ZU
YD

Digital CMAS Tracking System

Zuyd Applied
Data Science &
Artificial Intelligence

PR07-Project Report

Subtitle

Empowering Doctors and Children with JDM through a Java-Based CLI

Date

April 2025

Course

PR07- Introduction to Software Engineering

Institution

ZUYD University of ADSAI

Student Name

Zahra Amiri

2404000amiri@zuyd.nl

Z U
Y D

TABLE OF CONTENTS

ZU
YD

INTRODUCTION

In the evolving field of digital healthcare, solutions that improve accessibility and reduce the clinical workload are becoming increasingly essential. This project presents a targeted software solution for children diagnosed with Juvenile Dermatomyositis (JDM), a rare autoimmune muscle disorder that requires continuous monitoring of muscle strength, typically assessed through CMAS (Childhood Myositis Assessment Scale) exercises.
The goal of this project was to design a digital system using Java and SQLite, enabling patients or their caregivers to record CMAS test results remotely. Through a secure command-line interface, doctors can access patient data and monitor disease progression over time. By digitizing CMAS tracking, the system enhances convenience for families and provides structured, analysable data for healthcare professionals and clinical researchers.

BACKGROUND

Juvenile Dermatomyositis (JDM) is a rare autoimmune disease in children, causing chronic inflammation and muscle weakness. Accurate monitoring through CMAS scores is essential for evaluating progression and treatment effectiveness.
In the Dutch healthcare system, JDM patients often face difficulties in maintaining regular assessments due to travel needs and manual data handling. These challenges inspired us to design a digital tracking system to support families and provide better access to structured data.
Weekly planning, forum discussions, and the use of tools like Miro helped identify priorities like usability, data clarity, and long-term use. By combining CLI-based technology with real datasets such as PatientX and LabResult, the project supports both clinical decision-making and research opportunities.

METHODOLOGY

We followed a structured 10-day development plan, based on iterative and use-case-driven design. Weekly forum reflections and Miro planning boards were used to refine our direction, validate requirements, and prioritize features collaboratively. Our methodology can be broken down as follows:

1. Requirement Analysis: Reviewing the disease, understanding CMAS scoring, and interviewing potential users (e.g., parents, doctors).
2. Planning: Creating a Minimum Viable Product (MVP) plan and assigning roles within the group.
3. Design: Drawing UML class diagrams, mapping database tables, and structuring CLI menus.
4. Development: Implementing modular Java classes for each function using SQLite via JDBC.

1. Testing: Manually testing each feature with input variations and checking database results.
2. Documentation: Writing the final report, README, and preparing for GitHub submission.

Tools used include Visual Studio Code, GitHub, draw.io (for UML), and DB Browser for SQLite.

REQUIREMENTS AND USE CASES

FUNCTIONAL REQUIREMENTS

Patient Registration
Add a new patient with name and birth date.

CMAS Test Recording
Record a CMAS test result with date and score.

Progress Monitoring
View test results for a selected patient.

Secure Doctor Access
Allow doctors to log in using email and password.

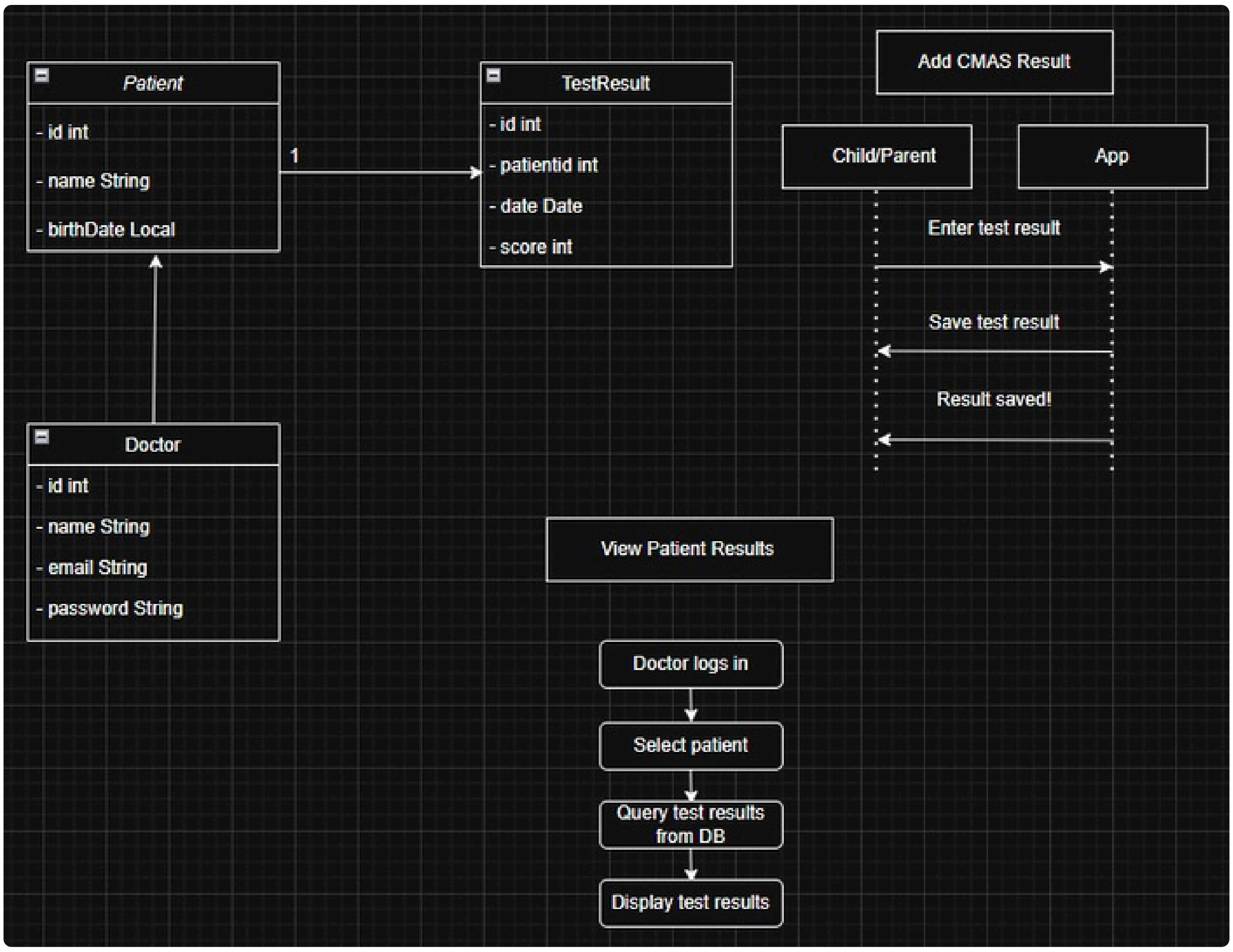
CLI-Based Interaction
Allow patients to interact via command line to enter or view their data.

USE CASE TABLE

Actor	Use Case	Description
Patient	Add new patient	Register their name and birth date
Patient	Add CMAS test result	Record a test score on a specific date
Patient	View test results	See all scores entered for them
Doctor	Log in	Access the system securely
Doctor	View patients & results	Review patient progress by ID

DESIGN

1. **System Design Overview :** UML, Sequence and Flow



This diagram summarizes the overall system design, including class relationships (UML), user interaction sequence, and the main data flow of the application. It shows how patients and doctors interact with the system to add and view CMAS test results.

2. Database Schema

The following database schema defines the structure of the system's local SQLite database, including tables for patients, test results, and doctors.

- `Patient(id, name, birth_date)`
- `TestResult(id, patient_id, date, score)`
- `Doctor(id, name, email, password)`

This schema was implemented in SQLite and used in the application via JDBC.

IMPLEMENTATION

The application is implemented as a modular Java program with a command-line interface (CLI). The codebase is organized into logical components as follows:

Entry point with menus
`Main.java`

Doctor login and functions
`DoctorMenu.java`

Patient-related tasks
`ViewTestResults.java`,
`AddTestResult.java`,
`AddPatient.java`

Login and database setup helpers
`InsertDoctor.java`,
`DoctorLogin.java`

All input is handled via Scanner, and data is stored persistently using SQLite. Error handling is used to catch SQL issues, and patient IDs are checked before adding results. The project is kept modular to allow easy expansion (e.g., adding biomarker tracking, visual graphs).

TESTING

We used manual testing to validate all use cases. Each function was tested by inputting valid and invalid data and comparing output to expectations.

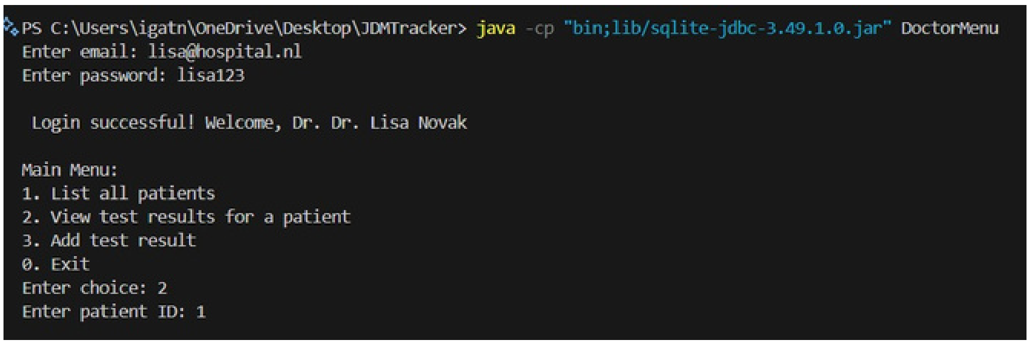
EXAMPLE TEST CASES

Test Case	Scenario	Input	Expected Output	Status
TC01	Add patient	Name + DOB	Patient successfully added	Pass
TC02	Add test result	Patient ID, Score	Error: no result found	Fail
TC03	Add test result	Patient ID, Date, Score	Result saved	Pass
TC04	View test results	Patient ID	List of scores shown	Pass
TC05	Doctor login	Valid email / password	Login success	Pass
TC06	Doctor login (invalid)	Wrong email / password	Login failed message	Pass

Test data was checked by opening the database directly via DB Browser and confirming inserts.
(Some edge cases failed, which helped us improve input validation.)

CLI EXECUTION EXAMPLE

The following screenshots demonstrate the working of the system. The doctor logs in using valid credentials and retrieves the CMAS score of a selected patient from the local database using the command-line interface.

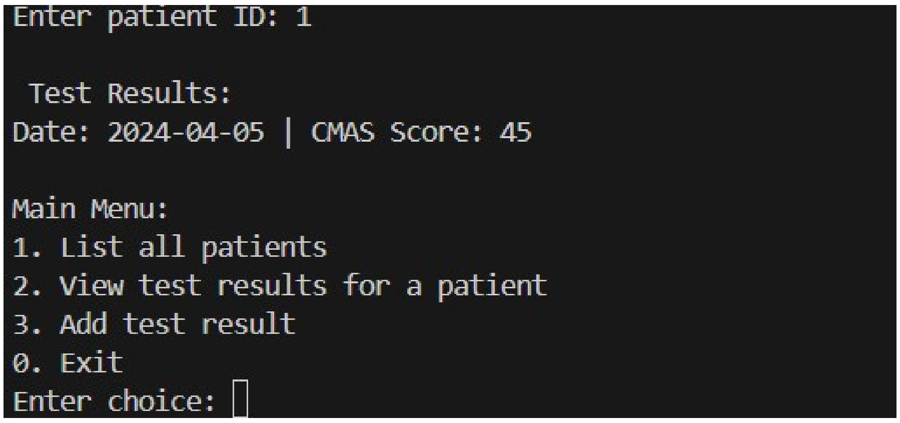


```
PS C:\Users\igatn\OneDrive\Desktop\JDMTracker> java -cp "bin;lib/sqlite-jdbc-3.49.1.0.jar" DoctorMenu
Enter email: lisa@hospital.nl
Enter password: lisa123

 Login successful! Welcome, Dr. Dr. Lisa Novak

Main Menu:
1. List all patients
2. View test results for a patient
3. Add test result
0. Exit
Enter choice: 2
Enter patient ID: 1
```

Successful doctor login using valid credentials.



```
Enter patient ID: 1

 Test Results:
Date: 2024-04-05 | CMAS Score: 45

Main Menu:
1. List all patients
2. View test results for a patient
3. Add test result
0. Exit
Enter choice:
```

CMAS test results displayed for a selected patient

RESULTS

Throughout the development process, we created a working command-line application that addresses the core functional requirements of the project. The system enables doctors and patients to interact with CMAS test data in a simple and structured way.

Key outcomes:

- Doctors can securely log in and monitor patient scores

- Patient Input and Review: Patients can enter and review their CMAS results

- Local database integration: All data is stored and retrieved from a local SQLite database

- Expandable Code Structure: Code is modular, readable, and ready for future improvements

The system supports all functional requirements, is easy to use via terminal, and can be packaged or distributed with little setup. Optional extensions like biomarker tracking and graphical output are possible in future iterations.

CONCLUSION

This project allowed us to apply software engineering principles to a real-world healthcare challenge. We developed a functional CLI-based Java application that supports CMAS result tracking for children with JDM. The system improves accessibility for patients and provides doctors with structured insights into patient progress.
Throughout the process, we practiced requirement analysis, UML design, modular coding, database integration, and testing. While the current version meets the basic goals, it can be further expanded with features like graphical dashboards, biomarker data, and a more user-friendly interface.

Most importantly, this project showed us how impactful digital solutions can be in healthcare. We realized that with the right software tools, and in the future, even AI-driven systems, critical tasks like patient monitoring and data management can become more efficient, less stressful, and more accessible to everyone involved. This experience not only strengthened our technical skills, but also gave us valuable insight into how software can improve lives and support medical professionals in real and meaningful ways.

APPENDIX

A) Project Folder Structure

```
JDMTracker
├──── src/                   Java source code
├──── bin/                   Compiled .class files
├──── database/              SQLite DB file (cmas_data.db)
├──── lib/                   JDBC driver
├──── README.md              GitHub README
├──── Report_Project.pdf     This report
```

B) Sample CLI Output

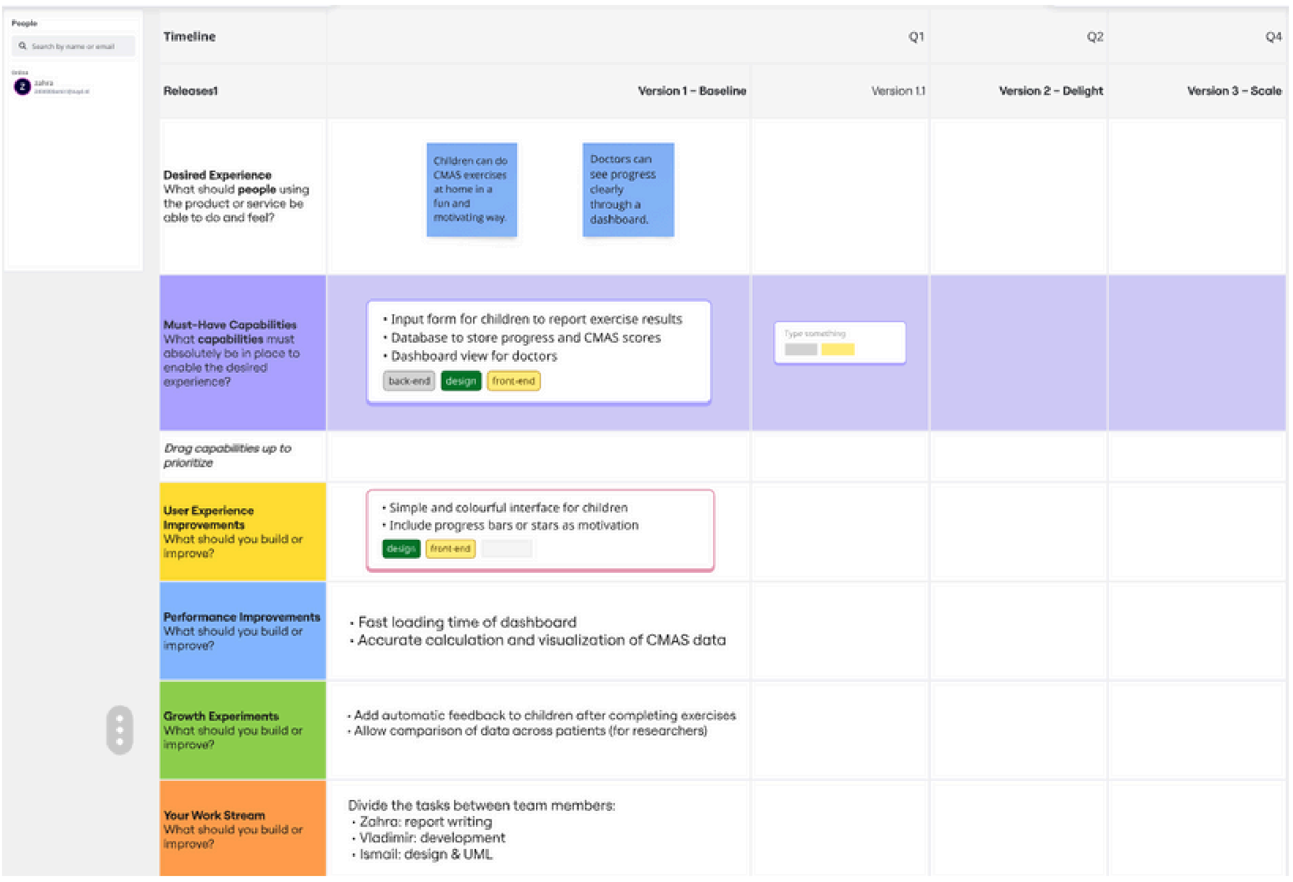
```
Main Menu:
1. I am a Doctor
2. I am a Patient
0. Exit

Doctor Login:
Email: lisa@hospital.nl
Password: secure123
→ Login successful

Patient Menu:
1. Add new patient
2. Add CMAS result
3. View results
```

C) Project Roadmap Table

This table was created using the Product Roadmap template in Miro and was shared in forum post . It outlines user experience goals, required capabilities, planned improvements, and task division among team members.



Timeline		Q1	Q2	Q4
Releases1	**Version 1 – Baseline**	Version 1.1	**Version 2 – Delight**	**Version 3 – Scale**
Desired Experience What should **people** using the product or service be able to do and feel?	Children can do CMAS exercises at home in a fun and motivating way. / Doctors can see progress clearly through a dashboard.			
Must-Have Capabilities What **capabilities** must absolutely be in place to enable the desired experience?	• Input form for children to report exercise results • Database to store progress and CMAS scores • Dashboard view for doctors back-end design front-end	Type something		
Drag capabilities up to prioritize				
User Experience Improvements What should you build or improve?	• Simple and colourful interface for children • Include progress bars or stars as motivation design front-end			
Performance Improvements What should you build or improve?	• Fast loading time of dashboard • Accurate calculation and visualization of CMAS data			
Growth Experiments What should you build or improve?	• Add automatic feedback to children after completing exercises • Allow comparison of data across patients (for researchers)			
Your Work Stream What should you build or improve?	Divide the tasks between team members: • Zahra: report writing • Vladimir: development • Ismail: design & UML			