

ITMB  
COC252  
F011321

**EHR System Modernization  
in the Republic of Moldova**

by

Calin Corcimaru

Supervisor: Dr. Georgina Cosma

Department of Computer Science

Loughborough University

May 2025

# **Abstract**

Abstract to be added

# Acknowledgements

Acknowledgements to be added

# List of Figures

2.1	Chosen stakeholders in the stakeholder influence-interest grid . . . . .	6
2.2	Updated stakeholders in the stakeholder influence-interest grid . . . . .	7
3.1	Software Development Life Cycle . . . . .	10
3.2	Waterfall model . . . . .	11
3.3	Scrum framework . . . . .	12
3.4	Stakeholder Influence/Interest matrix . . . . .	14
3.5	Medvalet screenshots . . . . .	25
3.6	Andaman7 screenshots . . . . .	27
3.7	Fasten Health screenshots . . . . .	29
4.1	UML Use Case Diagram . . . . .	31
4.2	UML Sequence Diagram — Upload Record Use Case . . . . .	35
4.3	UML Sequence Diagram — Share Records Use Case . . . . .	36
4.4	UML Activity Diagram - Upload Record Use Case . . . . .	37
4.5	UML Activity Diagram - Share Records Use Case . . . . .	38
4.6	Entity Relationship Diagram . . . . .	39
4.7	Desktop and Mobile version of the Dashboard screen . . . . .	40
4.8	Desktop and Mobile version of the Medical History screen . . . . .	41
4.9	Desktop and Mobile version of the Lab Test Results screen . . . . .	41
4.10	Proposed System Architecture . . . . .	42
4.11	Interaction of Frontend Components . . . . .	42
4.12	Interaction of Backend Components . . . . .	44
5.1	PrimeVue DatePicker component . . . . .	51
5.2	Vue Vaccine Card component . . . . .	55
5.3	Updated Entity Relationship Diagram Sprint # 2 . . . . .	56
5.4	Uploaded Vaccine Certificate . . . . .	57

5.5	Sequence Diagrams for File Upload and File View . . . . .	58
5.6	Desktop and Mobile version of the Allergies page . . . . .	61
5.7	Desktop and Mobile version of the Allergies page 2nd version . . . . .	64
5.8	Desktop and Mobile version of the Vitals page . . . . .	65
5.9	Desktop and Mobile version of the Vitals page, with Graph . . . . .	66
5.10	Desktop and Mobile version of the Dashboard page . . . . .	68
5.11	Updated Entity Relationship Diagram for Sprint #3 . . . . .	70
5.12	Desktop and Mobile version of the History page . . . . .	74
5.13	Desktop and Mobile version of the PDF viewer . . . . .	75
5.14	Filter Menu for Health Records . . . . .	76
5.15	Updated Entity Relationship Diagram Sprint #4 . . . . .	77
5.16	Updated Vitals View based on stakeholder feedback . . . . .	78
5.17	Updated Dashboard View based on stakeholder feedback . . . . .	79
5.18	UML Sequence Diagram — Extracting Lab Results . . . . .	81
5.19	Desktop and Mobile version of the new Add Document dialog . . . . .	82
5.20	Desktop dialog validation of extracted lab results . . . . .	84
5.21	Updated Entity Relationship Diagram Sprint #5 . . . . .	88
5.22	Desktop version of the new Lab View page . . . . .	90
5.23	Mobile version of the new Lab View page . . . . .	91
5.24	UML Sequence Diagram — Creating share link . . . . .	94
5.25	UML Sequence Diagram — Accessing shared link . . . . .	95
5.26	1st step of share link creation . . . . .	96
5.27	2nd step of share link creation . . . . .	97
5.28	3rd step of share link creation . . . . .	97
5.29	PIN prompt dialog . . . . .	98
5.30	Expired link dialog . . . . .	98
5.31	Share page — Desktop version . . . . .	99
5.32	Updated Entity Relationship Diagram Sprint #6 . . . . .	101
6.1	Test Scenario #1 . . . . .	104
6.2	Test Scenario #2 . . . . .	105
6.3	Test Scenario #3 . . . . .	106
6.4	PHR System Site Map . . . . .	107
6.5	Postman test — share token not found . . . . .	109
6.6	Postman test — share token expired . . . . .	110
6.7	Postman test — share token valid . . . . .	111
6.8	Synevo extraction results . . . . .	113

6.9	Alfalab extraction results . . . . .	114
6.10	Medpark extraction results . . . . .	115
B.1	Mobile version of the Vaccines screen . . . . .	140
B.2	Desktop and Mobile version of the Dashboard screen . . . . .	141
B.3	Desktop and Mobile version of the Doctor View screen . . . . .	142
B.4	Desktop and Mobile version of the Lab Test screen . . . . .	143
B.5	Desktop and Mobile version of the Medical History screen . . . . .	144
B.6	Desktop and Mobile version of the New Document screen . . . . .	145
B.7	Desktop and Mobile version of the Share Doctor screen . . . . .	146
B.8	Desktop and Mobile version of the Allergies screen . . . . .	147

# Contents

<b>Abstract</b>	ii
<b>Acknowledgements</b>	iii
<b>1 Introduction</b>	2
1.1 Background . . . . .	2
1.2 Problem Statement . . . . .	3
1.3 The Client . . . . .	3
1.4 Project Objectives . . . . .	4
<b>2 Research and Requirements</b>	5
2.1 Stakeholder Analysis . . . . .	5
2.1.1 Current situation analysis . . . . .	6
2.2 Requirements . . . . .	8
<b>3 Literature Review</b>	9
3.1 Software development methodologies . . . . .	9
3.1.1 Software Development Life Cycle . . . . .	9
3.1.2 SDLC Models . . . . .	11
3.1.3 A hybrid approach . . . . .	12
3.2 Requirements gathering . . . . .	13
3.2.1 Requirement types . . . . .	13
3.2.2 Stakeholders . . . . .	14
3.2.3 Requirement gathering techniques . . . . .	15
3.3 Tech stack . . . . .	16
3.3.1 Database . . . . .	16
3.3.2 Backend framework . . . . .	16
3.3.3 Frontend framework . . . . .	17

3.4	Large Language Models (LLMs) . . . . .	18
3.4.1	Multimodal LLMs . . . . .	18
3.4.2	LLMs in healthcare . . . . .	20
3.4.3	Prompt Engineering . . . . .	20
3.4.4	Challenges and concerns of using LLMs . . . . .	22
3.5	PHR Systems . . . . .	23
3.5.1	Existing Solutions . . . . .	24
<b>4</b>	<b>Project Planning and Design</b>	<b>30</b>
4.1	UML Diagrams . . . . .	30
4.1.1	Use Case Diagram . . . . .	31
4.1.2	Use Case Specifications . . . . .	32
4.1.3	Sequence Diagrams . . . . .	35
4.1.4	Activity Diagrams . . . . .	37
4.2	Database Design . . . . .	39
4.3	Wireframes . . . . .	40
4.4	Project Tech Stack . . . . .	42
4.4.1	Frontend . . . . .	42
4.4.2	Backend . . . . .	44
4.4.3	Database . . . . .	44
4.5	Project Management . . . . .	45
4.5.1	Methodology and Tools . . . . .	45
4.5.2	Sprint planning . . . . .	45
<b>5</b>	<b>Development</b>	<b>47</b>
5.1	Sprint #1 . . . . .	47
5.1.1	SQLModel Schemas & Database creation . . . . .	47
5.1.2	Authentication & APIs . . . . .	48
5.1.3	Frontend Setup . . . . .	51
5.1.4	Challenges . . . . .	52
5.1.5	Requirements completed . . . . .	53
5.2	Sprint #2 . . . . .	53
5.2.1	Vaccines, allergies and medication management . . . . .	53
5.2.2	File Uploads feature . . . . .	55
5.2.3	Challenges . . . . .	59
5.2.4	Requirements completed . . . . .	60
5.3	Sprint #3 . . . . .	62
5.3.1	User interface rework . . . . .	62

5.3.2	Filtering and sorting . . . . .	62
5.3.3	Vital signs monitoring . . . . .	63
5.3.4	Dashboard implementation . . . . .	67
5.3.5	Backend changes . . . . .	69
5.3.6	Challenges . . . . .	71
5.3.7	Requirements completed . . . . .	71
5.4	Sprint #4 . . . . .	72
5.4.1	Health Records . . . . .	72
5.4.2	Backend changes . . . . .	76
5.4.3	Work on stakeholder feedback . . . . .	78
5.4.4	Challenges . . . . .	79
5.4.5	Requirements completed . . . . .	79
5.5	Sprint #5 . . . . .	80
5.5.1	Lab Tests Extraction . . . . .	80
5.5.2	Results visualisation . . . . .	89
5.5.3	Challenges encountered . . . . .	89
5.5.4	Requirements completed . . . . .	89
5.6	Sprint #6 . . . . .	92
5.6.1	Share links . . . . .	92
5.6.2	User interface implementation . . . . .	96
5.6.3	Backend . . . . .	99
5.6.4	Challenges encountered . . . . .	100
5.6.5	Requirements completed . . . . .	100
<b>6</b>	<b>Evaluation</b>	<b>102</b>
6.1	Frontend Testing . . . . .	102
6.2	Backend Testing . . . . .	107
6.3	LLM Evaluation . . . . .	112
6.4	Stakeholders feedback . . . . .	116
<b>7</b>	<b>Conclusion and Future Work</b>	<b>118</b>
7.1	Areas of Improvement . . . . .	118
7.1.1	Security and data encryption . . . . .	118
7.1.2	Logging and monitoring . . . . .	119
7.1.3	Legal and compliance . . . . .	119
7.1.4	User experience and interface design . . . . .	120
7.2	Potential additions . . . . .	120
7.2.1	Mpass integration . . . . .	120

7.2.2	Expanding AI features . . . . .	120
7.2.3	Integration with lab providers . . . . .	121
7.2.4	Enhanced doctor-patient communication . . . . .	121
7.2.5	Integration with 3rd party providers . . . . .	121
7.3	Lessons Learned and Reflections . . . . .	122
7.3.1	Prioritisation and time management . . . . .	122
7.3.2	Importance of the team . . . . .	122
7.3.3	Agile & Documentation . . . . .	123
7.4	Final Thoughts . . . . .	124
	<b>References</b>	<b>125</b>
	<b>A Requirements</b>	<b>134</b>
	<b>B Wireframes</b>	<b>140</b>
	<b>C Full feedback from stakeholders</b>	<b>148</b>

# **Chapter 1**

## **Introduction**

### **1.1 Background**

The Republic of Moldova is a small country in Eastern Europe that borders Romania and Ukraine, with a current population of 2.4 million people (National Bureau of Statistics of the Republic of Moldova, 2024). Since its independence in 1991, Moldova has faced a number of challenges, including political instability, corruption, and economic difficulties which have left Moldova as one of the poorest countries in Europe (BBC, 2024).

Despite these challenges, Moldova has made significant progress in its digital transformation efforts, with the government launching a number of initiatives to modernize its public services and improve the quality of life for its citizens (E-Governance Agency, n.d.[a]). An example is the Citizen's Government Portal (MCabinet), which allows citizens to access personal information such as ‘valid identity documents, social contributions and benefits, own properties, information about the family doctor and the health institution where the person is registered, tax payments and other information about the citizen-government relationship’ (E-Governance Agency, n.d.[b]).

To continue supporting the existing transformation initiatives, Moldova’s Cabinet of Ministers has recently approved the ‘Digital Transformation Strategy of the Republic of Moldova for 2023–2030’, which aims to transform the country into a digital society by 2030, with the ultimate goal of having ‘all public services available in a digitalized format’ (United Nations Development Programme, 2023).

## 1.2 Problem Statement

The healthcare sector in Moldova has also seen some transformations, with the introduction of a new electronic health record system (EHR) in 15 hospitals across the country in 2017, called ‘Sistemul informațional automatizat „Asistența Medicală Spitalicească” (SIA AMS)’ (Ciurcă, 2021). While the system has been successful in helping doctors access patient information more efficiently such as medical history, examinations, test results, and prescriptions, the system hasn’t been updated since its inception in 2017 and there are still challenges that need to be addressed in 2024.

The main challenge with the current system lies in the user experience (UX) — SIA AMS feels old and isn’t user-friendly, with a clunky interface that is difficult to navigate, not adhering to modern accessibility standards and only accessible via Internet Explorer or legacy version of Microsoft Edge, with no support for other browsers or devices (Ciurcă, 2021).

Another big challenge with the system is its lack of interoperability within public and private medical institutions due to a lack of a nationally-wide integrated system — each hospital and clinic have their own, siloed, information system that contains the patient information, with no communication being made between systems in different hospitals (Ciurcă, 2021).

Finally, due to the current economic situation in Moldova, the government has not allocated any funds to upgrade the current or develop new systems, and the hospitals and clinics that use the system do not have the resources to update it themselves.

## 1.3 The Client

The client, ‘Nicolae Testemiteanu’ State University of Medicine and Pharmacy in Moldova (USMF), is a public university in Chisinau, Moldova, that offers a range of medical programs, including medicine, dentistry and pharmacy (USMF, 2023). Many of the faculty at USMF are also practicing doctors at hospitals and clinics across Moldova, and have first-hand experience with the current IT systems used in both public and private medical institutions. The USMF faculty members that the student will be interacting with during the project are part of an innovation team that researches potential opportunities to improve the healthcare sector in Moldova through the use of technology. As such, the client has expressed a need for a prototype that can act as a proof of concept for a modern system that could either replace or augment the current system in Moldova.

## 1.4 Project Objectives

This project aims to initially conduct some research on the current situation of the IT systems used in the healthcare sector in Moldova by interviewing several stakeholders from various healthcare-related institutions. Afterwards, the project will conduct a literature review on the most appropriate technologies and methodologies for developing a modernized EHR system, and an analysis of existing EHR systems to identify their existing functionality. Finally, based on the information gathered, the project will focus on designing and developing a working prototype, based on the requirements gathered and the feasibility of the chosen solution for the Moldovan healthcare sector. The student's hope is that the solution can then be used as a proof of concept to secure funding for a full-scale implementation of the new system in Moldova by the relevant authorities, such as the Ministry of Health.

As such, the objectives of the project are as follows:

1. Identify 2 to 4 stakeholders from various perspectives, such as healthcare institutions in Moldova and patients, that can provide insights into the current IT systems used in the healthcare sector in Moldova.
2. Conduct interviews with the identified stakeholders to gather information on the current IT systems used in the healthcare sector in Moldova.
3. Carry out a literature review to research the most appropriate technologies (frontend, backend and database) and project management methodologies for developing a modernized EHR system.
4. Explore at least 2 existing EHR systems and identify their strengths and weaknesses.
5. Design and develop a working web or mobile app for an EHR system based on the requirements gathered and the feasibility of the chosen solution for the Moldovan healthcare sector.
6. Offer the client the prototype to be used as a proof of concept to secure funding for a full-scale implementation of the new system in Moldova.

## Chapter 2

# Research and Requirements

To gain a more complete understanding of the current situation in Moldova, the existing problems and possible needs of the people involved, it is important to start with an analysis to identify the possible key stakeholders for this project. As previously mentioned in the literature review, a diverse group of stakeholders is essential to ensure that the current situation is reviewed from multiple perspectives.

Afterwards, the next step is to utilise the chosen stakeholders to gather as much information as possible from various perspective to ensure that the project is aligned with the needs of both patients and healthcare professionals in Moldova and solves the existing problems.

### 2.1 Stakeholder Analysis

The student has identified the following possible stakeholders for the project:

- Doctors and other medical staff working in hospitals
- Department head in a hospital
- IT staff members in hospitals
- Staff members at CNAM (National Health Insurance Company)
- Staff members at the Ministry of Health
- Patients

These stakeholders have been identified so that they can provide a wider picture on the needs and requirements of the project, and to ensure that the project is aligned with the

expectations of workers within the healthcare industry in Moldova from multiple perspectives.

The stakeholders have also been placed in the stakeholder influence-interest grid (which will be discussed in section 3.2.2) to help the student understand the level of influence and interest that each stakeholder has in the project:

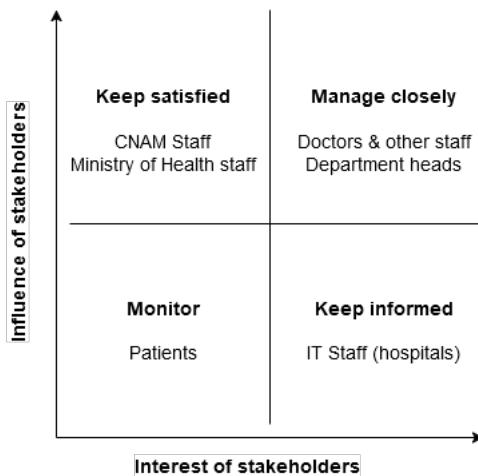


Figure 2.1: Chosen stakeholders in the stakeholder influence-interest grid

### 2.1.1 Current situation analysis

Following the analysis, the student has conducted several exploratory interviews with the chosen stakeholders to gather insights into current issues with the IT systems used in Moldova's healthcare sector. The student was able to reach out to every stakeholder, except for staff members at CNAM and the Ministry of Health.

After the conclusion of the interviews, three main issues and potential solutions have been identified:

1. Current EHR system is outdated and only accessible via Internet Explorer or legacy version of Microsoft Edge. A potential solution is to develop a new, modernized version of the existing system (retaining the core functionality) that is accessible via modern browsers, is more user friendly and has future upgrade capabilities.
2. Lack of interoperability between medical institutions due to a lack of a nationally-wide integrated system. A potential solution is to create a new system that acts as a patient history archive, where patients can upload their own medical records (such as lab tests, previous medical history, etc) and share them with any medical practitioner,

regardless of the institution they work at.

3. Lack of digitalization for some systems that still rely on paper-based records or very rudimentary data structures, such as the national transplant registry. A potential solution is digitalization of said system, as is in the case of the transplant registry, that can be accessed by any medical practitioner in Moldova.

Analysing the current issues and potential solutions, the student has determined that the solutions for issues #1 and #3 are too complex, as they require a complete overhaul and integration with existing systems. As such, the student has decided to focus on issue #2, as it is the most feasible and an MVP can be developed within the timeframe of the project.

Consequently, the stakeholder list has been updated to reflect the changed focus of the project:

- Doctors working in hospitals and clinics
- Staff members at the Ministry of Health
- Staff members at CNAM
- Patients that are using both public and private healthcare institutions
- Other medical staff members (nurses, pharmacists, etc)
- Staff members at CNPDCP (National Center for Personal Data Protection)

At the same time, the stakeholder influence-interest grid has been updated to reflect the changes in the project focus:

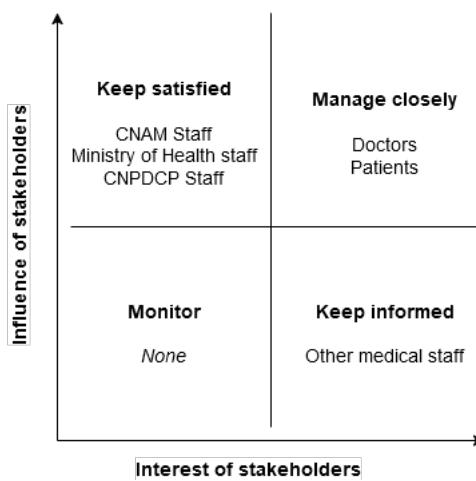


Figure 2.2: Updated stakeholders in the stakeholder influence-interest grid

## 2.2 Requirements

After the new stakeholders were identified, additional interviews were conducted to focus on the requirements for the chosen solution and enough information was gathered from the other stakeholders to identify the main requirements for the project. The student was unable to reach out to the staff members at CNAM, the Ministry of Health and CNDPC — instead legislation and regulations on their websites were reviewed (Compania Națională de Asigurări în Medicină, n.d.; Centrul Național pentru Protecția Datelor cu Caracter Personal al Republicii Moldova, n.d.; Ministerul Sănătății al Republicii Moldova, 2024).

All of the gathered requirements can be found in the appendix (section A), but the most important requirements have been summarized in the table below:

ID	Category	Requirement
1	Non-functional	The system must be accessible on all modern desktop and mobile-based browsers.
2	Non-functional	The system must store the data in a secure manner, ensuring that only the patient and the doctor can access the data.
3	Document upload	The system must allow patients to upload their own medical records in a variety of formats (PDF, DOC, etc.).
4	Personal cabinet	The system must display the patient's history in a chronological order in the form of a timeline.
5	Personal cabinet	The patient personal cabinet must provide an overview of the patient's history through 3 main sections: personal information, lab tests, and doctor consultations.
6	Personal Cabinet	The system must allow patients to add their own personal information, such as name, date of birth, or address.
7	Personal Cabinet	The system must allow patients to add their own allergies and vaccinations.
8	Shareable link	The system must allow the patient to generate a shareable link to provide access to their medical records.
9	Doctor view	When shared with the doctor, the system must allow the doctor to only view the patient's history, not edit it.
10	Patient medication	The system must allow patients to enter their current medication including details such as the name of the drug, dosage, frequency and start/end date.

Table 2.1: Summary of the Most Important Requirements

# Chapter 3

## Literature Review

This chapter will provide a review of the existing literature, which will be used guide the student in their planning and development efforts of the project.

As such, it will be covering the following areas:

- Software development methodologies
- Requirement and stakeholder management
- Tech stack (backend and frontend)
- Large Language Models (LLMs)
- PHR Systems

### 3.1 Software development methodologies

#### 3.1.1 Software Development Life Cycle

The Software Development Life Cycle (SDLC) is a process used to guide the development of software applications or systems (Ruparelia, 2010). The SDLC consists of multiple phases, each with its own set of activities and deliverables. 'Yas, Alazzawi, and Rahmatullah (2023) outline the phases of the SDLC as following:

1. **Requirement gathering and analysis phase** — the requirements are gathered saved in a document. Based on the requirements gathered, a development plan is created.

2. **Design phase** — requirements are written in a more technical manner and system designs are created.
3. **Implementation phase** — Actual development of the software occurs in this phase. Additionally, some smaller unit tests may be done during this phase.
4. **Testing phase** — may involve multiple types of testing, such as unit testing, integration testing, and system testing. ‘Luo (2001) describes the different types of tests:
  - Unit testing — testing individual units or components of the software.
  - Integration testing — performed on two or more units combined together, focusing on the interfaces between these components.
  - System testing — focuses on the ‘end-to-end quality of the entire system’, testing it as a whole.
5. **Maintenance phase** — involves the deployment and maintenance of the software. Additionally, this phase may include end-user acceptance testing, to ensure that it meets their needs (Luo, 2001).

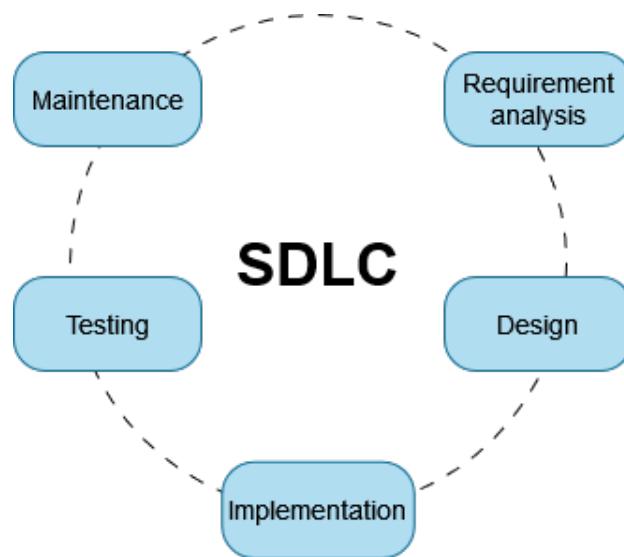


Figure 3.1: Software Development Life Cycle

### 3.1.2 SDLC Models

The literature describes several SDLC models that have been used in the development of software applications.<sup>4</sup> Ruparelia (2010) and Yas, Alazzawi, and Rahmatullah (2023) highlight the most common ones: Waterfall model, V model, Spiral model, Interative model, and Agile model.

#### Waterfall Model

The Waterfall Model is probably the most well-known SDLC model. It is a linear model, where the development process is divided into distinct, sequential phases (which can be seen in figure 3.2).

The Waterfall Model's strengths lie in its simplicty of use, ease of understanding and a clear, structured approach (Alshamrani and Bahattab, 2015). An additional strength that the authors note is its extensive documentation and planning, emphasizing quality and adherence to regulations.

On the other hand, one of Waterfall's main weaknesses is its lack of flexibility in regards to change (Mirza and Datta, 2019). Thus, this model is not suitable for projects where the requirements are not well understood or are likely to change. Additionally, the project deliverable is not avaialable until the end of the project, any changed or feedback cannot be done during its development (Mirza and Datta, 2019).

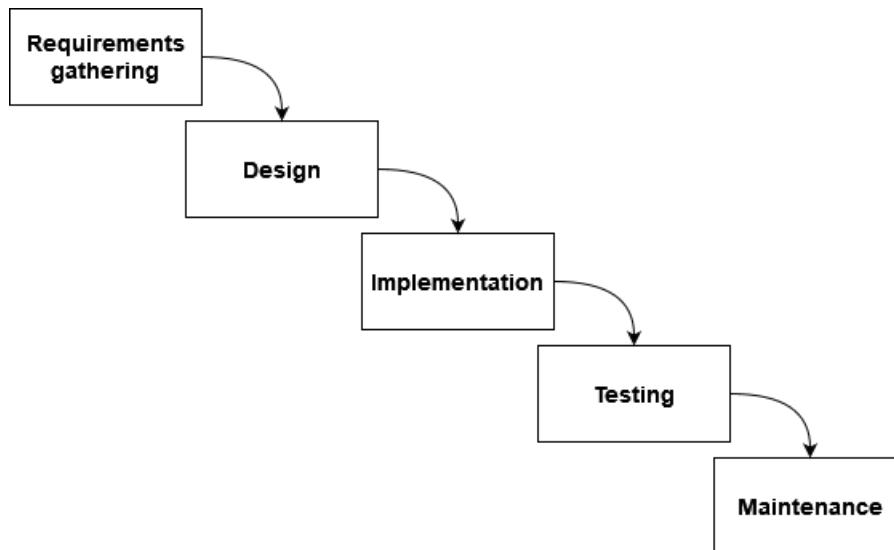


Figure 3.2: Waterfall model

## Agile Model

Another well-known model is Agile. It has multiple frameworks, with Scrum and Kanban being the most popular ones. Scrum is a framework where the project is divided into sprints, each lasting between 2–4 weeks, that aim on delivering value to the customer through incremental software features (Alqudah and Razali, 2018; Sunner, 2016). Kanban focuses on visualizing the project workflow by using a visual board with columns, cards and swimlanes. It uses column limits and a pull system to make the flow of work through the system more efficient (Sunner, 2016).

Agile has some drawbacks — its lack of documentation and formal planning, especially in the early stages of the project, may not be suitable for large scale projects (Sunner, 2016; Yas, Alazzawi, and Rahmatullah, 2023). Similarly, lack of knowledge on how to use the frameworks may be a barrier for some teams (Mirza and Datta, 2019; Yas, Alazzawi, and Rahmatullah, 2023).

Nowadays, the combined use of Scrum and Kanban is becoming quite popular, with many teams employing both frameworks in their projects, allowing them to adopt the appropriate practices and adapt them accordingly based on their needs (Alqudah and Razali, 2018).

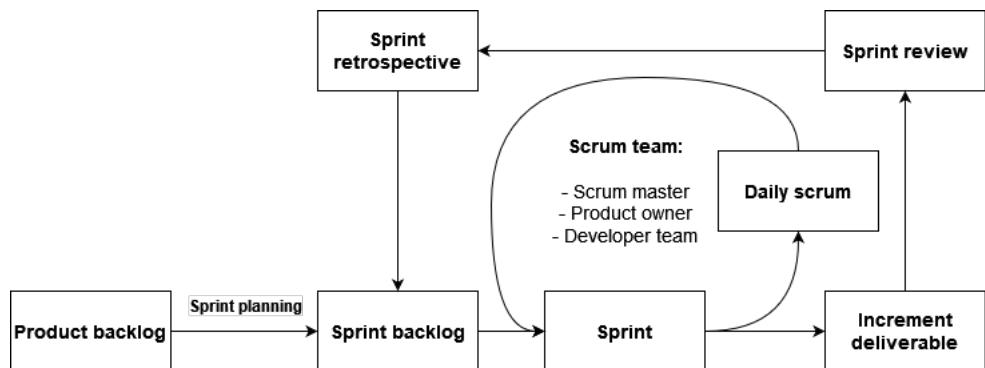


Figure 3.3: Scrum framework

### 3.1.3 A hybrid approach

A hybrid approach has also been emerging in software development projects. Various surveys report the most common combinations are Scrum, Iterative Development, Kanban, Waterfall and DevOps, with hybrid Waterfall and Scrum being the most popular one (Gemino, Reich, and Serrador, 2021; Prenner, Unger-Windeler, and Schneider, 2020). In this approach, the development part is done in an Agile way, with the rest of the project using Waterfall as a backbone (Prenner, Unger-Windeler, and Schneider, 2020).

Gemino, Reich, and Serrador (2021) note that projects using either Agile, traditional or hybrid approach show similar levels of success in terms of budget, time and quality. However, the authors have found that agile and hybrid approaches perform much better on customer satisfaction than the traditional ones.

## 3.2 Requirements gathering

Requirements gathering is the first step in any software development process. As described by Young (2002), a requirement is a ‘necessary attribute in a system...that identifies a capability, characteristic, or quality factor of a system in order for it to have value and utility to a user’. Multiple studies mention how proper requirement gathering plays a pivotal role in the project success, with many project failures attributed to poor requirements gathering (Hickey and A.M. Davis, 2003; S. Sharma and Pandey, 2013; Alan Davis et al., 2006a).

### 3.2.1 Requirement types

Requirements can be classified into 2 categories: functional and non-functional. Functional requirements describe the system’s behavior, while non-functional requirements describe the system’s quality attributes, such as performance, security, reliability, etc.’ ’(Laplante, 2019, p. 6).

When writing the requirements in a document, it is important to ensure clarity and conciseness to avoid ambiguity. Based on the recommendations of Laplante (2019, p. 112) and IEEE (2018), the following principles and practices can be followed:

- Write in a simple and consistent language.
- Avoid technical jargon, vague terms, and combining multiple requirements in a single statement.
- Ensure that requirements are necessary, appropriate, complete, feasible, and verifiable.
- Include attributes for each requirement, such as identification, owner, priority, risk, rationale, difficulty, and type (functional/non-functional).

Prioritizing requirements is another crucial task, especially in projects with numerous requirements. Some methods mentioned by Khan et al. (2015) include using a low to high priority, assigning a numerical value within a specific range or MoSCoW, which classifies requirements into four categories:

- Must have — must be implemented in the software before being released

- Should have — important but not necessary for the software to be released
- Could have — desirable but not necessary for the software to be released
- Won't have — requirements that are not included in the current release

### Requirements in Agile

In Agile projects, requirements are written in the form of User Stories, which are simple descriptions of a feature desired by the customer, using a specific format: ‘As a [user], I want to [action] so that [benefit]’ (Laplante, 2019, p. 191). Components of a User Story include a title, acceptance criteria, priority, story points and description. Epics are requirements that cannot be completed in a single sprint and can be broken down into user stories. Epics and User Stories are part of the Product and Sprint backlogs, which contain the requirements for the whole project and the current sprint, respectively.

#### 3.2.2 Stakeholders

Stakeholders are the individuals who have some interest in the success of the system or project, thus it is important to identify all possible stakeholders in the early stages of the project to avoid missing important requirements or constraints (Laplante, 2019, p. 34).

Stakeholder analysis can help understand their position within the project. One way of doing it is by using a stakeholder matrix, such as the Influence/Interest grid (see figure 3.4), which classifies stakeholders based on their influence and interest in the project (Morphy, 2020; Reddi, 2023).

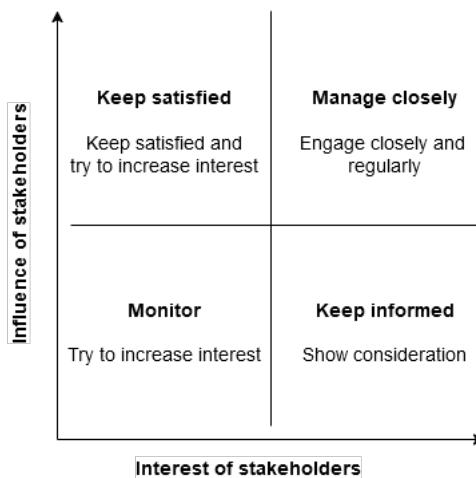


Figure 3.4: Stakeholder Influence/Interest matrix

### 3.2.3 Requirement gathering techniques

Multiple studies mention the most popular requirement gathering techniques are interviews, workshops, prototyping, modelling, brainstorming, storyboards and observing users (Hickey and A.M. Davis, 2003; Young, 2002; S. Sharma and Pandey, 2013; Tiwari, Rathore, and Gupta, 2012). In one of them, individuals with multiple years of experience in requirement gathering were interviewed, and the authors found that the most used techniques were collaborative meetings, interviews, ethnography and modelling (Hickey and A.M. Davis, 2003).

Multiple research papers recognise interviews as the most common technique for requirement gathering (Alan Davis et al., 2006b; Donati et al., 2017; Bano et al., 2019). Some studies have looked at best practices and common mistakes when conducting requirement gathering interviews. The recommended practices, based on Mohedas et al. (2022) and Loweth et al. (2021), and the common mistakes, from Donati et al. (2017) and Bano et al. (2019), are summarized in Table 3.1 below.

Common Mistakes	Recommended Practices
Wrong opening: failing to understand the context before discussing the problem.	State goals at the beginning and allow customer input at the end. <sup>11</sup>
Not leveraging ambiguity to reveal knowledge gaps.	Avoid ambiguity by asking clarifying questions. <sup>11</sup>
Lack of planning: unstructured sequence of questions.	Plan interviews with a structured sequence of questions. <sup>11</sup>
Failing to build rapport with the customer.	Building rapport through small talk or personal questions in the beginning. <sup>11</sup>
Implicit goals: failing to ask or clarify stakeholder goals.	Verify alignment and current interpretation with the customer's vision. <sup>11</sup>
Question omission: not asking about business processes or doing follow-up questions.	Be flexible by probing into relevant topics. <sup>11</sup>
Weak communication: too much technical jargon usage or not listening to the customer.	Use projective techniques like scenarios to encourage deep thinking. <sup>11</sup>
Poor question formulation: vague, technical, irrelevant, or too long.	Break down questions or responses into smaller parts and use story telling. <sup>11</sup>
Wrong closing sequence: skipping interview summaries or feedback.	Leaving time at the end for the stakeholder to offer any feedback or thoughts. <sup>11</sup>

Table 3.1: Comparison of Common Mistakes and Recommended Practices

### **3.3 Tech stack**

#### **3.3.1 Database**

There are several types of databases, such as: relational (SQL), NoSQL databases, graph databases or object-oriented databases (Oracle, 2020). The choice of database depends on the project or organisation requirements, such as the amount of data, the complexity of the data, the need for scalability, etc.

##### **Relational databases**

Relational databases store structured data in tables, linked through keys to create relationships between entries (Anderson and Nicholson, 2022). They use SQL (Structured Query Language) to create queries and schemas to help manage data efficiently. Anderson and Nicholson (2022) highlight that relational databases are used, thanks to their high data integrity, for industries like finance and healthcare. Relational databases are widely used, making it easier to find support and resources. However, the rigid schema limits adaptability to rapid data changes or usage of unstructured data. Examples include MySQL, PostgreSQL, Oracle, and Microsoft SQL Server.

##### **NoSQL databases**

NoSQL databases manage unstructured or semi-structured data without rigid schemas or relationships (Anderson and Nicholson, 2022). As the authors describe, NoSQL databases, such as key-value, document, column-family, and graph databases, excel in flexibility and scalability. NoSQL databases often prioritize performance over strict consistency, making them suitable for large datasets of unstructured data but less ideal for complex transactions. NoSQL systems lack SQL's mature standardization and support. Examples include MongoDB, Cassandra, Couchbase, and Redis.

#### **3.3.2 Backend framework**

Choosing the right backend framework is crucial — it is responsible for handling the business logic of the application, such as processing requests, interacting with the database, and returning responses to the client. A good framework also comes with the added benefit of included features such as security and authentication, database support, a big community and documentation.

Based on recent a recent survey by Vailshery (2024), the most popular backend frameworks are Express, Flask, Spring Boot, Django and Laravel. This data is also supported by the

Stack Overflow Developer Survey 2024, which lists the most popular programming languages as JavaScript (Express), Python (Flask, Django), Java (Spring Boot) and PHP (Laravel) (Stack Overflow, 2024).

Due to their high popularity among developers, the above frameworks will be compared. Using the information from Broadcom (2024), OpenJS Foundation (n.d.), Django (2024), and FastAPI (2024), table 3.2 will compare the frameworks based on the following criteria: programming language used, learning curve, community support, security features, database features, and project size suitability.

Framework	Django	FastAPI	Spring Boot	Express
Language	Python	Python	Java	JavaScript
Learning curve	Medium	Low	High	Low
Community Support	High	High	High	High
Security features	High	High	High	Medium
Database features	Medium	Low	High	Medium
Project size suitability	Small to medium	Small to medium	Medium to large	Small to medium

Table 3.2: Comparison of backend frameworks

### 3.3.3 Frontend framework

Similar to the backend frameworks, choosing a suitable frontend framework is equally important — it is responsible for the user interface of the application, such as displaying data, handling user interactions, and making requests to the backend.

Based on the same survey by Vailshery (2024), the most popular frontend frameworks are React, Angular, Vue.js, and Svelte. This data is also supported by the Stack Overflow Developer Survey 2024, where those frameworks rank among the highest for desirability and admirability among developers (Stack Overflow, 2024).

Table 3.3 will use the information from Meta Platforms (n.d.), Google (2024b), You (2024), and Svelte (n.d.) to compare the frameworks based on the following criteria: learning curve, community and documentation, ecosystem and tooling support, performance, state management, and project size suitability.

Framework	React	Angular	Vue.js	Svelte
<b>Learning curve</b>	Low	High	Medium	Low
<b>Community and documentation</b>	High	High	High	Medium
<b>Ecosystem and tooling support</b>	High	High	Medium	Low
<b>Performance</b>	High	Medium	Medium	High
<b>State management</b>	High	High	Medium	Low
<b>Project size suitability</b>	Small to large	Medium to large	Small to medium	Small to medium

Table 3.3: Comparison of frontend frameworks

## 3.4 Large Language Models (LLMs)

Large language models (LLMs) are artificial intelligence systems that are used for natural language processing (NLP) tasks such as text generation, translation, summarization and question answering (Naveed et al., 2023; Nazi and Peng, 2024). Additionally, LLMs have been found to have emergent capabilities, like reasoning, planning, decision-making and in-contenxt learning (Naveed et al., 2023). These extraordinary capabilities are achieved through extensive training on large corpus of text data, high parameter count (in the billions) and usage of techniques such as fine-tuning or prompt engineering to improve their performance (Naveed et al., 2023; Nazi and Peng, 2024).

LLMs are built on the transformer architecture, which allows them to understand text by learning and remembering the relationships between words (Vaswani et al., 2017). These models are first pre-trained on large amounts of unlabeled data using, allowing them to excel in a wide variety of tasks (Zhou et al., 2023; Naveed et al., 2023). These pre-trained models, known as foundation models such as the GPT or Llama families, can then be fine-tuned for specific tasks, improving their performance and accuracy even further (OpenAI et al., 2024; Dubey et al., 2024; Naveed et al., 2023).

### 3.4.1 Multimodal LLMs

One advancement in the field of LLMs has been the addition of multimodal abilities, allowing them to process, understand and generate text and images, audio or videos (Yin et al., 2024; D. Zhang et al., 2024). These new multimodal LLMs (MLLMs) utilise existing reasoning capabilities of LLMs, which are connected to an encoder that can processes images, audio or videos and a generator that helps with generating multimodal outputs (Yin et al., 2024). This integration of new modalities allows MLLMs to become versatile tools, expanding their possible use cases and bridging the gap between human and machine interaction (Nazi and Peng, 2024).

## API model providers

Running and hosting LLMs locally can be a challenge, considering their big model sizes and high computational requirements. As such, many platforms offer APIs that allow users to access LLMs through the cloud. A list of some free API providers, the models offered and their rate limits has been compiled by Cheah and Markham (2024) and some are listed in the table below.

Provider	Model name(s)	Free tier limits
Groq	Llama 3.2 11B Vision	7,000 requests/day, 7,000 tokens/minute
	Llama 3.2 90B Vision	3,500 requests/day, 7,000 tokens/minute
OpenRouter	Llama 3.2 11B Vision Instruct	
	Llama 3.2 90B Vision Instruct	20 requests/minute, 200 requests/day
	Gemini 2.0 Flash Experimental	
Google AI Studio	Gemini 2.0 Flash	4,000,000 tokens/minute, 10 requests/minute
	Gemini 1.5 Flash	1,000,000 tokens/minute, 1,500 requests/day, 15 requests/minute
	Gemini 1.5 Pro	32,000 tokens/minute, 50 requests/day, 2 requests/minute
GitHub Models	OpenAI GPT-4o	Rate limits dependent on Copilot subscription tier
	OpenAI GPT-4o mini	
Cloudflare Workers AI	Llama 3.2 11B Vision Instruct	10,000 tokens/day
glhf.chat	Any model on Hugging Face that fits on an A100 node ( 640GB VRAM)	480 requests/8 hours

Table 3.4: API providers for LLMs

### 3.4.2 LLMs in healthcare

One application of MLLMs is in healthcare, where the growing volume and complexity of data creates the need for more advanced tools to process and analyze it. LLMs and MLLMs have found use in various healthcare applications, either by using existing models or by developing new, specialized medical models such as Med-PaLm2, BioMistral or Med-Gemini (Labrak et al., 2024; Google, 2024a; Singhal et al., 2023). Some of these applications include:

- **Improving medical diagnosis:** By combining patient records, existing symptoms, and medical history, LLMs can use their reasoning capabilities and memory to assist in diagnosing or preventing health conditions (Nazi and Peng, 2024; Niu et al., 2024; Xiao et al., 2024).
- **Medical Imaging and Multimodal Capabilities:** In diagnostic imaging, multimodal models can assess both text and images (such as X-rays and MRIs) to offer comprehensive analysis. Clinicians can input medical images and contextual information, making MLLMs valuable assistants in the real-time diagnostic processes (Niu et al., 2024).
- **Virtual Health Assistants:** LLMs can also be deployed as virtual assistants, helping patients with personalised care and general health inquiries (Nazi and Peng, 2024; Niu et al., 2024). Patients in areas with limited healthcare access can benefit from these assistants, which also supports healthcare providers by lightening their workloads.
- **Administrative Support:** LLMs can assist in generating Electronic Health Records (EHRs), allowing healthcare providers to focus more on patient interaction (Xiao et al., 2024). Additionally, they can also help translate complex medical terms into more simple language, assist in administrative tasks, and more.

### 3.4.3 Prompt Engineering

The success of LLMs depends not only on the model itself — but also on how it's effectively used by the users, using techniques like prompt engineering, which involves the constant designing and refining of prompts to guide the output of LLMs (Meskó, 2023; Amatriain, 2024). Prompts represent instructions given to the model to guide its output, such as providing context, examples, or constraints to the model (Giray, 2023; Schulhoff et al., 2024; Amatriain, 2024).

There are multiple techniques for prompt engineering, ranging from simple to more advanced. The tables and subsections below outlines some of the most common techniques.

### **Zero-Shot Prompting**

Zero-shot prompting are techniques where the LLM is given a prompt without any examples, allowing it to generate an output based on the prompt alone (Schulhoff et al., 2024).

Technique	Description	Source
Role prompting	Assigning a specific role to the LLM in the prompt. The authors note that generally it provides mixed results but may be useful in certain settings.	Zheng, Pei, and Jurgens (2023)
Style prompting	Specifying the desired style or tone in the prompt.	Lu et al. (2023)
Emotion prompting	Incorporating phrases of psychological relevance to humans in the prompt.	Cheng Li et al. (2023)
Re-reading	Adding the phrase ‘Read the question again’ to the prompt in addition to repeating the question.	Xu et al. (2023)
Self-Ask	Prompting the LLM to decide if it needs to ask any follow-up questions for a given prompt.	Press et al. (2022)

Table 3.5: Zero-Shot Prompt Techniques

### **Few-Shot Prompting**

Few-shot prompting are techniques where the LLM learns how to complete a task based on a few examples given in the input prompt (Schulhoff et al., 2024).

Technique	Description	Source
Self-Generated	Using the LLM to automatically generate examples	Kim et al.
In-Context Learning	when training/example data is not available.	(2022)
Prompt Mining	Scanning the training data to discover common formats that can be used as prompt templates.	Jiang et al. (2020)

Table 3.6: Few-Shot Prompt Techniques

### **Multimodal and Multilingual Prompting**

Multimodal and multilingual prompting are techniques which aim to improve an LLM’s performance by leveraging multiple modalities or languages in the prompt (Schulhoff et al.,

2024).

Technique	Description	Source
Translate-first prompting	Translating the input prompt into English to leverage LLMs strengths in dealing with English inputs, compared to non-English inputs.	Etxaniz et al. (2023)
English prompting	Writing the prompt in English may usually be more effective than using the task language for multilingual tasks. The authors argue it may because of the predominance of the English language in the pre-training data.	Ahuja et al. (2023)
JSON/XML output formatting	Asking the LLM to format the response in a JSON or XML format and providing the expected schema has been found to improve the accuracy of LLM outputs	Hada et al. (2023)
Image-as-Text	Generating or writing a textual description of an image that can then be included in a text-based prompt.	Hakimov and Schlangen (2023)

Table 3.7: Multimodal and Multilingual Techniques

### 3.4.4 Challenges and concerns of using LLMs

While LLMs bring many benefits when applied to the healthcare domain, it is important to note that their use does come with several challenges:

- **Data Privacy and Compliance:** Patient data is highly sensitive, thus ensuring compliance with standards is essential, requiring data anonymization and secure handling practices to ensure patient data safety.(Nazi and Peng, 2024; Harrer, 2023; Xiao et al., 2024).
- **Transparency and Explainability:** LLMs are often described as ‘black boxes’, making it difficult to explain their decision-making processes. In healthcare, transparency is crucial — lack of it poses risks and raises ethical concerns about relying on such systems in high-stakes scenarios (Nazi and Peng, 2024; Harrer, 2023; Xiao et al., 2024)..
- **Bias and Fairness:** LLMs trained on vast datasets can inherit biases in the data, leading to skewed or unfair outcomes (Harrer, 2023).

- **Hallucinations:** LLMs sometimes generate false or fabricated outputs, also known as ‘hallucinations’. In healthcare, this poses significant risks, as incorrect or misleading information could jeopardize patient safety and trust in the technology (Xiao et al., 2024; Nazi and Peng, 2024).
- **Accountability:** Responsibility must be clearly communicated and understood by all parties involved in the development and use of the model (Harrer, 2023). The author recommends the usage of clear guidelines, policies and code of conducts to ensure that all parties are aware of their obligations.
- **High Costs and Infrastructure Needs:** Training and operating LLMs requires extensive computational resources, which can be a limiting factor for healthcare institutions (Xiao et al., 2024).

### 3.5 PHR Systems

A Personal Health Record (PHR) is an electronic resource used by patients to manage their own health information (Hosseini et al., 2023; Lee et al., 2021). PHRs are different from Electronic Health Records (EHRs) and Electronic Medical Records (EMRs) which are inter-organisational or internal systems to organise patient health records (Heart, Ben-Assuli, and Shabtai, 2017; Lee et al., 2021). Three different types of PHRs are described by Hosseini et al. (2023): stand-alone, which require manual entry to update the records; institution-specific, which are connected to a specific healthcare institution; and integrated, which can connect to multiple healthcare systems to aggregate data from multiple sources.

Usage of PHRs can bring many benefits to patients, such as: empowering patients to manage their health, improving patient outcomes, decreasing the cost of healthcare and improving the taking of medication (Hosseini et al., 2023).

PHRs contain highly sensitive health information, so it is important to ensure that the data is secure and private. Based on a survey of health information management and medical informatics experts, Hosseini et al. (2023) identified 7 dimensions that need to be addressed when developing a PHR system:

1. Confidentiality
2. Availability
3. Integrity
4. Authentication

5. Authorization
6. Non-repudiation
7. Access rights

The authors recommend mechanisms to ensure adherence to the above-mentioned dimensions, such as encrypting the data in the database, using backups or defining user access to data and access rights.

### 3.5.1 Existing Solutions

PHR systems have been implemented nation-wide in many developed countries, such as the NHS App in the UK (Lee et al., 2021). Additionally, there are many private solutions that offer similar features to the one proposed in this project. The next sections will provide a brief overview of 3 existing systems: Medvalet, Andaman<sup>7</sup> and Fasten Health.

#### Medvalet

A mobile app developed in Romania that allows patients to upload their medical history as PDFs or scanned documents (Medvalet, 2024). See Table 3.8 for a summary of its features and limitations and figure 3.5 for a screenshot of the app.

Key Features/Benefits	Limitations/Drawbacks
<ul style="list-style-type: none"> <li>• Upload medical history as PDFs or scanned documents.</li> <li>• Categorize documents by type (e.g., prescriptions, lab results).</li> <li>• Graphically track vitals like blood pressure and weight over time.</li> <li>• Patients can input personal details such as name, age, and weight.</li> <li>• Doctors can access patient history directly via the app.</li> </ul>	<ul style="list-style-type: none"> <li>• Requires doctors to create accounts, which may deter use.</li> <li>• Doctors can access patient history without explicit consent, raising privacy concerns.</li> <li>• App acts as document storage, which can be cumbersome to access for lengthy histories.</li> <li>• Lacks data extraction or summarization features from uploaded documents.</li> <li>• Only available as a mobile app, limiting accessibility for desktop-only users.</li> </ul>

Table 3.8: Medvalet Features and Limitations

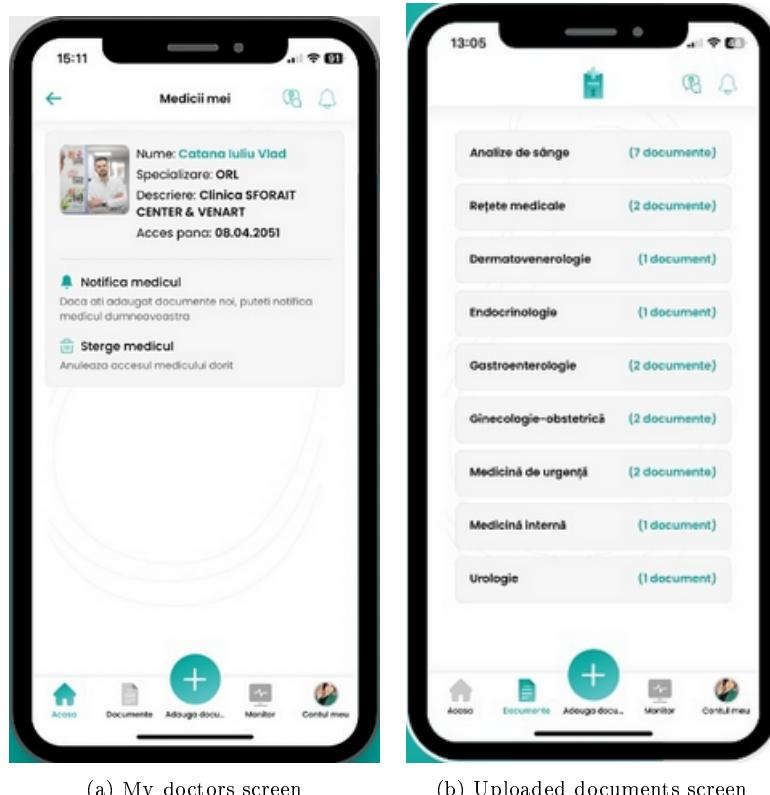


Figure 3.5: Medvalet screenshots

### Andaman7

A mobile app developed by a Belgian-American eHealth company with the goal to improve doctor-patient communication, compliant with GDPR and HIPAA (A7 Software, 2022). See Table 3.9 for a summary of its features and limitations and figure 3.6 for a screenshot of the app.

Key Features/Benefits	Limitations/Drawbacks
<ul style="list-style-type: none"> <li>• Offers sections for personal information, medical history, allergies, vaccinations, medications, etc.</li> <li>• Automatically collects health data from over 300 hospitals and clinics in the US and Europe.</li> <li>• Supports input from diverse sources like hospitals, labs, smart devices or even manual input.</li> <li>• Stores data locally on patients' devices, ensuring privacy.</li> <li>• Data sharing with QR codes and revokable access.</li> <li>• AI tools for summarization, translation, and simplifying medical jargon.</li> </ul>	<ul style="list-style-type: none"> <li>• Requires patients and doctors to both create accounts.</li> <li>• Does not extract data or values from uploaded documents like lab results.</li> <li>• Limited to mobile platforms, which may limit usability for desktop-only users.</li> </ul>

Table 3.9: Andaman7 Features and Limitations

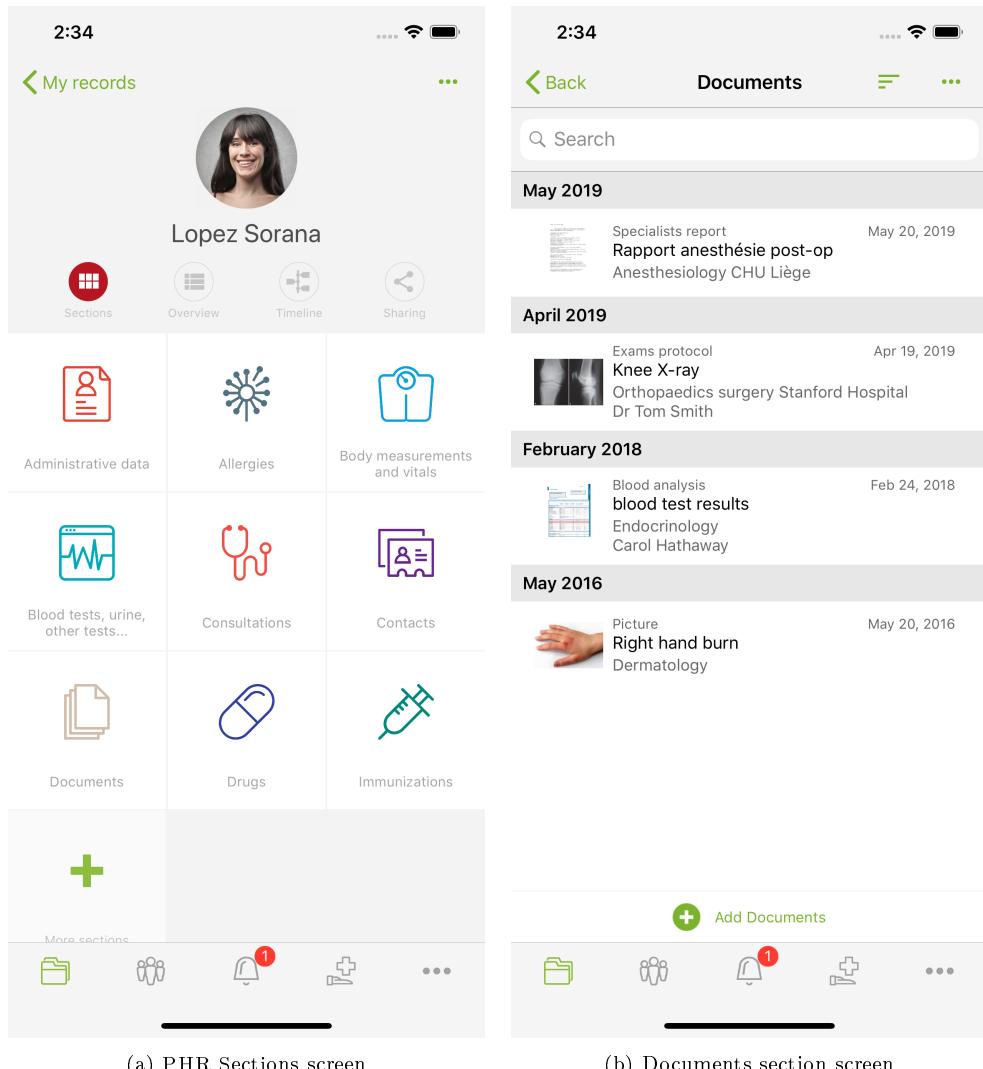


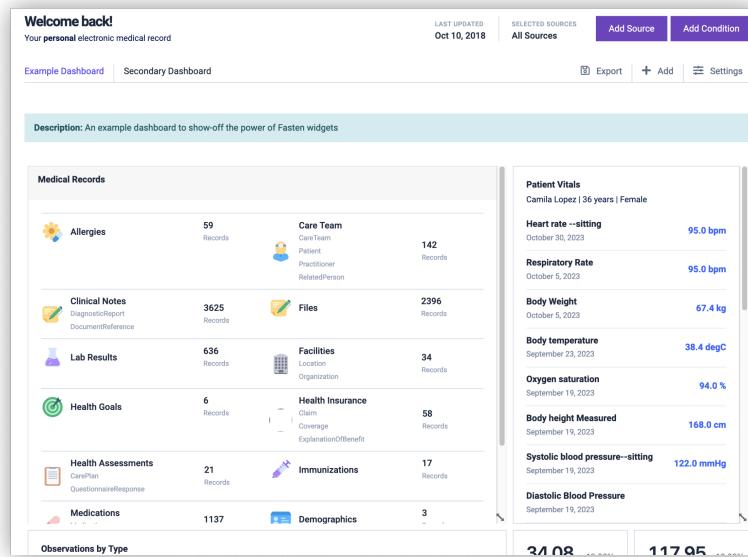
Figure 3.6: Andaman7 screenshots

## Fasten Health

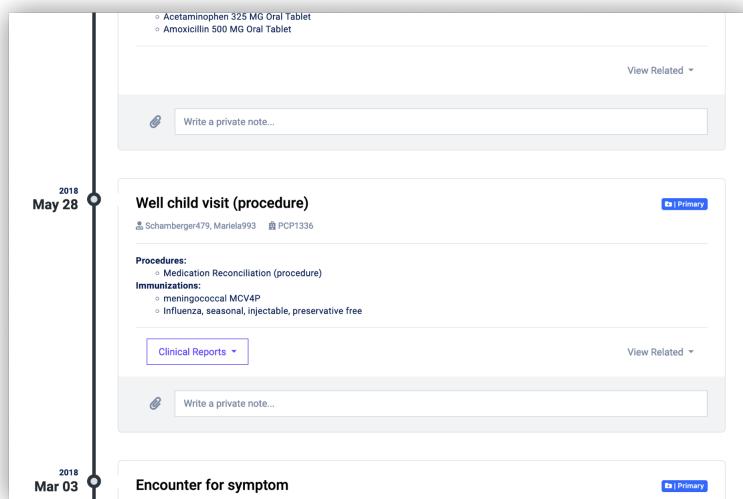
An open-source, self-hosted electronic medical record aggregator with optional paid desktop versions for Windows and Mac (Fasten Health, 2022). See Table 3.10 for a summary of its features and limitations and figure 3.7 for a screenshot of the app.

Key Features/Benefits	Limitations/Drawbacks
<ul style="list-style-type: none"><li>• Automatically aggregates records from multiple providers, like hospitals and labs.</li><li>• Supports self-hosting for complete control over data, stored locally.</li><li>• Compatible with protocols such as DICOM, FHIR, and OAuth2.</li><li>• Allows manual entry for allergies, vaccinations, and medications.</li><li>• Offers multiple dashboards with graphs to visualize health data.</li><li>• Supports multi-user functionality for families.</li></ul>	<ul style="list-style-type: none"><li>• Paid desktop versions may deter users.</li><li>• Manual data entry limited to new or existing encounters, complicating usage.</li><li>• Does not support OCR or automatic data extraction from documents.</li><li>• Lacks data-sharing capabilities with doctors.</li><li>• Requires technical expertise for self-hosting.</li><li>• Restricted to healthcare providers in the United States.</li></ul>

Table 3.10: Fasten Health Features and Limitations



(a) Dashboard screen



(b) Visit history screen

Figure 3.7: Fasten Health screenshots

## Chapter 4

# Project Planning and Design

### 4.1 UML Diagrams

UML, or Unified Modeling Language, is a standardised modeling language that consists of a set of diagrams used for modeling business processes and documenting software systems (Paradigm, 2024). It helps to better communicate potential designs and architectural decisions. The most common UML diagram types include:

- **Use Case Diagram** — Illustrates the system's intended functionality in terms of actors, use cases, and their relationships, showing how the system delivers value to users. The diagram can also be accompanied by a use case specifications document, which provides a detailed description of each use case.
- **Class Diagram** — Depicts the structure of the system by showing classes, attributes, operations, and static relationships between classes.
- **Sequence Diagram** — Demonstrates how objects interact in a particular, timed sequence scenario, focusing on the messages passed between objects.
- **Activity Diagram** — Represents the workflow of a target use case or business process through a series of activities, emphasizing steps, choices, iterations, and concurrency.

These diagrams were used to present the stakeholders with a visual representation of the system's design and functionalities. They can be found below, under their respective subsections.

#### 4.1.1 Use Case Diagram

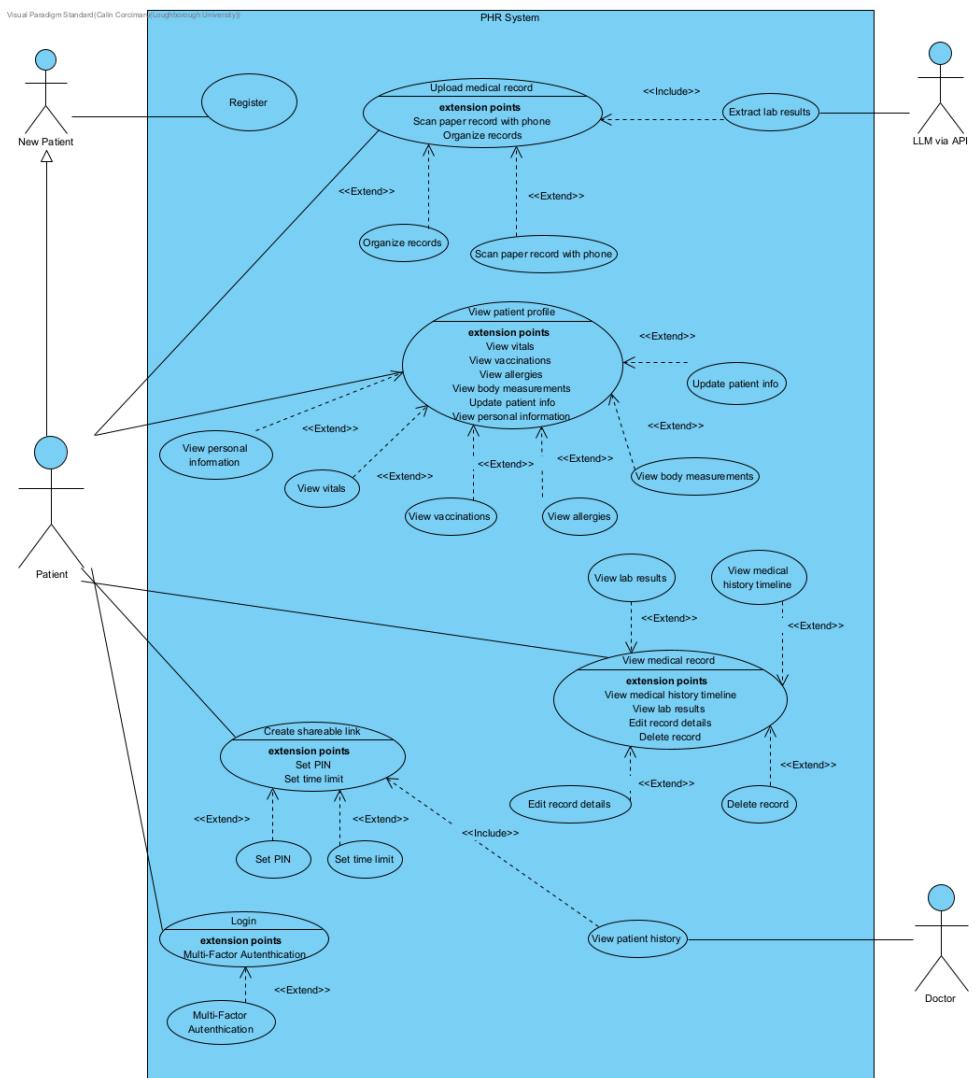


Figure 4.1: UML Use Case Diagram

#### **4.1.2 Use Case Specifications**

##### **Login**

Description	The Login use case allows the app user to log in to their existing account via their credentials, with an optional use of MFA.
Actors	Patient
Preconditions	Must have an existing account
Steps	<ol style="list-style-type: none"><li>1. User enters user and password</li><li>2. User clicks on the login button</li><li>3. App validates user credentials</li><li>4. Patient enters MFA code if enabled</li><li>5. App logs user in</li></ol>

##### **Register**

Description	The Register use case allows the app user to create a new account.
Actors	New Patient
Preconditions	No existing account with the email used to register
Steps	<ol style="list-style-type: none"><li>1. User enters user and password</li><li>2. User clicks on the register button</li><li>3. App validates user credentials</li><li>4. App logs user in</li><li>5. App sends verification email</li><li>6. User verifies email</li></ol>

## **Upload Record**

Description	The Upload Record use case allows the patient to upload their medical records to the app.
Actors	Patient, LLM
Preconditions	Must be logged in
Steps	<ol style="list-style-type: none"><li>1. User selects file upload (or camera scan)</li><li>2. User selects the record to upload</li><li>3. User clicks on the upload button</li><li>4. App validates the record</li><li>5. User selects the appropriate record type</li><li>6. If record type is lab result, app sends the record to LLM via API for processing into JSON format</li><li>7. App adds the record to the database</li><li>8. App shows confirmation to user</li></ol>

## **Share Records**

Description	The Share Records use case allows the patient to share their medical records with doctors.
Actors	Patient, Doctor
Preconditions	Must be logged in and have records uploaded
Steps	<ol style="list-style-type: none"><li>1. User selects option to create a share link</li><li>2. OPTIONAL: User selects the records to share</li><li>3. OPTIONAL: User adds a PIN to the share link</li><li>4. User selects time limit for the share link</li><li>5. App generates the share link</li><li>6. App sends the share link to the doctor via email</li><li>7. App shows confirmation to user</li><li>8. Doctor clicks on the share link</li><li>9. Doctor enters the PIN (if required)</li><li>10. App validates the PIN</li><li>11. App shows the records to the doctor</li></ol>

## **View Records**

Description	The View Records use case allows the patient to view and edit their uploaded medical records.
Actors	Patient
Preconditions	Must be logged in and have records uploaded
Steps	<ol style="list-style-type: none"><li>1. App provides a list of records or medical history</li><li>2. User selects record to view from list or medical history</li><li>3. App retrieves the record from the database</li><li>4. App shows the record to the user</li><li>5. OPTIONAL: User can view the record in a graphical format if lab result</li><li>6. OPTIONAL: User edits the record</li><li>7. OPTIONAL: User deletes the record</li></ol>

## **View Patient Profile**

Description	The View Patient Profile use case allows the patient to view and edit their profile information, including health information such as allergies, medications, vaccinations and recorded health data like blood pressure, glucose levels, etc.
Actors	Patient
Preconditions	Must be logged in
Steps	<ol style="list-style-type: none"><li>1. User selects the profile section</li><li>2. App retrieves the profile information from the database</li><li>3. App shows the profile information to the user — vaccinations, allergies, medications, health data</li><li>4. OPTIONAL: User edits the profile information</li><li>5. OPTIONAL: User adds new health data</li><li>6. OPTIONAL: User deletes health data</li><li>7. OPTIONAL: User can view health data in a graphical format</li></ol>

### 4.1.3 Sequence Diagrams

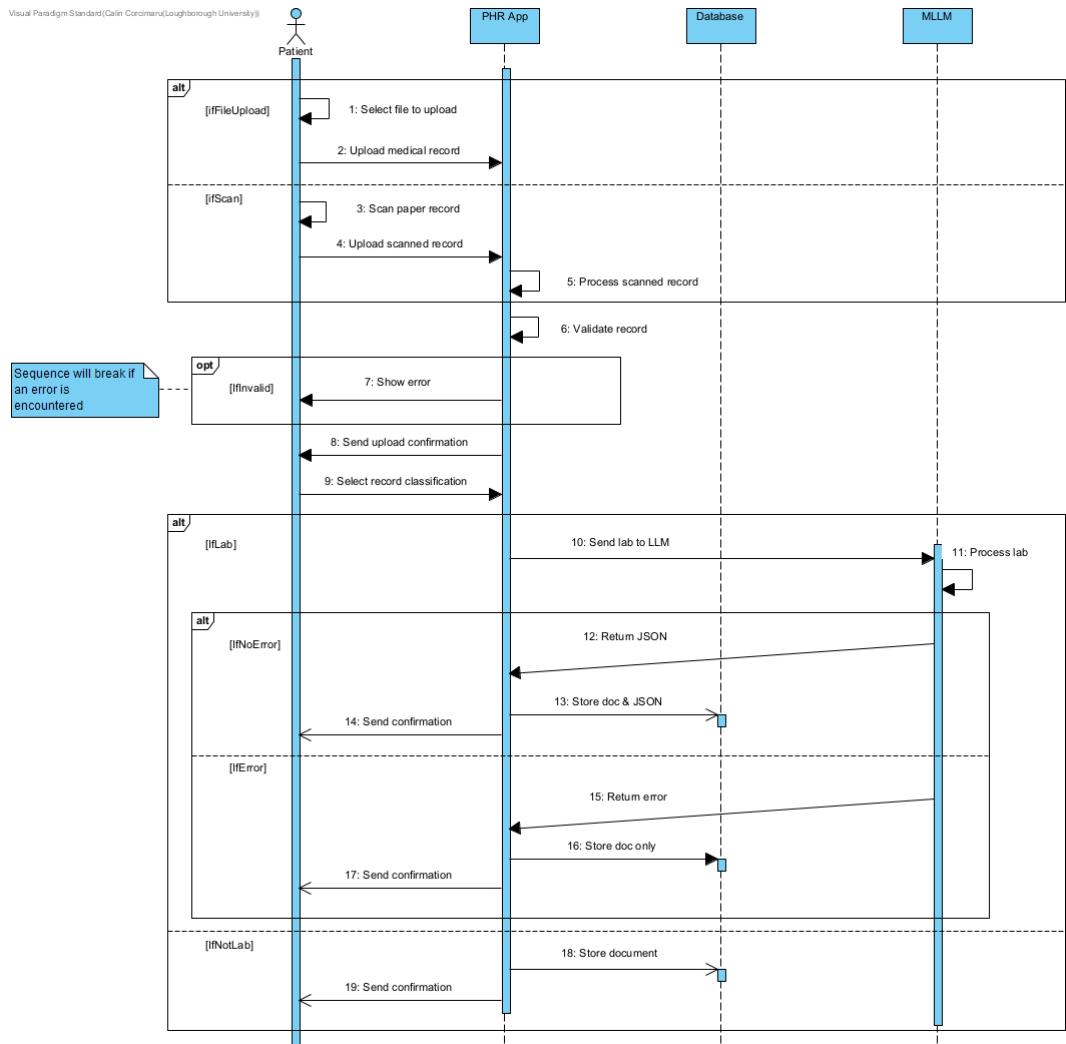


Figure 4.2: UML Sequence Diagram — Upload Record Use Case

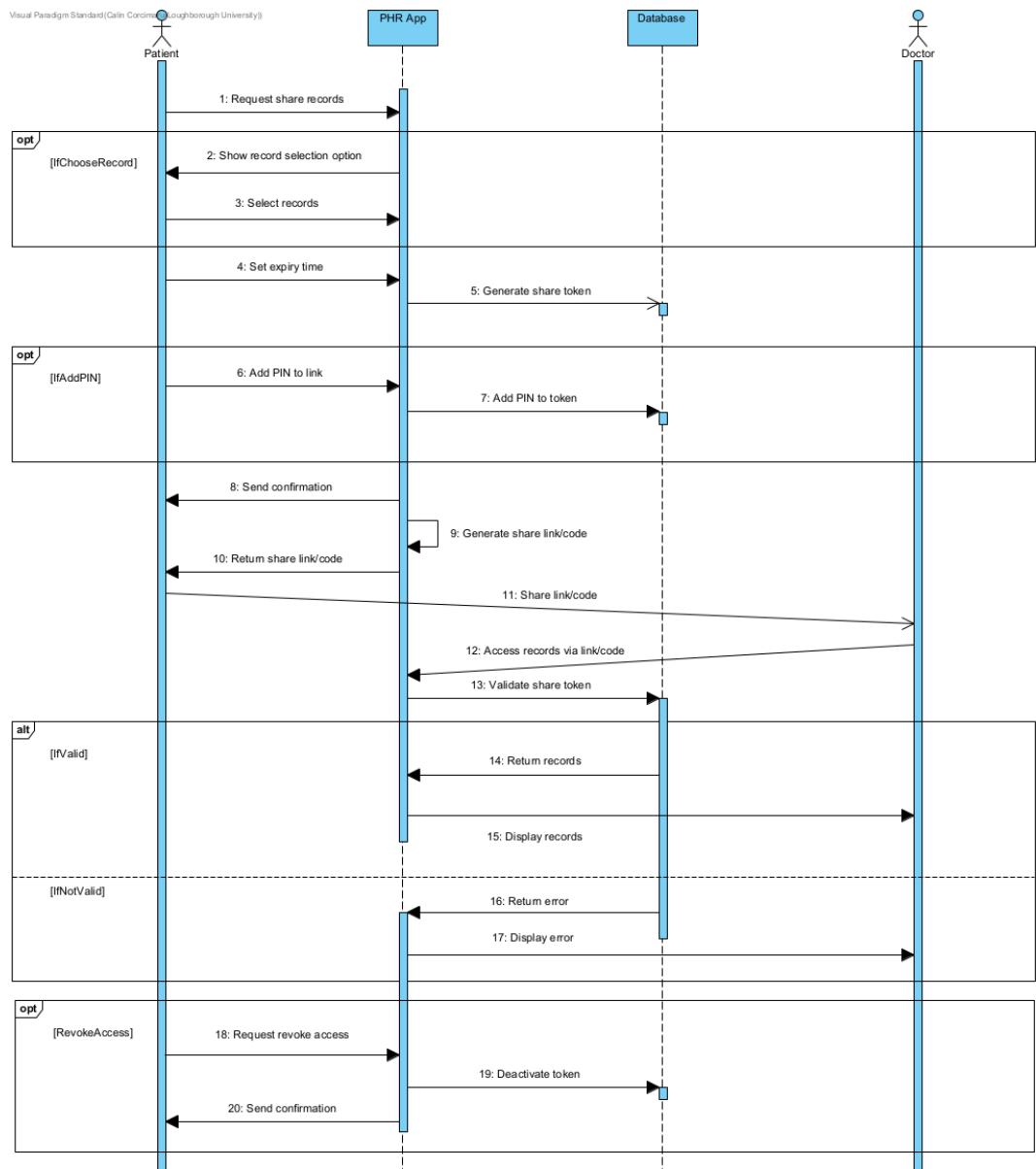


Figure 4.3: UML Sequence Diagram — Share Records Use Case

#### 4.1.4 Activity Diagrams

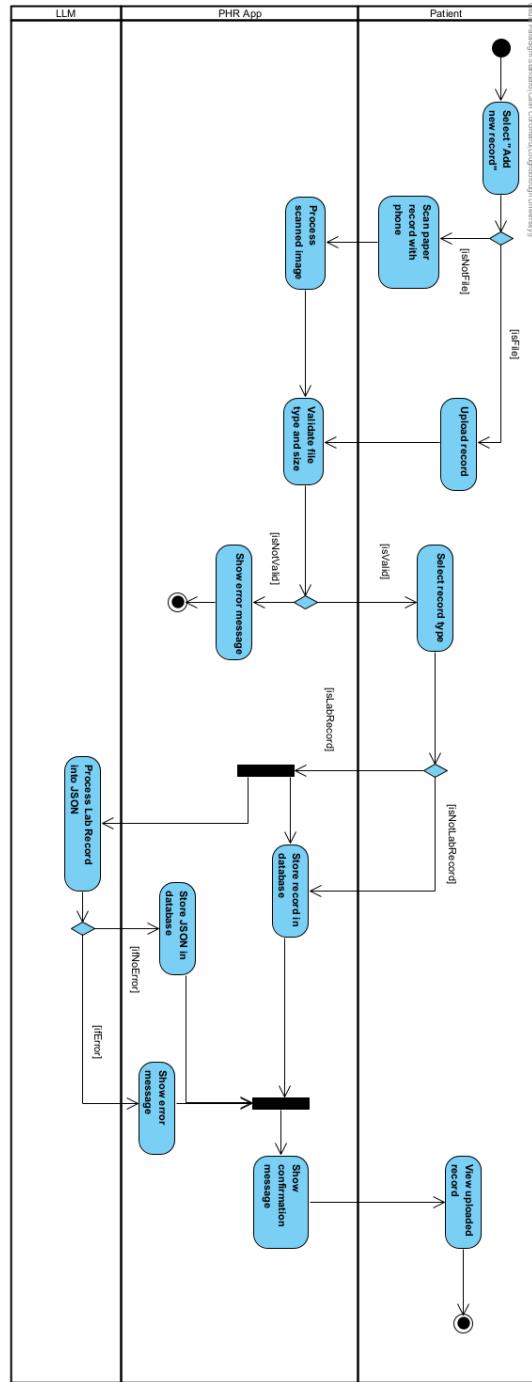


Figure 4.4: UML Activity Diagram - Upload Record Use Case

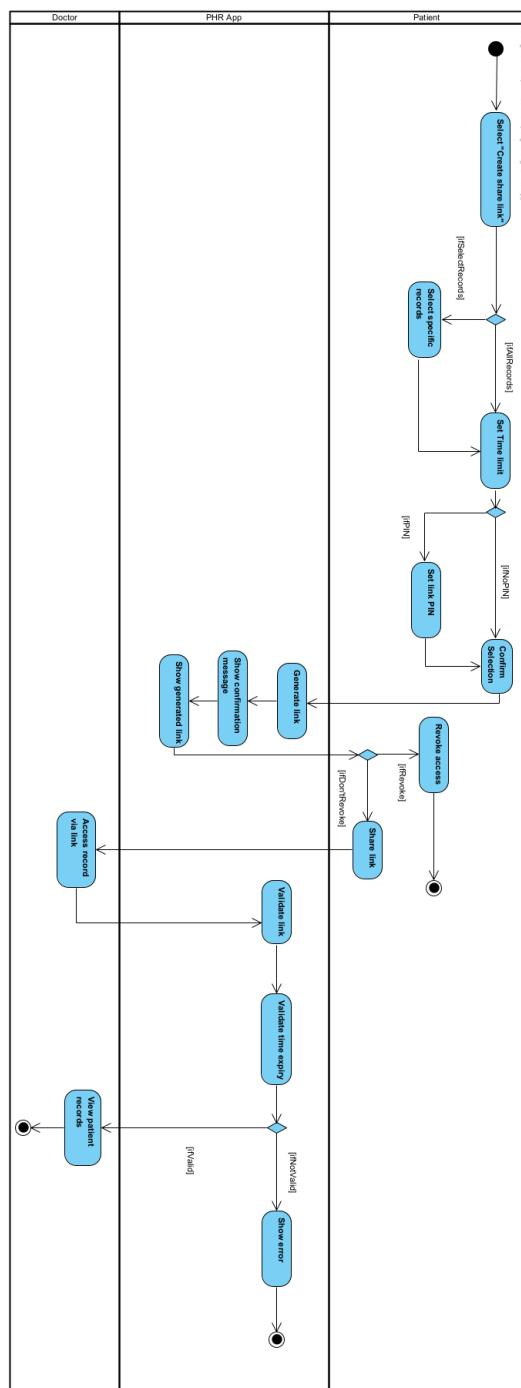


Figure 4.5: UML Activity Diagram - Share Records Use Case

## 4.2 Database Design

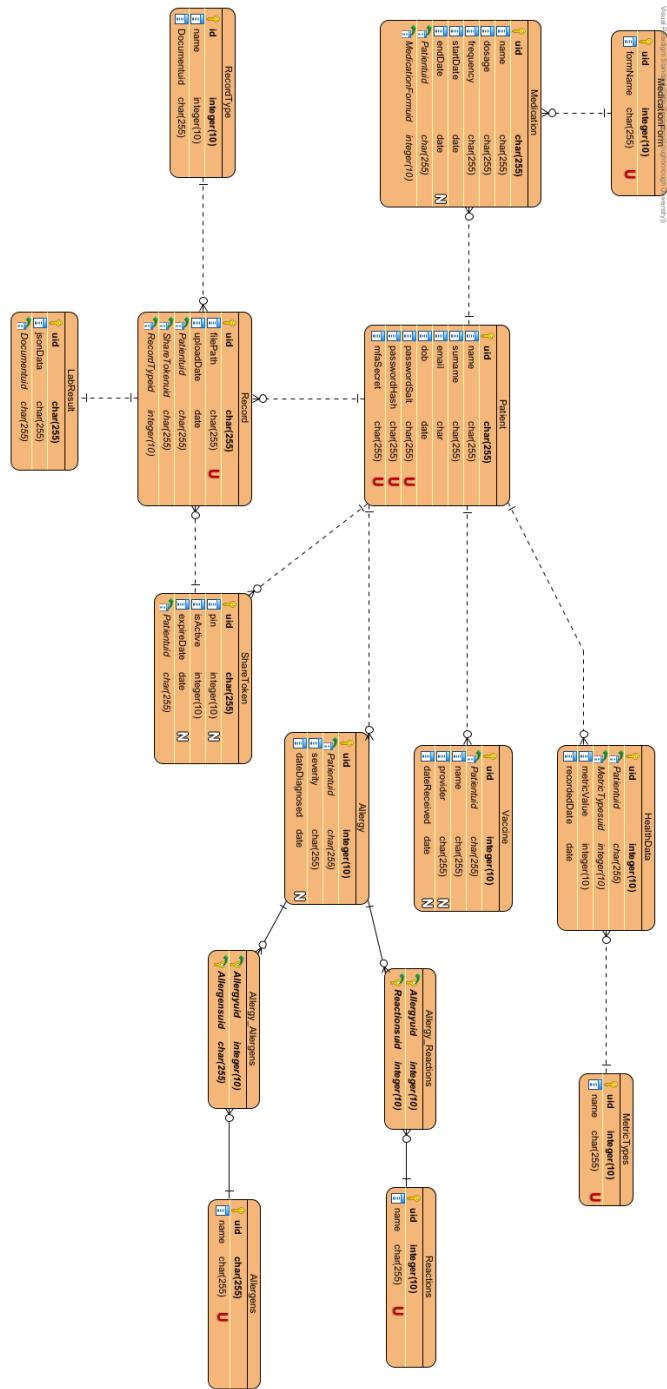


Figure 4.6: Entity Relationship Diagram

## 4.3 Wireframes

Wireframes have been used to provide a high-level overview of how the web application would look like and present some of its key functionalities. They have been created using Figma and later shown to key stakeholders. Based on the feedback received, the wireframes have been adjusted and some examples can be seen below in figures 4.7, 4.8 and 4.9, while the full set of wireframes can be found in appendix B.

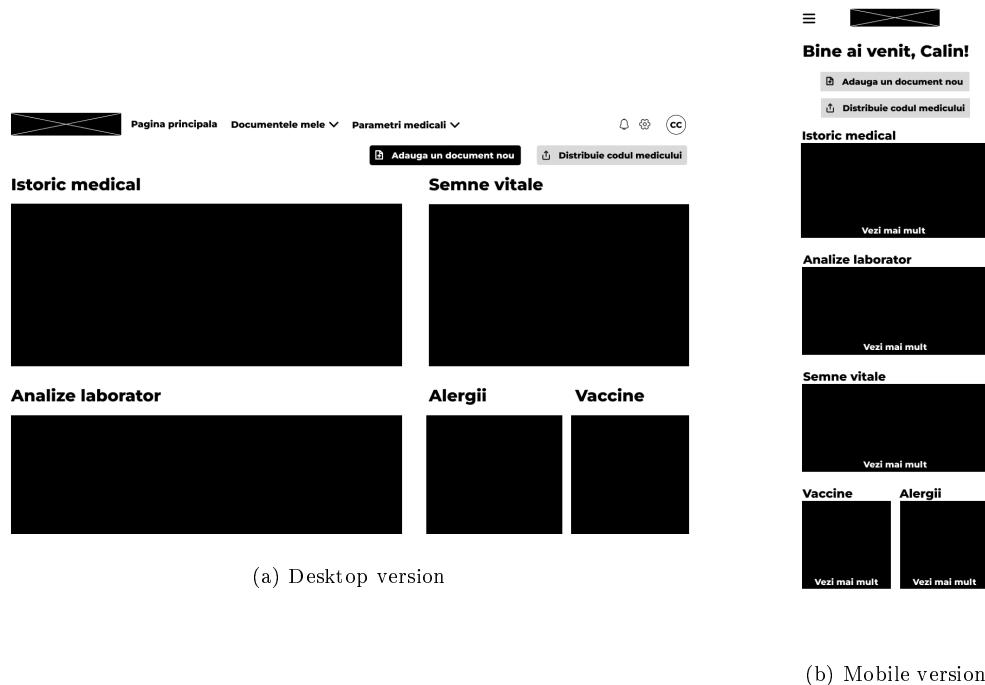


Figure 4.7: Desktop and Mobile version of the Dashboard screen

**Istoric medical**

Numele documentului	Categorie	Sub-Categoria	Numele doctorului	Data	Actiuni
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	

**Istoric medical**

**Adauga un document nou**

**Cauta document..**

- Vizita cardiolog**  
Consultatie | Cardiologie  
16.01.2025 [Detali >](#)
- Vizita cardiolog**  
Consultatie | Cardiologie  
16.01.2025 [Detali >](#)
- Vizita cardiolog**  
Consultatie | Cardiologie  
16.01.2025 [Detali >](#)
- Vizita cardiolog**  
Consultatie | Cardiologie  
16.01.2025 [Detali >](#)
- Vizita cardiolog**  
Consultatie | Cardiologie  
16.01.2025 [Detali >](#)
- Vizita cardiolog**  
Consultatie | Cardiologie  
16.01.2025 [Detali >](#)

(a) Desktop version
(b) Mobile version

Figure 4.8: Desktop and Mobile version of the Medical History screen

**Analize laborator**

**Tabel** **Grafic**

Selecteaza analiza: Hemoglobina

Selecteaza perioada: Ultimile 7 zile | Ultimile 30 zile | Ultimul an

Perioada custom: 08.01.2025 - 16.01.2025

Detalii analiza: Analiza sange | Sange

Valori de referinta: Maximum - 6 mg/L | Minimum - 10 mg/L

Pentru mai multe informatii: [Link]

Data colectarii	Valoare	Unitate	Statut	Document referinta
15.01.2025	13.01	g/L	Normal	Analize sange   Sange
15.01.2025	13.01	g/L	Normal	Analize sange   Sange
15.01.2025	13.01	g/L	Normal	Analize sange   Sange
15.01.2025	13.01	g/L	Normal	Analize sange   Sange
15.01.2025	13.01	g/L	Normal	Analize sange   Sange

**Analize laborator**

**Tabel** **Grafic**

Filtre: Selecteaza analiza: Hemoglobina | Selecteaza perioada: 08.01.2025 - 16.01.2025 | Ultimile 7 zile | Ultimile 30 zile | Ultimul an

**Detali analiza**

13.01 g/L | Normal | Analiza sange | Sange | Detali documental

13.01 g/L | Normal | Analiza sange | Sange | Detali documental

13.01 g/L | Normal | Analiza sange | Sange | Detali documental

13.01 g/L | Normal | Analiza sange | Sange | Detali documental

13.01 g/L | Normal | Analiza sange | Sange | Detali documental

(a) Desktop version
(b) Mobile version

Figure 4.9: Desktop and Mobile version of the Lab Test Results screen

## 4.4 Project Tech Stack

Based on the research conducted in sections 3.3.1, 3.3.2 and 3.3.3, the following tech stack was chosen for the project, which can be seen in diagram 4.10.

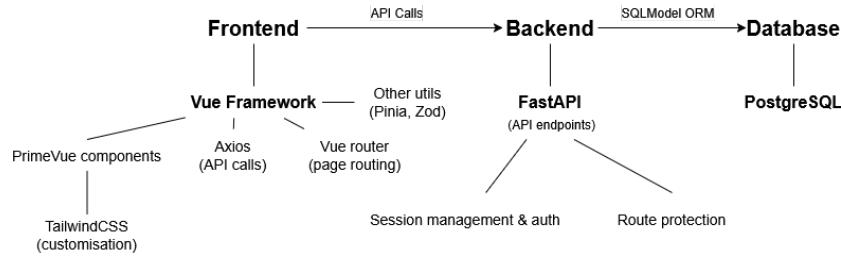


Figure 4.10: Proposed System Architecture

### 4.4.1 Frontend

For the frontend Vue was used, which is a JavaScript framework for building single-page applications (SPAs) (You, 2024). Its strengths lie in the ease of use and flexibility, with features like components, element reactivity and Single File Components (SFCs) that allows HTML, CSS and JS to be used in a single .vue file. Vue also boasts a large and active community, with many libraries and guides available for developers. Previous experience working with Vue made it an ideal choice for this project. Diagram 4.11 showcases how different elements of the frontend interact. The next subsections will briefly discuss each.

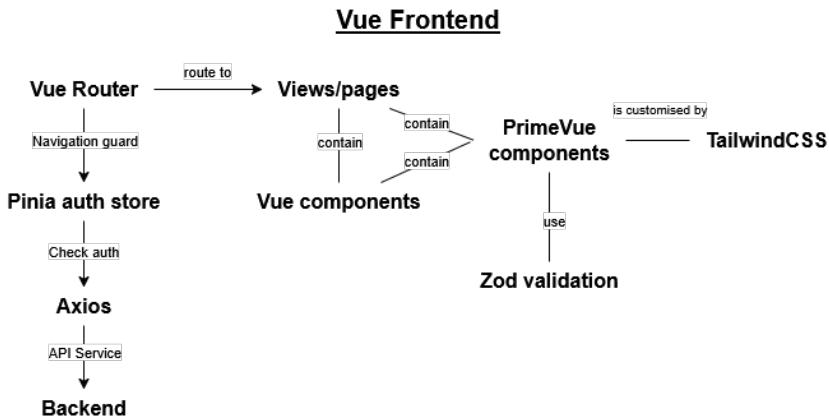


Figure 4.11: Interaction of Frontend Components

## **Vue Router**

Vue Router is the official client-side router for Vue.js, which allows the user to navigate between application pages without a full reload (Vue Router, n.d.). It makes full use of the SPA capabilities of Vue, providing a seamless user experience. Vue Router can also be used to create navigation guards to protect certain routes from unauthorized access, which is crucial for the security of the application.

## **Pinia**

Pinia is a store library for Vue made by the same team (Pinia, n.d.). Its main functionality is to provide a way to share states across Vue components and pages. In this project, Pinia will be used to store the user's authentication status and other global states that need to be shared within the application.

## **Axios**

Axios is a promise-based HTTP client for the browser and Node.js (Axios, n.d.). It is used to make HTTP requests to the backend. Similarly, it allows for automatic interaction with cookies in requests and responses, which is important for API communication.

## **Vue views and components**

Vue components and views represent the foundation of the frontend. Views represent the different pages of the application, which can be navigated to using the Vue Router. Components allow to break down the UI into independent and reusable elements (Vue.js, n.d.). Each component can be stored in a separate `.vue` file and imported within a view/page or another component.

## **PrimeVue and TailwindCSS**

PrimeVue is a component library for Vue, which provides a set of pre-built components that can be used to easily create the frontend of the application (PrimeTek, n.d.). TailwindCSS is a CSS framework that provides many classes that can be used to style different elements of the application (Tailwind Labs Inc., 2025). When combined, PrimeVue provides the basic components like buttons, forms, and tables, while TailwindCSS is used to style these components.

#### 4.4.2 Backend

For the backend, FastAPI was chosen as the main framework. It is a modern framework used in building APIs with Python (FastAPI, 2024). Based on the research, FastAPI seemed to be a good choice due to its ease of use, simplicity and documentation available. Previous experience with Python made a Python-based framework an easy choice for this project. Finally, a separate backend framework allows for more flexibility in the future, as it can be developed independently from the frontend and be re-used for different platforms, such as mobile. Diagram 4.12 shows how different components of the backend will interact with each other.

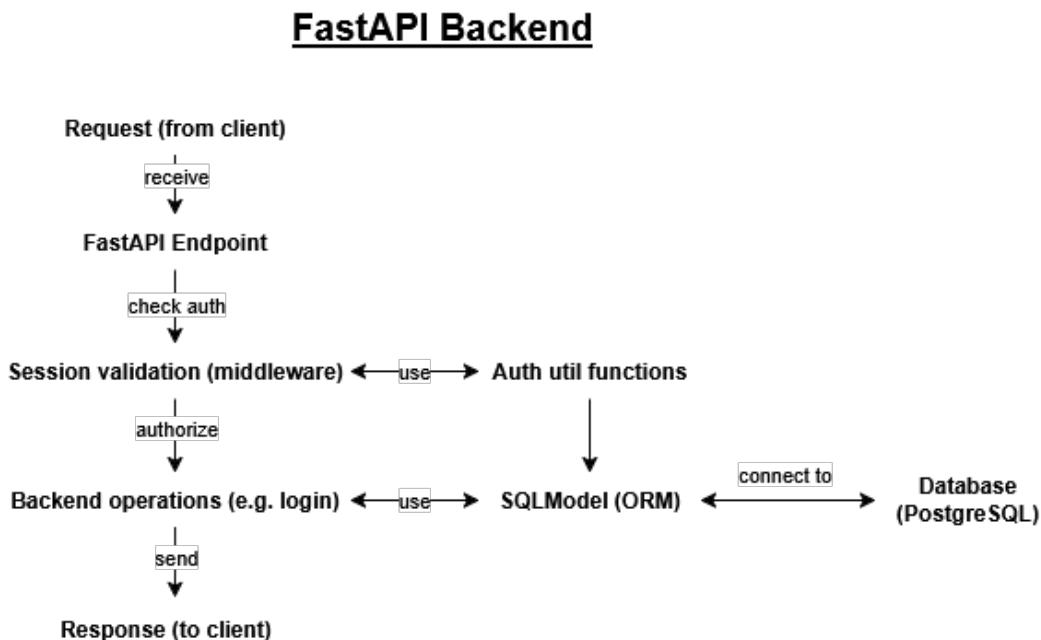


Figure 4.12: Interaction of Backend Components

#### 4.4.3 Database

PostgreSQL was the database of choice, a relational database system that boasts a good reputation for its reliability, wide feature set and performance (PostgreSQL Global Development Group, 2025). It supports a wide range of data types, including JSON and UUID, which are important for this project. Similarly, PostgreSQL offers robust security features like an access-control system or column And row-level security.

SQLModel, a Python library built on top of SQLAlchemy and Pydantic, will be used to

interact with the database. SQLModel provides a way to define database tables using classes, which are then used to interact with the database (SQLModel, n.d.).

## 4.5 Project Management

### 4.5.1 Methodology and Tools

Based on the research in section 3.1, a hybrid approach has been chosen, with Waterfall as the main methodology for planning and managing the initial part of the project, such as requirements gathering and the design of the system. The development part of the project will be done using a more Agile approach. There are several reasons for this choice:

1. The nature of the project — the project has a limited timeframe (about 6–7 months) and is on a smaller scale.
2. Documentation requirements — the project progress needs to be documented in this report, including the requirements gathered, design considerations and implementation decisions and outcomes.
3. Customer involvement — development will be done closely with stakeholders, who will be providing feedback throughout the project.

Notion will be used as the project management tool, which allows for task management, documentation and knowledge sharing. It provides a wide range of features, including templates for Agile project management, creation of user stories, backlogs, sprints and boards (Notion Labs, 2025).

### 4.5.2 Sprint planning

Based on the time remaining after the completion of previous project phases, it was decided that only 6 sprints will be able to be run before the deadline. The sprints will be planned as follows:

- **Sprint 1** — Will focus on installing the necessary tools, frameworks and setting up initial project files. Additionally, a basic frontend and backend will be created, with a focus on basic functionalities like user authentication and registration.
- **Sprint 2** — Will focus on the patient profile, vaccines, allergies, medications and health data. The dashboard will also be created, with a basic layout and design.
- **Sprint 3** — Will focus on finishing the main dashboard, adding the upload feature and medical history.

- **Sprint 4** — Will focus on processing lab results with the MLLM API and displaying them in tabular or graphical format.
- **Sprint 5** — Will focus on sharing records with doctors through various methods like email, share links or QR codes.
- **Sprint 6** — Will address any missing features or functionalities that need to be added, as well as final testing.

# Chapter 5

# Development

This chapter discusses the development process of the project, structured on a sprint-by-sprint basis. Each sprint section will outline the key features implemented, challenges faced and the requirements completed in each sprint.

## 5.1 Sprint #1

The first sprint focused on establishing the foundation for this system. Its main objectives involved setting up the database schema, creating the authentication API endpoints, basic authentication flow and the frontend of the application.

### 5.1.1 SQLModel Schemas & Database creation

```
1  class DateFormattingModel(SQLModel):
2      dob: date | None = None
3      @field_serialiser('dob')
4      def serialise_dob(self, value: date) -> str:
5          return value.strftime("%d-%m-%Y")
6
7  class User(DateFormattingModel, table=True):
8      id: uuid.UUID = Field(default_factory=uuid.uuid4, primary_key=True)
9      name: str
10     email: EmailStr = Field(index=True, unique=True)
11     hashed_password: str
```

Listing 5.1: SQLModel User Schema

SQLModel played a pivotal role in creating the database tables. The code above is an example of a schema that was used to create the User table in the database. It was also used to create specific models for the endpoints to validate the incoming or outgoing data. The following snippet is an example of a model that was used to validate the data that was being sent to the register endpoint.

```
1  class UserAuth(SQLModel):
2      email: EmailStr
3      password: str
4      name: str | None = None # Will only be used for registration
```

Listing 5.2: SQLModel Auth Schema

### 5.1.2 Authentication & APIs

As the application would handle sensitive health data, its security was determined as an important concern for both the system and its users. An evaluation of modern authentication methods was conducted, identifying three possible approaches:

- JWT Tokens
- Session-based authentication
- Using 3rd party authentication services like Auth0

JWT (JSON Web Tokens) authentication involves generating tokens containing user information for client-side storage (Akanksha and Chaturvedi, 2022). Its main advantages are its statelessness and minimal server-side management. However, tokens can be vulnerable to XSS and CSRF attacks and cannot be invalidated once created (Akanksha and Chaturvedi, 2022; Papathanasaki, Maglaras, and Ayres, 2022). The authors recommend using methods like pairs of access and refresh tokens to mitigate these risks, or to store the tokens inside cookies.

An alternative, session-based authentication, stores session information server-side, using cookies to maintain the sessions (Akanksha and Chaturvedi, 2022; Papathanasaki, Maglaras, and Ayres, 2022). While facing similar security risks as JWT, they can be secured through flags like `HttpOnly` and `SameSite` (Mozilla, 2025). Sessions can be invalidated server-side, providing better security control.

Third-party authentication options include OAuth2 through providers like Google (Papathanasaki, Maglaras, and Ayres, 2022; Okta, Inc., 2025b) or services like Auth0 (Okta, Inc., 2025a). While these offer quick implementation, they may introduce dependencies or security concerns due to third-party data storage.

Consideration was also given to Mpass, which is a Single Sign On (SSO) authentication system that offers a secure and easy way to access electronic government services in Moldova (Agenția de guvernare electronică, 2025). While it may not be possible to currently integrate MPass into the system, it will be documented as a possible future integration, which will be discussed in the 7.2 section.

Following the evaluation of authentication methods, session was selected based on its ease of implementation and balanced level of security. The ability to control the session on the server side was also an important factor, as it allows the system to safely log out users or invalidate sessions due to inactivity or suspicious activity.

The implementation used a Session database table containing a UUID, which was passed to the user via cookies. Upon successful authentication, the backend would configure the cookie and use it with every subsequent backend request. To ensure safety from common attacks such as Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF), best practices from Cheat Sheets Series Team, 2025; Mozilla, 2025 were used to configure the cookies:

```

1 # Set the session cookie in the response and send it to the client
2 response.set_cookie(
3     "session_id",
4     str(session_id), # Using str() to convert the UUID to a string
5     httponly=True,
6     max_age=3600, # 1 hour
7     samesite="strict",
8     secure=True)

```

Listing 5.3: Session Cookie Flags

The system's API endpoints were protected through a session validation middleware implemented via FastAPI's dependency injection system, which would check if the user was authenticated before processing the request. An example of this can be seen below:

```

1 @app.post("/logout")
2 # Logout endpoint example with validation dependency
3 async def logout(response: Response, request: Request, user_id: uuid.
4     UUID = Depends(validate_session), session: Session = Depends(
5         get_session))
6
7 # Validation function to check if the user is authenticated
8 async def validate_session(request: Request, session: Session =
9     Depends(get_session)):
10    session_id = request.cookies.get("session_id")

```

```

8     if not session_id:
9         raise HTTPException(
10            status_code=status.HTTP_401_UNAUTHORIZED,
11            detail="Session cookie not found"
12        )
13
14    existingAuthSession = session.exec(
15        select(AuthSession)
16        .where(AuthSession.id == uuid.UUID(session_id))
17        .where(AuthSession.expires_at > datetime.now())
18    ).first()
19
20    if not existingAuthSession:
21        raise HTTPException(
22            status_code=status.HTTP_401_UNAUTHORIZED,
23            detail="Session not found"
24        )
25
26    return existingAuthSession.user_id

```

Listing 5.4: FastAPI Dependency for Session Validation

The system also used a hashing algorithm to hash the passwords before storing them in the database. Three hashing algorithms were identified as the best choices: bcrypt, scrypt and Argon2.

Bcrypt is a popular hashing algorithm widely used in the industry (Naiakshina et al., 2020; Ntantogian, Malliaros, and Xenakis, 2019). It offers an optimal balance between security and speed, making it a popular choice for real-world implementation (A. Sharma et al., 2024). Bcrypt was ranked on the mid-to-high end of the security scale, with algorithms such as scrypt and Argon2id scoring higher in the dimension of security (A. Sharma et al., 2024; Naiakshina et al., 2020). However, while scrypt and Argon2id are more secure, they are also more computationally expensive, and may require more configuration to be used effectively (A. Sharma et al., 2024).

In the end, the decision was made to use bcrypt due to its simplicity of use, wide adoption and its balance between security and performance. The passlib library was used to hash the passwords before storing them in the database. An example of this can be seen below:

```

1  def verify_hash(plaintext_password: str, hashed_password: str):
2      return bcrypt.verify(plaintext_password, hashed_password)
3

```

```

4     def create_hash(plaintext_password: str):
5         return bcrypt.hash(plaintext_password)

```

Listing 5.5: Hashing passwords with bcrypt

### 5.1.3 Frontend Setup

The frontend development done in this sprint focused on creating the login and register pages. PrimeVue components were used to quickly and easily create the form elements, maintaining visual consistency and a good user experience. Figure 5.1 shows an example of the DatePicker component implemented in the Register page.

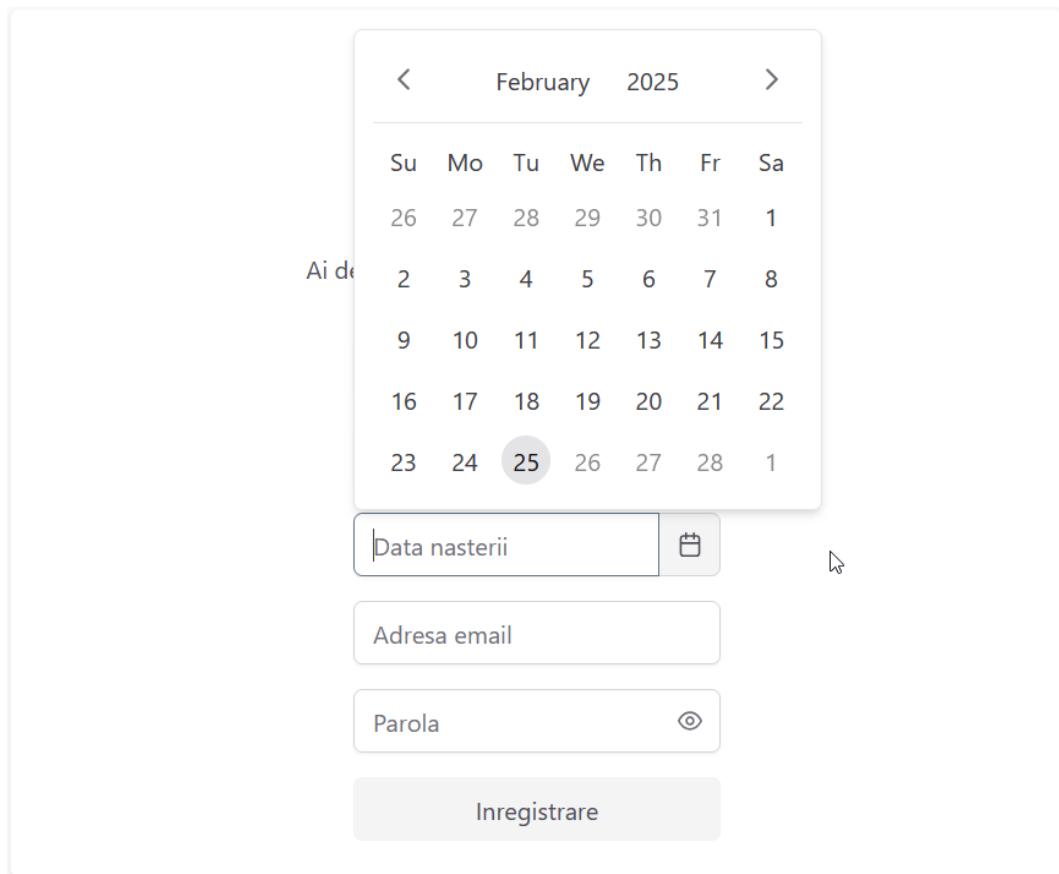


Figure 5.1: PrimeVue DatePicker component

To ensure control of route access, a simple navigation guard was created using Vue Router. Its role is to protect the routes flagged as requiring authentication, performing validation before navigation. An example of this can be seen below:

```

1 router.beforeEach(async (to) => {
2   const authStore = useAuthStore()
3   if (to.meta.requiresAuth) {
4     await authStore.checkAuth() // Check if user is authenticated
5     if (!authStore.isAuthenticated) {
6       return { name: 'Login' } // Redirect to login if not
7       authenticated
8     }
9   } // More code to handle other cases
10 })
11
12 export default router

```

Listing 5.6: Navigation Guard

Pinia store was used to manage the global authentication state across the application. It enabled access of authentication states across multiple application components, and was used by the Navigation Guard to validate a user's authentication status. The authentication store can be seen below:

```

1 export const useAuthStore = defineStore('auth', () => {
2   const isAuthenticated = ref(false)
3   const user = ref(null)
4
5   async function checkAuth() { // check if user is authenticated
6     try {
7       const response = await api.get('/me') // return user ID if user
8       is_authenticated.value = true // user is authenticated
9       user.value = response.data.user
10    }
11    // More code to handle errors and no authentication
12  }
13
14  return { isAuthenticated, user, checkAuth }

```

Listing 5.7: Pinia Store for Authentication

#### 5.1.4 Challenges

##### Lack of experience

A primary challenge encountered in this sprint was the lack of experience with some of the frameworks and libraries. As such, the development progress moved slower than expected

due to the need to consult documentation or researching guides online. In the end, this learning process enabled to establish a solid foundation that would benefit the next sprints' development.

### **Implementation decisions**

Another challenge was evaluating implementation options for different elements of the application, such as authentication. Additional time was spent researching the best practices for each feature analysing their advantages, disadvantages and ways to implement them in the current system.

#### **5.1.5 Requirements completed**

This initial sprint enabled the completion of some of the foundational requirements for this application:

- The system must provide a secure login mechanism for patients by using a combination of login and password.
- The database must store the user credentials in a secure manner.
- The system must be accessible on all modern desktop and mobile-based browsers.
- The system must allow patients to add their own personal information, such as name or date of birth.

## **5.2 Sprint #2**

The second sprint focused on implementing system functionality, specifically managing vaccines, allergies and medications. This approach enabled further familiarisation with the frameworks and libraries used in the project, while also establishing design and implementation patterns used in further sprints.

### **5.2.1 Vaccines, allergies and medication management**

Database schemas were first created for vaccines, allergies and medications by using SQLModel, followed by corresponding API endpoints to add, update, delete and view the data. The frontend development used Vue components to create reusable elements, that would be used to display the user added records. These components used other Vue features such as props and emits to pass data and events between elements. An example of the Vaccine Card can be seen below:

```

1 <Card class="w-full" :pt="cardStyles">
2   <template #title>
3     <span class="font-bold text-2xl">{{ name }}</span>
4   </template>
5   <template #subtitle>
6     <div class="flex items-center justify-between">
7       <div>
8         <span class="font-bold">{{ provider }}</span>
9         <span>{{ date_received }}</span>
10      </div>
11      <div>
12        <Button icon="pi pi-eye" class="p-button-rounded p-button-text"
13          @click="emit('showFile', props.id)" v-if="hasCertificate"/>
14        <Button icon="pi pi-ellipsis-h" class="p-button-rounded p-button-
15          -text"
16            @click="toggle"/>
17        <Menu ref="menu" :model="items" :popup="true" />
18      </div>
19    </div>
20  </template>
21 </Card>

```

Listing 5.8: Vue Vaccine Card Component Example

The VaccineCard component was then easily integrated in its respective page:

```

1 <VaccineCard
2   v-for="vaccine in vaccines"
3   :key="vaccine.id"
4   v-bind="vaccine"
5   :has-certificate="!!vaccine.certificate"
6   @delete="deleteVaccine"
7   @open-edit="openEditDialog"
8   @show-file="showCertificate"
9 />

```

Listing 5.9: Using VaccineCard Component



Figure 5.2: Vue Vaccine Card component

Vue's reactivity system was used to ensure the data was automatically updated whenever any change was made. Examples of the reactivity system in use can be seen below:

```
1 // Will add a vaccine to the vaccines array, which will automatically
2 // create a new VaccineCard element
3 const addVaccine = (vaccine) => {
4     vaccines.value.push(vaccine)
5 }
6
7 // Will remove a vaccine from the vaccines array, which will
8 // automatically remove the VaccineCard element
9 const deleteVaccine = (id) => {
10     vaccines.value = vaccines.value.filter((vaccine) => vaccine.id !==
11         id)
```

Listing 5.10: Vue Reactivity System

### 5.2.2 File Uploads feature

A file upload feature was also implemented in this sprint. It was initially intended for vaccination certificates, but designed to be easily extensible for other future records. This required for a new table to be created in the database, FileUpload, which would contain file metadata. Similarly, a link table for allergies and severities was created to enforce consistency in the names used and allow for easy filtering and sorting on the frontend. The updated ERD diagram, with the new tables, can be seen in figure 5.3:

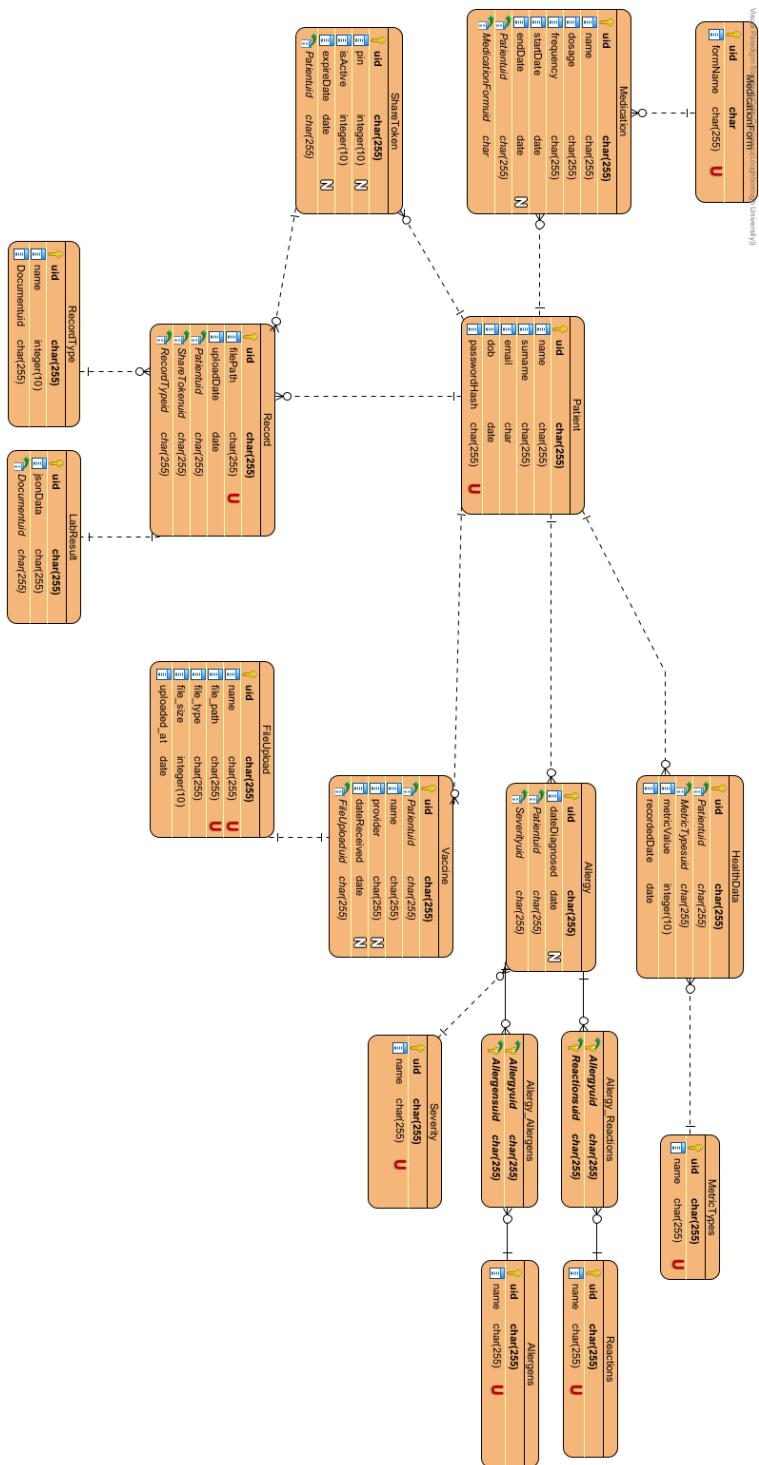


Figure 5.3: Updated Entity Relationship Diagram Sprint # 2

File security was implemented by using AES as the encryption algorithm. It was selected based on a comparative research of existing encryption methods, with AES consistently offering the best balance between security and performance among other symmetric and asymmetric encryption algorithms (Nurgaliyev and H. Wang, 2021; Marqas, Almufti, and Ihsan, 2020; Alenezi, Alabdulrazzaq, and Mohammad, 2020). A symmetric algorithm was used for its implementation simplicity and to prevent users losing access to their files by losing their private key. The key used to encrypt the files was stored in the environment variables. In the interest of time, no rotation system for the key was implemented, however this would be a good feature to add in the future.

The file upload process can be seen in figure 5.5 below. It encompasses the following steps:

1. File upload to server via multipart form data
2. Server-side validation of file type and size
3. File encryption and storage in the file system and database
4. For retrieval, the file would be streamed via a new endpoint

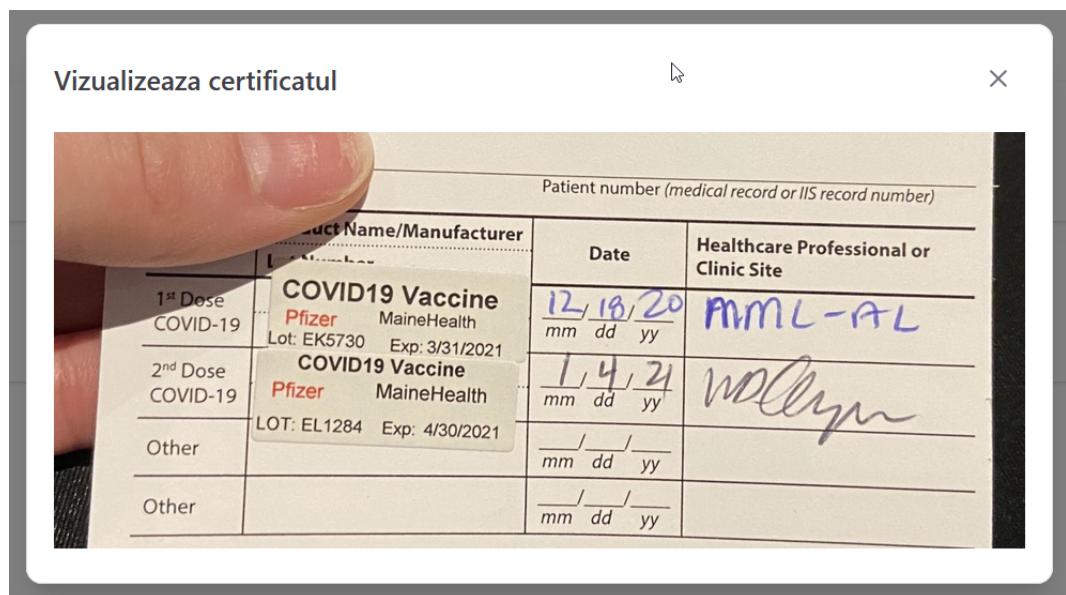
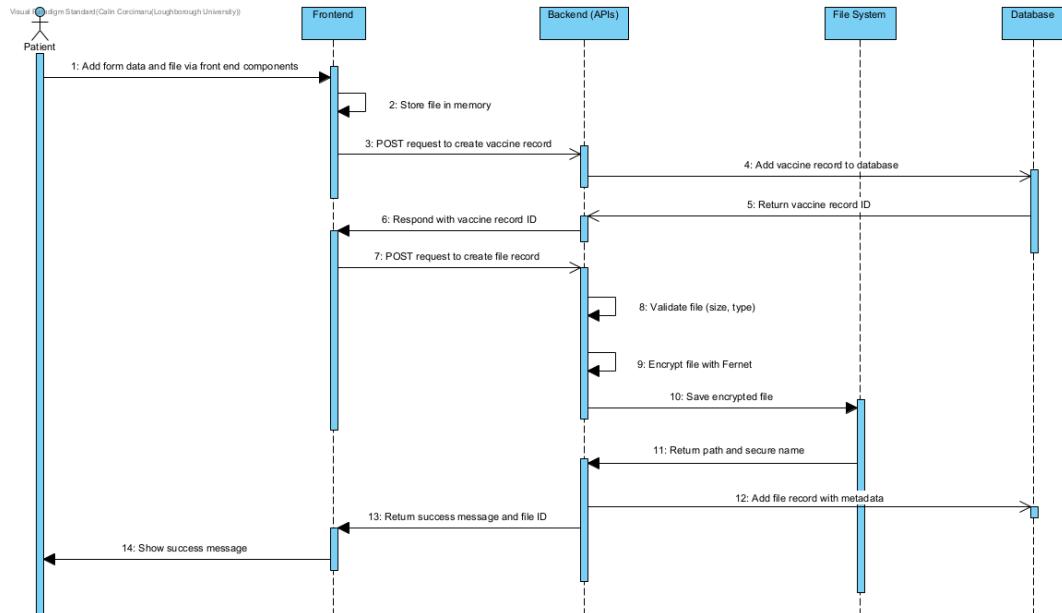
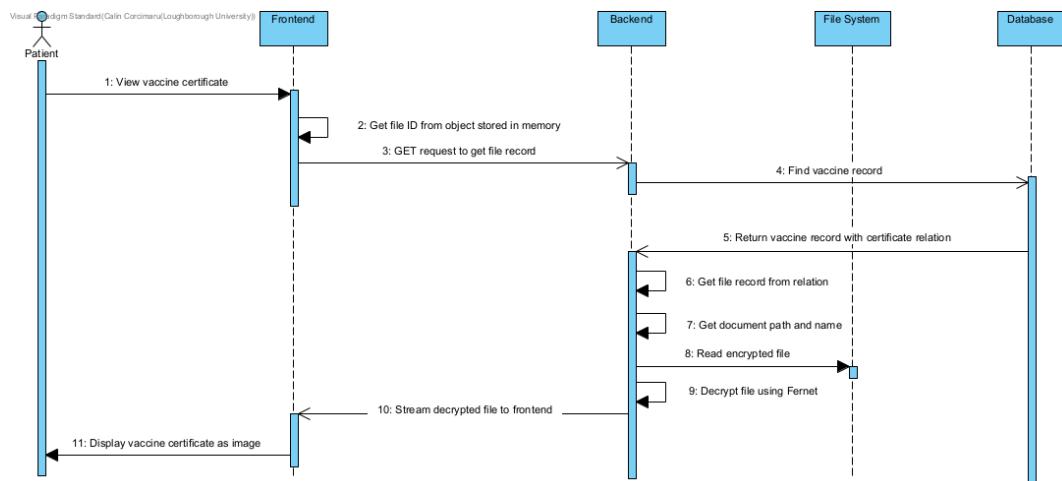


Figure 5.4: Uploaded Vaccine Certificate



(a) File Upload Sequence Diagram



(b) File View Sequence Diagram

Figure 5.5: Sequence Diagrams for File Upload and File View

```

1 # Get a file by ID
2 @app.get("/files/{record_type}/{record_id}")
3 async def get_file(record_type: str, record_id: uuid.UUID, user_id:
4     User = Depends(validate_session), session: Session = Depends(
5         get_session)):
6     # Database query omitted for brevity
7     async def get_data_from_file():
8         with open(file_record.file_path, "rb") as f:
9             encrypted_content = f.read()
10            decrypted_content = decrypt_file(encrypted_content)
11
12            yield decrypted_content
13
14    return StreamingResponse(
15        content=get_data_from_file(),
16        media_type=file_record.file_type,
17        status_code=status.HTTP_200_OK,
18        headers={"Content-Disposition": f"inline; filename={file_record.
19            name}"})
20

```

Listing 5.11: File Download Endpoint

The frontend was created with both mobile and desktop browsers in mind. This was achieved through using a combination of PrimeVue components TailwindCSS styling. Examples of the allergies page on both desktop and mobile can be seen in figure 5.6.

### 5.2.3 Challenges

This sprint encountered its share of challenges. However, tackling them showed that undertaking a more agile approach to the project was beneficial, as the project allowed to embrace change and to be flexible during the development process.

#### New functionality integration

Introducing new functionality in the system, while ensuring compatibility with both the frontend and backend, was the main challenge this sprint. Additional complexity was added by changing existing code or database schemas to accommodate the new features, which required careful planning and testing to ensure that the new features did not break any existing functionality.

## **File Uploads**

Lack of prior experience working with server and client side file handling features was another challenge encountered this sprint. It required additional research, development and testing time to ensure proper implementation and compatibility with the existing system.

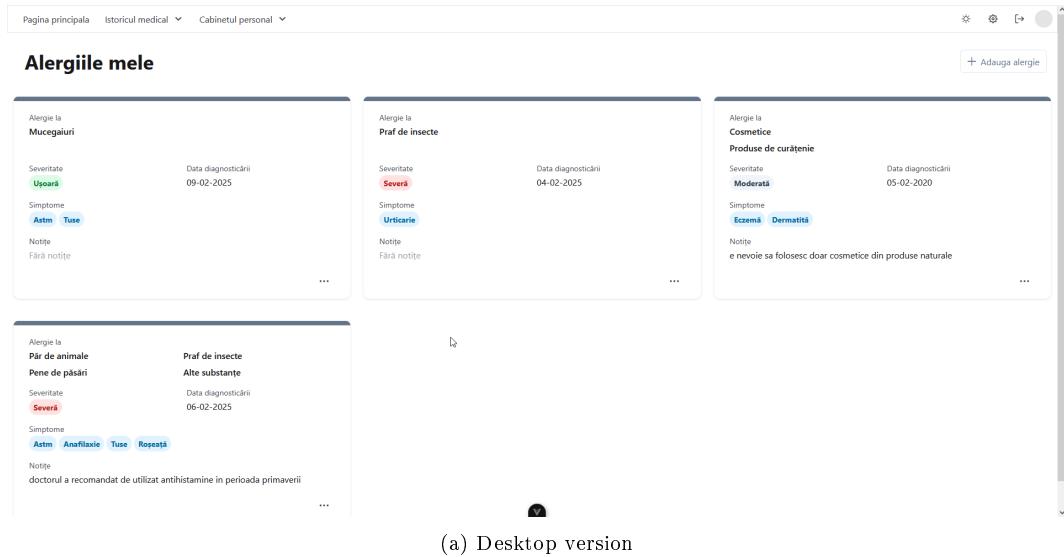
## **Poor planning & time management**

The additional complexity brought by the integration of this sprint's feature required more time than initially estimated. As such, poor planning and time management resulted in moving some of the sprint's planned features to the next one, ensuring a rushed development would be avoided.

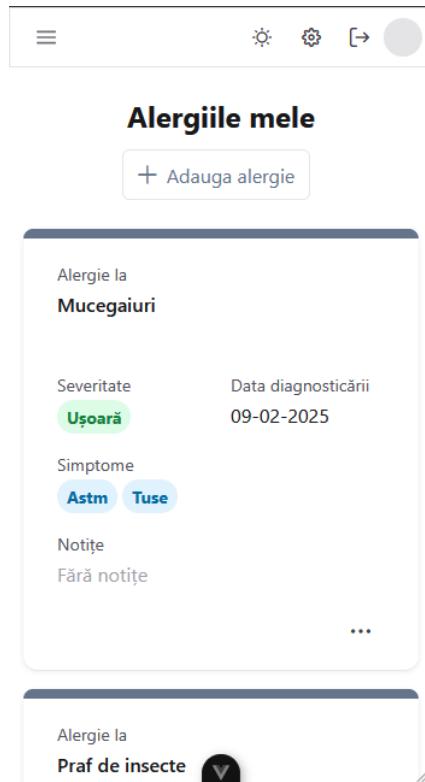
### **5.2.4 Requirements completed**

The main requirements completed in this sprint were:

- The system must store the data in a secure manner, ensuring only the patient and doctor can access the data.
- The system must allow patients to upload records in multiple file formats.
- The system should allow users to assign uploaded files for all record types.
- The system must allow the patient to add their own allergies.
- The system must allow the patient to add their own vaccinations.
- The system must allow patients to enter their current medication details.
- The system must allow patients to add new medication to their list.



(a) Desktop version



(b) Mobile version

Figure 5.6: Desktop and Mobile version of the Allergies page

## 5.3 Sprint #3

The third sprint focused on implementing the main dashboard and the vital signs functionality of the system, which would allow patients to add information such as weight or blood pressure in both a tabular and graphical format. Stakeholder feedback from previous sprints also had to be addressed, which included reworking some frontend elements to ensure consistency across application pages.

The features were chosen to be completed this sprint as their implementation would help build proficiency with the frameworks and libraries before tackling more complex functionality in subsequent sprints.

### 5.3.1 User interface rework

During a presentation of previous sprint results, the stakeholders identified inconsistencies in how information was presented across pages in the application, caused by the different information each record contained. To ensure a consistent view of medical records, Prime-Vue's Data View component was implemented. It provided both a grid and list visualisation option, while allowing for customisation of how individual records were displayed. The information displayed was also simplified, to ensure important details were conveyed without distractions.

An example of the updated allergies page can be seen in figure 5.7. The page now offers both a grid and list view, which can be toggled by the user, creating a more consistent view across different medical records.

### 5.3.2 Filtering and sorting

Stakeholder feedback also identified the need to filter and sort the information displayed in the application. This was implemented by using Vue's reactivity system, ref and computed functions. It allowed to create complex relationships, which would update variables based on given input or data change. An example of the sorting and filtering functionality can be seen below:

```
1  const filteredMedicine = computed(() => {
2      return props.medications.filter((medication) => {
3          return medication.name.toLowerCase().includes(searchQuery.value.
4              toLowerCase())
5      })
6  })
```

Listing 5.12: Medication Sorting and Filtering

```

1  const filteredAllergies = computed(() => {
2
3    if (selectedAllergens.value.length > 0) {
4      return props.allergies.filter((allergy) =>
5        allergy.allergens.some((allergen) => selectedAllergens.value.
6          includes(allergen))
7      )
8
9    if (selectedReactions.value.length > 0) {
10      return props.allergies.filter((allergy) =>
11        allergy.reactions.some((reaction) => selectedReactions.value.
12          includes(reaction))
13    )
14
15    if (selectedSeverities.value.length > 0) {
16      return props.allergies.filter((allergy) =>
17        selectedSeverities.value.includes(allergy.severity)
18      )
19    }
20
21    return props.allergies.filter((allergy) =>
22      allergy.allergens.some((allergen) =>
23        allergen.toLowerCase().includes(searchQuery.value.toLowerCase())
24      )
25    )
26  })

```

Listing 5.13: Allergies Sorting and Filtering

### 5.3.3 Vital signs monitoring

The vital signs feature was implemented to add and track basic health metrics such as weight, blood pressure or pulse over time. This addition allowed for the data to be displayed in both a tabular and graphical format to enable track progress over time. To display the data in a tabular and graphical view, the system uses PrimeVue's Data Table component and Chart.js library respectively. To further enhance the feature, current value indicators and trend markers were added to the component to provide a better user experience. Examples of how the Vitals page looks can be seen in figure 5.8 and 5.9.

Pagina principala Istoricul medical Cabinetul personal

## Alergiile mele

+ Adauga alergie

Caută alergie... Sorteaza după Alergeni Simptome Severitati

Alergie la	Praf de acarieni	Păr de animale	Pene de păsări
Severitate	<b>Ușoară</b>	Data diagnosticării 04-02-2025	
Simptome	<b>Urticarie</b> Rinită Astm		
Notte	Fără notite		

Alergie la	Latex	Cosmetice	Producție curățenie
Severitate	<b>Severă</b>	Data diagnosticării 06-02-2025	
Simptome	<b>Dermatită</b> Edem Roseată		
Notte	nu se poate lăsa!		

Alergie la	Mucegaiuri	Păr de animale	Praf de insecte	Pene de păsări	Alimente
Severitate	<b>Moderată</b>	Data diagnosticării 24-01-2020			
Simptome	<b>Urticarie</b> Eczemă Dermatită Rinită Conjunctivitate Astm				
Notte	asdasdasdasdasdasda				

Alergie la	Praf de acarieni
Severitate	<b>Moderată</b>
Simptome	<b>Eczemă</b>
Notte	asdasda

...

(a) Desktop version

Alergiile mele

+ Adauga alergie

Caută alergie... Sorteaza după Alergeni Simptome Severitati

Alergie la	Praf de acarieni	Păr de animale	Pene de păsări
Severitate	<b>Ușoară</b>	Data diagnosticării 09-07-2024	
Simptome	<b>Conjunctivitate</b>		
Notte	...		

Alergie la	Praf de acarieni	Păr de animale
Severitate	<b>Severă</b>	Data diagnosticării 09-07-2024
Simptome	<b>Conjunctivitate</b>	
Notte	...	

Alergie la	Păr de animale
Severitate	<b>Ușoară</b>
Simptome	<b>Urticarie</b>
Notte	...

...

(b) Mobile version

Figure 5.7: Desktop and Mobile version of the Allergies page 2nd version

Pagina principala Istoricul medical Cabinetul personal

### Semne vitale

+ Adauga semn vital nou

Valori curente		Valori istorice	
Greutate	80.5 kg	înălțime	180 cm
Tensiune arterială	120/80 mmHg	Puls	70 bpm
Nivelul de zahăr din sânge	90 mg/dL	BMI	24.8

Interval de referinta: 60 - 80 kg

Data adaugarii	Valoarea	Notite
12-03-2025	80.5 kg	...
10-03-2025	81.2 kg	...
07-03-2025	80.8 kg	...
03-03-2025	81.5 kg	...
27-02-2025	82 kg	...

(a) Desktop version

Greutate	80.5 kg	înălțime	180 cm
Tensiune arterială	120/80 mmHg	Puls	70 bpm
Nivelul de zahăr din sânge	90 mg/dL	BMI	24.8

**Valori istorice**

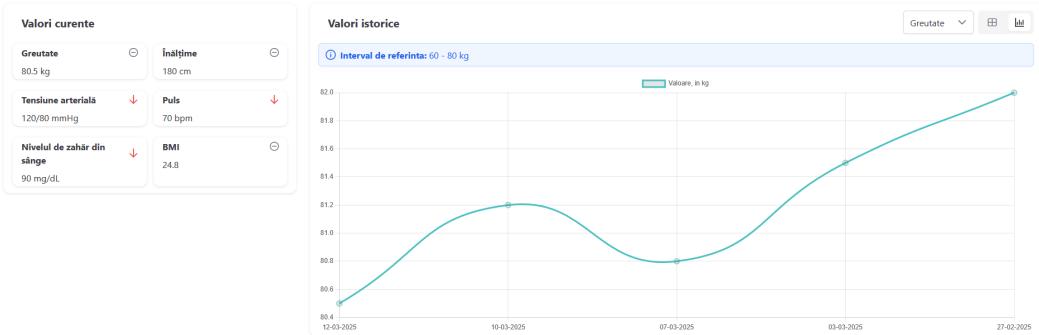
Interval de referinta: 60 - 80 kg

Data adaugarii	Valoarea	Notit
12-03-2025	80.5 kg	

(b) Mobile version

Figure 5.8: Desktop and Mobile version of the Vitals page

## Semne vitale



(a) Desktop version



(b) Mobile version

Figure 5.9: Desktop and Mobile version of the Vitals page, with Graph

### 5.3.4 Dashboard implementation

The dashboard feature was implemented to provide an overview of recent records added to the system. It was designed to present up to 5 recent records for each section, encouraging users to visit the respective page to view more records and details. The display format for each record was kept consistent with their individual pages.

Data Table from PrimeVue was used to create the individual tables for the dashboard sections. Each section was an individual and modular Vue component, which was then imported into the dashboard main file. An example of a section code can be seen below:

```
1 <template>
2   <Card :pt="cardStyles" class="h-full">
3     <template #title><h2 class="text-xl font-bold p-4">Semne vitale
4       recent adaugate</h2></template>
5     <template #content>
6       <DataTable :value="props.vitals" removableSort>
7         <Column field="name" header="Numele" sortable />
8         <Column field="value" header="Valoarea" sortable>
9           <template #body="{data}">
10             <span v-if="data.value">{{ data.value }} {{ data.unit }}</span>
11             <span v-else>{{ data.value_systolic }}/{{ data.
12               value_diastolic }} {{ data.unit }}</span>
13           </template>
14         </Column>
15         <Column field="date_recorded" header="Data adaugarii" sortable
16           >
17           <template #body="{data}">{{ data.original_date_recorded }}</
18             template>
19           </Column>
20         </DataTable>
21       </template>
22       <template #footer><RouterLink to="/vitale" class="p-
23         button-text">Vezi toate semnele vitale</RouterLink></template>
24     </Card>
25   </template>
```

Listing 5.14: Vital Signs Dashboard Section Component

Bine ai venit, Calin Corcimaru!

[Adauga un document nou](#) [Distribuie codul medicalului](#)

Istoric medical					
Lorem ipsum dolor sit, amet consectetur adipisciing elit. Asperiores ullam maiores et illo omnis? Quasi optio qui, soluta adipisci sed deserunt veniam labore atque perspicatis expedita sapiente quae at deleniti.					
Semne vitale recent adaugate					
Numele ↑↓	Valoarea ↑↓	Data adaugarii ↑↓			
Tensiune arterială	122/81 mmHg	22-02-2025			
Tensiune arterială	119/78 mmHg	25-02-2025			
Tensiune arterială	127/84 mmHg	28-02-2025			
Tensiune arterială	121/80 mmHg	01-03-2025			
Tensiune arterială	125/83 mmHg	04-03-2025			
<a href="#">Vezi toate semnele vitale</a>					
Medicamentele recent adaugate					
Numele ↑↓	Doza ↑↓	Frecventa ↑↓	Forma ↑↓	Data prescrierii ↑↓	Durata tratamentului ↑↓
Paracetamol	400mg	la nevoie	Comprimat	06-02-2025	30 zile
Suprastin	200mg	1 la fiecare 2 zi	Comprimat	26-02-2025	10 zile
Ibuprofen	200mg	2 pe zi	Capsulă	12-02-2025	10 zile
<a href="#">Vezi toate medicamentele</a>					
Analize laborator					
Lorem ipsum dolor sit, amet consectetur adipisciing elit. Asperiores ullam maiores et illo omnis? Quasi optio qui, soluta adipisci sed deserunt veniam labore atque perspicatis expedita sapiente quae at deleniti.					
Vaccinurile recent adaugate					
Numele ↑↓	Furnizorul ↑↓	Data primirii ↑↓			
Test	Test	04-03-2025			
COVID-19	Pfizer	5-02-2025			
Alergii recent adaugate					
Alergenii ↑↓	Reactii ↑↓	Severitate ↑↓			
Păr de animale	Urticarie	Ujară			
Praf de acarieni	Conjunctivită	Severă			

(a) Desktop version

### Semne vitale recent adaugate

Numele ↑↓	Valoarea ↑↓	Data adau
Tensiune arterială	122/81 mmHg	22-0
Tensiune arterială	119/78 mmHg	25-0
Tensiune arterială	127/84 mmHg	28-0
Tensiune arterială	121/80 mmHg	01-0
Tensiune arterială	125/83 mmHg	04-0

[Vezi toate semnele vitale](#)

(b) Mobile version

Figure 5.10: Desktop and Mobile version of the Dashboard page

### 5.3.5 Backend changes

Several database changes were implemented to support the new frontend functionalities. The updated backend diagram can be seen in figure 5.11. The main changes included:

- Addition of a new database table for vital signs, with a separate data type to standardise units and references ranges across values.
- Addition of a new date\_added fields to all database tables to differentiate between the clinical date it represents and the date it was added to the system, enabling the ability to sort and query the recently added records.

To accommodate the newly added table, changes were made to the vital sign API endpoints. Additional endpoints were also created for the dashboard sections to display their most recent results by using the new date\_added field. The dashboard endpoint can be seen below:

```
1 @app.get("/dashboard", response_model=UserDashboard)
2 async def get_dashboard(user_id: uuid.UUID = Depends(validate_session),
3                         session: Session = Depends(get_session)):
4     user = session.get(User, user_id)
5
6     if user_id != user.id:
7         raise HTTPException(status_code=403, detail="You do not have
8 permission to access this endpoint!")
9
10    newest_vaccines = session.exec(select(Vaccine).where(Vaccine.
11                                         user_id == user_id).order_by(col(Vaccine.date_added).desc()).limit
12                                         (5)).all()
13
14    # Similar code for allergies, medications and vitals...
15    # Response transformation code omitted for brevity
16
17    user_dashboard = UserDashboard(
18        id = user.id,
19        name = user.name,
20        vaccines = vaccines_response,
21        allergies = allergies_response,
22        vitals = healthdata_response,
23        medications = medications_response
24    )
25
26    return user_dashboard
```

Listing 5.15: Dashboard Endpoint

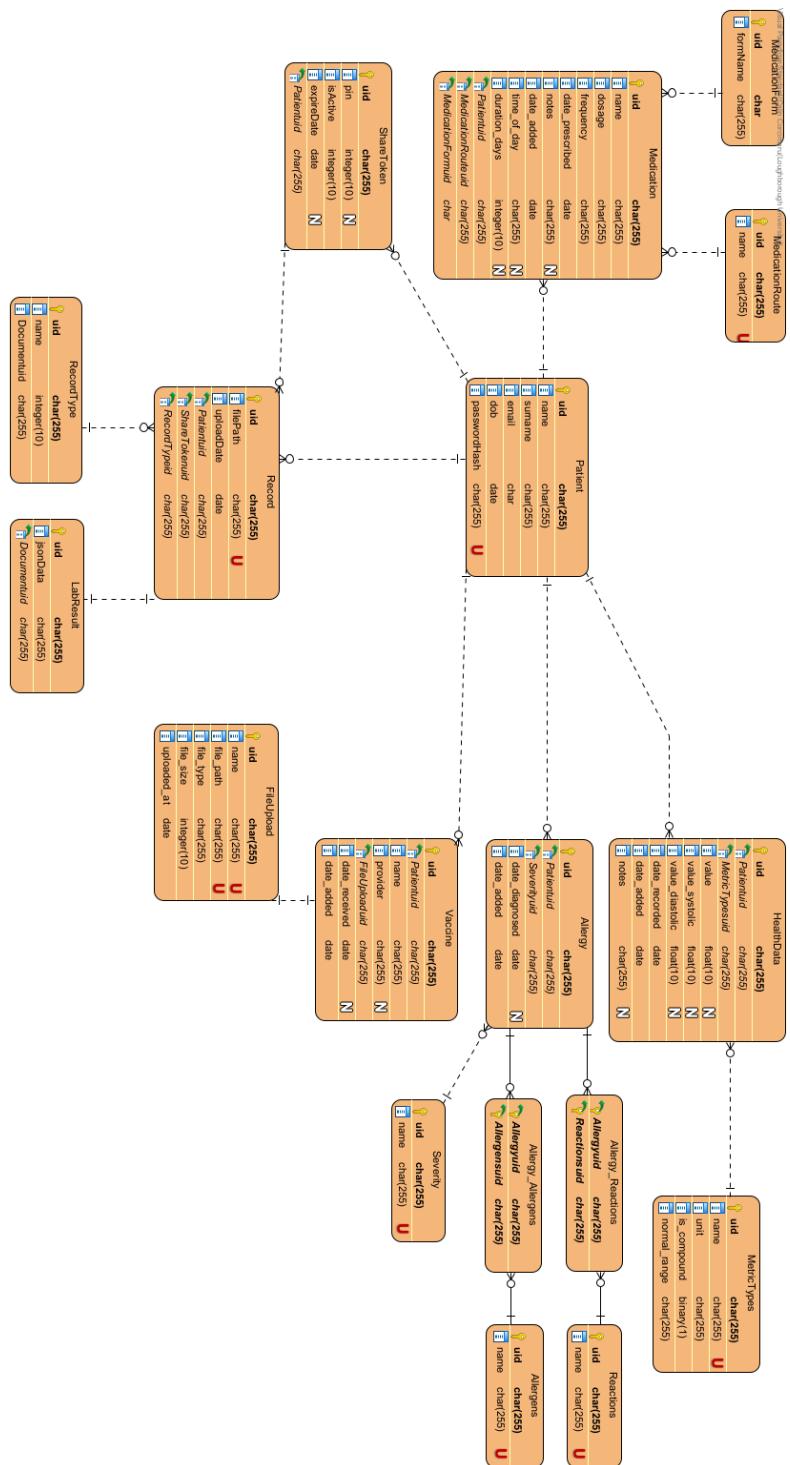


Figure 5.11: Updated Entity Relationship Diagram for Sprint #3

### **5.3.6 Challenges**

#### **Scope Creep**

Stakeholder feedback introduced at the start of the sprint led to additional work being done beyond the initial workload planned. Both backend and frontend changes were required to ensure adherence to the new requirements, sapping valuable time and resources from the main sprint objectives. This issue emphasises both the importance of frequent and clear communication with stakeholders, but also the need to establish a clear scope and priority for each sprint.

#### **Infrequent communication with stakeholders**

Infrequent stakeholder interaction was another issue encountered in this and previous sprints. The unavailability of some stakeholders due to busy schedules as doctors led to some decisions made that later required revision, adding additional workload. As such, more consistent communication with stakeholders was identified as a key improvement necessary for the success of subsequent sprints.

#### **Testing limitations**

The lack of experience with automated testing frameworks for the framework resulted in limited testing capability and coverage. Even though testing was still done, albeit manually, not all issues would be immediately caught. This results in intermittent bugs appearing during the development process, which sidetracked the development efforts from the sprint objectives. Additionally, due to time constraints, testing was usually left out at the end or not done at all, leading to even more issues arising unexpectedly in the future.

### **5.3.7 Requirements completed**

- The system must have a dashboard view which displays an overview of the most recent information added to the system (latest lab tests, doctor consultations, vaccinations etc).
- The system should display the health records in both a list or grid view.
- The system should allow the patient to enter vitals information, such as height, weight, blood pressure, etc.
- When multiple vital entries are made, the system could display a historical graph of the patient's vitals.

## 5.4 Sprint #4

The fourth sprint of the project focused on implementing the health records functionality, enabling users to add records such as lab tests, doctor consultations and imaging results. It addresses one of the core requirements of this application, which is to view and manage your medical history in one place. Additional feedback was received from stakeholders at the end of sprint #3, which included:

- Integration of reference ranges for vitals sign graphs
- Implementation of date filtering for vital signs data
- Adjustment of recent vital signs comparison logic to compare with normal ranges instead
- Refinement of dashboard display for allergies and vital signs sections: allergies with moderate and above severities and the most recent value for each vital sign type

### 5.4.1 Health Records

The medical history feature was implemented using PrimeVue's DataTable component to display the data in a tabular format. This allowed for a more structured view of the data, but also enabled the ability to sort and filter by column data thanks to DataTable's built-in functionalities. As such, it was possible to ensure an efficient organisation of data, with consistency across record types. An example of the health records page can be seen in figure 5.12.

Advanced filtering options were enabled through DataTable's filtering API, allowing for both global and column-specific filtering options. An example of a filter menu can be seen in figure 5.14.

Finally, to provide an even better user experience, the document management functionality was enhanced with the ability to display PDF files. This allowed patients to attach files to specific medical records for future reference. On desktop, the files were displayed using the browser's embedded PDF viewer, while on mobile the application would provide a download option to avoid compatibility issues. The code for the PDF viewer can be seen below, with a mobile and desktop example shown in figure 5.13.

```

1  <Dialog
2    v-model:visible="visible"
3    modal
4    header="Vizualizeaza fisierul"
5    class="w-full md:w-3/4 md:h-3/4"
6    @hide="emit('close')">
7    <div v-if="metadata?.file_type == 'application/pdf'>
8      <object
9        :data="`http://localhost:8000/files/medicalhistory/${props.
10       historyId}`"
11        type="application/pdf"
12        class="w-full h-[70vh] hidden md:block"/>
13      <div class="block md:hidden text-center mt-2">
14        <a
15          :href="`http://localhost:8000/files/medicalhistory/${props.
16       historyId}`"
17          target="_blank"
18          class="bg-blue-500 text-white px-4 py-2 rounded-md inline-
19          block mb-4">
20          Deschide certificatul PDF
21        </a>
22      </div>
23    </div>
24    <Image
25      v-else
26      :src="`http://localhost:8000/files/medicalhistory/${props.
27       historyId}`"
28      alt="Fisier consultatie"
29      class="w-full h-screen"
30      preview/>
31  </Dialog>
32 </template>

```

Listing 5.16: PDF viewer Function for Health

Pagina principala Istoricul medical Cabinetul personal

**Istoric Medical**

+ Adauga un istoric medical nou

Căuta...

Data efectuării	Descriere	Numele Doctorului	Locația	Categorie	Subcategoria	Notite	Optiuni
15-02-2025	Consultație anuală	Dr. Andreea Popescu	Clinica MedLife	Consultatie	Medicina Internă	Evaluare generală. Stare de sănătate bună.	...
20-01-2025	ECG	Dr. Mihai Ionescu	Spatialul Județean	Consultatie	Cardiologie	Ritm cardiac normal.	...
10-02-2025	Analize sângel	Dr. Elena Dumitrescu	Synevo	Laborator	Hematologie	Hemoglobină și leucocite în limite normale.	...
05-01-2025	RMN genunchi	Dr. Adrian Stancu	Regina Maria	Imagistica		Ruptură menisc medial.	...
01-03-2025	Evaluare dermatologică	Dr. Diana Vasilescu	Clinica DermaPro	Consultatie	Dermatologie	Verificare alunite. Nicio modificare suspectă.	...
25-02-2025	Radiografie toracică	Dr. Victor Popa	Spatialul Militar	Imagistica		Plămâni în limite normale.	...
05-03-2025	Test glicemie	Dr. Sorin Neagu	Central Medical Nova	Laborator	Endocrinologie	Valori normale de zahăr în sânge.	...
15-01-2025	Control oftalmologic	Dr. Ana Marin	Vista Vision	Consultatie	Oftalmologie	Acutitate vizuală normală.	...
28-03-2025	Testing	test	Spatialul Republican	Laborator	Biochimie	-	...

<< < > >>

(a) Desktop version



## Istoric Medical

+ Adauga un istoric medical nou

Căuta...

Data efectuării ↑ Descriere ↓ Num Doc

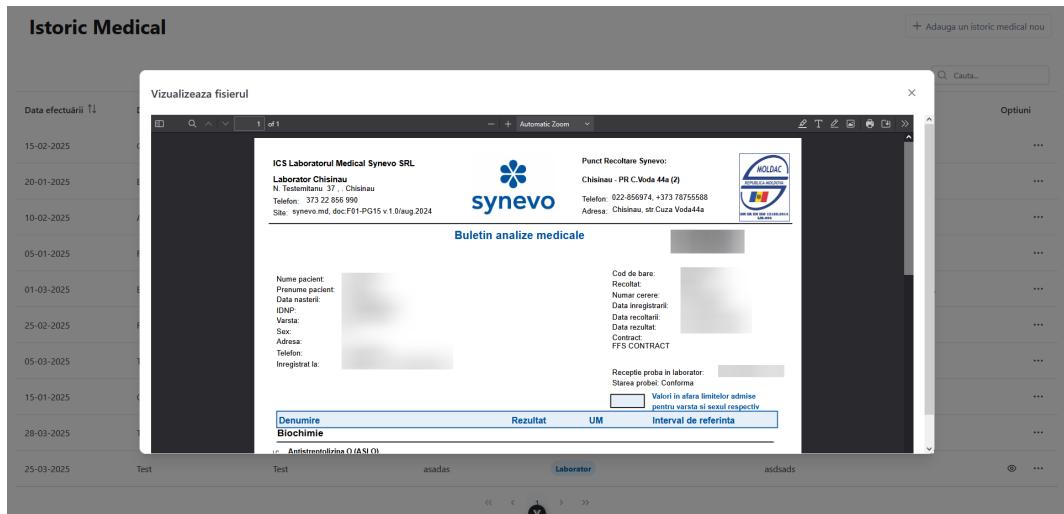
15-02-2025 Consultație anuală Dr. A Popescu

20-01-2025 ECG Dr. M Ionescu

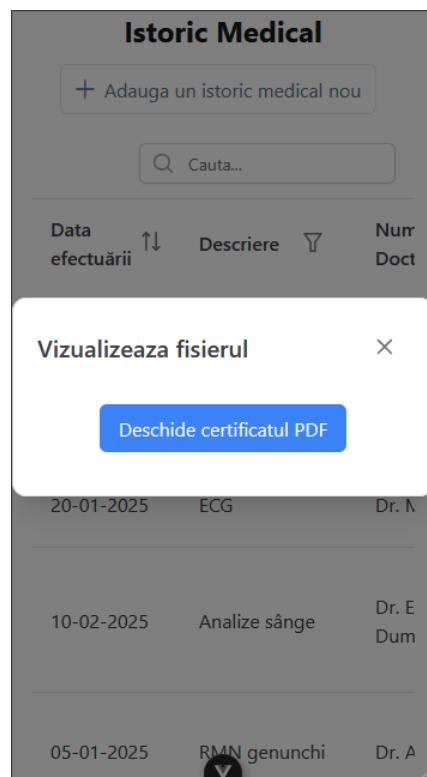
10-02-2025 Analize sângel Dr. E Dumitrescu

(b) Mobile version

Figure 5.12: Desktop and Mobile version of the History page



(a) Desktop version



(b) Mobile version

Figure 5.13: Desktop and Mobile version of the PDF viewer

The screenshot shows a web-based medical record system. At the top, there are navigation links: 'Pagina principală', 'Istoricul medical' (with a dropdown arrow), and 'Cabinetul personal' (also with a dropdown arrow). On the right side of the header are icons for settings, search, and refresh.

The main title is 'Istoric Medical'. In the top right corner, there is a button '+ Adauga un istoric medical nou' and a search bar with the placeholder 'Cauta...'. Below the title, there is a small note: '+ Adauga un istoric medical nou'.

The main content area is a table listing medical records. The columns are: 'Data efectuării' (Date), 'Descriere' (Description), 'Numele Doctorului' (Doctor's Name), 'Locația' (Location), 'Categoria' (Category), 'Subcategoria' (Subcategory), 'Notite' (Notes), and 'Optiuni' (Options). A filter modal is open over the table, specifically for the 'Numele Doctorului' column. The modal has a dropdown menu set to 'Contains' with the value 'Dr. Elena Dumitrescu'. It also includes a search input 'Caută după descriere', a 'Clear' button, and an 'Apply' button.

Data efectuării ↑↓	Descriere	Numele Doctorului	Locația	Categoria	Subcategoria	Notite	Optiuni
15-02-2025	Consultație anuală	Contains Dr. Elena Dumitrescu	Clinica MedLife	Consultație	Medicina Internă	Evaluare generală. Stare de sănătate bună.	...
20-01-2025	ECG		Spatialul Județean	Consultație	Cardiologie	Ritm cardiac normal.	...
10-02-2025	Analize sângel	Dr. Elena Dumitrescu	Synevo	Laborator	Hematologie	Hemoglobină și leucocite în limite normale.	...
05-01-2025	RMN genunchi	Dr. Adrian Stancu	Regina Maria	Imagistică		Ruptură menisc medial.	...
01-03-2025	Evaluare dermatologică	Dr. Diana Vasilescu	Clinica DermaPro	Consultație	Dermatologie	Verificare alunite. Nicio modificare suspectă.	...
25-02-2025	Radiografie toracică	Dr. Victor Popa	Spatialul Militar	Imagistică		Plămâni în limite normale.	...
05-03-2025	Test glicemie	Dr. Sorin Neagu	Centrul Medical Nova	Laborator	Endocrinologie	Valori normale de zahăr în sânge.	...
15-01-2025	Control oftalmologic	Dr. Ana Marin	Vista Vision	Consultație	Oftalmologie	Acutitate vizuală normală.	...
28-03-2025	Testing	test	Spatialul Republican	Laborator	Biochimie	-	...

At the bottom of the table, there are navigation arrows: '<<', '<', 'V' (highlighted in blue), '>', and '>>'.

Figure 5.14: Filter Menu for Health Records

### 5.4.2 Backend changes

To accommodate the medical history feature, the database was extended with new tables to support record organisation, which include a core table for record metadata, a table for primary classification (into consultation, laboratory or imagining) and secondary tables for consultation specialties and test types. The new ERD diagram can be seen in figure 5.15.

Additionally, API endpoints were created to allow the user to add, edit and delete health records. To promote reusability within the system, the file management endpoints created in sprint #2 were reused for the medical history feature.

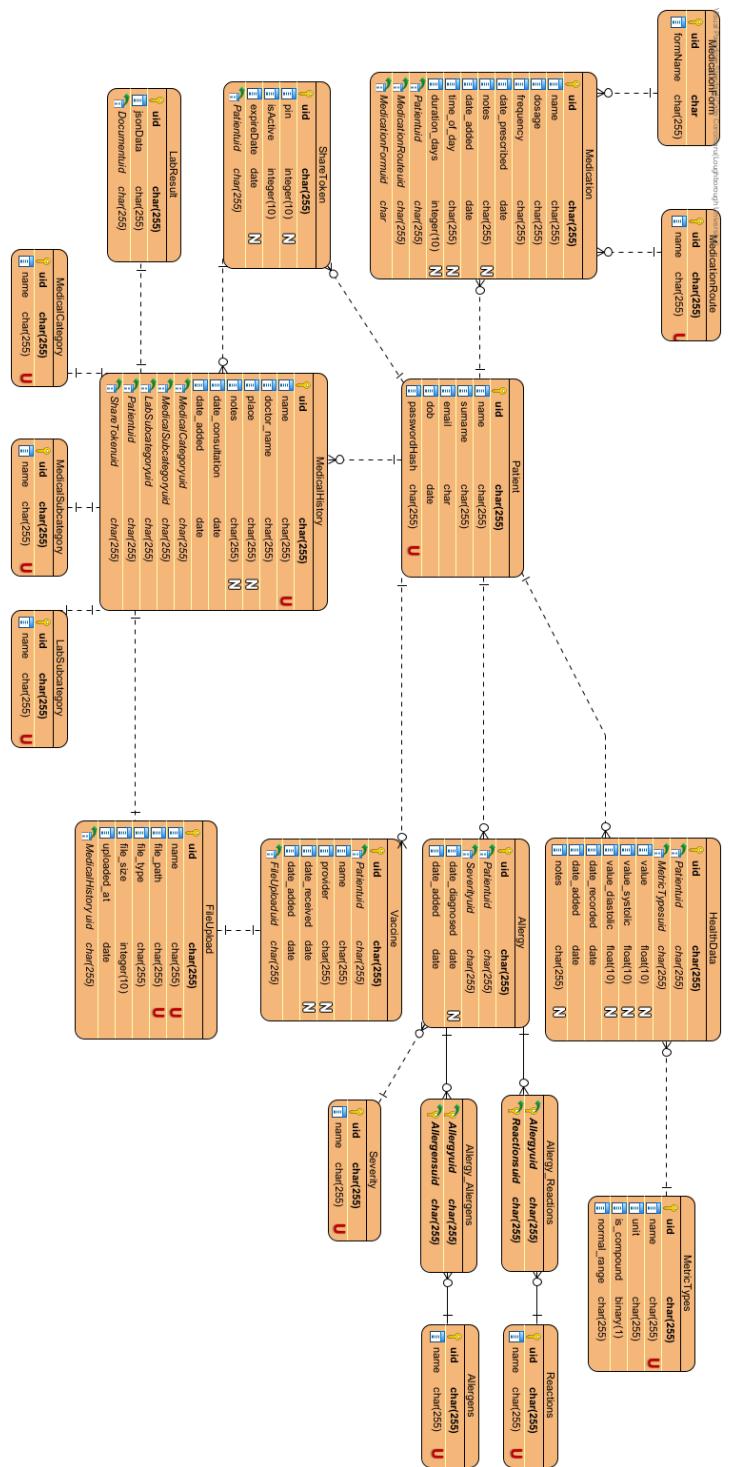


Figure 5.15: Updated Entity Relationship Diagram Sprint #4

### 5.4.3 Work on stakeholder feedback

Several changes to the user interface were implemented based on stakeholder feedback. The biggest change was the addition of ‘normal range’ boundaries to the graph views for some vital sign types, providing immediate context and feedback for vital sign results. Additionally, a date filter was introduced on the page, allowing users to filter the data based on selected date ranges. Finally, the computation logic for recent vital sign trends was changed to utilise their respective normal ranges to dictate their trend, instead of the previous value. Overall, these changes were made to improve the user experience and provide more context for the data displayed. Their implementation can be seen below in figure 5.16.

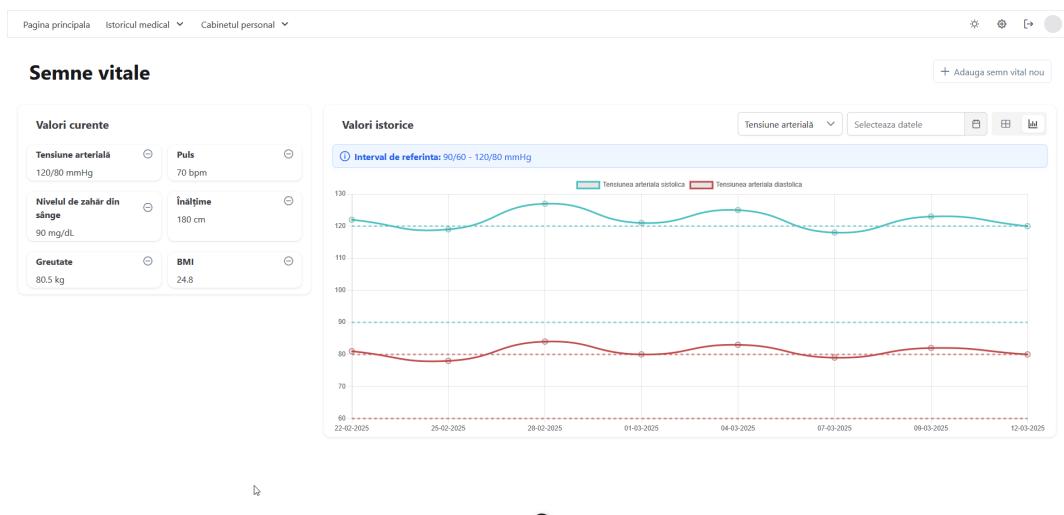


Figure 5.16: Updated Vitals View based on stakeholder feedback

The main dashboard view was also optimised to display more accurate and relevant information in some of the existing sections. For vital signs, only the most recent value for each type was now shown in the dashboard, alongside their trend indicator. For the allergies section, it was changed to only display records with a severity of ‘moderate’ and above to prioritise showing more severe (and more clinically important) cases of a patient’s allergies. The new dashboard can be seen in figure 5.17.

**Istoricul recent adaugat**

Numele	Doctor	Categorie	Subcategoria	Data consult
Testing	test	Laborator	Biochimie	28-03-2025
Control oftalmologic	Dr. Ana Marin	Consultatie	Oftalmologie	15-01-2025
Test glicemie	Dr. Sorin Neagu	Laborator	Endocrinologie	05-03-2025
Radiografie toracică	Dr. Victor Popa	Imagistică		25-02-2025
Evaluare dermatologica	Dr. Diana Vasilescu	Consultatie	Dermatologie	01-03-2025

[Vezi tot istoricul medical](#)

**Semne vitale recent înregistrate**

Numele	Valoarea	Adaugat	Trend
Greutate	80.5 kg	12-03-2025	⊖
Înălțime	180 cm	12-03-2025	⊖
Puls	70 bpm	12-03-2025	⊖
Nivelul de zahăr din sânge	90 mg/dL	12-03-2025	⊖
Tensiune arterială	120/80 mmHg	12-03-2025	⊖

[Vezi toate semnele vitale](#)

**Medicamentele recent adăugate**

Numele	Doza	Frecvența	Prescris	Durata
testing	1000mg	continuu	30-03-2025	14 zile
Paracetamol	400mg	la nevoie	06-02-2025	30 zile
Suprastin	200mg	1 la fiecare 2 zile	26-02-2025	10 zile
Ibuprofen	200mg	2 pe zi	12-02-2025	10 zile

[Vezi toate medicamentele](#)

**Analyze laborator**

Quasi optio qui, soluta adipisci sed deserunt veniam labore atque perspicatis expedita sapiente quae at defeniti.

**Vaccinurile recent adăugate**

Numele	Furnizorul	Data primirii
test	test	27-03-2025
test2	blablablablublu	22-03-2025
Test	test	04-03-2025

**Alergiile severe/moderate recent adăugate**

Alergenii	Reacții	Severitate
Praf de acarieni	Conjunctivită	Severă
Praf de acarieni	Eczemă + 3 reacții	Severă

[Vezi toate alergiile](#)

Figure 5.17: Updated Dashboard View based on stakeholder feedback

#### 5.4.4 Challenges

The primary challenge encountered during this sprint was, interestingly enough, unrelated to any technical implementation issues. Instead, an unexpected period of illness resulted in approximately one week lost of development time, leading to a reduced feature output for this sprint. As such, some of the unfinished worked had to be transferred to the next sprint's backlog.

#### 5.4.5 Requirements completed

- The system must allow the patient to specify and categorise the type of document they are uploading (lab test, doctor consultation, etc).
- The system must allow patients to upload their own medical records in a variety of formats (PDF, DOC, etc).
- The system must display the patient's history in a chronological order in the form of a timeline.
- When viewing doctor consultations, the system should divide them into categories based on the domain of the doctor (cardiology, neurology, etc).
- The system must allow the patient to add a description of the document they are uploading.
- The system should allow the patient to add a date for the document they are uploading.

- The system should allow the patient to add a location for the document they are uploading.
- The system should allow the patient to add the doctor name for the document they are uploading.
- The system should allow the patient to sort and filter the documents based on the type of document, date, location, and doctor name.

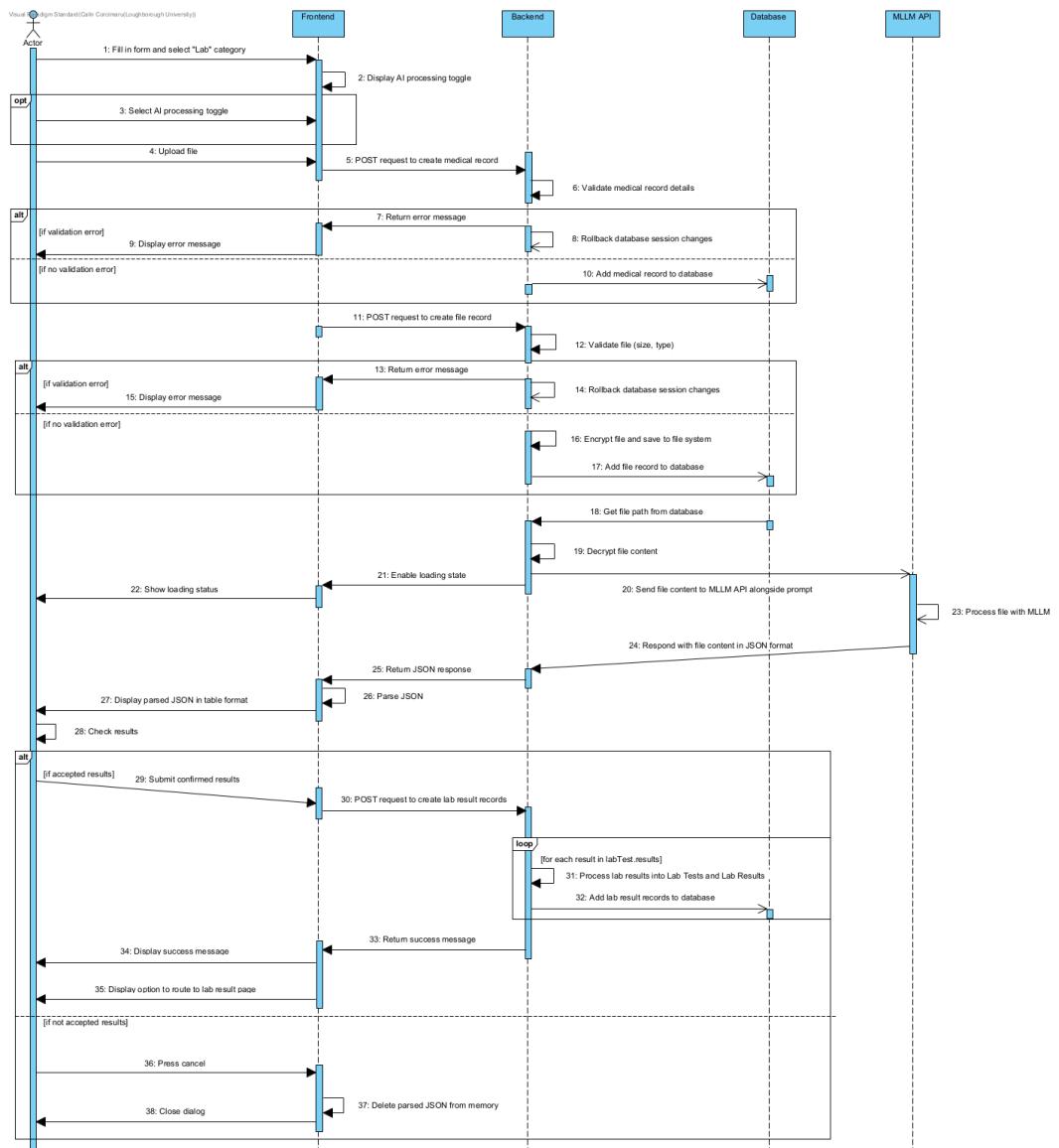
## 5.5 Sprint #5

The fifth sprint focused on implementing an automated lab test data extraction system that utilised a multimodal LLM to process uploaded lab result documents. This feature represented an incredible innovation within this application, as it enabled automatic extraction of test results from documents, with minimal manual intervention. Being a continuation of the work done in the previous sprint, some of the initial work required for the implementation of this feature was already completed. This allowed for easier and faster development of the lab extraction functionality, which was completed as per the initially agreed requirements and wireframes, with minor changes to how the lab results were displayed on the frontend.

The sequence diagram for the lab tests extraction process can be seen in figure 5.18. It displays how a user would interact with the system when uploading a lab result document, how the system processes the file and sends it to the MLLM API, and how the response is then received and displayed to the user.

### 5.5.1 Lab Tests Extraction

The implementation of initial lab extraction steps leveraged previously added components in sprint #4, to streamline the user experience and maintain a single entry point for document uploads with optional AI processing. As such, the ‘Add Health Record’ component and the ‘Laboratory’ health record type were reused to expand the document upload flow. Upon selecting the ‘Laboratory’ category, the user would get the option to toggle document AI processing, which will send the file to the MLLM API. This design choice closely matched the initial wireframe design for the ‘New Document’ dialog, which can be seen in figure B.6. As such, the updated ‘Add Health Record’ dialog can be seen in figure 5.19.



**Înregistrare medicală nouă**

Adaugă informații pentru o nouă înregistrare medicală.

\* Câmpurile marcate sunt obligatorii

Descriere *	<input type="text"/>
Numele doctorului *	<input type="text"/>
Locație	<input type="text"/>
Categorie *	Laborator
Subcategorie laborator	Hematologie
Extragă date laborator cu AI?	<input checked="" type="checkbox"/>
Data efectuării*	<input type="text"/> Data <input type="button" value="..."/>
Notite	<input type="text"/> Adaugă note sau observații
<input type="button" value="+ Alege documentul medical"/> No file chosen	
<input type="button" value="Anulează"/> <input type="button" value="Salvează"/>	

(a) Desktop version

**Înregistrare medicală nouă**

Adaugă informații pentru o nouă înregistrare medicală.

\* Câmpurile marcate sunt obligatorii

Descriere *	<input type="text"/>
Numele doctorului *	<input type="text"/>
Locație	<input type="text"/>
Categorie *	Laborator
Subcategorie laborator	Hematolo...
Extragă date laborator cu AI?	<input checked="" type="checkbox"/>

(b) Mobile version

Figure 5.19: Desktop and Mobile version of the new Add Document dialog

Google's Gemini 2.0 Flash was selected as the application's multimodal LLM, based on research of available free tiers with API access from table 3.4. However, it was noted that a local MLLM could've also been used for the project, a point which will be discussed in more detail in section 7.2. The MLLM implementation mainly involved using Prompt Engineering techniques to provide explicit instructions and format the output consistently. Prompting techniques from section 3.4.3 were used to write the following prompt, which is passed alongside the file to the API:

```

1 def extract_with_llm(file_content: bytes, file_type: str):
2     prompt = """You have been given a document that contains lab results
3         that is written in the Romanian language. Your job is to extract
4         all lab results from this document in JSON format with the
5         following fields: test_name, test_code, value, unit,
6         reference_range, method. Sometimes the code of the test will be in
7         the name itself, and it is your job to determine if the code is
8         there, for example in brackets or separated by a comma, and
9         separate the name and the code. Sometimes the lab result will not
10        have a method specified, and in that case you return an empty
11        string. The document is in Romanian, however the JSON keys should
12        be in English.
13
14 EXTREMELY IMPORTANT FORMATTING INSTRUCTIONS:
15 1. Return ONLY the raw JSON array
16 2. DO NOT use code blocks, backticks, or markdown formatting
17 3. DO NOT include ``json or `` anywhere in your response
18 4. DO NOT include any explanations or text before or after the JSON
19 5. Your response must start with the '[' character and end with the
20    ']' character
21 6. The output should be valid JSON that can be parsed directly
22 7. Use period (.) as the decimal separator, not comma (,)
23
24 Example of how your output should look, starting from the very first
25 character:
26 [{"test_name": "Hemoglobina", "test_code": "HGB", "value": "14.3", "unit": "mg/dL", "reference_range": "13.2-17.3", "method": "Chemiluminiscenta"}]
27
28 if there is no method specified, return an empty string:
29 [{"test_name": "Hemoglobina", "test_code": "HGB", "value": "14.3", "unit": "mg/dL", "reference_range": "13.2-17.3", "method": ""}]""]"""

```

Listing 5.17: Prompt for Lab Results Extraction

Upon receiving the results in a JSON format, the system would parse the response and display it in a table format to the user. Next, a validation step was added to ensure result accuracy before submitting the data to the backend, by utilising PrimeVue's DataTable row editing feature to enable the user to correct the results if necessary. This step was deemed necessary to ensure any possible MLLM hallucination could be corrected by the user. An example of how the extracted lab results are displayed can be seen in figure 5.20.



Figure 5.20: Desktop dialog validation of extracted lab results

The extracted lab results also leverage DataTable's dynamic column generation feature, ensuring the ability to accommodate any set of columns passed by the result data. This is important, as it may not be possible to predict the number of columns beforehand, but it also greatly simplifies the code needed to create the table. A code snippet of the DataTable can be seen below:

```

1 <DataTable
2   :value="extractionResult"
3   v-model:editingRows="editingRows"
4   class="w-full"
5   :rows="10"
6   editMode="row"
7   paginator

```

```

8   dataKey="test_name"
9   @row-edit-save="onRowEditSave"
10  v-model:filters="filters"
11  :globalFilterFields="['test_name', 'test_code']">
12  <template #header>
13    <h2 class="m-0">Analizati rezultatele extrase si schimbati daca
14    sunt gresite</h2>
15    <div class="flex justify-end">
16      <IconField>
17        <InputIcon>
18          <i class="pi pi-search" />
19        </InputIcon>
20        <InputText
21          size="small"
22          v-model="filters['global'].value"
23          placeholder="Cauta..."/>
24      </IconField>
25    </div>
26  </template>
27  <Column
28    v-for="col of columns"
29    :key="col.field"
30    :field="col.field"
31    :header="col.header">
32    <template #editor="{ data, field }">
33      <InputText v-model="data[field]" class="w-full" fluid />
34    </template>
35  </Column>
36  <Column
37    :rowEditor="true"
38    style="width: 10%; min-width: 8rem"
39    bodyStyle="text-align:center"/>
</DataTable>
```

Listing 5.18: Dynamic DataTable for Lab Results

### Backend changes

To accommodate the new lab extraction feature, the database was extended with two new tables: LabTest, which contained the core test metadata (name and code) and LabResult, which contained individual test results with values, units and reference ranges. A dual

table approach was chosen to prevent data duplication and facilitate historical tracking by allowing to associate multiple results to one test type. The updated database schema can be seen in figure 5.21. The new database schema includes a direct patient-result relationship to optimise queries for cases when retrieving all lab results for a patient is necessary, as is the case for the dashboard.

A code snippet of the endpoint that handles the processing of lab results can be seen below:

```

1  # Create the lab test records in the database
2  @router.post('/me/labtests/')
3  async def create_lab_tests(extraction_result: LabsCreate, user_id:
4      uuid.UUID = Depends(validate_session), session: Session = Depends(
5          get_session)):
6      medhistory = session.get(MedicalHistory, extraction_result.
7          medicalhistory_id)
8      user = session.get(User, user_id)
9
10     if not medhistory:
11         raise HTTPException(status_code=status.HTTP_404_NOT_FOUND,
12             detail="Medical history not found")
13
14     if medhistory.user_id != user_id:
15         raise HTTPException(status_code=status.HTTP_403_FORBIDDEN,
16             detail="Not authorised to access this medical history")
17
18     for lab_item in extraction_result.lab_tests:
19         lab_test = session.exec(select(LabTest).where(LabTest.name ==
20             lab_item.name)).first()
21
22         if not lab_test:
23             lab_test = LabTest(
24                 name = lab_item.name,
25                 code = lab_item.code,)
26             session.add(lab_test)
27             session.flush()
28
29         lab_result = LabResult(
30             value = lab_item.value,
31             is_numeric = check_is_numeric(lab_item.value),
32             unit = lab_item.unit,
33             reference_range = lab_item.reference_range,
```

```
28     date_collection = extraction_result.date_collection,
29     test = lab_test,
30     medicalhistory = medhistory,
31     user = user,
32     method = lab_item.method)
33
34     session.add(lab_result)
35     session.flush()
36
37     session.commit()
38
39     return {"status": status.HTTP_201_CREATED, "message": "Lab tests
  created successfully"}
```

Listing 5.19: Lab Results Endpoint

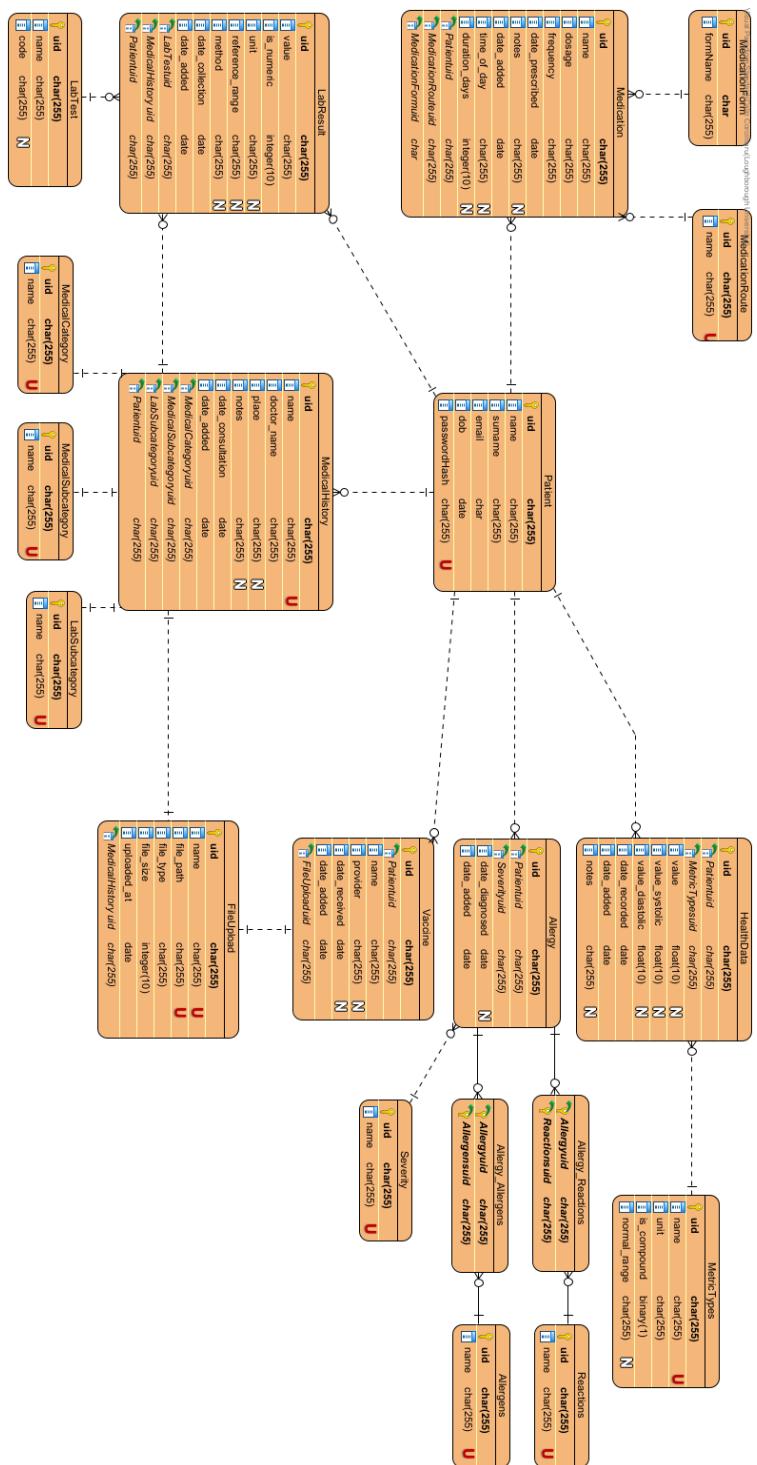


Figure 5.21: Updated Entity Relationship Diagram Sprint #5

### **5.5.2 Results visualisation**

The initial idea to display the extracted lab results was to reuse the existing design from the vitals page, developed in sprint #3, which would've also followed the initially agreed wireframes and design. However upon discussing with stakeholders, a new design was proposed which used a hierarchical format, with expandable rows for each test type. Alongside each test type a small graph would be displayed to show the evolution of test results over time, with a visual indicator on the side which would inform patients if the recent result was outside the reference range. An example of the new lab results page can be seen in figures 5.22 and 5.23.

This new design offered several advantages:

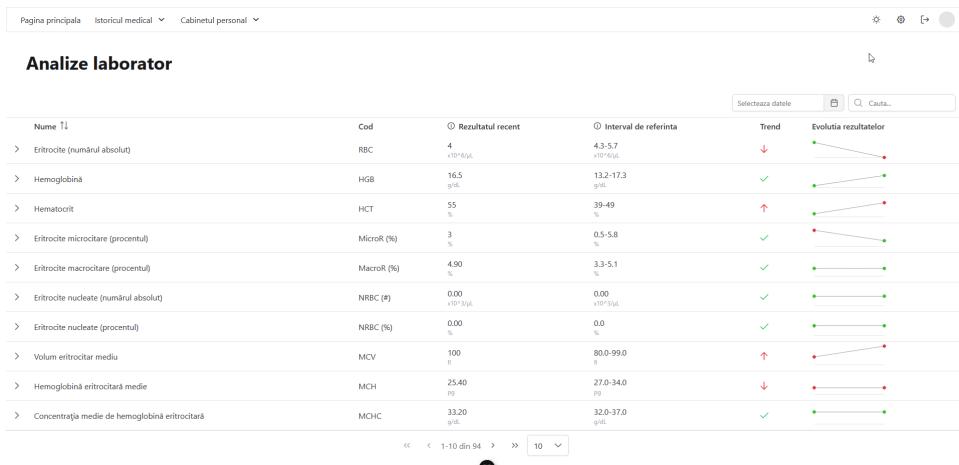
1. A compact presentation of large amounts of test results
2. Efficient filtering for existing test types
3. Integrated trend visualisation through sparklines and reference range indicator
4. A more responsive design for mobile users

### **5.5.3 Challenges encountered**

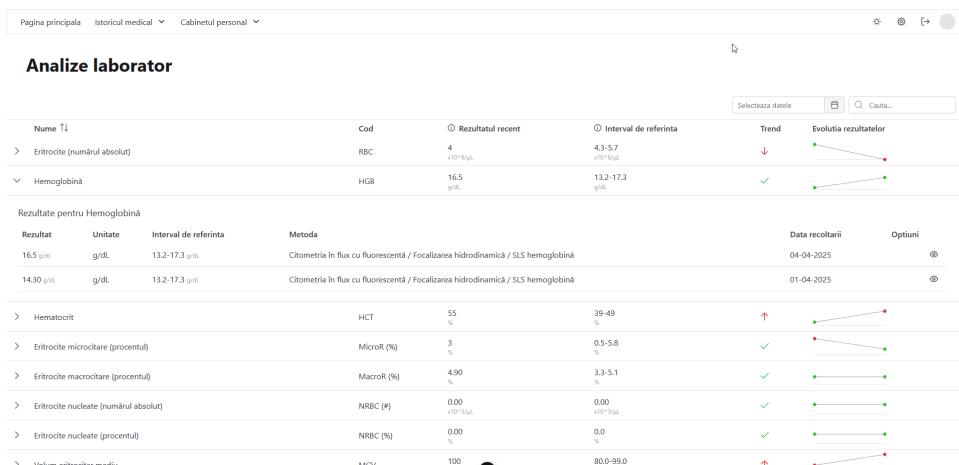
The primary challenge encountered this sprint was adapting to the mid-sprint lab result visualisation design change that was requested by the stakeholders. While it introduced additional design and research work, the agile approach taken and PrimeVue's extensive component library facilitated an effective implementation. In the end, the design actually streamlined the development as it leveraged the DataTable's capabilities for complex data presentation. Another challenge was the project's timeline constraints, which resulted in core functionality being prioritised over implementing more experimental features.

### **5.5.4 Requirements completed**

- The system must provide an overview of the patient history through 3 main sections: personal information, lab tests, and doctor consultations.
- The system must allow the user to view blood tests in a graphical format.
- The system must allow the user to view blood tests in a numerical, tabular format.
- For blood test results, the system should display the normal range values for each test.
- The system could allow the user to view the document from which the lab results were extracted.



(a) Desktop version

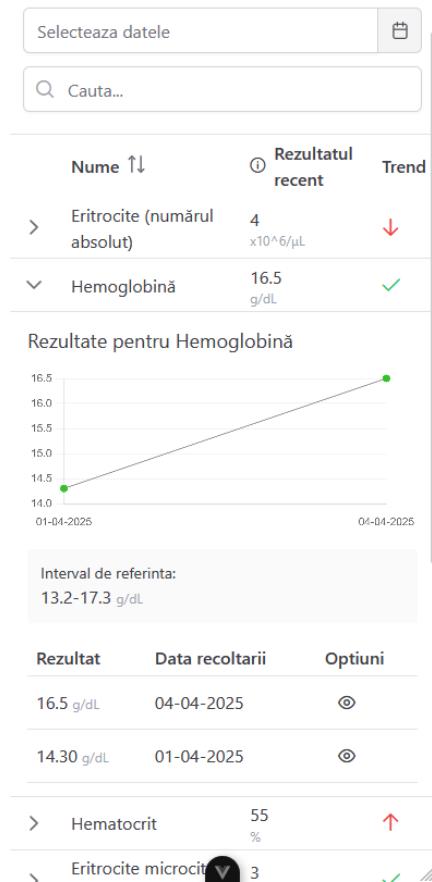


(b) Desktop version #2

Figure 5.22: Desktop version of the new Lab View page

Analize laborator			
	Număr ↑↓	Rezultatul recent	Trend
>	Eritrocite (numărul absolut)	4 x10^6/µL	⬇️
>	Hemoglobină	16.5 g/dL	✓
>	Hematocrit	55 %	⬆️
>	Eritrocite microcitare (procentul)	3 %	✓
>	Eritrocite macrocitare (procentul)	4.90 %	✓
>	Eritrocite nucleate (numărul absolut)	0.00 x10^3/µL	✓
>	Eritrocite nucleate (procentul)	0.00 %	✓
>	Volum eritrocitar mediu	100 fl	⬆️
>	Hemoglobină eritrocitară medie	25.40 pg	⬇️

(a) Mobile version



(b) Mobile version #2

Figure 5.23: Mobile version of the new Lab View page

## 5.6 Sprint #6

The final sprint focused on implementing the record sharing functionality, a critical feature that would enable patients to securely share their medical records with healthcare practitioners. This feature addressed one of the core objective of this project, which was to eliminate information siloing across Moldova's multiple healthcare institutions. Instead, this feature would empower the user, creating a patient-controlled approach that complements existing institutional systems. It also represented a consolidation of previously implemented features, leveraging existing components and endpoints to display records to healthcare practitioners.

### 5.6.1 Share links

During discussion with stakeholders, links were quickly identified as the preferred method of sharing medical information. Links were chosen due to their accessibility, as most doctors in Moldova have access to a computer at their desk. Similarly, links can be very versatile in how they are communicated: via email, dictated or even generated as a QR code.

To ensure a safe sharing process, the mechanism was designed with three security layers in mind. The first layer would be a random 8-character alphanumeric code, implemented to facilitate easy sharing while maintaining reasonable security. An example of a share link would be <https://example.com/share/7zSaQbBd>, with the code for the link generation and token model available below:

```
1  # Helper function to generate random strings for share codes
2 def generate_random_string(length: int) -> str:
3     characters = string.ascii_letters + string.digits
4     return ''.join(secrets.choice(characters) for _ in range(length))
5
6 # Share Token model for database
7 class ShareToken(SQLModel, table=True):
8     id: uuid.UUID = Field(default_factory=uuid.uuid4, primary_key=True)
9     share_code: str = Field(index=True, unique=True, default_factory=
10         lambda: generate_random_string(8))
11     expiration_time: datetime
12     created_at: datetime = Field(default_factory=datetime.now)
13     hashed_pin: str
14     shared_items: dict = Field(default_factory=dict, sa_column=Column(
15         JSON))
16     user_id: uuid.UUID = Field(foreign_key="user.id")
17     user: "User" = Relationship(back_populates="share_tokens")
```

Listing 5.20: Share Link Generation

The second layer was a configurable expiration period, which by default was 1 hour but could be set to a maximum of 24 hours. It was implemented to limit any potential exposure of the shared information, aligning with the single-use nature of the links during consultations. Finally, the third layer was a user-defined 6-character PIN, set during the initial steps of the share process. The PIN would be hashed and stored in the database for increased security. When accessed, the system first validates the link's expiration time, then prompts for the correct PIN before displaying any medical data. The verification endpoint is shown below:

```

1  @router.post("/share/{share_code}/verify", status_code=status.
2      HTTP_200_OK, response_model=ShareItemsResponse)
3  @limiter.limit("5/minute")
4  async def verify_share_token(
5      request: Request, share_code: str, pin: Annotated[str, Body(embed=
6          True)],
7      session: Session = Depends(get_session)):
8      share_token = session.exec(select(ShareToken).where(ShareToken.
9          share_code == share_code)).first()
10
11     if not share_token:
12         raise HTTPException(status_code=status.HTTP_404_NOT_FOUND,
13             detail="Share token not found")
14
15     if not verify_hash(pin, share_token.hashed_pin):
16         raise HTTPException(status_code=status.HTTP_403_FORBIDDEN,
17             detail="Invalid PIN")
18
19     user = session.exec(select(User).where(User.id == share_token.
20         user_id)).first()
21     user_data = {"name": user.name, "dob": user.dob.strftime("%d-%m-%Y")
22         } if user.dob else None}
23     items_data = get_item_data(share_token.shared_items, session)
24
25     return ShareItemsResponse(
26         expiration_time=share_token.expiration_time,
27         patient=user_data,
28         items=items_data)

```

Listing 5.21: Share Link Endpoint

The whole record sharing flow is illustrated in sequence diagrams 5.24 and 5.25.

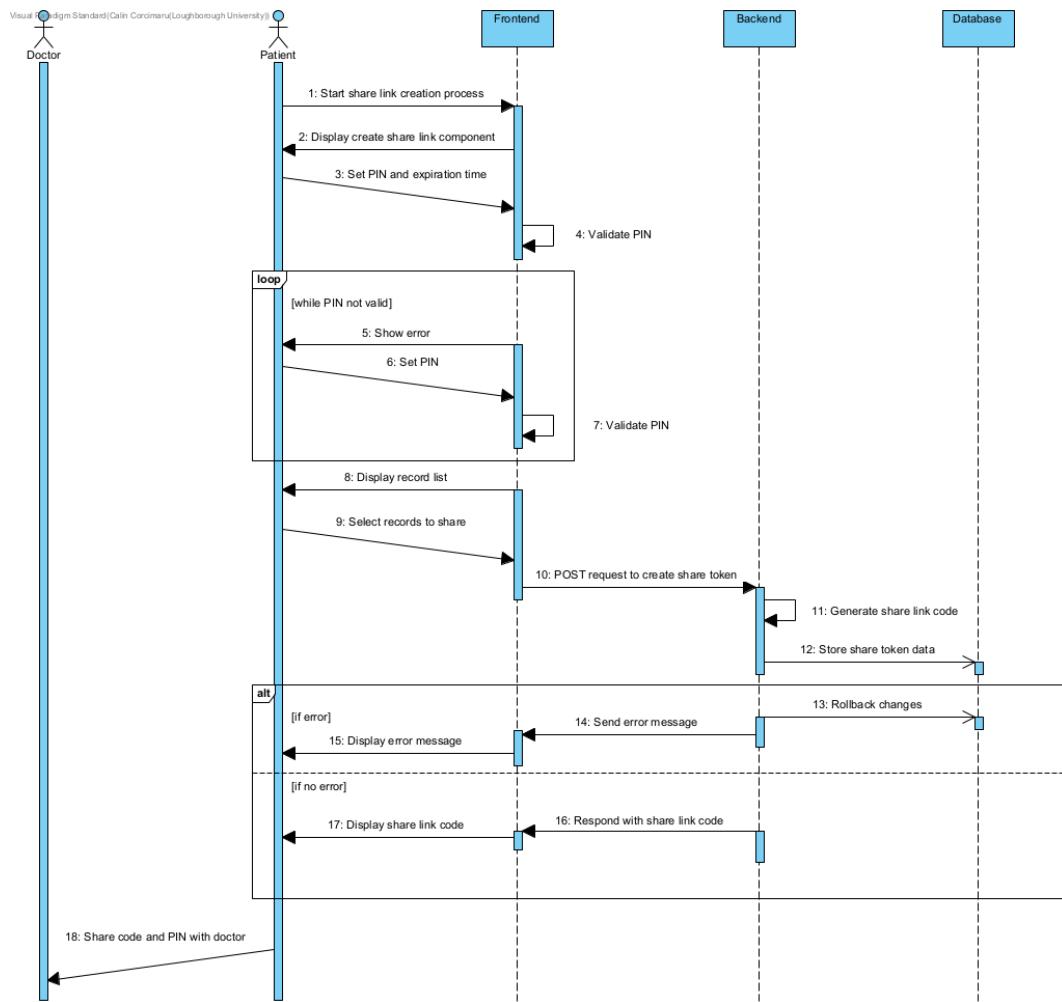


Figure 5.24: UML Sequence Diagram — Creating share link

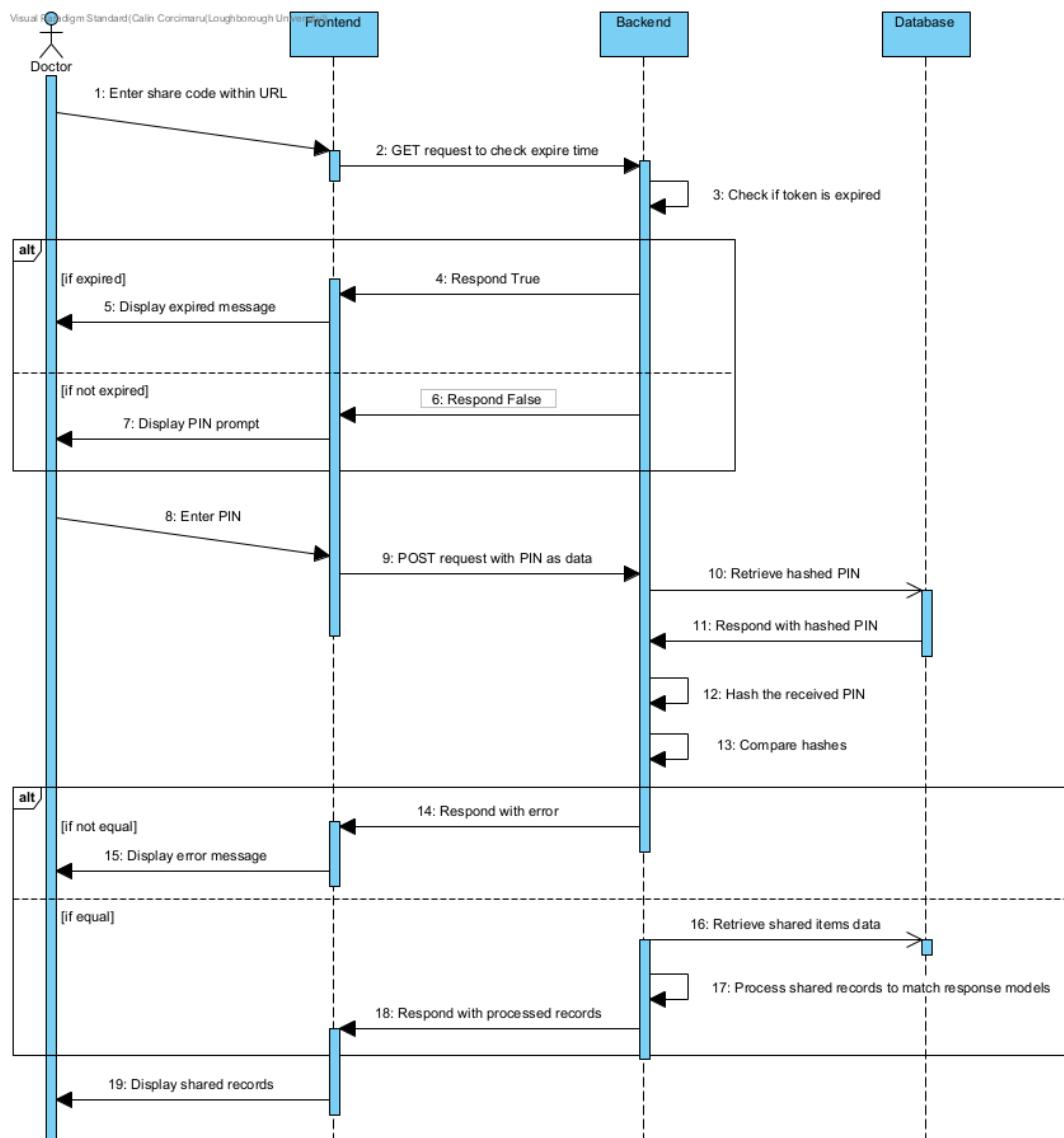


Figure 5.25: UML Sequence Diagram — Accessing shared link

### 5.6.2 User interface implementation

Based on initial wireframes designs, the link creation component was located in the dashboard. To ensure a streamlined user experience, the share creation process was structured as an easy to follow and guided three-step process. The first step involved configuring the expiration time and setting the PIN, which can be seen in figure 5.26. The second step allowed users to choose the specific records to share with healthcare practitioners, displayed in figure 5.27. The final step confirmed the share link creation and displayed the generated link, as shown in figure 5.28.

The screenshot shows a modal window titled 'Creeaza un link de partajare'. It displays a three-step process: 1. Creeaza link-ul de partajare, 2. Alege ce vrei sa partajezi, and 3. Partajeaza link-ul cu doctorul. The current step is 1. The sub-step 'Alege un PIN pentru link-ul de partajare' is active, showing a 6-field input field for a PIN. Below it, a note says 'PIN-ul trebuie sa contine 6 caractere'. A message below states 'Acest PIN va fi folosit pentru a accesa link-ul de partajare.' The sub-step 'Alege duratia link-ului de partajare' is shown below, with 'Ore' selected and three options: '1 ora', '2 ore', and '30 min'. A 'Next →' button is at the bottom right.

Figure 5.26: 1st step of share link creation

The frontend counterpart to the security measures was implemented next. As described in the previous section, the system would check the validity of the share link and if valid, would prompt the user to enter the PIN code. Both valid and invalid link outcomes can be seen in figures 5.30 and 5.29.

The actual results page was developed last. One challenge quickly encountered was how to display the shared information in a single page. The amount of information displayed would not work with previously used methods such as PrimeVue's DataView or DataTable. Instead, TabView was used to support multiple record types onto one page.

Creeaza un link de partajare

- 1 Creeaza link-ul de partajare
- 2 Alege ce vrei sa partajezi
- 3 Partajeaza link-ul cu doctorul

Ultimul an     Ultimele 6 luni     Ultimele 3 luni     Selecteaza datele   

Tip	Numar de elemente	Elemente selectate	
<input checked="" type="checkbox"/> Semne vitale	21	21	>
<input checked="" type="checkbox"/> Medicamente	7	7	>
<input checked="" type="checkbox"/> Vaccine	8	8	>
<input checked="" type="checkbox"/> Alergii	4	4	>
<input checked="" type="checkbox"/> Istoric medical	14	14	>
<input checked="" type="checkbox"/> Rezultate laborator	105	105	>

Back   

Figure 5.27: 2nd step of share link creation

Creeaza un link de partajare

- 1 Creeaza link-ul de partajare
- 2 Alege ce vrei sa partajezi
- 3 Partajeaza link-ul cu doctorul

**Link-ul de partajare a fost creat cu succes!**  
Poți transmite aceste informații doctorului tău pentru a accesa datele tale medicale

**Link de acces:**  
<http://localhost:5173/share/BQTvc1FK>

**PIN de acces:**  
1 1 1 1 1 1

① Link-ul va expira la: 23 Apr 2025 - 23:20

Trimite pe email     Genereaza QR code

Back

Figure 5.28: 3rd step of share link creation

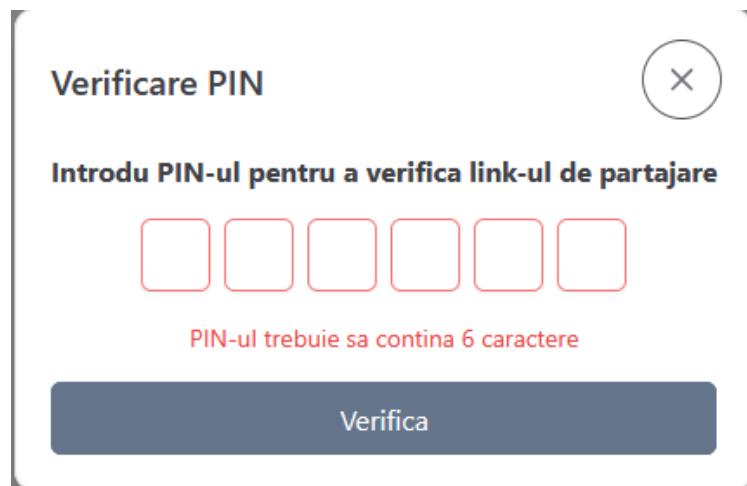


Figure 5.29: PIN prompt dialog



Figure 5.30: Expired link dialog

A combination of TabView and DataTable was used to organise the different record categories — the former would allow the user to switch different tabs representing different records (Vaccines, Allergies, etc), while the latter would contain and display the actual record data within each tab. This ensured that the user could navigate between different records without requiring a page load or additional API calls, which also greatly simplified the development process. The tabs and table columns were dynamically created based on the data received from the backend to accommodate any combination of record data.

To further expand the user experience on the shared page, basic user information and a countdown timer was added at the top of the page. An example of share page can be seen in figure 5.31.

Date pacient							
		00:24:19					
		Semne vitale	Medicamente	Vaccinuri	Alergii	Istoric medical	Analize de laborator
>	Nume	Cod	Rezultatul recent	Interval de referinta	Trend	Evolutia rezultatelor	
>	Acid folic		8.6 mg/mL	3.0-17.0 mg/mL	✓	•	
>	Vitamina B12		545 pg/mL	193.0-982.0 pg/mL	✓	•	
>	Insulina		4.6 µU/mL	2.6-24.9 µU/mL	✓	•—————•	
>	Feritina		176 ng/mL	28-365 ng/mL	✓	•	
>	Triiodtironina liberă	FT3	2.74 pg/mL	2.0-4.4 pg/mL	✓	•	
>	Tiroxina liberă	FT4	1.51 ng/dL	0.93-1.7 ng/dL	✓	•	
>	Hormonul tireotrop	TSH	2.42 µU/mL	0.27-4.20 µU/mL	✓	•	
>	Prolactina	PRL	409 mIU/L	53.0-360.0 mIU/L	↑	•	
.							

Figure 5.31: Share page — Desktop version

### 5.6.3 Backend

A new table was created in the database to store the share tokens, which contained metadata such as the hashed PIN, share code, expiration time and the shared items. Two different methods of storing the shared items were considered: a new table containing the record type and its respective UUID, or using a JSON field to store the shared records in the token table. The first method was quickly dismissed due to complex processing and database queries needed to be executed to retrieve and process the items into their response models. Instead, the second method was chosen, as it offered several advantages:

1. Reduced processing overhead and number of database queries

2. Storage of records in a ready-to-use format (response model)
3. Simplified record retrieval process
4. Future compatibility with new record types

The updated ERD can be seen in figure 5.32.

#### **5.6.4 Challenges encountered**

Two notable challenges influenced this sprint's development process. Firstly, the approaching project deadline resulted in a shift in focus towards core functionality and away from additional features or refinements. This particularly affected the user interface of the share page, which was simplified to accommodate the time constraints.

Multilingual conflicts was another challenge encountered. The combination of a Romanian frontend with an English-based database and codebase created name inconsistencies, which were especially noticeable in dynamically generated content such as table headers. This discrepancy stemmed from developing the application within a English-based academic context, while targeting a Romanian-speaking audience. While acceptable for a prototype implementation, a production system would require comprehensive internationalisation support to ensure a consistent user experience.

#### **5.6.5 Requirements completed**

- The system must allow the patient to generate a shareable link to provide access to their medical records.
- When creating the shareable link, the system must allow the patient to set an expiration date for the link.
- When creating the shareable link, the system should allow the patient to set an access password for the link.
- When creating the shareable link, the system should allow the patient to select which records to share with the doctor.

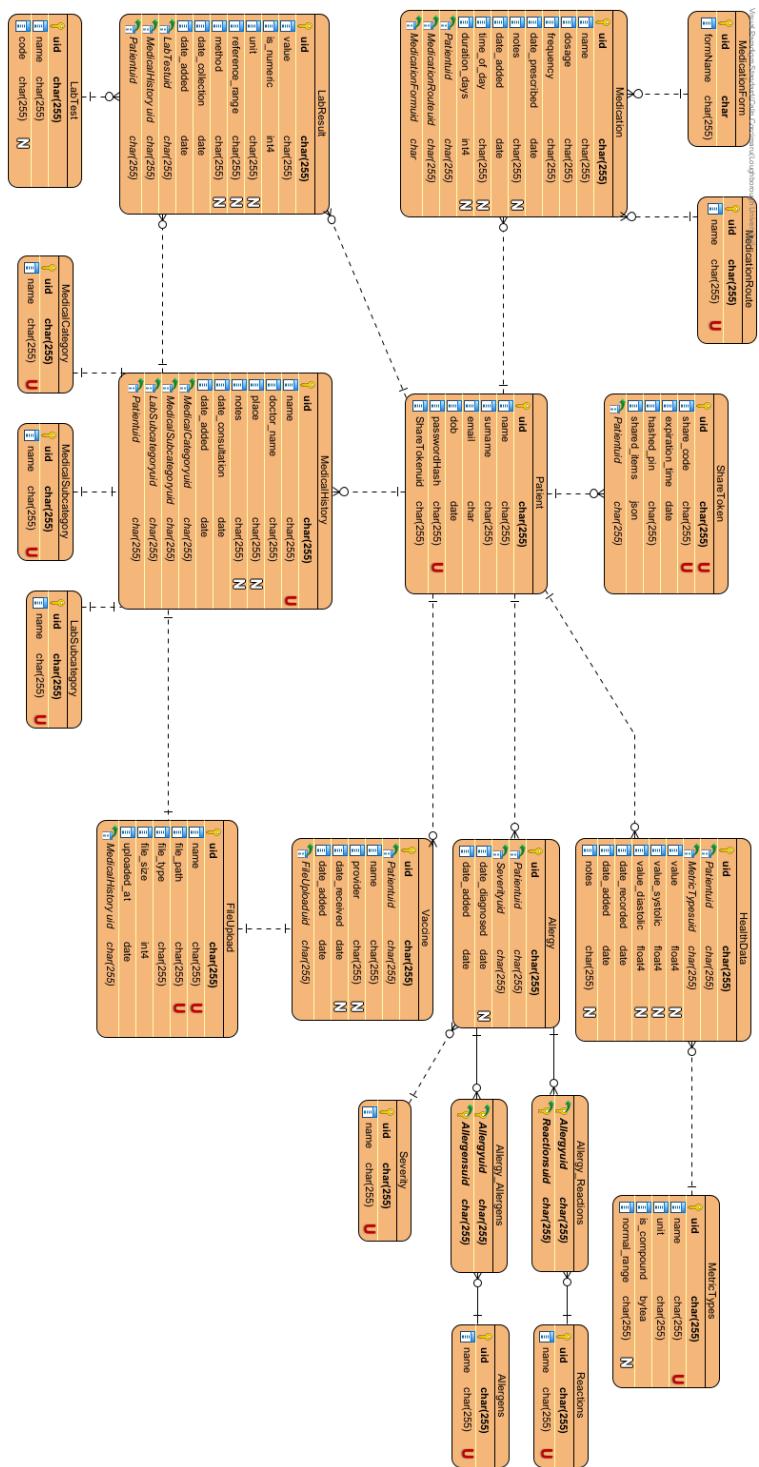


Figure 5.32: Updated Entity Relationship Diagram Sprint #6

# Chapter 6

## Evaluation

This chapter will discuss the evalution and testing of the system and its components. The next sections will cover the evaluation of the frontend, backend and the LLM used. This chapter will then conclude with a discussion on the client feedback received at the end of the project.

### 6.1 Frontend Testing

Frontend testing was mainly conducted through manual testing, done at the same time as the development of said components. While initially the plan was to use automated testing, the time constraints, complexity of components and lack of experience with any testing frameworks led to the decision to use manual testing instead.

To ensure some level of robustness in the testing procedure, a checklist or a list of test cases was created for every created component. After finishing development, this list would then be used to ensure the component wasa working as intended. To have a more structured approach, the test cases were usually written using the Gherkin syntax, which follows the Given-When-Then format, allowing for better definition of the test cases (The Cucumber Open Source Project, 2025).

An example of some test cases for the selection of records to be shared in a share link is shown below:

```
1  Feature: Record selection for share link
2
3  Scenario: User selects all record types to be shared
```

```

4      Given the user is on the share link page
5      When the user selects the top checkbox to select all record
6      types
7          Then all record types should be selected for sharing
8          And the all children checkboxes should be selected as well
9
10     Scenario: User selects specific record types to be shared
11         Given the user is on the share link page
12         When the user selects a specific record type to be shared
13         Then only the selected record types should be selected for
14         sharing
15         And the respective children checkboxes should all be selected
16         as well
17
18     Scenario: User selects specific children records to be shared
19         Given the user is on the share link page
20         When the user selects specific children records to be shared
21         Then only the selected children records should be selected for
22         sharing
23         And the respective parent checkbox should not be selected
24         And the top child checkbox should not be selected

```

Listing 6.1: Test cases for selecting records to be shared in a share link

And the respective frontend result of these test cases can be seen in figures 6.1, 6.2 and 6.3. The first figure shows the selection of all record types, the second figure shows the selection of specific record types and the third figure shows the selection of specific children records.

**Creeaza un link de partajare**

1 Creeaza link-ul de partajare    2 Alege ce vrei sa partajezi    3 Partajeaza link-ul cu doctorul

Ultimul an     Ultimele 6 luni     Ultimele 3 luni     Selecteaza datele   

Tip	Numar de elemente	Elemente selectate
<input checked="" type="checkbox"/> Semne vitale	21	21 >
<input checked="" type="checkbox"/> Medicamente	7	7 >
<input checked="" type="checkbox"/> Vaccine	5	5 <

Cauta...

name	provider	certificate	original_date_received
Test Vaccine File	Test Provider	true	16-04-2025
Test vaccine	Test provider	false	04-03-2025
COVID-19 Dose 3	Pfizer	false	15-02-2025
COVID-19 Dose 2	Pfizer	false	23-02-2025
COVID-19 Dose 1	Astrazeneca	false	09-02-2025

<< < 1-5 din 5 > >> 10 <

Figure 6.1: Test Scenario #1

**Creeaza un link de partajare**

1 Creeaza link-ul de partajare    2 Alege ce vrei sa partajezi    3 Partajeaza link-ul cu doctorul

Ultimul an     Ultimele 6 luni     Ultimele 3 luni     Selecteaza datele   

Tip	Numar de elemente	Elemente selectate
<input type="checkbox"/> Semne vitale	21	0 >
<input type="checkbox"/> Medicamente	7	0 >
<input checked="" type="checkbox"/> Vaccine	5	5 ▾

Cauta...

	name	provider	certificate	original_date_received
<input checked="" type="checkbox"/>	Test Vaccine File	Test Provider	true	16-04-2025
<input checked="" type="checkbox"/>	Test vaccine	Test provider	false	04-03-2025
<input checked="" type="checkbox"/>	COVID-19 Dose 3	Pfizer	false	15-02-2025
<input checked="" type="checkbox"/>	COVID-19 Dose 2	Pfizer	false	23-02-2025
<input checked="" type="checkbox"/>	COVID-19 Dose 1	Astrazeneca	false	09-02-2025

<< < 1-5 din 5 > >> 10 ▾

Figure 6.2: Test Scenario #2

**Creeaza un link de partajare**

1 Creeaza link-ul de partajare    2 Alege ce vrei sa partajezi    3 Partajeaza link-ul cu doctorul

Tip	Numar de elemente	Elemente selectate
<input type="checkbox"/> Semne vitale	21	0 >
<input type="checkbox"/> Medicamente	7	0 >
<input type="checkbox"/> Vaccine	5	4 ▼

	name	provider	certificate	original_date_received
<input type="checkbox"/>	Test Vaccine File	Test Provider	true	16-04-2025
<input checked="" type="checkbox"/>	Test vaccine	Test provider	false	04-03-2025
<input checked="" type="checkbox"/>	COVID-19 Dose 3	Pfizer	false	15-02-2025
<input checked="" type="checkbox"/>	COVID-19 Dose 2	Pfizer	false	23-02-2025
<input checked="" type="checkbox"/>	COVID-19 Dose 1	Astrazeneca	false	09-02-2025

<< < 1-5 din 5 > >> 10 ▼

Figure 6.3: Test Scenario #3

In addition to the test cases, a site map was created to track all the pages and their respective components, with relationships showing how each page and component is related to each other. This was done to ensure that all components were created and tested properly. In the end, it allowed for a better, high-level overview of the system and its parts, but also for tracking of the overall progress of the project. The site map can be seen in figure 6.4.

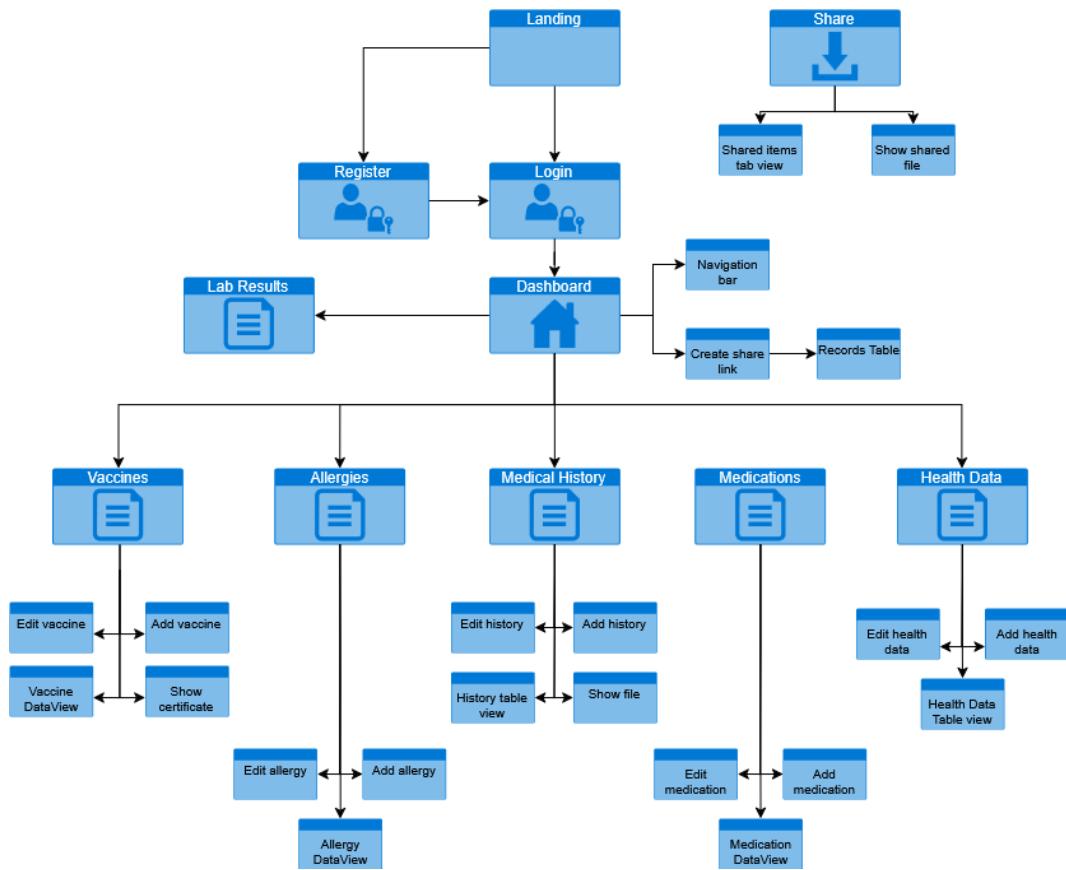


Figure 6.4: PHR System Site Map

## 6.2 Backend Testing

The backend testing part was also done manually. Thankfully, the backend was less complex than the frontend, so the testing was easier and faster to do. Additionally, much of backend's functionality was already tested simultaneously with the frontend testing, as it was dependent on the backend to work properly. As such, many of the test cases were already covered in the frontend testing.

However, to ensure that the backend was working as intended, additional tests were done using Postman while the API was being developed. This allowed for testing of the endpoints without relying on the frontend components being created. The tests were done by sending requests to the API and checking the responses, and comparing them with the expected results.

An example of how an endpoint was tested will be shown now. An example endpoint can be seen below:

```

1     @router.get("/share/{share_code}", status_code=status.HTTP_200_OK)
2     @limiter.limit("5/minute")
3     async def check_share_token(
4         request: Request,
5         share_code: str,
6         session: Session = Depends(get_session)
7     ):
8         share_token = session.exec(select(ShareToken).where(ShareToken.
9             share_code == share_code)).first()
10
11        if not share_token:
12            raise HTTPException(status_code=status.HTTP_404_NOT_FOUND,
13                detail="Share token not found")
14
15        if share_token.expiration_time < datetime.now():
16            raise HTTPException(status_code=status.HTTP_410_GONE, detail="Share
17                token has expired")
18
19        return {
20            "valid": True
21    }

```

Listing 6.2: Example endpoint for checking a share link for expiration time

This endpoint checks if a share link is valid and if it has expired. The endpoint takes in a share code and checks if it exists in the database. If it does, it checks if the expiration time is still valid. If both checks pass, it returns a valid response. If either check fails, it raises an HTTP exception with the appropriate status code and message.

As such, for this endpoint, 3 test cases can be created: one where the share code is not valid, another where the code is valid but the expiration time is not valid and a third one where everything is valid. The test cases in Postman can be seen in figures 6.5, 6.6 and 6.7.

The screenshot shows a Postman test interface. At the top, there is a header bar with 'GET' selected, a URL field containing 'http://127.0.0.1:8000/share/BQTvc1Fu', and a 'Send' button. Below the header are tabs for 'Params', 'Authorization', 'Headers (7)', 'Body', 'Scripts', and 'Settings'. The 'Params' tab is active. Under 'Query Params', there is a table with two columns: 'Key' and 'Value'. The first row has 'Key' and 'Value' both empty. The second row has 'Key' empty and 'Value' also empty. In the bottom section, there are tabs for 'Body', 'Cookies', 'Headers (4)', and 'Test Results'. The 'Body' tab is active. It shows a status bar with 'Status: 404 Not Found Time: 13 ms Size: 166 B'. Below the status bar are buttons for 'Pretty', 'Raw', 'Preview', and 'JSON'. The JSON preview area contains the following code:

```
1  {
2      "detail": "Share token not found"
3 }
```

Figure 6.5: Postman test — share token not found

The screenshot shows a Postman test interface. At the top, the URL is set to `http://127.0.0.1:8000/share/BQTvc1FK`. The method is selected as `GET`. Below the URL, there are tabs for `Params`, `Authorization`, `Headers (7)`, `Body`, `Scripts`, and `Settings`. The `Headers` tab is currently active, showing seven header entries. To the right of the headers are `Code` and `Cookies` buttons. A `Send` button is located at the top right of the main input area. Below the input area, there are tabs for `Body`, `Cookies`, `Headers (4)`, and `Test Results`. The `Body` tab is active, showing a JSON response with three numbered lines. The response body is:`1 [ {  
2 "detail": "Share token has expired"  
3 } ]`The status bar at the bottom indicates `Status: 410 Gone Time: 18 ms Size: 163 B`.

Figure 6.6: Postman test — share token expired

The screenshot shows a Postman test interface. At the top, the URL is set to `http://127.0.0.1:8000/share/mWX1lYvy`. The method is set to `GET`. There are tabs for `Params`, `Authorization`, `Headers (7)`, `Body`, `Scripts`, and `Settings`. The `Headers` tab is selected, showing seven headers. Below the headers is a `Query Params` table with one row, where the key is "Key" and the value is "Value". In the bottom section, the `Body` tab is selected, showing the response body in JSON format:

```
1 [ { 2 "valid": true 3 } ]
```

Figure 6.7: Postman test — share token valid

## 6.3 LLM Evaluation

As previously mentioned in the Development chapter, the MLLM used in this project was Gemini 2.0 Flash. One of the main reasons for using this LLM was its generous free tier, which allowed for up to 15 requests per minute with a maximum of 1,000,000 tokens per minute. Alongside the generous free tier, the model also sported robust multimodal capabilities, which was crucial for processing lab results, which were often in PDF or image format.

However, it is also important to note that while the free tier and its multimodal capabilities were a big factor in choosing this model, it was also chosen for its performance and speed, which rivaled that of other, similarly sized models. As of writing this section, Gemini 2.0 Flash occupies a spot in the top 10 of the Chatbot Arena leaderboard for both language and vision tasks, authored by Chiang et al. (2024). As described by its authors, the leaderboard is an ‘open platform for crowdsourced AI benchmarking’, which generates live leaderboards based on user feedback from comparing responses from different models. As such, it is quite likely that this leaderboard is a good indicator of the performance of the models, and good place to compare them.

To ensure the MLLM was working as intended and was up to the task, the same prompt used in the actual system was used to test the model across multiple lab result documents from various institutions. Documents from different institutions such as labs and hospitals were used to ensure that the model was able to handle different formats of the document and the language used within it.

Three different documents were used for evaluation — 2 from labs and 1 from a private hospital. The labs and the hospital were chosen for their popularity and a wide network of testing sites across the whole country (ALFA diagnostica, n.d.; Synevo Moldova, n.d.; Medpark, n.d.). The anonymised documents, alongside their extracted results, can be seen in the following figures: 6.8, 6.9 and 6.10.

Denumire	Rezultat	UM	Interval de referinta
<b>Biochimie</b>			
LC Antistreptolizina O (ASLO) Ser / metoda imunoturbidimetrica	40.9	UI/mL	≤ 200
LC Factor reumatoid Ser / metoda imunoturbidimetrica	< 10	UI/mL	< 14
LC Proteina C reactivă (CRP) Ser / metoda latex-imunoturbidimetrica	< 0.6	mg/L	< 5

(a) Lab document

**Înregistrare medicală nouă**

Analizați rezultatele extrase și schimbăți dacă sunt gresite

Test	Cod	Valoare	Unitate	Interval de referință	Metodă
Antistreptolizina O	ASLO	40.9	UI/mL	≤ 200	imunoturbidimetrica
Factor reumatoid		< 10	UI/mL	< 14	imunoturbidimetrica
Proteina C reactivă	CRP	< 0.6	mg/L	< 5	latex-imunoturbidimetrica

<< < 1-3 > >> 5 ▾

Salvează Anulează

(b) Extraction result

Figure 6.8: Synevo extraction results

## Hemograma cu formula leucocitară

Sângel integral. Metoda: Citometria în flux cu fluorescentă / Focalizarea hidrodinamică / SLS hemoglobină

Parametru		Rezultat	Valori de referință	Unități de măsură
Eritrocite (numărul absolut)	RBC	5,62	4,3 — 5,7	$\times 10^6/\mu\text{L}$
Hemoglobină	HGB	14,30	13,2 — 17,3	g/dL
Hematocrit	HCT	43,10	39 — 49	%
Eritrocite microcitare (procentul)	MicroR (%)	8,60	0,5 — 5,8	%
Eritrocite macrocitare (procentul)	MacroR (%)	4,90	3,3 — 5,1	%
Eritrocite nucleate (numărul absolut)	NRBC (#)	0,00	0,00	$\times 10^3/\mu\text{L}$
Eritrocite nucleate (procentul)	NRBC (%)	0,00	0,0	%
Volum eritrocitar mediu	MCV	76,70	80,0 — 99,0	fL

(a) Lab document

Înregistrare medicală nouă

Analizați rezultatele extrase și schimbați dacă sunt gresite

Cauta...

Test	Cod	Valoare	Unitate	Interval de referință	Metodă
Eritrocite (numărul absolut)	RBC	5.62	$\times 10^6/\mu\text{L}$	4.3-5.7	Ø
Hemoglobină	HGB	14.30	g/dL	13.2-17.3	Ø
Hematocrit	HCT	43.10	%	39-49	Ø
Eritrocite microcitare (procentul)	MicroR (%)	8.60	%	0.5-5.8	Ø
Eritrocite macrocitare (procentul)	MacroR (%)	4.90	%	3.3-5.1	Ø
Eritrocite nucleate (numărul absolut)	NRBC (#)	0.00	$\times 10^3/\mu\text{L}$	0.00	Ø
Eritrocite nucleate (procentul)	NRBC (%)	0.00	%	0.0	Ø
Volum eritrocitar mediu	MCV	76.70	fL	80.0-99.0	Ø

(b) Extraction result

Figure 6.9: Alfalab extraction results

BULETIN DE ANALIZE MEDICALE LABORATOR MEDPARK				
Data	Denumire	Rezultat	Interval de referință	UM
<b>BIOCHIMIE</b>				
<b>BIOCHIMIE GENERALA din sange si urina</b>				
	Lipaza serică	19.8	13 - 60	u/l
	Alfa-amilaza pancreatică serica	19.1	0 - 53	u/l
<b>MICROBIOLOGIE</b>				
Examen coproparazitologic pentru oua de helminti				
		Negativ	Negativ	<>
Giardia lamblia- antigen in materii fecale				
		0.04	0 - 0,28	<>
<b>IMUNOLOGIE</b>				
<b>MARKERI ENDOCRINI</b>				
	T4 free (tiroxina libera) (Ser/CHEMILUMINESCENTA (CLIA))	16.6	11,5 - 22,7	pmol/l
<b>MARKERI BOLI AUTOIMUNE</b>				
	Anti- TPO ( anticorpi anti-peroxidaza) (Ser/CHEMILUMINESCENTA (CLIA))	17.3	0 - 34	IU/mL

lab

(a) Lab document

### Înregistrare medicală nouă

Analyzezi rezultatele extrase și schimbă dacă sunt gresite

Test	Cod	Valoare	Unitate	Interval de referință	Metodă
Lipaza serică		19.8	u/l	13-60	<input checked="" type="checkbox"/>
Alfa-amilaza pancreatică serica		19.1	u/l	0-53	<input checked="" type="checkbox"/>
Examen coproparazitologic pentru oua de helminti		Negativ	<>	Negativ	<input checked="" type="checkbox"/>
Giardia lamblia- antigen in materii fecale		0.04	<>	0 -0.28	<input checked="" type="checkbox"/>
T4 free (tiroxina libera)	T4	16.6	pmol/l	11,5 - 22,7	Ser/CHEMILUMINESCENTA (CLIA) <input checked="" type="checkbox"/>

<< < 1-5 > >> 5 <

Salvează
Anulează

(b) Extraction result

Figure 6.10: Medpark extraction results

As it can be seen by the results, the model was able to extract the lab results and their details such as code, reference range, unit and even method. These excellent results were achieved with a single, common prompt and delivered consistent results across all 3 documents, whether they were edited in any way or not. In conclusion, the model seemed to be an excellent choice for the task at hand, and these results confirmed that Gemini 2.0 Flash was indeed a suitable choice for extracting lab results from documents.

## 6.4 Stakeholders feedback

To round up this section and the development of the system, feedback was collected from involved stakeholders regarding its performance and usability. The feedback was collected through virtual meetings, where the system was presented in a live demo format. After the presentation, stakeholders were allowed to ask any further questions and at the end they were provided feedback on the developed system and its future potential in the digital healthcare industry in Moldova.

There was no formal feedback collection process, nor a structured questionnaire, with stakeholders being encouraged to express their opinions in a free format. Some feedback excerpts can be seen below, with the names of the stakeholders anonymised. The full feedback can be found in the appendix C.

Feedback 1, from a medical doctor specialised in digital healthcare applications:

'I appreciated the concept of the project and the way all essential components of a Personal Health Record were integrated: general, clinical, and paraclinical data, consultations, physician access, and data security. The platform is strongly patient-centered, enhancing access to personal medical data and significantly contributing to patient empowerment... From my perspective as a medical doctor specialized in public health and an expert in digital transformation of healthcare, I see great potential in this project and I am committed to offering the necessary support for its further development'

Feedback 2, from a patient and healthcare user:

'I was really impressed with the application on the digital health records book you have developed and presented to me. As a BA you have managed to capture the most important needs and then translated them into useful features included in the solution developed... By developing this solution you have given a new opportunity to Moldova to make one big step forward to the digitisation, and hope it will be largely used by moldovans as the country offers them also

other related nice opportunities like high speed internet access and good coverage by 5G network'

# Chapter 7

## Conclusion and Future Work

The last chapter of this report will touch upon the conclusion of the project, discussing areas of improvement, future opportunities to further develop this system, lessons learned from all the stages of the work done, and final thoughts on the final year project experience.

### 7.1 Areas of Improvement

As it can be seen by the length of the report, the amount of sections written and the number of features that were implemented, the system created is quite complex. Although the feedback received was mainly positive, due to lack of time or experience, some areas were still in need of improvement. The next subsections will briefly touch upon these areas and how they can be improved in the future.

#### 7.1.1 Security and data encryption

One of the main area that was of concern were the security measures within the system. Even though some measures were already implemented, as discussed in the development chapter, it was definitely not enough for a live product.

One of the biggest issue with the security of the system was the storage of data at rest. All of the patient data is currently stored in a PostgreSQL database and in a plaintext format. If access to the database was compromised, all of the data would be easily exposed. As the data used is medical in nature, encrypting the data before being stored in the database would be crucial and non-negotiable first step to improve the security of the system. However, this would come with other issues, such as needing to decrypt the data everytime it is accessed

or even limiting some querying or other operations on the data. Another solution would be to secure the database itself by encrypting the database file or using a more secure database, such as AWS S3 or other cloud storage.

Another issue is the authentication method. The system currently uses a basic session-based authentication method, which may not be secure enough for a deployed system. Better options could include using a series of JWT tokens, or even using a secure 3rd party such as Auth0 or Mpass for authentication (Okta, Inc., 2025a; Agentia de guvernare electronică, 2025). This would also allow for better user management and access control, as well as better security measures such as 2FA or passwordless authentication.

Finally, data in transit is also a crucial area that needs to be considered. Since the system was never deployed, it was not possible to generate an SSL certificate and enable HTTPS for secure API communication. While not as hard to implement as the other potential security improvements, in a real-life deployment scenario this would be one of the first things to do.

### **7.1.2 Logging and monitoring**

One area that was not touched upon at all during development was logging and monitoring of system events. While it allows for better debugging and error handling, a more important role it would play would be in the auditing, security and compliance of the system. By having proper logging in place, it would be possible to track all changes made to any patient data, which is a critical element when dealing with such sensitive data. This would ensure that regardless of who made a change, whether that is a doctor, a patient or even a company employee, the change would be logged and allow for better accountability and auditing. Finally, it would also allow for better tracking of user activity, making it easier to spot any suspicious or malicious activity.

### **7.1.3 Legal and compliance**

Any system that works with sensitive data, especially medical data, needs to comply with a series of complex legal requirements and regulations. It is totally a different beast to tackle, and was not considered during development due to the lack of knowledge in the field and the prototype nature of the system. However, if idea were to be pursued, it would be crucial to ensure that the system complies with necessary regulations, both from the Republic of Moldova but also the EU, such as GDPR. This would not only involve writing extensive policies that show how the data is used, stored, processed and many other aspects, but it would also mean actually implementing them and being subject to regular audits and checks.

Usage of MLLMs within the system may also raise legal, compliance and even ethical issues

of how the patient data is processed and used. As such, it may be necessary to consider whether using a 3rd party MLLM is even possible or if it would be necessary to use a locally hosted MLLM, which may add additional complexity and cost to the release of the system.

#### **7.1.4 User experience and interface design**

The final area that was not fully addressed during the development of this project was the interface and user experience design. This was primarily due to the prototype nature of the project, but also the limited resources and time available that limited the time that could be dedicated to this area. It is an important area to consider, as it is very likely to have a big impact on users' first impression and experience. For a final product, it would be necessary for the system to be available in multiple formats, which includes web for both desktop and mobile devices, as well as a native mobile application for both iOS and Android. This would allow for better accessibility and usability of the system, as well as better user experience. Accessibility needs should also be addressed in the final system, to ensure it can be used by all users, regardless of their abilities.

## **7.2 Potential additions**

While the previous section discussed general areas of improvement for existing features within the system, this section will instead focus on specific additions that were either proposed by the stakeholders or were considered during the development process but not added. These features would mainly serve as additional quality of life improvement for the system by making it easier to use and improving the overall experience of the users.

### **7.2.1 Mpass integration**

While extensively discussed in multiple sections of the report, the system would greatly benefit from implementing Mpass as an authentication method. This would not only provide better authentication and user management, but it will also allow to use patient information that is already stored in government databases, as Mpass is closely linked to other government services. This would open up many possibilities for future integration with government services and greatly simplify the process of creating and accessing an account.

### **7.2.2 Expanding AI features**

It may be possible to further extend the existing AI features of the system. For example, the medical history record creation process could be further automated by eliminating the manual input of information by allowing the patient to just upload the file, which would

then process the document, extract the relevant information and label it accordingly. The user would only need to confirm and edit the information, if necessary, at the end. This would greatly improve the user experience and make it easier to use the system.

### **7.2.3 Integration with lab providers**

The current system expects the user to manually upload the file and create a new record instance to extract and store the lab results. This could be avoided by integrating directly with lab providers, which could send the results directly to the system. This would allow for much more accurate and faster processing of the results. One way thi could be achieved is by creating a custom mailbox for each patient, which they can use with the lab provider to send the results. However, this might create additional attack vectors for malicious actors, as they would be able to send emails to the system and potentially exploit it. Another solution would be to integrate via the lab provider's API and allow them to send the results directly to the system. This would be a more secure solution, but it would require a lot of work and effort to implement, as well as the cooperation of the lab providers.

### **7.2.4 Enhanced doctor-patient communication**

One feature requested by one the stakeholders during the final presentation was to add the ability for doctors to create a new records and add notes, comments when viewing the patient's records. This proposal was intended to reduce the likelihood of patients misunderstanding doctor diagnoses or directions during the consultations, allowing doctors direct input to ensure accuracy and clarity. This may also open up future possibilities for the system, where doctors and patients could communicate directly through the system, further reducing any friction in the process of consultation or treatment.

### **7.2.5 Integration with 3rd party providers**

Some of the existing features in the system could be improved by integrating 3rd party providers. An example would be the vitals data tracking, where currently users have to manually input any data. This could be improved by integrating with health tracker apps such as Google Fit, allowing for automatic syncing of relevant data. Another exampel of 3rd party integration could be the display of imaging results, such as X-rays or MRIs, which usually use different unconventional formats such as DICOM. This would allow for better display and management of the imaging results, as well as better integration with the rest of the system.

## 7.3 Lessons Learned and Reflections

The previous two sections of this chapter focused more on future technical considerations for this project. On the other hand, this section will more focus on the lessons learned based on the experience worked on this project. This will include both technical and non-technical aspects of the project, as well as the overall experience of working on a final year project.

### 7.3.1 Prioritisation and time management

One of the key lessons learned for this project is how important prioritisation of work and time management for complex projects is. This was a lesson learned the hard way, which was only realised towards the end of the project. A clear example of this was the improper prioritisation done for the project requirements, mainly the lack of priority order for the system features. As such, features like vaccines, allergies and medication were implemented first, when it was clear that the most important features were the medical history, lab extraction and sharing features. As such, the prototype could've been created without the vaccines, allergies and medication features, which could've been added later due to their simplicity and thus would have allowed for more time to be spent on the other elements, such as increased security, better user experience and interface design, as well as better testing and documentation.

The improper prioritisation also led to worse time management, as there was too much time spent on unnecessary features, and too little time was allocated for the more complex ones, but also for testing, deploying and other similarly important elements of the project.

As such, the lesson learned here is that prioritisation must be done properly, and it must be constantly revisited to ensure that the priorities remain the same throughout the project. Prioritisation does not mean just using a framework such as MoSCoW, but it also may mean ordering things in an order of importance or at least order of implementation, which must be agreed by all stakeholders.

An example of this can be seen in Scrum, which uses Product Backlogs to ensure all the requirements are properly prioritised in their correct order. Scrum also employs Backlog Refinement to ensure the requirements and their priority are constantly revisited and updated.

### 7.3.2 Importance of the team

Speaking of stakeholders, another key lesson learned was how important a team and its diverse roles and members are for the success of a project, especially for one as complex as

the one in this report. Even in the beginning, it was becoming clear how useful a separate roles would lessen the burden of the various tasks for this project — a Business Analyst to manage the requirements, a Designer to manage the user interface and experience, a QA engineer to manage the testing, and many other roles that are present in a team.

The lesson learned in this case was to respect all and any roles within a team that does software development. Each role and person has its importance and brings value in its own way, even though sometimes it may not be as clear or easily seen. Working together as a team will always bring better results than working alone, so it is important to have good teamworking and communications skill to ensure that the team can work together and achieve its goals.

### **7.3.3 Agile & Documentation**

To finish this section on a more positive note, this lesson was not something learned the hard way or something that created issues during the project. Instead, it was more an observation that was made many times during the project, especially the development phase.

The lesson learned here is how different Agile and more traditional methodologies are in terms of documentation. This can be clearly seen when comparing the project planning and development chapters of this report — the first contains multiple diagrams, wireframes, and other diagrams, whereas the development chapter contains noticeably less. This was a result of a more Agile approach used in the development phase, which involved iterations and changes based on constant stakeholder feedback. This resulted in less documentation such as diagrams made, as it was redundant due to constant changes made to the system.

However, one thing was clear — the documents that were used during the project could be described as ‘alive’. They were constantly changing and evolving as the project progressed. This can be clearly seen in some of the UML diagrams created, such as the ERD diagram or even some sequence diagrams, which are very different to their first version during the project planning.

As such, an important lesson learned here is that documents should not be seen as static, where they are created once and forgotten. Instead, they should always be revisited and ensured they reflect the most up to date state of the system and project. This is especially important in today’s world, where software development is often done in an Agile way, where requirements, designs and other elements are constantly changing and evolving.

## **7.4 Final Thoughts**

In conclusion, this project was a great experience that allowed for a lot of learning and growth. As a student that is doing a course that combines both Computer Science and Business elements, it was a great opportunity to apply the knowledge gained in the past 4 years. Additionally, it also served as a great practical example of how a real-life projects would be done, especially when working on real problems and with real stakeholders.

Going forward, this experience will be invaluable, as it had provided both a high and low level overview of most of the stages of a software development project. Considering my next role is in Product Management, this experience will allow me to better understand the technical aspects of the role, better communicate with team members and understand the challenges and issues that any member may face during the project. I will gladly take this experience with me in my future career and I hope to be able to apply the lessons learned in my future projects.

# References

- A7 Software (2022). *Who are we ?* URL: <https://www.andaman7.com/en/who-are-we>. (Accessed: 11 Nov 2024).
- Agenția de guvernare electronică (2025). *Întrebări frecvente*. URL: <https://mpass.gov.md/info/faq>. Accessed: 25 Feb 2025.
- Ahuja, Kabir et al. (2023). “MEGA: Multilingual Evaluation of Generative AI”. In: *ArXiv abs/2303.12528*. URL: <https://api.semanticscholar.org/CorpusID:257663467>.
- Akanksha and Akshay Chaturvedi (2022). “Comparison of Different Authentication Techniques and Steps to Implement Robust JWT Authentication”. In: *2022 7th International Conference on Communication and Electronics Systems (ICCES)*, pp. 772–779. DOI: [10.1109/ICCES54183.2022.9835796](https://doi.org/10.1109/ICCES54183.2022.9835796).
- Alenezi, Mohammed N, Haneen Alabdulrazzaq, and Nada Q Mohammad (2020). “Symmetric encryption algorithms: Review and evaluation study”. In: *International Journal of Communication Networks and Information Security* 12.2, pp. 256–272.
- ALFA diagnostica (n.d.). *Despre noi*. URL: <https://alfalab.md/ro/pagins/alfa-diagnostica-6015>. Accessed: 25 Apr 2025.
- Alqudah, Mashal and Rozilawati Razali (2018). “An empirical study of Scrumban formation based on the selection of scrum and Kanban practices”. In: *Int. J. Adv. Sci. Eng. Inf. Technol* 8.6, pp. 2315–2322.
- Alshamrani, Adel and Abdullah Bahattab (2015). “A comparison between three SDLC models waterfall model, spiral model, and Incremental/Iterative model”. In: *International Journal of Computer Science Issues (IJCSI)* 12.1, p. 106.
- Amatriain, Xavier (2024). *Prompt Design and Engineering: Introduction and Advanced Methods*. arXiv: 2401.14423 [cs.SE]. URL: <https://arxiv.org/abs/2401.14423>.
- Anderson, Benjamin and Brad Nicholson (2022). *SQL vs. NoSQL Databases: What's the difference?* URL: <https://www.ibm.com/think/topics/sql-vs-nosql>. (Accessed: 28 Oct 2024).

- Axios (n.d.). *Getting Started*. URL: <https://axios-http.com/docs/intro>. Accessed: 09 Feb 2025.
- Bano, Muneera et al. (2019). “Teaching requirements elicitation interviews: an empirical study of learning from mistakes”. In: *Requirements Engineering* 24, pp. 259–289.
- BBC (2024). *Moldova country profile*. URL: <https://www.bbc.co.uk/news/world-europe-17601580>. (Accessed: 22 Oct 2024).
- Broadcom (2024). *Why Spring?* URL: <https://spring.io/why-spring>. (Accessed: 14 Nov 2024).
- Centrul Național pentru Protecția Datelor cu Caracter Personal al Republicii Moldova (n.d.). *Legi*. URL: <https://datepersonale.md/legislation/national-legislation/legi/>. Accessed: 17 Dec 2024.
- Cheah, Jun Siang and Kevin Markham (2024). *Free LLM API resources*. URL: <https://github.com/cheahjs/free-llm-api-resources>. Accessed: 18 Dec 2024.
- Cheat Sheets Series Team (2025). *Session Management Cheat Sheet*. URL: [https://cheatsheetseries.owasp.org/cheatsheets/Session\\_Management\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html). Accessed: 24 Feb 2025.
- Chiang, Wei-Lin et al. (2024). *Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference*. arXiv: 2403.04132 [cs.AI].
- Ciurcă, Aliona (2021). *Sistemul informational automatizat din spitale: cum funcționează în R. Moldova și ce cred medicii că ar trebui ajustat*. URL: <https://www.zdg.md/stiri/stiri-sociale/sistemul-informational-automatizat-din-spitale-cum-functioneaza-in-r-moldova-si-ce-cred-medicii-ca-ar-trebui-ajustat/>. (Accessed: 22 Oct 2024).
- Compania Națională de Asigurări în Medicină (n.d.). *Legi*. URL: <http://cnam.md/legislatie/legi/>. Accessed: 17 Dec 2024.
- Davis, Alan et al. (2006a). “Effectiveness of Requirements Elicitation Techniques: Empirical Results Derived from a Systematic Review”. In: *14th IEEE International Requirements Engineering Conference (RE'06)*, pp. 179–188. DOI: 10.1109/RE.2006.17.
- (2006b). “Effectiveness of Requirements Elicitation Techniques: Empirical Results Derived from a Systematic Review”. In: *14th IEEE International Requirements Engineering Conference (RE'06)*, pp. 179–188. DOI: 10.1109/RE.2006.17.
- Django (2024). *Why Django?* URL: <https://www.djangoproject.com/start/overview/>. (Accessed: 14 Nov 2024).
- Donati, Beatrice et al. (2017). “Common mistakes of student analysts in requirements elicitation interviews”. In: *Requirements Engineering: Foundation for Software Quality: 23rd International Working Conference, REFSQ 2017, Essen, Germany, February 27–March 2, 2017, Proceedings* 23. Springer, pp. 148–164.

- Dubey, Abhimanyu et al. (2024). *The Llama 3 Herd of Models*. arXiv: 2407.21783 [cs.AI]. URL: <https://arxiv.org/abs/2407.21783>.
- E-Governance Agency (n.d.[a]). *About EGA*. URL: <https://egov.md/en/about-ega>. (Accessed: 22 Oct 2024).
- (n.d.[b]). *Citizen's Government Portal*. URL: <https://egov.md/en/content/citizens-government-portal>. (Accessed: 22 Oct 2024).
- Etxaniz, Julen et al. (2023). “Do Multilingual Language Models Think Better in English?” In: *North American Chapter of the Association for Computational Linguistics*. URL: <https://api.semanticscholar.org/CorpusID:260378999>.
- FastAPI (2024). *FastAPI framework, high performance, easy to learn, fast to code, ready for production*. URL: <https://fastapi.tiangolo.com/>. (Accessed: 31 Jan 2025).
- Fasten Health (2022). *What is Fasten?* URL: <https://docs.fastenhealth.com/>. (Accessed: 12 Nov 2024).
- Gemino, Andrew, Blaize Horner Reich, and Pedro M. Serrador (2021). “Agile, Traditional, and Hybrid Approaches to Project Success: Is Hybrid a Poor Second Choice?” In: *Project Management Journal* 52.2, pp. 161–175. DOI: 10.1177/8756972820973082. eprint: <https://doi.org/10.1177/8756972820973082>. URL: <https://doi.org/10.1177/8756972820973082>.
- Giray, Louie (2023). “Prompt engineering with ChatGPT: a guide for academic writers”. In: *Annals of biomedical engineering* 51.12, pp. 2629–2633.
- Google (2024a). *Advancing medical AI with Med-Gemini*. URL: <https://research.google/blog/advancing-medical-ai-with-med-gemini/>. Accessed: 17 Nov 2024.
- (2024b). *What is Angular?* URL: <https://angular.dev/overview>. (Accessed: 14 Nov 2024).
- Hada, Rishav et al. (2023). “Are large language model-based evaluators the solution to scaling up multilingual evaluation?” In: *arXiv preprint arXiv:2309.07462*.
- Hakimov, Sherzod and David Schlangen (July 2023). “Images in Language Space: Exploring the Suitability of Large Language Models for Vision & Language Tasks”. In: *Findings of the Association for Computational Linguistics: ACL 2023*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, pp. 14196–14210. DOI: 10.18653/v1/2023.findings-acl.894. URL: <https://aclanthology.org/2023.findings-acl.894>.
- Harrer, Stefan (2023). “Attention is not all you need: the complicated case of ethically using large language models in healthcare and medicine”. In: *eBioMedicine* 90.104512. DOI: <https://doi.org/10.1016/j.ebiom.2023.104512>.
- Heart, Tsipi, Ofir Ben-Assuli, and Itamar Shabtai (2017). “A review of PHR, EMR and EHR integration: A more personalized healthcare and public health policy”. In: *Health Policy*

- and Technology* 6.1, pp. 20–25. ISSN: 2211-8837. DOI: <https://doi.org/10.1016/j.hlpt.2016.08.002>. URL: <https://www.sciencedirect.com/science/article/pii/S2211883716300624>.
- Hickey, A.M. and A.M. Davis (2003). “Elicitation technique selection: how do experts do it?” In: *Proceedings. 11th IEEE International Requirements Engineering Conference, 2003.* Pp. 169–178. DOI: 10.1109/ICRE.2003.1232748.
- Hosseini, Azamossadat et al. (2023). “Integrated personal health record (PHR) security: requirements and mechanisms”. In: *BMC Medical Informatics and Decision Making* 23.1, p. 116.
- IEEE (2018). “ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering”. In: *ISO/IEC/IEEE 29148:2018(E)*, pp. 1–104. DOI: 10.1109/IEEESTD.2018.8559686.
- Jiang, Zhengbao et al. (July 2020). “How Can We Know What Language Models Know?” In: *Transactions of the Association for Computational Linguistics* 8, pp. 423–438. ISSN: 2307-387X. DOI: 10.1162/tacl\_a\_00324. eprint: [https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl\\_a\\_00324/1923867/tacl\\_a\\_00324.pdf](https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00324/1923867/tacl_a_00324.pdf). URL: [https://doi.org/10.1162/tacl%5C\\_a%5C\\_00324](https://doi.org/10.1162/tacl%5C_a%5C_00324).
- Khan, Javed Ali et al. (2015). “Comparison of requirement prioritization techniques to find best prioritization technique”. In: *International Journal of Modern Education and Computer Science* 7.11, p. 53.
- Kim, Hyuhng Joon et al. (2022). “Self-Generated In-Context Learning: Leveraging Auto-regressive Language Models as a Demonstration Generator”. In: *ArXiv* abs/2206.08082. URL: <https://api.semanticscholar.org/CorpusID:249712501>.
- Labrak, Yanis et al. (2024). *BioMistral: A Collection of Open-Source Pretrained Large Language Models for Medical Domains*. arXiv: 2402.10373 [cs.CL]. URL: <https://arxiv.org/abs/2402.10373>.
- Laplante, Phillip A (2019). *Requirements engineering for software and systems*. 3rd. Auerbach Publications.
- Lee, Jisan et al. (2021). “Review of national-level personal health records in advanced countries”. In: *Healthcare Informatics Research* 27.2, pp. 102–109.
- Li, Cheng et al. (2023). “Large Language Models Understand and Can be Enhanced by Emotional Stimuli”. In: URL: <https://api.semanticscholar.org/CorpusID:260126019>.
- Loweth, Robert P. et al. (Mar. 2021). “A Comparative Analysis of Information Gathering Meetings Conducted by Novice Design Teams Across Multiple Design Project Stages”. In: *Journal of Mechanical Design* 143.9, p. 092301. ISSN: 1050-0472. DOI: 10.1115/1.4049970. eprint: <https://asmedigitalcollection.asme.org/mechanicaldesign/>

- [article-pdf/143/9/092301/6667332/nd\143\9\092301.pdf](https://article-pdf/143/9/092301/6667332/nd\143\9\092301.pdf). URL: <https://doi.org/10.1115/1.4049970>.
- Lu, Albert et al. (2023). "Bounding the Capabilities of Large Language Models in Open Text Generation with Prompt Constraints". In: *ArXiv* abs/2302.09185. URL: <https://api.semanticscholar.org/CorpusID:257039031>.
- Luo, Lu (2001). "Software testing techniques". In: *Institute for software research international Carnegie mellon university Pittsburgh, PA* 15232.1-19, p. 19.
- Marqas, Ridwan B, Saman M Almufti, and Rasheed Rebar Ihsan (2020). "Comparing Symmetric and Asymmetric cryptography in message encryption and decryption by using AES and RSA algorithms". In: *Xi'an Jianzhu Keji Daxue Xuebao/Journal of Xi'an University of Architecture & Technology* 12.3, pp. 3110–3116.
- Medpark (n.d.). *Despre noi*. URL: <https://www.medpark.md/cine-suntem/>. Accessed: 25 Apr 2025.
- Medvalet (2024). *Despre noi*. URL: <https://medvalet.ro/>. (Accessed: 11 Nov 2024).
- Meskó, Bertalan (2023). "Prompt engineering as an important emerging skill for medical professionals: tutorial". In: *Journal of medical Internet research* 25, e50638.
- Meta Platforms (n.d.). *React. The library for web and native user interfaces*. URL: <https://react.dev/>. (Accessed: 14 Nov 2024).
- Ministerul Sănătății al Republicii Moldova (2024). *Legislație Națională*. URL: <https://ms.gov.md/legislatie/sanatate/legislatie-nationala/>. Accessed: 17 Dec 2024.
- Mirza, Mahrukh Sameen and Soma Datta (2019). "Strengths and Weakness of Traditional and Agile Processes-A Systematic Review." In: *J. Softw.* 14.5, pp. 209–219.
- Mohedas, Ibrahim et al. (2022). "The use of recommended interviewing practices by novice engineering designers to elicit information during requirements development". In: *Design Science* 8, e16. DOI: 10.1017/dsj.2022.4.
- Morphy, Tamzin (2020). *Stakeholder Analysis / Definition and best method*. URL: <https://www.stakeholdermap.com/stakeholder-analysis.html>. (Accessed: 31 Oct 2024).
- Mozilla (2025). *Using HTTP cookies*. URL: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies>. Accessed: 24 Feb 2025.
- Naiakshina, Alena et al. (2020). "On Conducting Security Developer Studies with CS Students: Examining a Password-Storage Study with CS Students, Freelancers, and Company Developers". In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI '20. Honolulu, HI, USA: Association for Computing Machinery, pp. 1–13. ISBN: 9781450367080. DOI: 10.1145/3313831.3376791. URL: <https://doi.org/10.1145/3313831.3376791>.

- National Bureau of Statistics of the Republic of Moldova (2024). *Population*. URL: [https://statistica.gov.md/en/statistic\\_indicator\\_details/25](https://statistica.gov.md/en/statistic_indicator_details/25). (Accessed: 22 Oct 2024).
- Naveed, Humza et al. (2023). “A comprehensive overview of large language models”. In: *arXiv preprint arXiv:2307.06435*.
- Nazi, Zabir Al and Wei Peng (2024). “Large Language Models in Healthcare and Medical Domain: A Review”. In: *Informatics* 11.3. ISSN: 2227-9709. DOI: 10.3390/informatics11030057. URL: <https://www.mdpi.com/2227-9709/11/3/57>.
- Niu, Qian et al. (2024). *From Text to Multimodality: Exploring the Evolution and Impact of Large Language Models in Medical Practice*. arXiv: 2410.01812 [cs.CY]. URL: <https://arxiv.org/abs/2410.01812>.
- Notion Labs (2025). *Want Agile results? Use Notion for Agile management*. URL: <https://www.notion.com/use-case/agile-management>. (Accessed: 11 Feb 2025).
- Ntantogian, Christoforos, Stefanos Malliaros, and Christos Xenakis (2019). “Evaluation of password hashing schemes in open source web platforms”. In: *Computers & Security* 84, pp. 206–224. ISSN: 0167-4048. DOI: <https://doi.org/10.1016/j.cose.2019.03.011>. URL: <https://www.sciencedirect.com/science/article/pii/S0167404818308332>.
- Nurgaliyev, Alibek and Hua Wang (2021). “Comparative study of symmetric cryptographic algorithms”. In: *2021 International Conference on Networking and Network Applications (NaNA)*, pp. 107–112. DOI: 10.1109/NaNA53684.2021.00026.
- Okta, Inc. (2025a). *Auth0 Overview*. URL: <https://auth0.com/docs/get-started/auth0-overview>. Accessed: 25 Feb 2025.
- (2025b). *What is OAuth 2.0?* URL: <https://auth0.com/intro-to-iam/what-is-oauth-2>. Accessed: 25 Feb 2025.
- OpenAI et al. (2024). *GPT-4 Technical Report*. arXiv: 2303.08774 [cs.CL]. URL: <https://arxiv.org/abs/2303.08774>.
- OpenJS Foundation (n.d.). *Fast, unopinionated, minimalist web framework for Node.js*. URL: <https://expressjs.com/>. (Accessed: 14 Nov 2024).
- Oracle (2020). *What is a Database?* URL: <https://www.oracle.com/uk/database/what-is-database/>. (Accessed: 28 Oct 2024).
- Papathanasaki, Maria, Leandros Maglaras, and Nick Ayres (2022). “Modern Authentication Methods: A Comprehensive Survey”. In: *AI, Computer Science and Robotics Technology*. DOI: 10.5772/acrt.08. URL: <https://doi.org/10.5772/acrt.08>.
- Paradigm, Visual (2024). *What is Unified Modeling Language (UML)?* URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/#sequence-diagram>. (Accessed: 07 Nov 2024).

- Pinia (n.d.). *Introduction*. URL: <https://pinia.vuejs.org/introduction.html>. Accessed: 09 Feb 2025.
- PostgreSQL Global Development Group (2025). *About*. URL: <https://www.postgresql.org/about/>. Accessed: 09 Feb 2025.
- Prenner, Nils, Carolin Unger-Windeler, and Kurt Schneider (2020). “How are hybrid development approaches organized? A systematic literature review”. In: *Proceedings of the International Conference on Software and System Processes*, pp. 145–154.
- Press, Ofir et al. (2022). “Measuring and Narrowing the Compositionality Gap in Language Models”. In: *ArXiv* abs/2210.03350. URL: <https://api.semanticscholar.org/CorpusID:252762102>.
- PrimeTek (n.d.). *Introduction*. URL: <https://primevue.org/introduction/>. Accessed: 09 Feb 2025.
- Reddi, Latha Thamma (2023). *Stakeholder Analysis using the Power Interest Grid*. URL: <https://www.projectmanagement.com/wikis/368897/Stakeholder-Analysis--using-the-Power-Interest-Grid>. (Accessed: 16 Dec 2024).
- Ruparelia, Nayan B. (May 2010). “Software development lifecycle models”. In: *SIGSOFT Softw. Eng. Notes* 35.3, pp. 8–13. ISSN: 0163-5948. DOI: 10.1145/1764810.1764814. URL: <https://doi.org/10.1145/1764810.1764814>.
- Schulhoff, Sander et al. (2024). *The Prompt Report: A Systematic Survey of Prompting Techniques*. arXiv: 2406.06608 [cs.CL]. URL: <https://arxiv.org/abs/2406.06608>.
- Sharma, Akshat et al. (2024). “A Secure Mechanism for Password Hash Value Generator with the Security Analysis of Various Hashing Algorithms”. In: *2024 4th International Conference on Computer, Communication, Control & Information Technology (C3IT)*, pp. 1–6. DOI: 10.1109/C3IT60531.2024.10829444.
- Sharma, Shreeta and SK Pandey (2013). “Revisiting requirements elicitation techniques”. In: *International Journal of Computer Applications* 75.12.
- Singhal, Karan et al. (2023). *Towards Expert-Level Medical Question Answering with Large Language Models*. arXiv: 2305.09617 [cs.CL]. URL: <https://arxiv.org/abs/2305.09617>.
- SQLModel (n.d.). *SQLModel, SQL databases in Python, designed for simplicity, compatibility, and robustness*. URL: <https://sqlmodel.tiangolo.com/>. Accessed: 09 Feb 2025.
- Stack Overflow (2024). *Technology*. URL: <https://survey.stackoverflow.co/2024/technology#most-popular-technologies-language>. (Accessed: 14 Nov 2024).
- Sunner, Daminderjit (2016). “Agile: Adapting to need of the hour: Understanding Agile methodology and Agile techniques”. In: pp. 130–135. DOI: 10.1109/ICATCCT.2016.7911978.

- Svelte (n.d.). *Overview*. URL: <https://svelte.dev/docs/svelte/overview>. (Accessed: 14 Nov 2024).
- Synevo Moldova (n.d.). *Despre noi*. URL: <https://synevo.md/despre-noi/>. Accessed: 25 Apr 2025.
- Tailwind Labs Inc. (2025). *Styling with utility classes*. URL: <https://tailwindcss.com/docs/styling-with-utility-classes>. Accessed: 09 Feb 2025.
- The Cucumber Open Source Project (2025). *Reference*. URL: <https://cucumber.io/docs/gherkin/reference>. Accessed: 24 Apr 2025.
- Tiwari, Saurabh, Santosh Singh Rathore, and Atul Gupta (2012). “Selecting requirement elicitation techniques for software projects”. In: *2012 CSI Sixth International Conference on Software Engineering (CONSEG)*, pp. 1–10. DOI: 10.1109/CONSEG.2012.6349486.
- United Nations Development Programme (2023). *A 100% digital state: The strategy for digital transformation of the Republic of Moldova for 2023-2030, approved by the Executive*. URL: <https://www.undp.org/moldova/press-releases/100-digital-state-strategy-digital-transformation-republic-moldova-2023-2030-approved-executive>. (Accessed: 22 Oct 2024).
- USMF (2023). *Brief history*. URL: <https://usmf.md/en/brief-history-usmf>. (Accessed: 22 Oct 2024).
- Vailshery, Lionel Sujay (2024). *Most used web frameworks among developers worldwide, as of 2024*. URL: <https://www.statista.com/statistics/1124699/worldwide-developer-survey-most-used-frameworks-web/>. (Accessed: 14 Nov 2024).
- Vaswani, Ashish et al. (2017). “Attention is all you need”. In: *Advances in neural information processing systems*.
- Vue Router (n.d.). *Getting Started*. URL: <https://router.vuejs.org/guide/>. Accessed: 09 Feb 2025.
- Vue.js (n.d.). *Components Basics*. URL: <https://vuejs.org/guide/essentials/component-basics.html>. Accessed: 09 Feb 2025.
- Xiao, Hanguang et al. (2024). *A Comprehensive Survey of Large Language Models and Multimodal Large Language Models in Medicine*. arXiv: 2405.08603 [cs.CL]. URL: <https://arxiv.org/abs/2405.08603>.
- Xu, Xiaohan et al. (2023). “Re-Reading Improves Reasoning in Large Language Models”. In: *Conference on Empirical Methods in Natural Language Processing*. URL: <https://api.semanticscholar.org/CorpusID:261696483>.
- Yas, Qahtan, Abdulbasit Alazzawi, and Bahbibi Rahmatullah (Nov. 2023). “A Comprehensive Review of Software Development Life Cycle methodologies: Pros, Cons, and Future Directions”. In: *Iraqi Journal For Computer Science and Mathematics* 4.4, pp. 173–190.

- DOI: 10.52866/ijcsm.2023.04.04.014. URL: <https://journal.esj.edu.iq/index.php/IJCM/article/view/664>.
- Yin, Shukang et al. (2024). "A survey on multimodal large language models". In: *National Science Review*, nwae403.
- You, Evan (2024). *What is Vue?* URL: <https://vuejs.org/guide/introduction.html>. (Accessed: 14 Nov 2024).
- Young, Ralph R (2002). "Recommended requirements gathering practices". In: *CrossTalk* 15.4, pp. 9–12.
- Zhang, Duzhen et al. (2024). "Mm-llms: Recent advances in multimodal large language models". In: *arXiv preprint arXiv:2401.13601*.
- Zheng, Mingqian, Jiaxin Pei, and David Jurgen (2023). "When "A Helpful Assistant" Is Not Really Helpful: Personas in System Prompts Do Not Improve Performances of Large Language Models". In: URL: <https://api.semanticscholar.org/CorpusID:265220809>.
- Zhou, Ce et al. (2023). "A comprehensive survey on pretrained foundation models: A history from bert to chatgpt". In: *arXiv preprint arXiv:2302.09419*.

## Appendix A

# Requirements

ID	Requirement	Priority
1.1	The system must be accessible on all modern desktop and mobile-based browsers.	Must Have
1.2	The system must be accessible from any location by using an internet connection.	Must Have
1.3	The system must store the data in a secure manner, ensuring that only the patient and the doctor can access the data.	Must Have
1.4	When shared with the doctor via a link, the system must load within 3 to 5 seconds when accessed via a desktop browser.	Should Have
1.5	When shared with the doctor via a link, the system should secure the data with a unique token or PIN that expires after the specified time frame.	Could Have
1.6	The system could be accessible on all modern mobile devices via a mobile application.	Could Have

Table A.1: Non-functional Requirements

ID	Requirement	Priority
2.1	The system must provide a secure login mechanism for patients by using a combination of login and password.	Must Have
2.2	The database must store the user credentials in a secure manner.	Must Have
2.3	The system could provide an option for multi factor authentication.	Could Have
2.4	The system could provide an option for password recovery.	Could Have
2.5	If used on mobile, the system could allow the patient to use biometric authentication for logging in.	Should Have

Table A.2: Login Requirements

ID	Requirement	Priority
3.1	The system must allow patients to upload their own medical records in a variety of formats (PDF, DOC, etc).	Must Have
3.2	The system should allow to upload files/documents for other types of records (vaccine certificates, etc).	Should Have
3.2	The system must allow the patient to specify and categorise the type of document they are uploading (lab test, doctor consultation, etc).	Must Have
3.3	The system must allow the patient to add a description of the document they are uploading.	Must Have
3.4	The system should allow the patient to add a date for the document they are uploading.	Should Have
3.5	The system should allow the patient to add a location for the document they are uploading.	Should Have
3.6	The system should allow the patient to add the doctor name for the document they are uploading.	Should Have
3.7	The system should allow the patient to sort and filter the documents based on the type of document, date, location, and doctor name.	Should Have
3.8	If used on mobile, the system should allow the patient to take a picture of the document and upload it.	Could Have

Table A.3: Document Upload Requirements

<b>ID</b>	<b>Requirement</b>	<b>Priority</b>
4.1	The system must allow the patient to generate a shareable link to provide access to their medical records.	Must Have
4.2	When creating the shareable link, the system must allow the patient to set an expiration date for the link.	Must Have
4.3	When creating the shareable link, the system should allow the patient to set an access password for the link.	Should Have
4.4	When creating the shareable link, the system should allow the patient to select which records to share with the doctor.	Should Have

Table A.4: Patient Shareable Link Requirements

ID	Requirement	Priority
5.1	The patient personal cabinet must provide an overview of the patient's history through 3 main sections: personal information, lab tests, and doctor consultations.	Must Have
5.2	The system must display the patient's history in a chronological order in the form of a timeline.	Must Have
5.3	The system must have a dashboard view which displays an overview of the most recent information added to the system (latest lab tests, doctor consultations, vaccinations etc).	Must Have
5.4	The system must allow patients to add their own personal information, such as name or date of birth.	Must Have
5.5	The system must allow the patient to add their own allergies.	Must Have
5.6	The system must allow the patient to add their own vaccinations.	Must Have
5.7	When viewing doctor consultations, the system should divide them into categories based on the domain of the doctor (cardiology, neurology, etc).	Should Have
5.8	The system should allow the patient to enter vitals information, such as height, weight, blood pressure, etc.	Should Have
5.9	The system should display the information in both a list or grid view.	Should Have
5.10	When multiple vital entries are made, the system could display a historical graph of the patient's vitals.	Could Have
5.11	The system could allow the patient to switch between viewing the lab tests in the document format or in a tabular, numerical format.	Could Have

Table A.5: Patient Personal Cabinet Requirements

<b>ID</b>	<b>Requirement</b>	<b>Priority</b>
6.1	The system must provide an overview of the patient history through 3 main sections: personal information, lab tests, and doctor consultations.	Must Have
6.2	When shared with the doctor, the system must allow the doctor to only view the patient's history, not edit it.	Must Have
6.3	The system must allow the doctor to view blood tests in a graphical format.	Must Have
6.4	The system must allow the doctor to view blood tests in a numerical, tabular format.	Must Have
6.5	The system must allow the doctor to view the patient's history in a chronological order.	Must Have
6.6	The system must display the doctor consultation and every lab test, except for blood tests, in a free text or document format.	Must Have
6.7	When viewing blood test results, the system should show the source document of the blood test value.	Should Have
6.8	For blood test results, the system should display the normal range values for each test.	Should Have

Table A.6: Shared Patient Information Requirements (Doctor View)

<b>ID</b>	<b>Requirement</b>	<b>Priority</b>
7.1	The system must allow patients to enter their current medication including details such as the name of the drug, dosage, frequency and start/end date.	Must Have
7.2	The system must allow patients to add new medication to their list.	Must Have
7.3	When adding medication, the system should have 2 options: add a simplified version of the medication or add a detailed version of the medication.	Should Have
7.4	When choosing the simplified version, the system should allow the patient to just add the name, dosage and duration of the medication.	Should Have
7.5	When choosing the detailed version, the system should allow the patient to add the name, dosage, frequency, start/end date, and the reason for taking the medication.	Should Have
7.6	The system should allow patients to add their past medication	Should Have
7.7	The system should allow patients to set medication reminders.	Should Have
7.8	After entering the medication, the system could allow the patient to track the medication intake.	Could Have

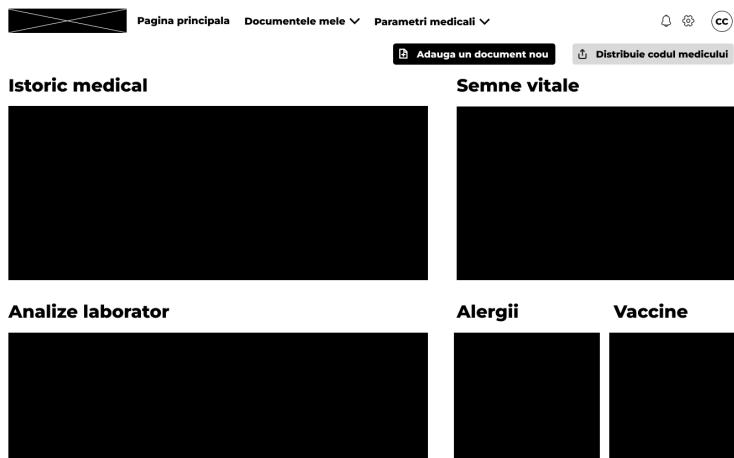
Table A.7: Patient Medication Requirements

## Appendix B

### Wireframes



Figure B.1: Mobile version of the Vaccines screen

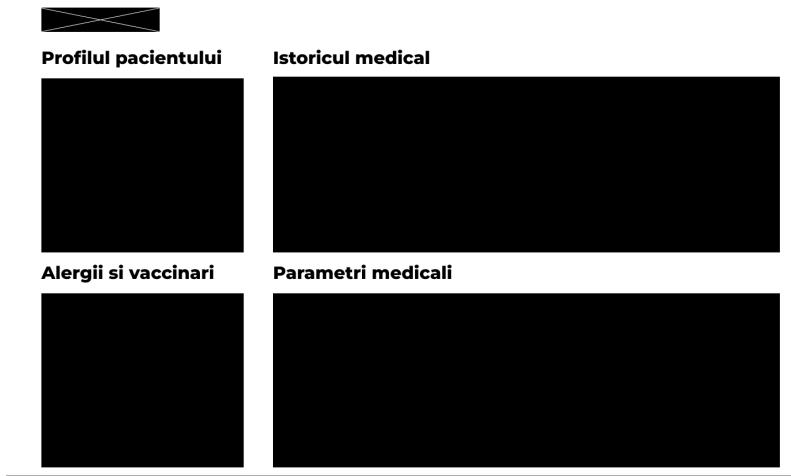


(a) Desktop version

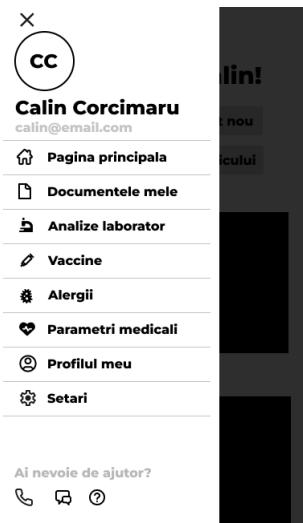


(b) Mobile version

Figure B.2: Desktop and Mobile version of the Dashboard screen



(a) Desktop version



(b) Mobile version

Figure B.3: Desktop and Mobile version of the Doctor View screen

Pagina principală Documentele mele ▾ Parametri medicali ▾

**Analize laborator**

Tabel Grafic

Data colectarii	Valoare	Unitate	Status	Document referinta
15.01.2025 10:29 GMT+2	13.01	g/L	Normal	Analize sange
15.01.2025 10:29 GMT+2	13.01	g/L	Normal	Analize sange
15.01.2025 10:29 GMT+2	13.01	g/L	Normal	Analize sange
15.01.2025 10:29 GMT+2	13.01	g/L	Normal	Analize sange
15.01.2025 10:29 GMT+2	13.01	g/L	Normal	Analize sange

**Selecteaza analiza**: Hemoglobina

**Selecteaza perioada**: Ultimile 7 zile, Ultimile 30 zile, Ultimul an

**periodea custom**: 08.01.2025 - 16.01.2025

**Detaли analiza**:  
 Valori de referinta:  
 Maximum - 6 mg/dL  
 Minimum - 10 mg/dL  
 Pentru mai multe informații:  
 Lorem ipsum Lorem ipsum Lorem ipsum  
 Lorem ipsum Lorem ipsum Lorem ipsum

(a) Desktop version

**Analize laborator**

Tabel Grafic

**Filtre**

**Selecteaza analiza**: Hemoglobina

**Selecteaza perioada**: 08.01.2025 - 16.01.2025

**Detaли analiza**

<b>13.01</b> g/L	Normal
15.01.2025	10:21 GMT+2
Analiza sange	

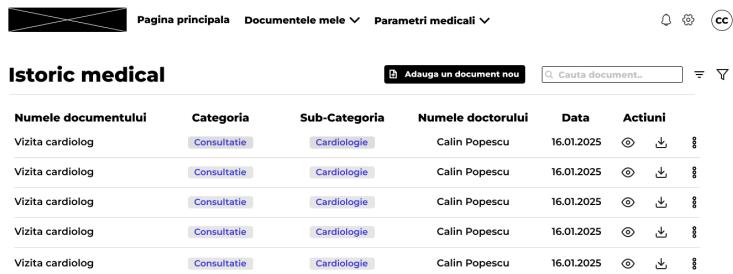
<b>13.01</b> g/L	Normal
15.01.2025	10:21 GMT+2
Analiza sange	

<b>13.01</b> g/L	Normal
15.01.2025	10:21 GMT+2
Analiza sange	

<b>13.01</b> g/L	Normal
15.01.2025	10:21 GMT+2
Analiza sange	

(b) Mobile version

Figure B.4: Desktop and Mobile version of the Lab Test screen



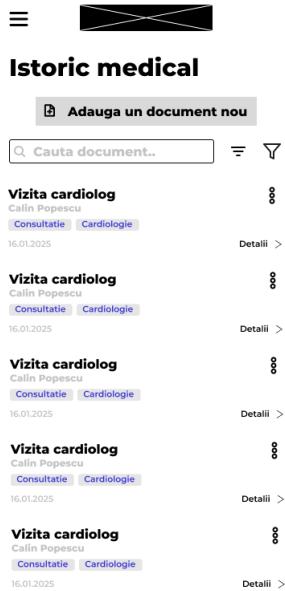
Pagina principala Documentele mele Parametri medicali

**Istoric medical**

Adauga un document nou Cauta document..

Numele documentului	Categoria	Sub-Categoria	Numele doctorului	Data	Actiuni
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	
Vizita cardiolog	Consultatie	Cardiologie	Calin Popescu	16.01.2025	

(a) Desktop version



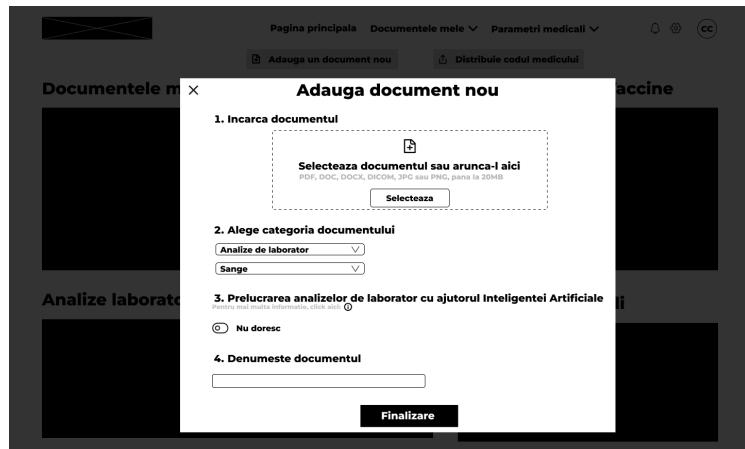
**Istoric medical**

Adauga un document nou Cauta document..

<b>Vizita cardiolog</b> Calin Popescu Consultatie Cardiologie 16.01.2025	 <a href="#">Detalii &gt;</a>
<b>Vizita cardiolog</b> Calin Popescu Consultatie Cardiologie 16.01.2025	 <a href="#">Detalii &gt;</a>
<b>Vizita cardiolog</b> Calin Popescu Consultatie Cardiologie 16.01.2025	 <a href="#">Detalii &gt;</a>
<b>Vizita cardiolog</b> Calin Popescu Consultatie Cardiologie 16.01.2025	 <a href="#">Detalii &gt;</a>
<b>Vizita cardiolog</b> Calin Popescu Consultatie Cardiologie 16.01.2025	 <a href="#">Detalii &gt;</a>

(b) Mobile version

Figure B.5: Desktop and Mobile version of the Medical History screen

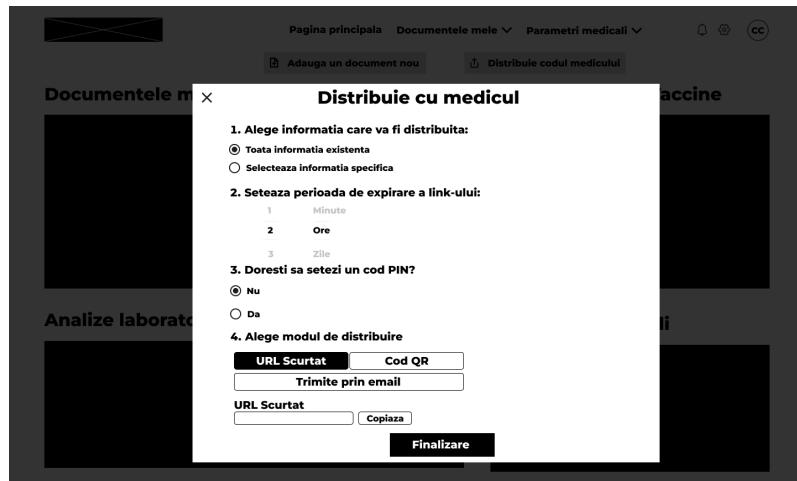


(a) Desktop version

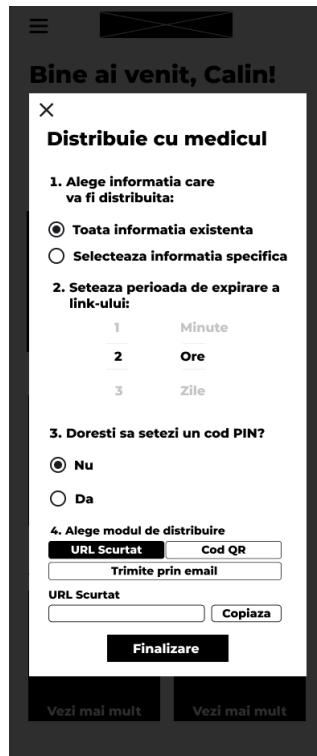


(b) Mobile version

Figure B.6: Desktop and Mobile version of the New Document screen



(a) Desktop version



(b) Mobile version

Figure B.7: Desktop and Mobile version of the Share Doctor screen

**Alergii**

Gestionează și vizualizează alergiile tale

+ Adaugă alergie

<b>Tip</b> <b>Medicament</b> Alergie la <b>Penicilină</b>  Severitate <b>Severă</b>  Simptome <b>Dificultăți de respirație, urticarie</b>  Data diagnosticării <b>2022-03-15</b>  Note <b>Evitați toate antibioticele din familia penicilinelor</b>	<b>Tip</b> <b>Alimentar</b> Alergie la <b>Arahide</b>  Severitate <b>Moderată</b>  Simptome <b>Urticarie, măncărini</b>  Data diagnosticării <b>2021-06-20</b>  Note <b>Evități toate produsele care conțin sau pot conține urme de arahide</b>	<b>Tip</b> <b>Mediu</b> Alergie la <b>Polen</b>  Severitate <b>Ușoară</b>  Simptome <b>Strânut, nas infundat, ochi iritați</b>  Data diagnosticării <b>2020-04-10</b>  Note <b>Se manifestă în special primăvara</b>
--	--	---

(a) Desktop version



## Alergiile mele

**Adaugă alergie nouă**

### Arahide

Alergen: **Arahide, nuci, alte nuci**

Severitate: **Mediu**

Diagnosticat la **16.01.2025**

### Arahide

Arahide, nuci, alte nuci

### Arahide

Arahide, nuci, alte nuci

(b) Mobile version

Figure B.8: Desktop and Mobile version of the Allergies screen

## Appendix C

# Full feedback from stakeholders

Full feedback for stakeholder 1:

'I appreciated the concept of the project and the way all essential components of a Personal Health Record were integrated: general, clinical, and paraclinical data, consultations, physician access, and data security. The platform is strongly patient-centered, enhancing access to personal medical data and significantly contributing to patient empowerment.'

I would emphasize the importance of adhering to interoperability standards, both for future integration with hospital software systems and for enabling the upload of data from mobile devices.

From my perspective as a medical doctor specialized in public health and an expert in digital transformation of healthcare, I see great potential in this project and I am committed to offering the necessary support for its further development'

Full feedback for stakeholder 2:

'I was really impressed with the application on the digital health records book you have developed and presented to me. As a BA you have managed to capture the most important needs and then translated them into useful features included in the solution developed.'

Even though it is one the first releases, it looks already complete and covers all the journey of a citizen fascinated with the use of the technology for managing the comprehensive records related to its health status in a very efficient and effective way. The application is user friendly, it looks so intuitive as the user

can easily find any data or add something new to the app. It offers a lot of flexibility in terms of types of data that can be added and monitored. The dashboards and reports generated in the app are easy to read and understand, as well as provide insightful information on the evolution of the status. I also would like to mention the professional approach as you have provided for notifications related to personal data protection, as the applications seem to use very sensitive information.

By developing this solution you have given a new opportunity to Moldova to make one big step forward to the digitisation, and hope it will be largely used by moldovans as the country offers them also other related nice opportunities like high speed internet access and good coverage by 5G network'