Software Requirements Specification

for

PawMed

Version 1.3

Authors: Dawid Grobert
Julia Boczkowska
Rafał Synoradzki
Bartosz Ryczek
Daniel Marek
Almerino Buce
Aibak Aljdayah

Modified: 28.06.2022

Created: 31.03.2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Dawid Grobert	31.03.2022	The document has been created	1.0
Daniel Marek Rafał Synoradzki	07.04.2022	Sections 1, 2 and 3 have been added	1.1
Rafał Synoradzki	26.06.2022	Sections 4 and 5 have been added	1.2
Rafał Synoradzki	28.06.2022	Sections 3 and 4 has been modified	1.3

1. Introduction

1.1 Purpose

The purpose of this project is to create a product that incorporates a steady workflow and proper internal communication for medical facilities.

1.2 Intended Audience and Reading Suggestions

The product is directed towards medical staff employees, specifically registrars, doctors, lab technicians and supervisors. On top of that, the following documentation is being directed at system administrators wanting to get accommodated to the system's behavior behind the scenes.

1.3 Product Scope

This platform is designed for usage across many different kinds of medical facilities through its universal construction.

1.4 References

- https://www.luxmed.pl/ - one of the largest medical centers in Poland.
- https://medchart.pl/ - software designed for clinic management

2. Overall Description

2.1 Product Perspective

This website is a standalone product, it is not a part of a larger application ecosystem. It is being provided as an alternative to already existing services of a similar type (i.e. an internal lab clinic system).

2.2 Product Functions

The product functionalities are based around the inner workings of the clinic. Additionally, an admin panel will be available to the system administrator of the clinic. The list of functionalities available to the users is as follows:

- creating a patient profile
- registering visits
- ordering and revising laboratory tests
- access to a patient's visit and test history

2.3 User Classes and Characteristics

- Lab Technician - can perform ordered laboratory tests
- Lab Supervisor/Manager - can manage the list of ordered laboratory tests (corrections, removal, etc.)
- Doctor - can handle patient visits, is able to include information about a performed physical examination to the patient's profile, can order laboratory tests
- Registrar - can create patient profiles and register doctor visits

2.4 Operating Environment

The project is a web based application, therefore It only requires a Django-compatible web server to operate. From the user's side, a modern web browser is required.

2.5 Design and Implementation Constraints

The team is composed of young software developers without real experience. The team must plan carefully during the design process to go through the implementation phase smoothly.

2.6 User Documentation

As a website application, we don't plan to prepare technical documentation for the user. However, we will prepare some simple guides which will help use our product.

3. External Interface Requirements

3.1 User Interfaces

3.1.1 Navigation Bar

This element is common for every view, contains the following elements:

- option to log into an account,
- option to log out of an account,
- notification displaying,
- search bar with filtering options.

3.1.2 Registration Page

Registration page is available to users with accounts labeled as 'registrar'. This view allows for creating new database entries and updating already existing ones regarding patients data, along with creating new entries regarding doctor visits and laboratory tests.

3.1.3 Listed Tests Page

This page contains all planned and already conducted laboratory tests with brief description. This page also contains the option to sort and filter all tests.

3.1.4 Detailed Test Page

This view contains all information about planned and already conducted laboratory tests. Informations regarding tests are as follows:

- person performing test,
- doctor which ordered the test,
- type of test (blood, hormones etc.)
- patients id whose sample is being tested,
- test outcome (if already performed).
- additional notes,
- test validation outcome.

3.1.5 Lab Manager Page

The lab manager page contains information concerning the work of the entire lab. Its supervisor has access to functionalities such as:

- displaying the currently performed tests
- displaying the queue of tests awaiting to be performed
- displaying the history of tests already performed through filters such as:
    - patient name
    - lab technician name
    - date
- managing the state of individual tests (modification / removal)

3.1.6 Starting page

The starting page contains contact information to the IT support at the header, an image to left and two buttons for logging in with an existing and approved account and for registering a new account for approval

3.2 Hardware Interfaces

Due to being a web application, the product is designed to be available to use on both computer and mobile devices' browsers with JavaScript enabled. The internal communication is going to be performed between the website and a PostgreSQL database.

3.3 Software Interfaces

The web application can be divided into three sections: frontend, backend and database. Each part use different libraries:

- Django for the frontend and backend
- PostgreSQL for the database

3.4 Communications Interfaces

The most used communication protocol is going to be HTTPS due to the web nature of this application. It will serve the majority of user and server requests. On top of that, the SMTP protocol may be used to send email notifications to users.

4. System Features

4.1 User - Register and login to an account

4.1.1 Description

The user can register a staff account at the website for manual verification by the site's administrator. Upon the administrator's approval and role assignment the account will gain access to its functionalities.

Priority: High

4.1.2 Technical Details

In order to support this feature several data tables have to be incorporated into the database. Next, an authentication system has to be created - this is done using Django's integrated authentication feature. On top of that, a logged-in user's view of the site has to be different from a regular visitor's one - a non authenticated user of the site should not be able to access any of the roles' functionalities.

4.2 Registrar - Access patient information

4.2.1 Description

The user with the registrar role can access the information about any patient registered in the clinic's database. The information includes:

- Name and surname
- Personal ID
- Gender
- Birthday date
- Phone Number
- City
- Zip code
- Registration date

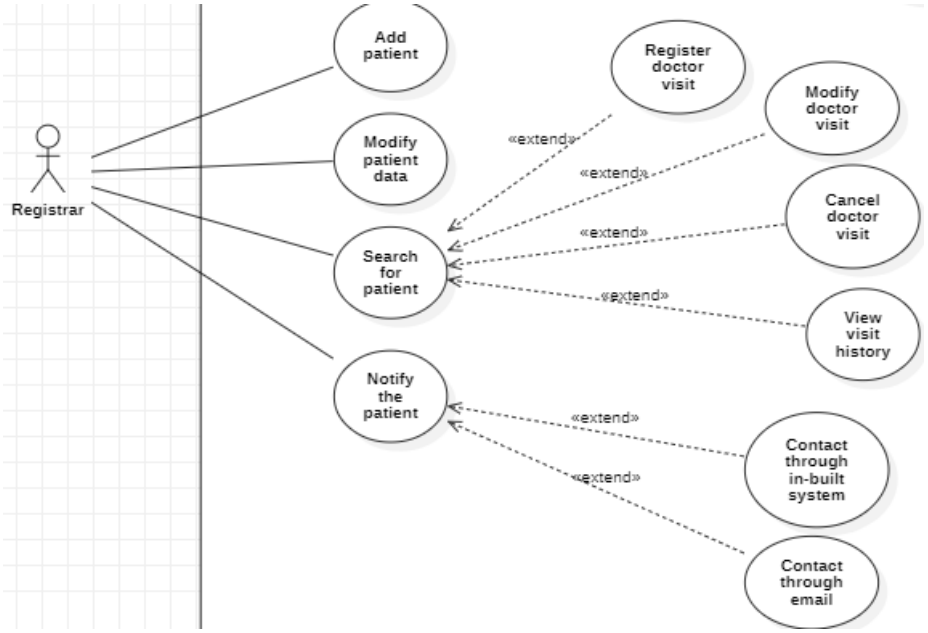
From there they can additionally schedule a new appointment with the doctor.

Priority: High

4.2.2 Technical Details

In order to support this feature, a data table for patient data has to be incorporated into the database. Calls to the backend to retrieve the data based on provided criteria need to be created. The Registrar can only view a patient's profile by entering the patient's name and ID - by making physical contact with the patient's ID card. Without this, registrar cannot view the patient's profile. Modifying the url also does not work to view other patients.

4.2.3 Use Case Diagram



4.3 Registar - Add a patient

4.3.1 Description

The user with the registrar role can add information about a new patient to the database. The information required about the patient includes: Name and surname, gender, birthday date, phone Number,city and zip code.

Priority: High

4.3.2 Technical Details

In order to support this feature, a new form for patient information has to be created. The registrar has to fill in all of the required fields for the patient's profile to be created and added to the database.

4.4 Registar - Management of appointments

4.4.1 Description

The Registrar, by accessing a user profile, can make appointments with doctors based on date range, specialty, or the doctor himself. Registrar can also cancel appointments.

Priority: High

4.4.2 Technical Details

The appointment selection is done through a special ajax form. First, the specialty is selected, then based on that registrar can choose a doctor from the list of doctors with a particular specialty. Registrar can also specify the date. Registrar is then redirected to the list of free appointments, which can also be booked via Ajax Button. Pagination makes it easier to navigate through the dates when there are too many free appointments.

4.5 Doctor - Manage and perform patient visits

4.5.1 Description

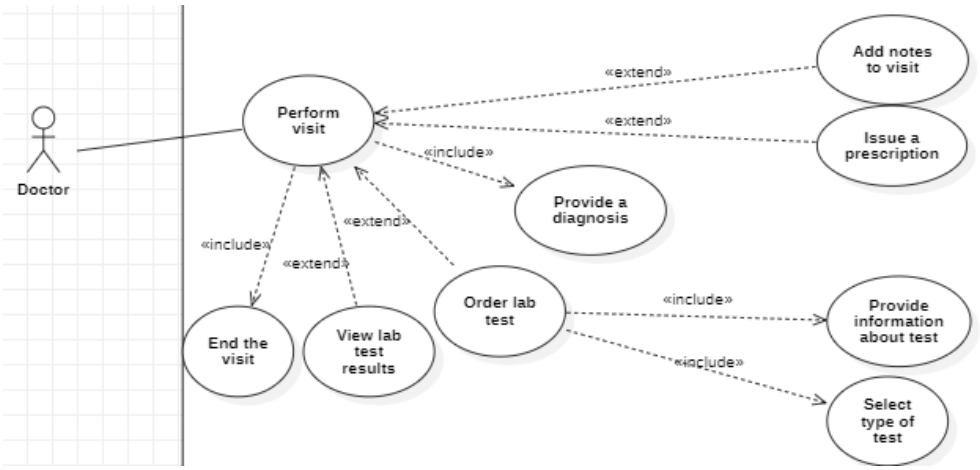
The user with the doctor role has access to the list of patients appointed by the registrar to them. From there they can postpone or cancel scheduled visits. They can also add prescriptions to patients. On top of that they have access to additional options, such as ordering tests and reviewing them.

Priority: High

4.5.2 Technical Details

In order to support this feature, a list view of patients along with a detailed view has to be created. A relation between the doctor and visits has to be formed in the database.

4.5.3 Use Case Diagram



4.6 Doctor - Order test

4.6.1 Description

The user with the doctor role can order a laboratory test after a performed visit. They have to include additional information concerning the test, such as the type of the test and additional remarks.

Priority: High

4.6.2 Technical Details

In order to support this feature, an additional form has to be created for the creation of new tests. A relation has to be included between the visit and the test. A doctor can only order a test after a visit has been performed.

4.7 Lab Manager - Manage the list of tests

4.7.1 Description

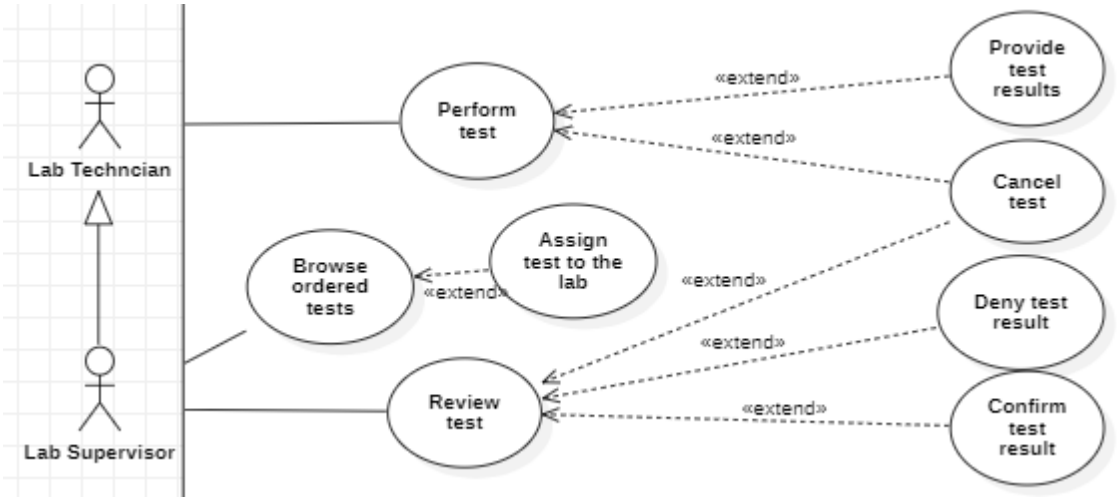
The user with the lab manager (supervisor) role has access to the list of incoming laboratory test requests. They can assign individual tests to available lab technicians along with a specific laboratory room.

Priority: High

4.7.2 Technical Details

In order to support this feature, a list view of tests along with a detailed view has to be created. A relation between the test and technician, as well as between the test and laboratory has to be included.

4.7.3 Use Case Diagram



4.8 Lab Manager - Provide feedback on tests

4.8.1 Description

The user with the lab manager role can provide feedback on test results from technicians - they can either accept the result, deny it and request another test or cancel the test.

Priority: Medium

4.8.2 Technical Details

In order to support this feature, a detailed test view needs to be created. Confirming the result of the test allows the doctor to view it from their perspective. Rejecting the result of the test allows the lab technician to view it and fill it in again from their perspective.

4.9 Lab Technician - Perform test and return results

4.9.1 Description

The user with the lab technician role can perform tests assigned to them by the lab manager. After the test they can attach results to it and send it over to the lab manager for approval. If the manager decides to reject the test results, they can send the test back to the technician who can then perform the test again and enter in new results.

Priority: High

4.9.2 Technical Details

In order to support this feature, a list view of assigned test requests along with a detailed view of the assigned test needs to be created. For the test to be sent from the technician’s side, it has to contain the results. If the test is reassigned after result denial, the same requirements apply.

4.10 Administrator - Manage roles

4.10.1 Description

The user with the administrator role can manage other users' accounts. After a new user registers an account, before using it they have to be manually approved by the administrator. On top of that they can also assign their own role, as well as manage the available users with the doctor role.

Priority: High

4.10.2 Technical Details

In order to support this feature, Django's automatic admin system is being used. The administrator has to manually verify a new employee account and assign them a role before the account can be used.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Server-side, the application should work efficiently for its users, regardless of the traffic to it. On the client's side, the website should not consume more system memory than modern systems can provide.

5.2 Safety Requirements

The website will have features designed for internal company usage. Thus, no moderation apart from the site's administrator manual one is required.

5.3 Security Requirements

The application will store patients' credentials for usage in the clinic system. As for the employees' accounts, upon registration the account has to be manually approved and assigned a role by the system administrator prior to usage.

5.4 Software Quality Attributes

Every team member who wishes to update the code within the repository's main branch will need to send a pull request, which will be reviewed by other developers and then accepted by the team leader. Additionally, each pull request will be required to contain unit tests.