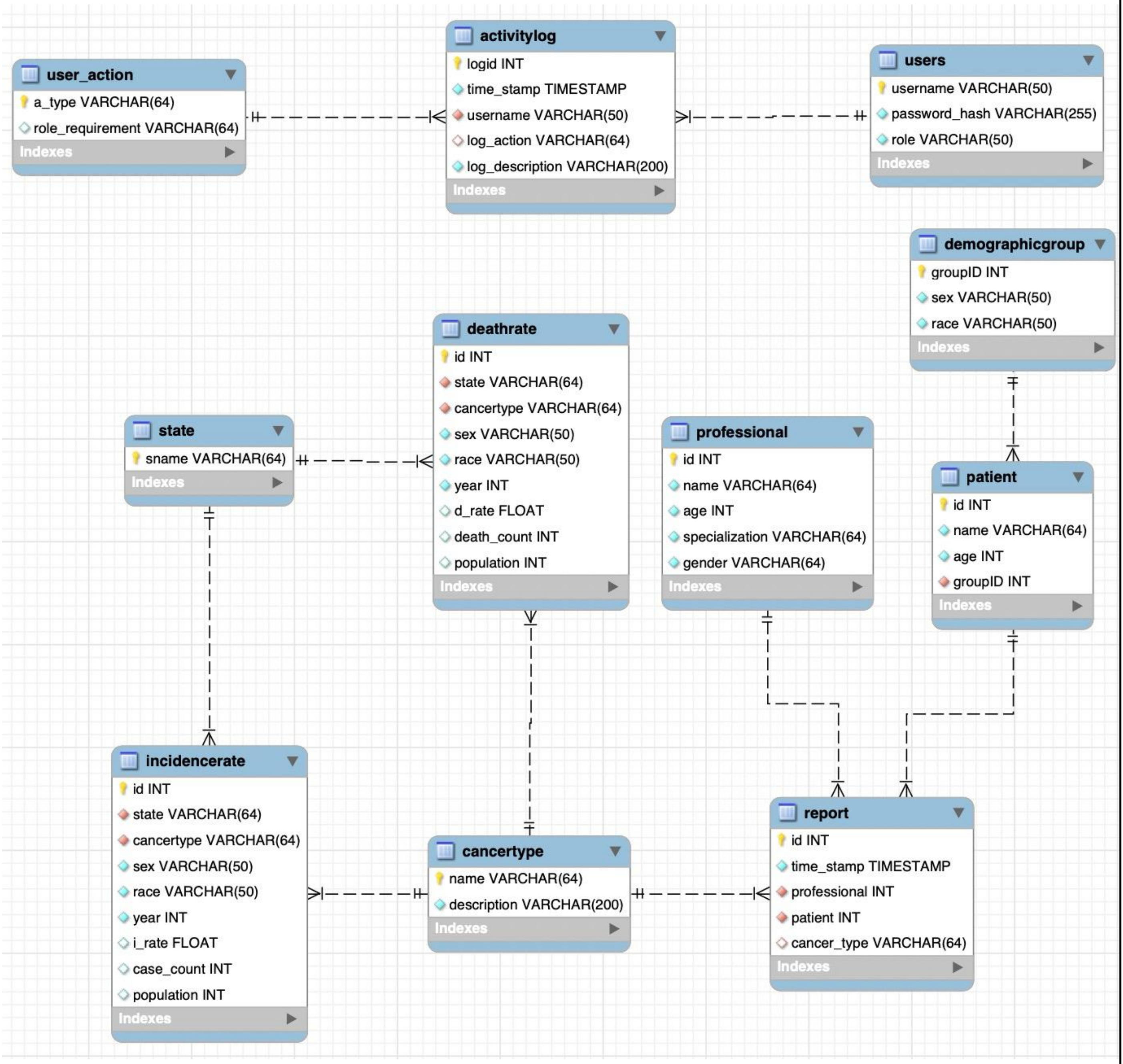
Development and Execution Environment

- **IDE Compatibility**: The project can be developed and executed in Java-supporting IDEs such as IntelliJ IDEA, Eclipse, or NetBeans.
- **Code Organization**: The **Frontend/src** directory contains the Java source files, structured as per Maven's conventions, facilitating easy navigation and development.
- **Configuration**: The application requires configuration for database connection parameters, likely managed within Java property files or directly in the code.

Conceptual Design



Logical Design for our Schema



User Flow Description:

The CancerStatsExplorer system is designed to provide users with an intuitive and efficient way to interact with and analyze cancer statistics. The user flow of the system is structured to facilitate easy access to data and features, ensuring that users can navigate and utilize the system effectively. The following is a detailed description of the user flow, including the commands or methods users perform to interact with the system.

1. Connect to Database:

- Concurrently with the application launch, the system establishes a connection to the database that holds user credentials and cancer statistics data.

2. User Authentication

Registration and Authentication:

- Upon starting, the user is presented with the option to either log in or register a new account.

Login:

- Users start by logging into the system using their credentials (username and password).
- The system validates the credentials against the database.
- Upon successful login, the user is directed to the main dashboard.

Logout:

- Users can log out of the system, which ends their session and returns them to the login page.

3. Dashboard Overview

- Dashboard Access:
  - Upon successful login, users are presented with a dashboard.
  - The dashboard provides a high-level overview of key statistics, recent updates, and quick navigation options.

4. Data Exploration and Analysis

- Viewing Statistics:
    - Users can view various cancer-related statistics, which may include incidence rates, mortality rates, and demographic data.
    - Options to filter and sort data based on criteria like region, cancer type, age group, etc.
- Data Visualization:
    - The system may offer graphical representations of data.
    - Users can interact with these visualizations to gain deeper insights.

5. Detailed Data Analysis

- Advanced Search:
    - Users can perform advanced searches using specific parameters.
    - This feature allows for detailed exploration of the data.
- Report Generation:
    - Users have the option to generate reports based on the data displayed.
    - Reports can be customized and downloaded in various formats (PDF, Excel, etc.).

6. Data Management (For Admins or Authorized Users)

- Adding Data:
    - Authorized users can add new data entries to the database.
    - This could involve entering details through a form or uploading data files.
- Updating Data:
    - Users with the necessary permissions can edit existing data entries.
    - This is crucial for keeping the database up-to-date and accurate.
- Deleting Data:
    - Authorized users can delete outdated or incorrect data entries.

7. User Account Management (For Admins)

- Creating User Accounts:
    - Admins can create new user accounts, assigning roles and permissions.
- Managing User Accounts:

    - Admins can manage existing user accounts, updating details or deactivating accounts as necessary.

8. System Settings and Preferences

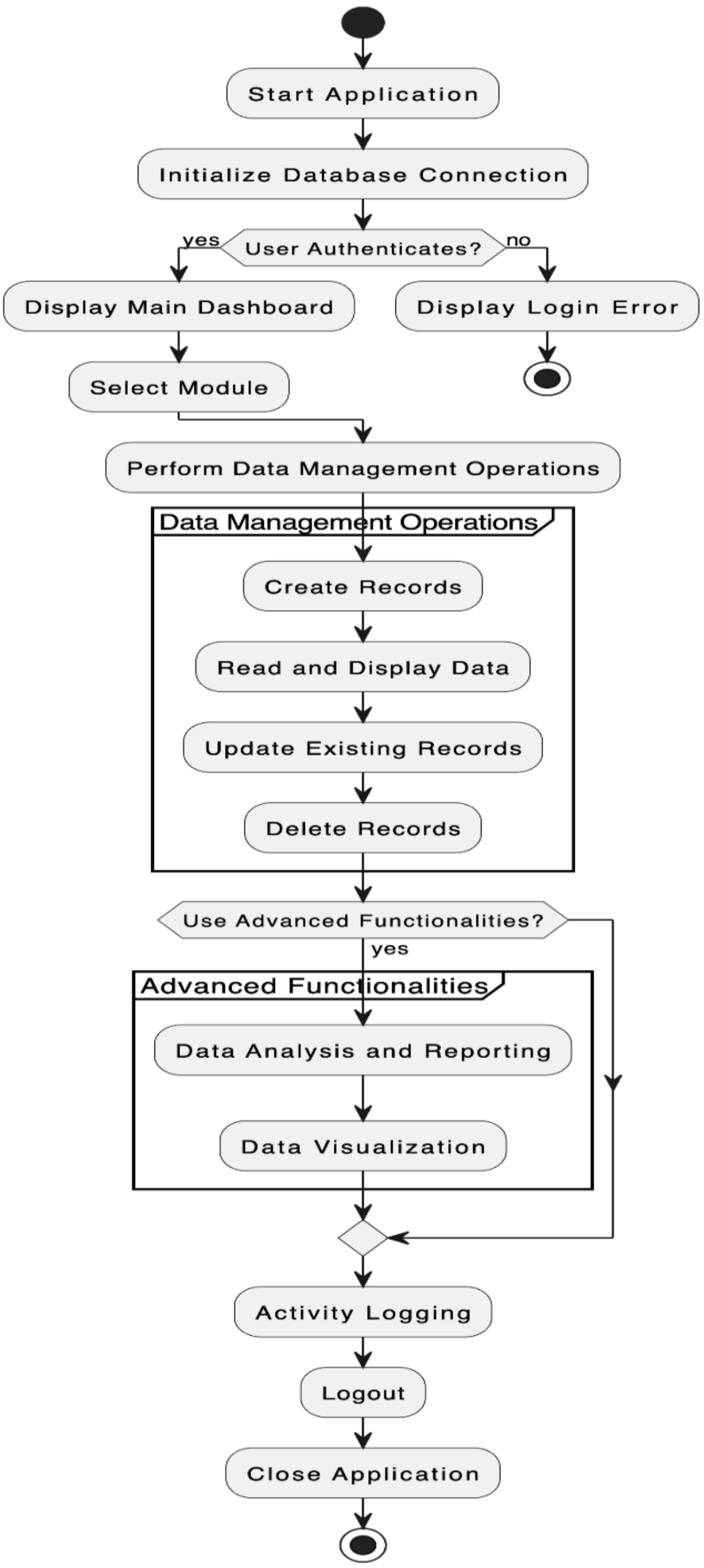
- Updating Preferences:
    - Users can update their system preferences, such as display settings, notification preferences, etc.
- Help and Support:
    - Access to help resources, FAQs, and support contact information.
- Feedback Submission:
    - Users can submit feedback or report issues directly through the system.

9. Logging Out/Ending Session

- Session Termination:
    - Users can securely log out of the system, ensuring their session is safely closed.

User Flow Diagram:

User Flow of CancerStatsExplorer System



Lessons Learned

Technical Expertise Gained

1. Java and Swing Framework:

- **Advanced Java Concepts**: Deepened understanding of advanced Java concepts such as multithreading, exception handling, and stream API which were crucial for efficient application development.
- **Swing for GUI**: Gained significant expertise in Swing, learning to create more responsive and user-friendly interfaces, and exploring advanced components and customization techniques.

2. Database Design and SQL Proficiency

- **Complex SQL Queries**: Developed the ability to write more complex SQL queries, enhancing skills in data manipulation and retrieval.
- **Database Optimization**: Learned techniques for optimizing database performance, such as indexing and query tuning.

3. JDBC and Database Connectivity

- **Connection Management**: Acquired a deeper understanding of managing database connections in Java using JDBC, including handling connection pooling for improved efficiency.
- **Transaction Management**: Learned the importance and implementation of transaction management to ensure data consistency and reliability.

4. Maven and Build Automation

- **Dependency Resolution**: Mastered Maven for complex dependency management and resolution.
- **Automated Build Processes**: Leveraged Maven for automating build and deployment processes, significantly improving development efficiency.

Insights

1. Project Management and Agile Methodology

   Agile Practices: Adopted Agile methodologies for better project management, learning to work in sprints, manage backlogs, and adapt to changing requirements.

   Risk Management: Gained insights into identifying potential risks early in the project lifecycle and developing mitigation strategies.

2. Collaborative Development

   Code Reviews and Pair Programming: Emphasized the importance of code reviews and pair programming sessions for improving code quality and team learning.

   Cross-functional Collaboration: Learned to effectively collaborate with team members with different expertise, leading to a more integrated and robust system.

Data Domain Insights

1. Healthcare Data Sensitivity

   Data Privacy and Security: Gained an in-depth understanding of the sensitivity and privacy concerns surrounding healthcare data, learning to implement stricter security measures.

   Compliance with Regulations: Learned about healthcare regulations like HIPAA, ensuring the system's compliance with data handling and privacy laws.

Alternative Designs and Approaches

1. Cloud-based Solutions

   Cloud Database Consideration: Explored the possibility of using cloud-based databases for scalability and high availability but remained with on-premise solutions for greater data control.

2. Frontend Technology Alternatives

   Web vs Desktop: Extensively debated between developing a web-based interface and a desktop application, weighing factors like accessibility, performance, and security.

   Modern UI Frameworks: Considered using modern UI frameworks like React or Angular for the frontend to enhance user experience but decided to continue with Swing for its integration with Java.

3. Data Visualization and Analytics

   Integration of Advanced Analytics: Explored integrating more sophisticated analytics and machine learning for predictive analysis but deferred due to project scope and time constraints.

4. API-First Approach

   Decoupling Backend and Frontend: Contemplated an API-first approach to decouple backend and frontend development, which would have allowed more flexibility and scalability.

Future Work

Planned Uses of the Database

1. Expanded Data Collection

   Broader Data Scope: Expand the database to include a wider range of cancer-related data, such as more detailed patient histories, treatment methods, and outcomes.

   Integration with Medical Records: Plan to integrate the system with electronic medical records (EMRs) for real-time data updates and comprehensive patient data management.

2. Enhanced Data Analytics

   Predictive Analytics: Implement predictive modeling tools to analyze trends and forecast cancer incidence and survival rates.

   Clinical Research Support: Utilize the database to support clinical research studies, providing a rich source of data for observational studies and clinical trials.

Potential Areas for Added Functionality

1. User Interface and Experience Improvements

   Modern UI/UX: Upgrade the Swing-based UI to a more modern framework, possibly transitioning to a web-based interface for broader accessibility and enhanced user experience.

   Customizable Dashboards: Add functionality for users to create customizable dashboards, allowing them to tailor the interface to their specific needs and preferences.

2. Advanced Reporting and Visualization

   Dynamic Reporting Tools: Develop dynamic reporting tools that allow users to generate custom reports based on various parameters.

   Interactive Data Visualization: Implement interactive data visualization tools for more intuitive analysis of complex data sets.

3. Cloud Migration and Scalability

   Cloud-Based Infrastructure: Migrate the system to a cloud-based infrastructure to improve scalability, reliability, and accessibility.

   API Development: Develop a RESTful API to facilitate easier integration with other systems and potential future expansions.

4. Security Enhancements

   Robust Data Security: Implement advanced security protocols and encryption methods to protect sensitive health data.

Compliance and Auditing: Strengthen compliance with healthcare regulations and introduce comprehensive auditing mechanisms.

5. Mobile Application

   Mobile Accessibility: Develop a mobile application to provide convenient access to the system for users on the go.

Justification for Future Work

The proposed future work aims to elevate the CancerStatsExplorer project from a functional cancer data management system to a more versatile and powerful tool for healthcare professionals and researchers. By expanding the database and enhancing functionality, the system can cater to a broader range of needs and adapt to the evolving landscape of healthcare data management. The focus on user experience, advanced analytics, and security ensures that the system remains relevant, secure, and efficient in meeting the challenges of managing sensitive health data. The justification for each area of future work is grounded in the goal of making the CancerStatsExplorer a more comprehensive, user-friendly, and secure platform for cancer data analysis and research.

Bonus Work done:

1. Application supports multiple user roles.

2. Additional front end functionality such as website or a GUI.

3. Interesting queries that can be used for analysis or visualization of the data .

4. Overly complicated translations from user operations to database operations.

5. Complicated schema – user data pull requires multi-joins, or many tables (> 10) due to the complexity of the data domain.

6. Visualization of the data.

THANKYOU