Bilkent University
Department of Computer Engineering

Senior Design Project
T2506
Hipograf

Detailed Design Report

Team Members:

Salih Furkan Göktaş, 22202620

Orhun Güder, 22202471

Şükrü Eren Gökırmak, 22203746

Ramiz Arda Ünal, 22202554

Artun Berke Gül, 22203316

Supervisor: Prof. Dr. Uğur Doğrusöz

Instructor: Mert Bıçakçı

Instructor: İlker Burak Kurt

13.03.2026

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

1. Introduction

1.1 Purpose of the System

Hipograf is a new age medical visualization and information graphing application that bridges the gap between humans and medical data across Türkiye. The purpose of Hipograf is to help doctors make informed decisions regarding their patients by providing them with easy-to-read medical history visualizations. It is also intended as a tool for patients to be more informed about their own conditions and health.

1.2 Design Goals

The four main design goals of Hipograf are Reliability, Performance, Security and Usability, in line with the application's non-functional requirements.

Reliability: Hipograf is designed to be used in the medical sector, where access should be guaranteed for as long periods as possible. Utmost care should be taken to avoid presenting the user with erroneously incomplete information. Hipograf should notify the user that its knowledge base is incomplete when there is an issue with the amount of information it is able to access.

Performance: One of Hipograf's main objectives is to increase efficiency, being designed to be used by medical facilities across all of Türkiye. This means the system should be able to work under heavy load, letting users quickly access whatever representation of data they require. It should also function properly on the computers available at medical facilities.

Security: Secure handling of personal and private data is critical to Hipograf's success. No medical data will be kept permanently on site unless directly uploaded by the user (concerning data that does not have a one-to-one correspondence with equivalent data in existing external medical databases that Hipograf connects to). All external data that is handled during the duration of a log-in session will be deleted immediately upon session termination.

Usability: As Hipograf's main purpose is to be an aid to medical information visualization, usability is one of the most important aspects of the system. The user interface should be easily understandable and usable, along with being accessible enough to maximize the number of people who can use it effectively.

1.3 Definitions, Acronyms, Abbreviations

Definitions

e-Nabız: A web-based application developed by the Türkiye Cumhuriyeti Sağlık Bakanlığı to give citizens a unified access portal for digitized personal medical data [1].

HIPPO: The LLM-powered chatbot deployed within Hipograf.

Abbreviations

IEC: Abbreviation for 'International Electrotechnical Commission', a standardization organization specialising in electrical engineering hardware [2].

ISO: Abbreviation for 'International Organization for Standardization', an international standardization organization that covers scientific, engineering and more general fields alike [2].

IEEE: Abbreviation for 'Institute of Electrical and Electronics Engineers', an organization aiming to further advancement and research in electrical engineering and related disciplines [2].

SRS: Abbreviation for 'Software Requirement Standards', a set of standards jointly published by the IEC/ISO/IEEE that states their expectations for requirements documentation of software projects [2].

W3C: Abbreviation for 'World Wide Web Consortium', a standardization organization specialising in the usage and guidelines of the internet [3].

WCAG: Abbreviation for 'Web Content Accessibility Guidelines', a series of recommendations published by the W3C to further the accessibility of the internet, especially for disability-based access [3].

OMG: Abbreviation for 'Object Management Group', a standardization organization specialising in data modelling [4].

UML: Abbreviation for 'Unified Modelling Language', a set of modelling standards that capture the structural and behavioural design of software systems [4].

WHO: Abbreviation for 'World Health Organization', an agency of the United Nations that concerns itself with the general livelihood, alongside physical and mental health of people worldwide [5].

ICD: Abbreviation for 'International Classification of Diseases', a globally accessible and standardized list of diseases monitored and maintained by the WHO [5].

ATC: Abbreviation for 'Anatomical Therapeutic Chemical', a classification system for medically prescribed drugs and medication maintained and updated by the WHO [5].

SKRS: Abbreviation for ‘Sağlık Kodlama Referans Sunucusu’, an online reference resource maintained by the Türkiye Cumhuriyeti Sağlık Bakanlığı that contains medical classification databases used in national hospitals and care facilities [6].

SUT: Abbreviation for ‘Sağlık Uygulama Tebliği’, a communique regularly published and updated by the Sosyal Güvenlik Kurumu detailing the pricing of medical services and equipment [7].

KVKK: Abbreviation for ‘Kişisel Verilerin Korunması Kanunu’, a set of legal limitations on the processing of information considered to be relating to a person’s private affairs [8].

1.4 Overview

Hipograf is a web application that allows users to connect their medical data, in order to receive detailed graphical analyzes and visualizations on it. It can be used by both patients and medical practitioners. In order to achieve its purpose, Hipograf features various pages. After logging in, users are able to view a dashboard with a general overview of their present medical situation, alongside various widgets. A Timeline page chronologically tracks medical events. The Graph page allows users to generate, combine and customize various graphs of their medical data. Finally, the List Pages allow users to view additional information like: existing blood tests and radiology images, prescribed medications, vaccines and more.

2. Current Software Architecture

2.1 General Data Management and Visualization Solutions

Two very prominent data visualization tools used in the industry are Microsoft Power BI and SAS Visual Analytics [9, 10, 11]. Both of these tools have vast data visualization capabilities with a high capacity of customization with support for a variety of data domains. However, the magnitude of capabilities provided by these tools can be a detriment when an industry professional does not use them. An important goal of Hipograf is for it to be intuitive and be easy to use by everyone, regardless of their background. The variety of options and ways to customize provided by these tools can overwhelm Hipograf’s target audience. On top of this, these tools are not designed to be compatible with the existing ecosystem of Turkish national medical information access portals like e-Nabız. A middleware between the systems provided by the Turkish government and these tools is still required, which imposes restrictions on this pipeline’s availability to every citizen of Türkiye. Hipograf does not have this issue because it is designed to be a part of this ecosystem by integrating with APIs like the one e-Nabız offers and be available to everyone who is a part of the Turkish health system. With the shortcomings of these tools stated, it can be concluded that general purpose data management and visualization tools are not fit to be used for the purposes of medical data visualization.

2.2 Domestic Medical Information Visualization Solutions

All health related information can be accessed through e-Nabız for all citizens of Türkiye. As a direct result of this, we assume that the amount of publicly available tools are not focused on private medical information. The existence of a government backed, centralized service like e-Nabız is likely to prevent the introduction of a software ecosystem consisting of comparable applications since it serves as a monolith that uniquely provides a significant number of capabilities. New competitors cannot sprout when one entity has already deeply rooted itself with practically all the needed functionality. However, when the existing system eventually encounters a weakness, the lack of alternatives leaves the citizens with no viable options to switch to. In the case of e-Nabız, this weakness is the visualization of the medical information. e-Nabız provides most data as text; historical information is presented as simple lists, which puts little emphasis on things like passage of time between data points [1]. Hipograf serves as the missing link between the textual data and the user's comprehension by being able to access all the necessary information to provide its users with strong visualization options for their private medical information, something currently lacking in the Turkish health application ecosystem.

2.3 International Medical Information Visualization Solutions

In the international ecosystem of medical information visualization tools, strong tools like Medihis, KeepTrackMed, CareClinic are available [12, 13, 14]. Even though these tools provide alternative or similar visualization capabilities to what is planned to be included in Hipograf, not a single one of these applications individually provides everything Hipograf plans to provide. A major problem with these applications is that they often have their own way of keeping data, relying on people uploading their own information. This goes against two important requirements we set out to achieve: Having a tool that is available to all citizens of Türkiye and limited possession of private medical information.

3. Proposed Software Architecture

3.1 Overview

Users will be able to interact with the application by connecting to the Hipograf webpage through the HTTPS protocol (a normal web browser will suffice). They will be presented with a login screen prompting them to log in with their Hipograf account. This account is used for three kinds of data persistence: The account credentials themselves, details of various display preferences of the user, and any data they have chosen to upload directly. After logging in, they will be presented with a dashboard that gives them a summary of their present medical history and conditions, alongside a simplified timeline of relevant events and a set of customizable widgets that give them an overview of various data fields. Users will be able to filter and choose the data they are presented with on this page. They also have the option of accessing other pages, including the Timeline, Graph and List pages.

The Timeline Page offers a more detailed view of the simplified timeline preview that is available on the Dashboard, including all medical events the patient has partaken in. These consist of information obtainable through E-Nabız that have dates attached to them and can be things like doctor visits, and blood tests. They also include information that the user has submitted themselves, like blood test results.

The Graph Page will give users the option of choosing which graphs and graph to create based on the user's preference, displaying data as a function of time. The user is able to create graphs on an interactive screen where one or more data types can be specified from a searchable dropdown menu. This graph is resizable and combinable with other graphs to create automatically normalized fusions plotting several data points at once.

The List Page actually refers to a set of pages, providing a wide variety of additional functionalities. Users should be able to list all patients (if they are a medical practitioner) under their purview, view (and upload) existing blood tests and radiology images, list prescribed medications, list vaccinations, list existing diagnoses, list former hospital visits and list medical operations that concern the patients.

3.2 Subsystem Decomposition

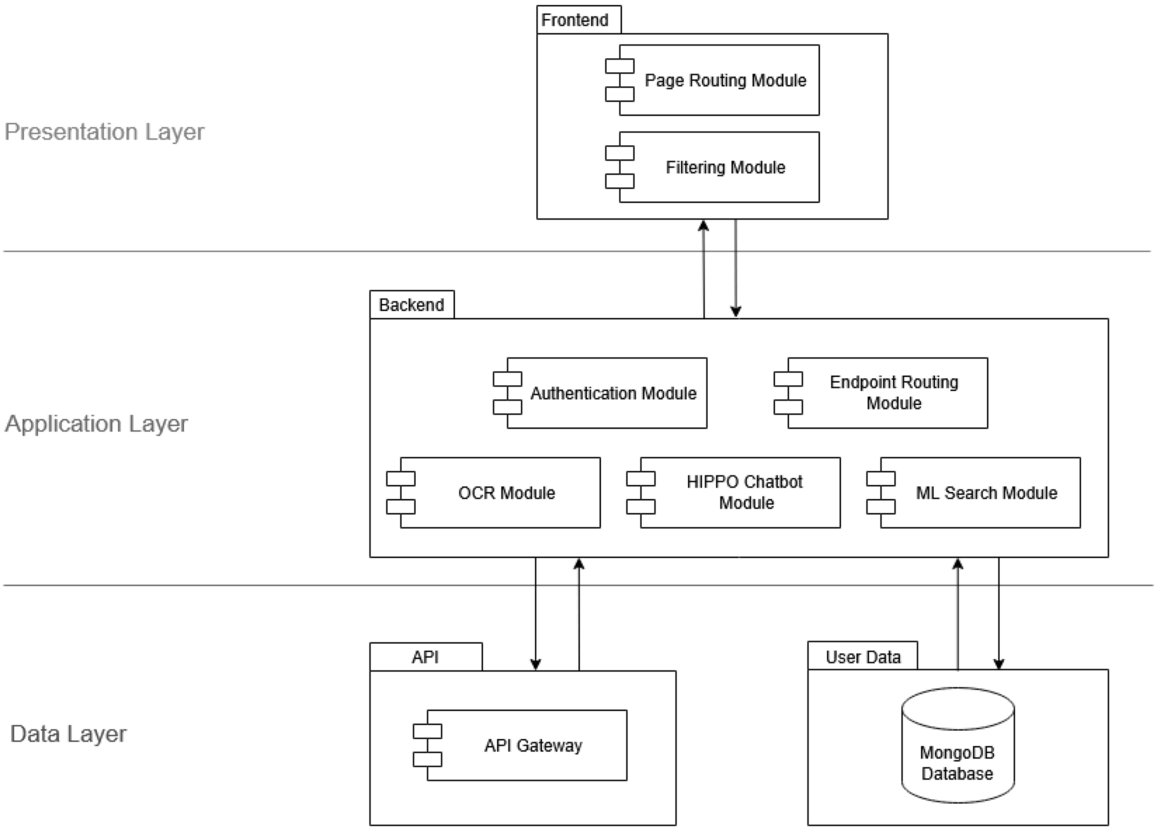

Fig. 1. Subsystem Decomposition Diagram

3.3 User Stories

The User Stories are included here for use in the test cases. The user stories that a particular test case satisfies will be included in the test case description, in the bottom row. User stories are numbered as such: USx, where x is a denary number incremented starting from 1.

General Management

US1 - Registration

User Story: As a medical practitioner or a patient, I want to register to Hipograf so that I can securely authenticate myself to the system in the future.

Acceptance Criteria:

- The user will provide a username and a password.
- The password will be checked to ensure it complies with the accepted standards.
- If successful, the user will be told that their registration was successful.
- The user will be redirected to the login page.

US2 - Login

User Story: As a medical practitioner or a patient, I want to log in to Hipograf so that I can use the application.

Acceptance Criteria:

- The user will provide a name and password.
- The username and the corresponding hash of the password will be checked to make sure it is equal to the one in the database.
- If the credentials are correct, the user will be logged into their account, and redirected to their home page.

US3 - Logout

User Story: As a medical practitioner or a patient, I want to log out of Hipograf so that I can close my session.

Acceptance Criteria:

- The user will click the “log out” button.
- If successful, the session of the user will end, and they will be logged out.
- If the server shuts down or experiences another issue, the valid cookies issued by the server will reset.

US4 - Account Deletion

User Story: As a medical practitioner or a patient, I want to be able to delete my account so that Hipograf servers will wipe my data.

Acceptance Criteria:

- The user will click the “Delete Account” button.
- On success, the information of the user, such as their login credentials and data viewing preferences, will be wiped from the Hipograf servers.
- On failure, a failure message will be shown, prompting the user to try again.

US5 - Password Change

User Story: As a medical practitioner or a patient, I want to be able to change my password so that I can change my login credentials.

Acceptance Criteria:

- While logged in, the user will click the “Change Password” button, and be prompted to enter their new password, and confirm it.
- On success, the password of the user will successfully be changed.
- On failure, a failure message will be shown, prompting the user to try again. Failure can happen due to network connectivity or an unsuitable password.

US6 - Account Information Update

User Story: As a medical practitioner or a patient, I want to update the information available on my account so that it stays up to date when my credentials change.

Acceptance Criteria:

- While logged in, the user will click the “Update Account Details” button, and be prompted with a page where they can update some of their credentials such as username, email, etc.
- On success, the account details of the user will be changed.
- On failure, a failure message will be shown, prompting the user to try again.

US7 - Reset Password

User Story: As a medical practitioner or a patient, I want to be able to reset my password in the login page, so that I can change it when I forget my password.

Acceptance Criteria:

- While logged out, the user will click the “Forgot password” button, and will be prompted to enter their registered email. After, a link will be sent to the user’s registered e-mail, where they can reset their password.
- On success, the user’s password will change and the user will be redirected back to the login page.
- On failure, a failure message will be shown, prompting the user to try again. Failure can happen due to the absence of an account related to the given email, or connectivity issues.

US8 - Clear Preferences Data

User Story: As a medical practitioner or a patient, I want to be able to clear my data preferences so that I can reset the information displayed in my dashboard to its default state.

Acceptance Criteria:

- While logged in, the user will click the “Clear preferences” button.
- On success, the data visualization preferences will be reset.

- On failure, a failure message will be shown.

US9 - Chat with HIPPO

User Story: As a medical practitioner or a patient, I want to be able to chat with HIPPO, so that I can be informed about the functionalities of Hipograf.

Acceptance Criteria:

- While logged in, the user will click the HIPPO icon, and either pick a predetermined chat option or type a query themselves.
- On success, HIPPO will answer the user’s query.
- On failure, a failure message will be shown.

Patient Selection System

US10 - List Patients

User Story: As a medical practitioner, I want to be able to list all patients appointed to me so that I can select a patient and access their medical history.

Acceptance Criteria:

- The system shall display a list containing the names of all patients who have granted me access to their medical records.
- Each patient name in the list must be selectable/clickable.

US11 - Select Patient Page

User Story: As a medical practitioner, I want to access the comprehensive, single set of pages for a selected patient, so that I can quickly review their entire medical history and current status.

Acceptance Criteria:

- Upon selecting a patient from the list, I shall be navigated to a detailed patient profile page.
- This page must clearly display Basic Personal Information (Name, Birthdate, City of Residence, etc.).
- This page must display a unified, filterable Medical Timeline compiling all major events.
- The page must include separate sections or widgets for Recent Blood Work, Medications, Hospital Visits, Diagnoses, Operations, Radiology Images, and Vital Signs data from wearables.

List Management System

US12 - List Blood Test

User Story: As a medical practitioner or a patient, I want to view and analyze all a patient's blood test results, so that I can monitor specific physiological changes and health trends over time.

Acceptance Criteria:

- I shall be able to view a list of all blood work instances and all radiology images.
- I must be able to filter blood work instances by criteria such as the presence of a specific parameter and the time/date the test was performed.
- I must be able to select a single blood work instance to view a detailed page showing the test date and all measured parameters.
- From the detailed blood work view, I must be able to click on a specific parameter to display a graph showing all historical measurements of that parameter across all blood work instances.

US13 - Upload Blood Test

User Story: As a medical practitioner or a patient, I want to upload blood test results to Hipograf so that I can use my blood test data points inside Hipograf.

Acceptance Criteria:

- I shall be able to upload a blood test file from an external API in PDF format to Hipograf, and get back the test results in table form.
- I must be able to see and edit the inferred test result data before submitting it.

US14 - List Diagnosis

User Story: As a medical practitioner or a patient, I want to view and filter a patient’s past diagnoses, so that I can analyze their complete health profile and history of conditions.

Acceptance Criteria:

- I shall be able to view a list of all past and current diagnoses.
- I must be able to filter the diagnoses list using criteria such as whether the diagnosis is chronic or acute, the diagnosis’s name/category, or the time frame when the diagnosis was given.

US15 - List Hospital Visit

User Story: As a medical practitioner or a patient, I want to list all of a patient’s hospital visits and apply filters, so that I can review past acute care episodes and dates of admission.

Acceptance Criteria:

- I shall be able to view a list of all documented hospital visits.
- I must be able to filter the list of hospital visits by criteria, such as the date of admission.

US16 - List Radiological Images

User Story: As a medical practitioner or a patient, I want to view, filter, and access detailed information about a patient's radiological images, so that I can review the findings from medical imaging procedures.

Acceptance Criteria:

- I shall be able to view a list of all radiological images (X-rays, CTs, MRIs, etc.).
- I must be able to filter this list by the type of imaging (e.g., "CT Scan") and the date it was taken.
- Selecting an image from the list must navigate me to a page with detailed information about that specific radiological exam (e.g., full report, findings).

US17 - List Operations

User Story: As a medical practitioner or a patient, I want to list all of a patient’s medical operations and apply various filters, so that I can understand their surgical history.

Acceptance Criteria:

- I shall be able to view a list of all recorded medical operations/surgeries.
- I must be able to filter the list of operations using various relevant criteria (e.g., date of operation, type of operation).

US18 - List Vaccinations

User Story: As a medical practitioner or a patient, I want to list all of a patient’s past vaccinations so that I can be informed and avoid repeat doses of vaccines.

Acceptance Criteria:

- I shall be able to view a list of all past vaccinations.

- I must be able to filter the list of vaccines using the date of the vaccine administered, and the contents of the vaccine.

US19 - List Prescriptions

User Story: As a medical practitioner or a patient, I want to list all of a patient's past and current prescriptions so that I can be informed and avoid repeat doses or conflicting prescriptions.

Acceptance Criteria:

- I shall be able to view a list of all past and current prescriptions.
- I must be able to filter the list of prescriptions by date, and the active chemical within the prescriptions.

US20 - List Wearable Data

User Story: As a medical practitioner or a patient, I want to list all of a patient's wearable data, if they have opted in to list them, so that I can be more informed about my patient's medical status.

Acceptance Criteria:

- I shall be able to view a patient's past wearable data.
- I must be able to list the wearable data by date, and relevant wearable data class.

Timeline Management

US21 - Filter Event Category

User Story: As a medical practitioner or a patient, I want to be able to choose the categories of kinds of events shown so that I can focus on events that are currently important to me.

Acceptance Criteria:

- The user will have access to a menu containing the currently available event categories on the same page as the timeline.
- The user will be able to toggle on or off these categories through the menu.
- The toggled off categories will be removed from the timeline.

US22 - Choose Timeframe

User Story: As a medical practitioner or a patient, I want to be able to choose a timeframe for the timeline so that I can focus on events that took place in that time window.

Acceptance Criteria:

- The user will be able to input a start date and end date for the timeline.
- These dates will decide the earliest and latest dates that will be shown on the timeline, respectively.

US23 - Edit Timeframe

User Story: As a medical practitioner or a patient, I want to be able to use the dedicated resize buttons or key combinations to change the length of the timeframe of the timeline so that I can prevent cluttering that can be caused by an accumulation of a high number of events in a timeframe.

Acceptance Criteria:

- The user will be able to zoom in and zoom out of the timeline.
- If the user zooms in, the earliest date shown on the screen will increase and the latest date shown on the screen will decrease, showing a narrower date interval on the screen.
- If the user zooms out, the earliest date shown on the screen will decrease and the latest date shown on the screen will increase, showing a wider date interval on the screen.

US24 - Scroll Timeline

User Story: As a medical practitioner or a patient, I want to be able to use the scroll wheel or a key combination to change the earliest and the latest dates shown on the timeline while keeping the length of the timeframe constant so that I can reach all events on the timeline even if I resized the timeline.

Acceptance Criteria:

- The user will be able to scroll the timeline horizontally towards either direction.
- If the timeline is scrolled to the right, both the earliest and the latest date shown on the screen will increase.
- If the timeline is scrolled to the left, both the earliest and the latest date shown on the screen will decrease.
- In both cases, the period of time shown on screen will be kept constant.

US25 - Obtain Event Information

User Story: As a medical practitioner or a patient, I want to be able to click on events shown on the timeline so that I can get detailed information about that specific event.

Acceptance Criteria:

- The user will click on a specific event stationed on the timeline.
- An info box connected to the chosen event will appear above or below depending on the circumstances of the timeline.
- This info box will contain detailed information related to the chosen event.

US26 - Save Preferences

User Story: As a practitioner or a patient, I want to be able to save the currently applied filtering and timeline viewing options as a preference so that I can retrieve it later and apply it instantly, preventing unnecessary burden.

- The user save will click to save their current preferences.
- They will be asked to give the preset a name.
- The currently applied customization choices will be saved under the given name.

US27 - Load Preferences

User Story: As a medical practitioner or a patient, I want to be able to load a preference set from a list I have previously saved before so that I would not need to reconfigure every option according to my needs everytime I login to the system.

Acceptance Criteria:

- The user opens the preset menu that lists all their presets.
- The user chooses the preset they would like to apply
- The timeline is adjusted according to the details of the chosen preset.

Graph Management

US28 - Create Graph

User Story: As a medical practitioner or a patient, I want to be able to create individual graphs that represent multiple different sets of data on a single graph so that I can compare and contrast their change relative to one another over time.

Acceptance Criteria:

- The user should be able to select multiple datasets when the graph is being created.
- The user should be able to select a timescale that fits for both of them on the same graph.
- The compounded data should be automatically normalized so that I can view the graphs with relative ease.

US29 - Search for Graph

User Story: As a medical practitioner or a patient, I want to be able to type out my desired graph set-up along with its relevant timeframe and have it appear for me.

Acceptance Criteria:

- The user should be able to type out, in normal English, their desired graph(s) along with the timeframes they remain relevant for.
- The user should then be able to immediately access these created graphs such that they can perform the rest of the normal graph operations on them.

US30 - Edit Graph

User Story: As a medical practitioner or a patient, I want to be able to edit, scale and move the graphs that represent my data so that I can gain a holistic understanding of the interaction of datasets.

Acceptance Criteria:

- The user should be able to edit graphs to change the timescale setting or the data being graphed.
- The user should be able to resize the graphs around the screen.
- The user should be able to move the graphs around on the screen without being limited to specific spots on the screen.

US31 - Recolor Graph

User Story: As a medical practitioner or a patient, I want to be able to recolor a graph so that I can have better color contrast with other present graphs.

Acceptance Criteria:

- The user should be able to change the color of any individual graph by picking from a color picker with several dozen color options

US32 - Combine Graph

User Story: As a medical practitioner or a patient, I want to be able to combine graphs to combine the data they are representing in a united space.

Acceptance Criteria:

- The user should be able to combine existing graphs to result in graphs that graph the data simultaneously.
- The user should see an overview that automatically normalizes graph data.

US33 - Split Graph

User Story: As a medical practitioner or a patient, I want to be able to split combined graphs to once again see them on their own.

Acceptance Criteria:

- The user should be able to split combined graphs to get the original graphs before the combination.

US34 - Save Preferences

User Story: As a medical practitioner or a patient, I want to be able to save my preferences for a graph so that I can refer to them at a later date.

Acceptance Criteria:

- The user should be able to save their preferences for time or data type, across single or multivariate graphs, in the local database.
- The user should not have the option of saving any medical data.

US35 - Load Preferences

User Story: As a medical practitioner or a patient, I want to be able to restore previously saved graph preferences so that I can receive a familiar overview without having to consistently reconfigure it.

Acceptance Criteria:

- The user should be able to reproduce the previously saved data onto the graphing interface.
- The user should be able to do this consistently, as many times as is required.
- The user should have the option of editing and deleting preferences in addition to restoring them.

3.4 Behavioral UML Diagrams

Activity Diagrams

Saving Preference

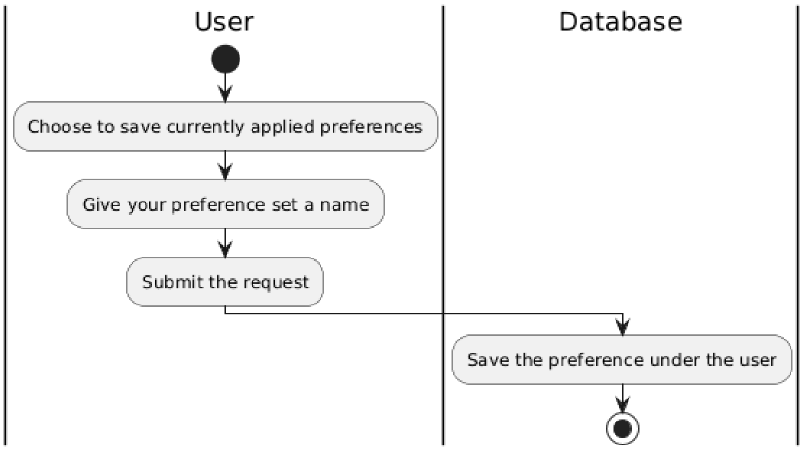
User
Database
Choose to save currently applied preferences
Give your preference set a name
Submit the request
Save the preference under the user

Fig. 2. Activity Diagram for Preference Saving

Loading Preference

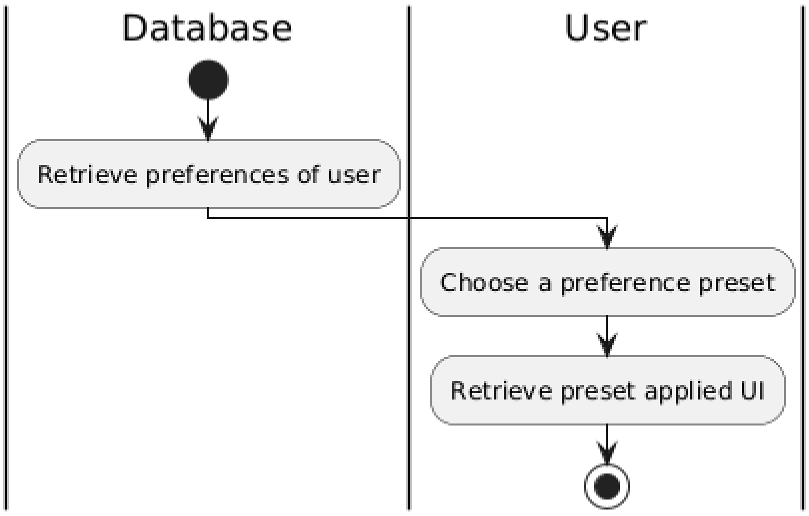
Database
User
Retrieve preferences of user
Choose a preference preset
Retrieve preset applied UI

Fig. 3. Activity Diagram for Preference Loading

Login

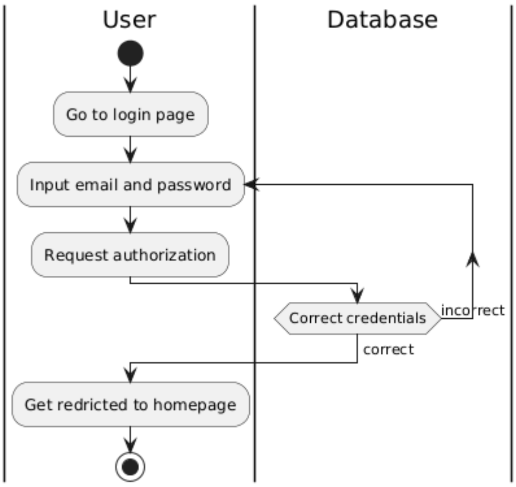

Fig. 4. Activity Diagram for Hipograf Login

Reset Password

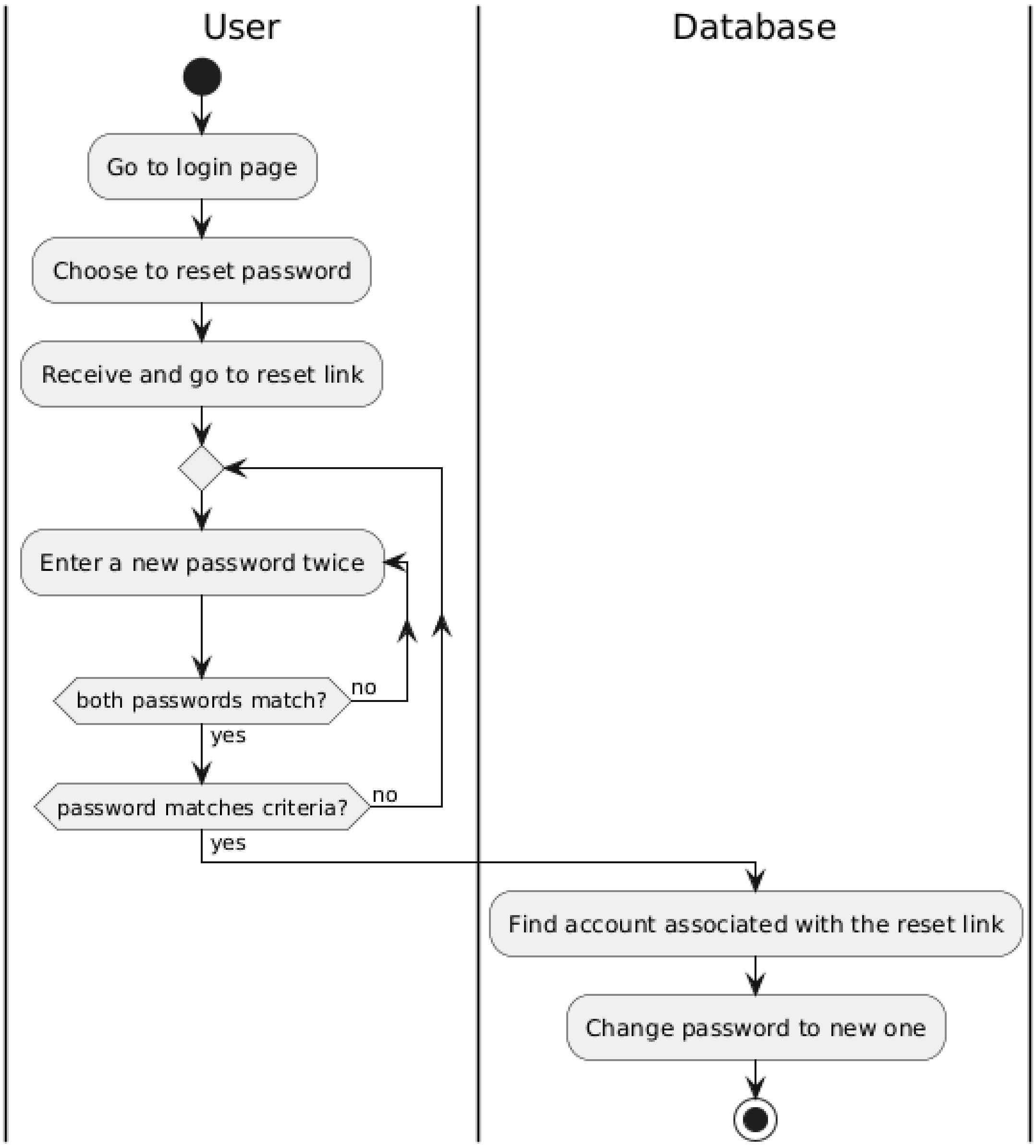
User
Database
Go to login page
Choose to reset password
Receive and go to reset link
Enter a new password twice
both passwords match?
no
yes
password matches criteria?
no
yes
Find account associated with the reset link
Change password to new one

Fig. 5. Activity Diagram for Password Reset

Register

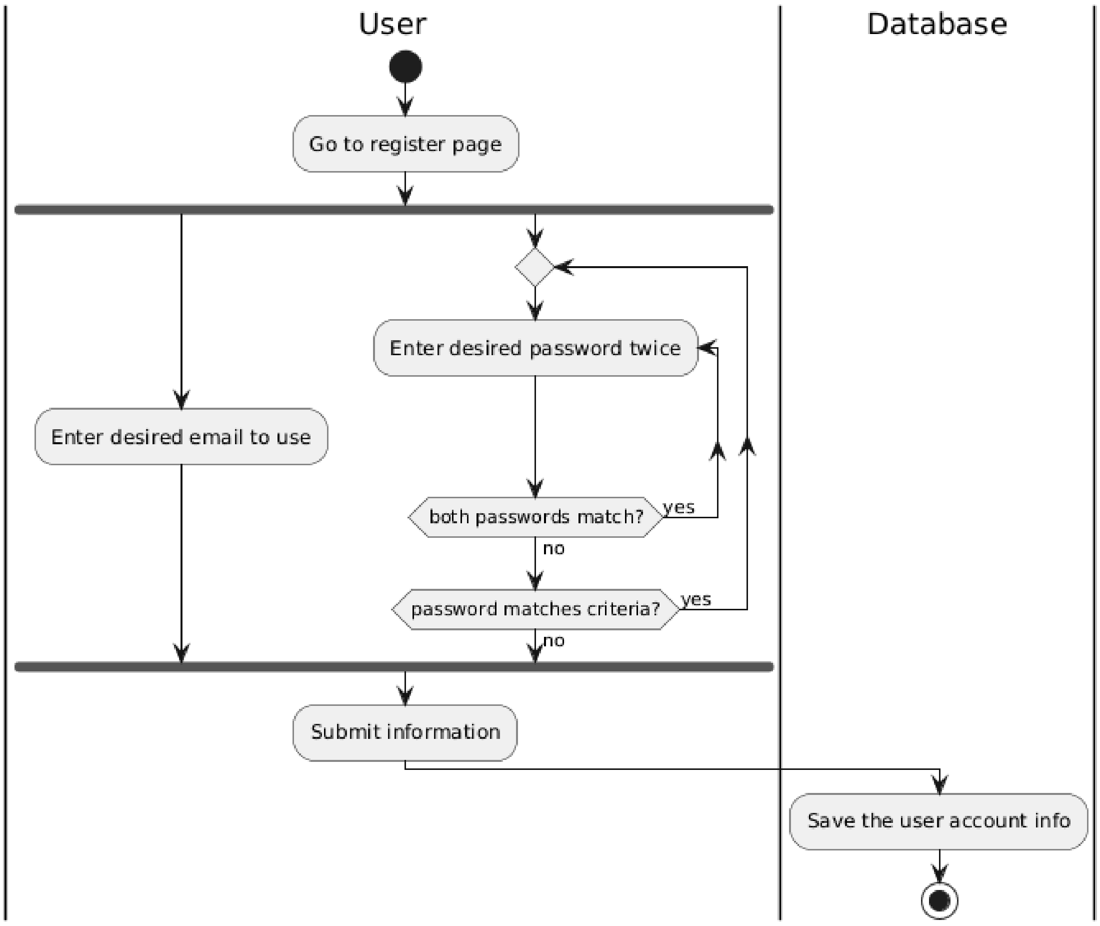
User
Database
Go to register page
Enter desired email to use
Enter desired password twice
both passwords match?
yes
no
password matches criteria?
yes
no
Submit information
Save the user account info

Fig. 6. Activity Diagram for Registration

Doctor Choosing a Patient

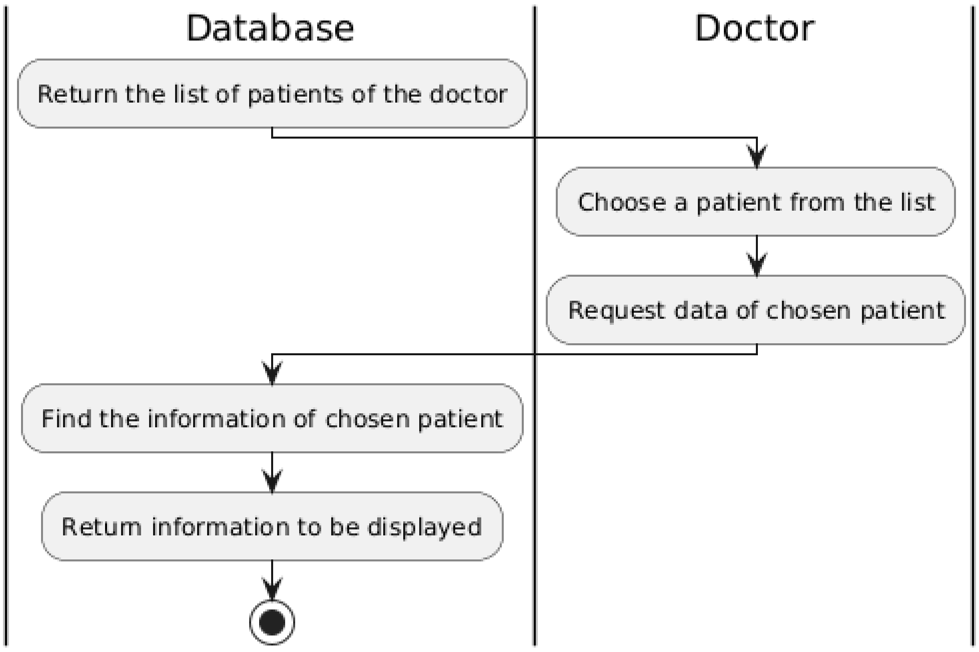
Database
Doctor
Return the list of patients of the doctor
Choose a patient from the list
Request data of chosen patient
Find the information of chosen patient
Return information to be displayed

Fig. 7. Activity Diagram for Patient Selection

Filter Timeline

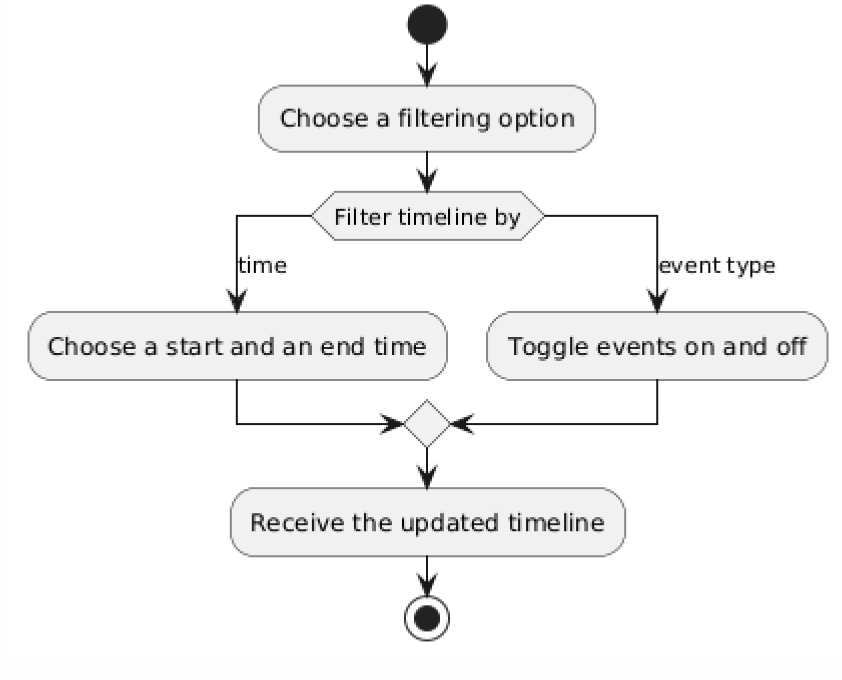
Choose a filtering option
Filter timeline by
time
event type
Choose a start and an end time
Toggle events on and off
Receive the updated timeline

Fig. 8. Activity Diagram for Timeline Filtering

Visual Formatting of Timeline

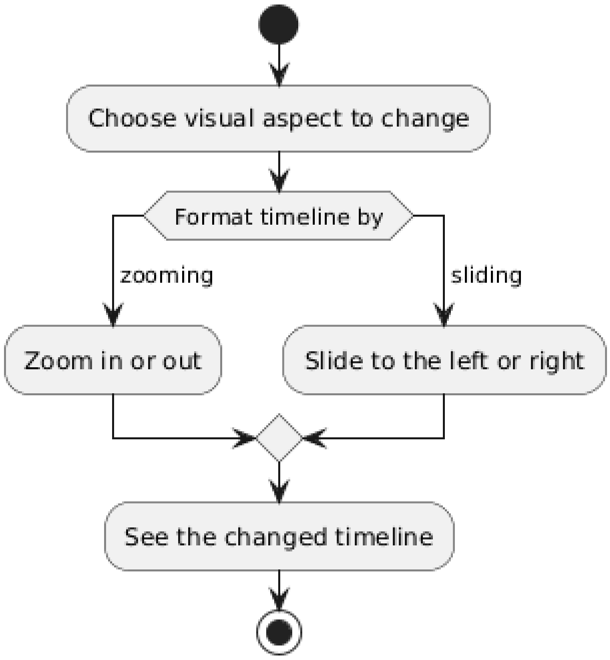

Fig. 9. Activity Diagram for Timeline Visual Formatting

Timeline Graph Creation

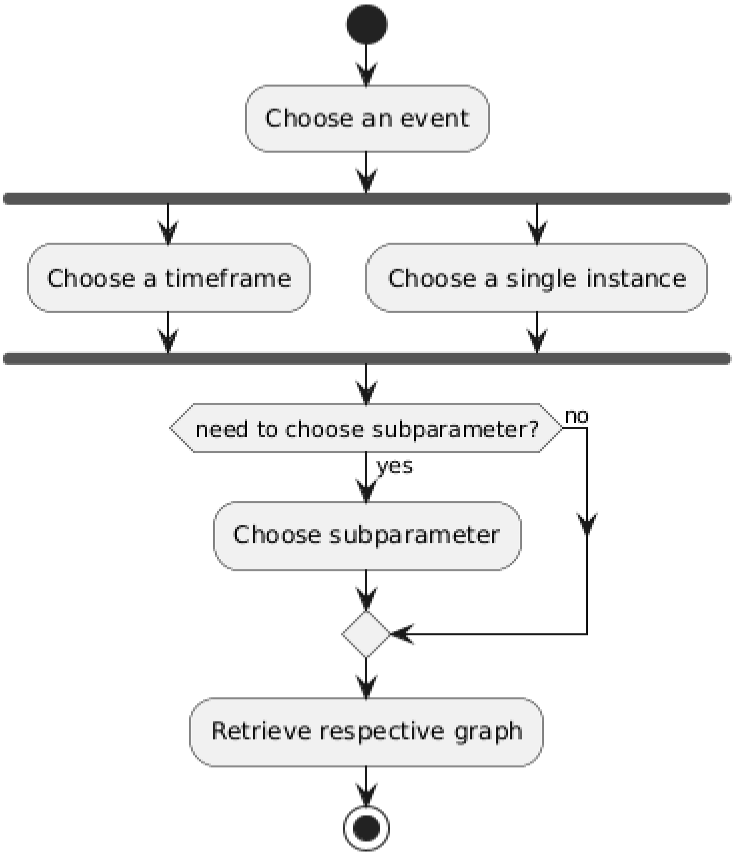

Fig. 10. Activity Diagram for Timeline Graph Creation

Graph Creation

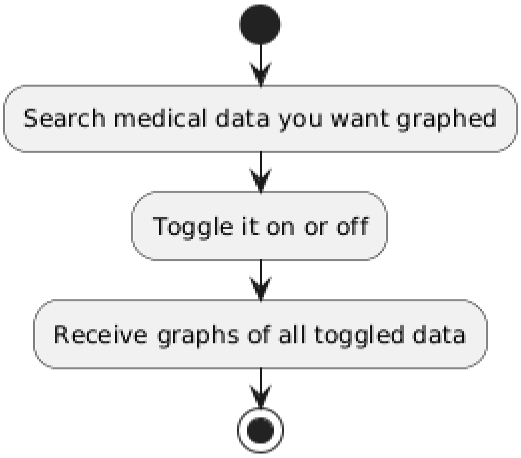
Search medical data you want graphed
Toggle it on or off
Receive graphs of all toggled data

Fig. 11. Activity Diagram for Graph Creation

Graph Customization

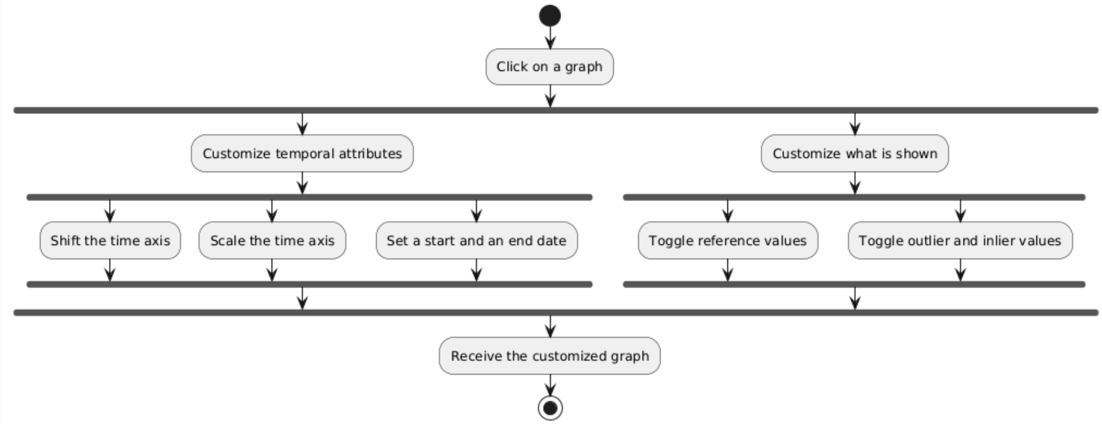
Click on a graph
Customize temporal attributes
Customize what is shown
Shift the time axis
Scale the time axis
Set a start and an end date
Toggle reference values
Toggle outlier and inlier values
Receive the customized graph

Fig. 12. Activity Diagram for Graph Customization

Graph Merging and Splitting

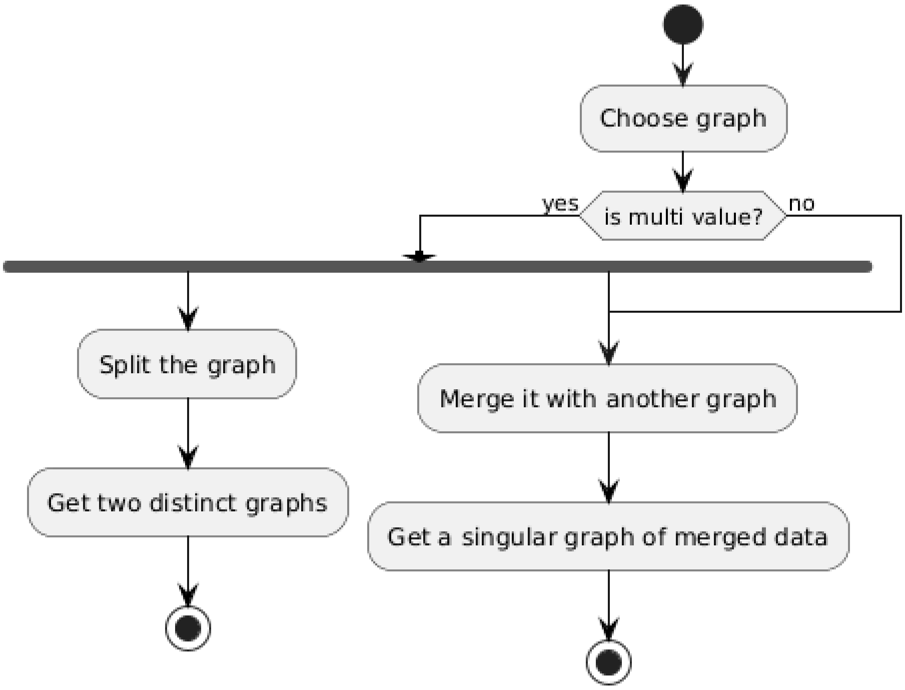

Fig. 13. Activity Diagram for Merging/Splitting

Export Timeline to Graphs Page

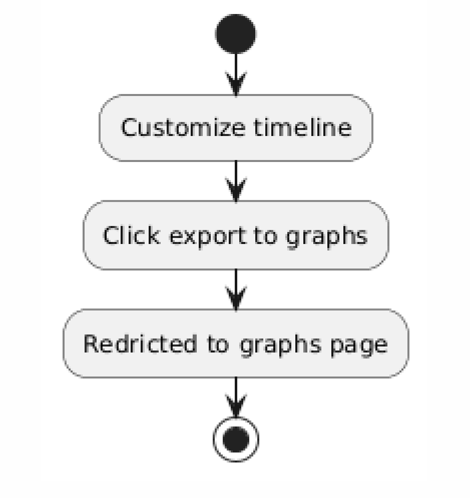

Fig. 14. Activity Diagram for Exporting Timelines to the Graph Page

Export Graphs to Timeline Page

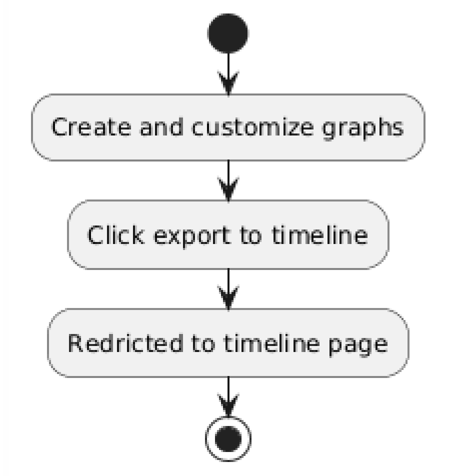

Fig. 15. Activity Diagram for Exporting Graphs to Timeline Page

Generic List Page Usage

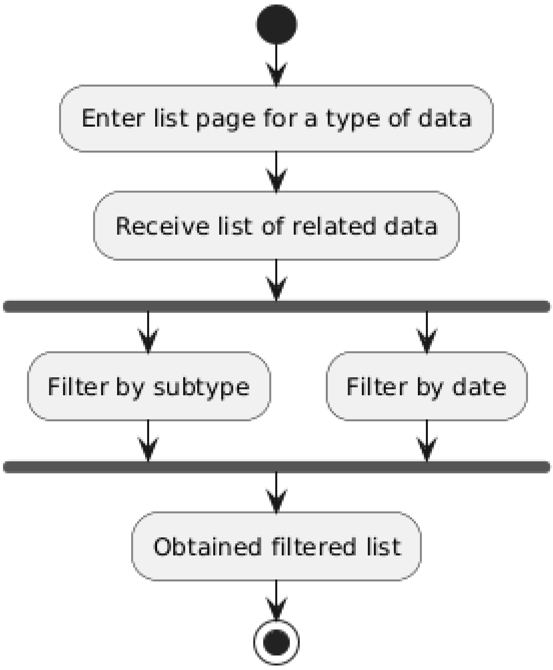
Enter list page for a type of data
Receive list of related data
Filter by subtype
Filter by date
Obtained filtered list

Fig. 16. Activity Diagram for List Page Usage

Sequence Diagrams

Register

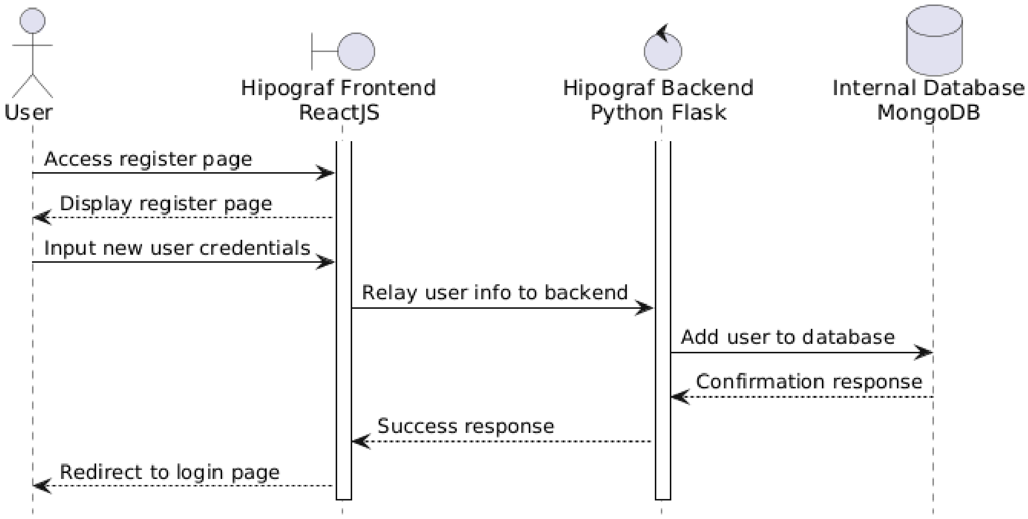
User
Hipograf Frontend ReactJS
Hipograf Backend Python Flask
Internal Database MongoDB
Access register page
Display register page
Input new user credentials
Relay user info to backend
Add user to database
Confirmation response
Success response
Redirect to login page

Fig. 17. Sequence Diagram for User Registration

Login

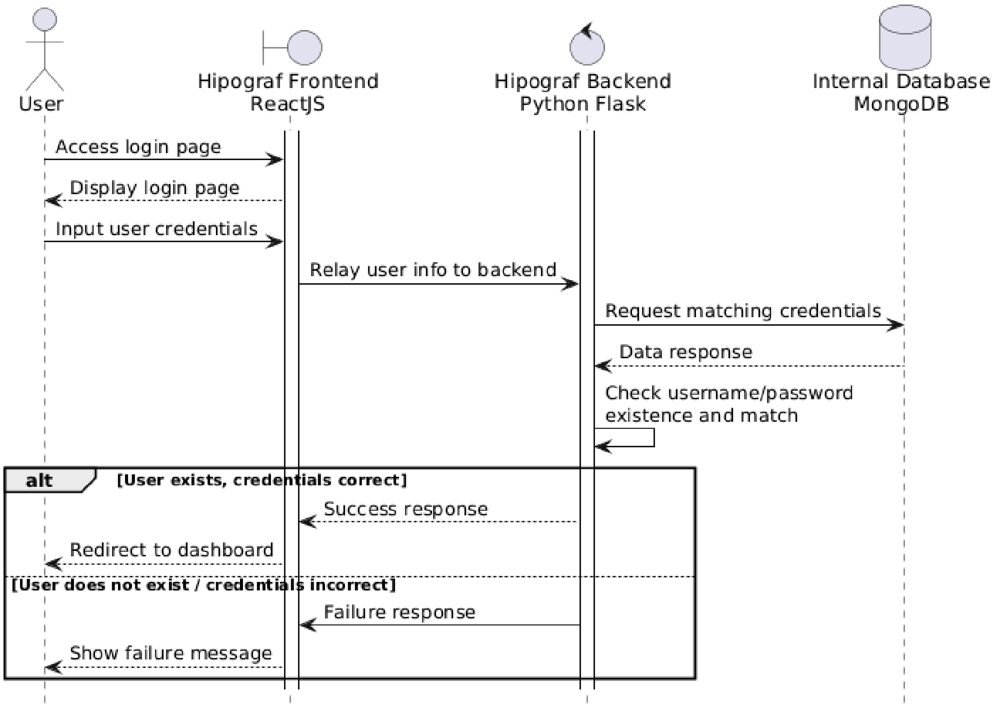
User
Hipograf Frontend ReactJS
Hipograf Backend Python Flask
Internal Database MongoDB
Access login page
Display login page
Input user credentials
Relay user info to backend
Request matching credentials
Data response
Check username/password existence and match
alt
[User exists, credentials correct]
Success response
Redirect to dashboard
[User does not exist / credentials incorrect]
Failure response
Show failure message

Fig. 18. Sequence Diagram for User Login.

Logout

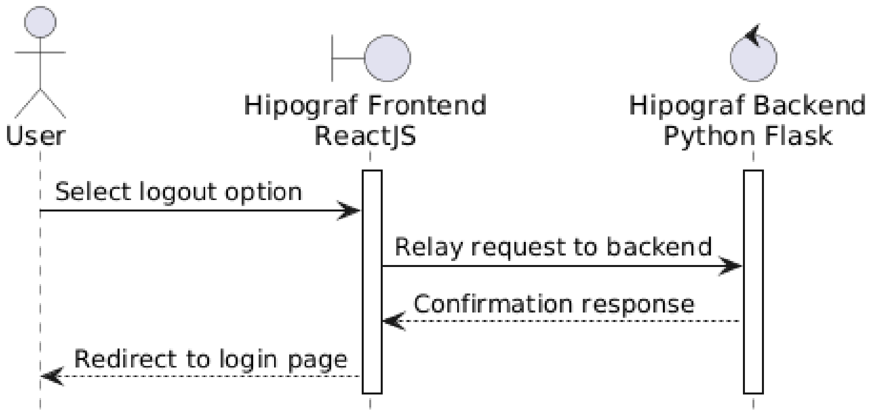

Fig. 19. Sequence Diagram for User Logout.

List Page Access

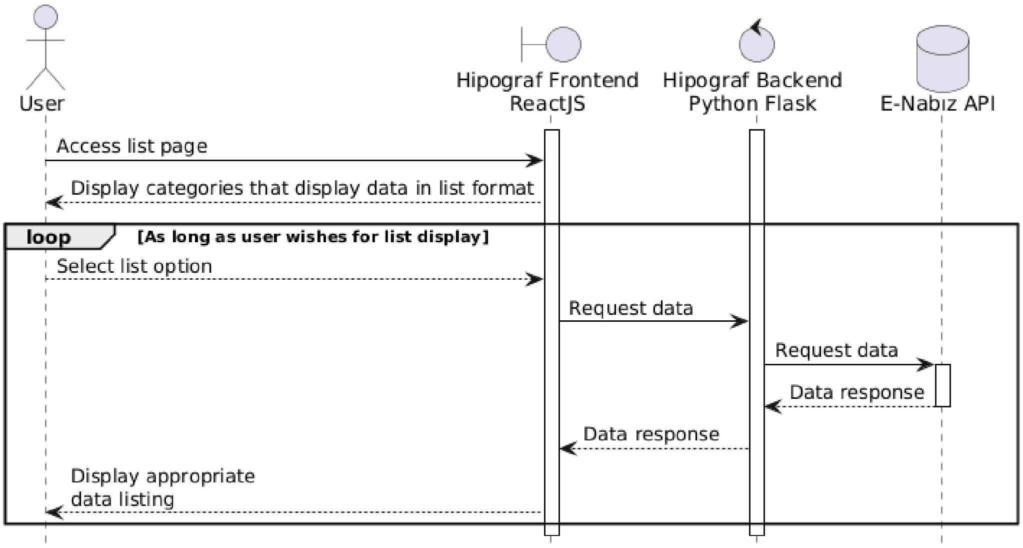

Fig. 20. Sequence Diagram for Initial Access of a List Page.

List Filtering

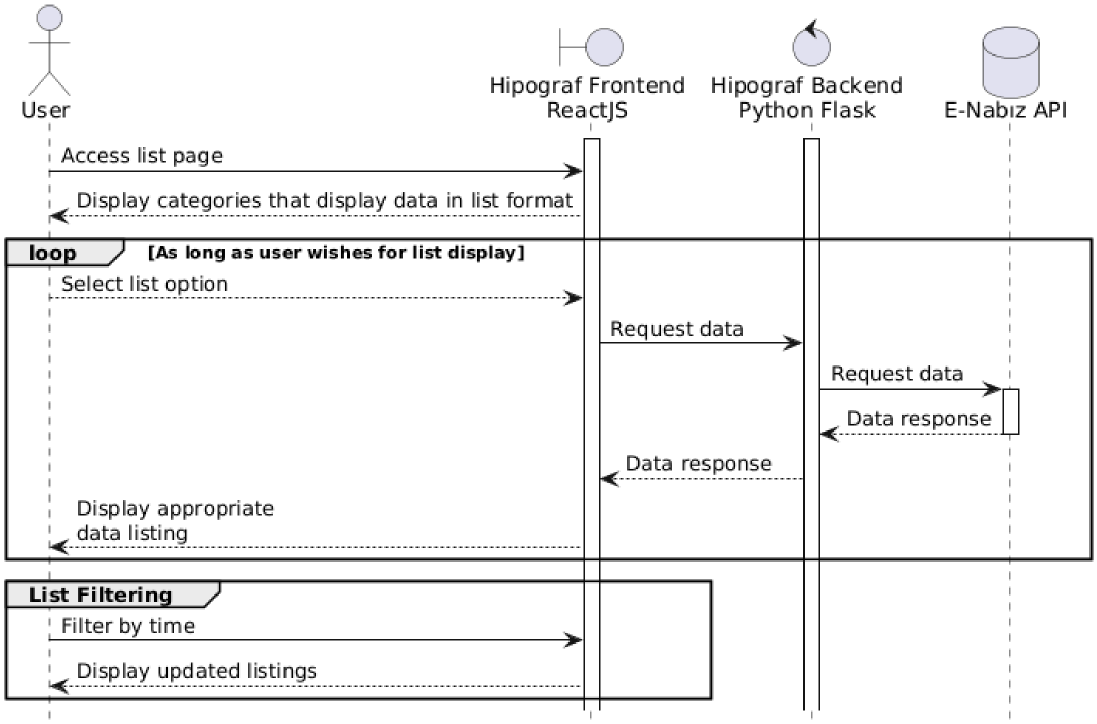

Fig. 21. Sequence Diagram for Data Filtering on the List Page.

Timeline Page Access

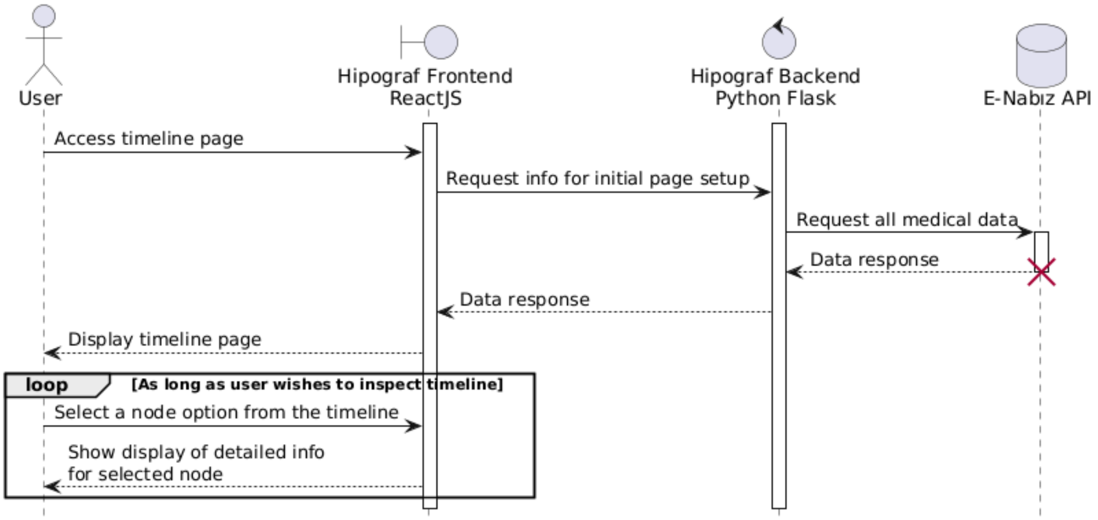

Fig. 22. Sequence Diagram for Initial Access of a Timeline Page.

Timeline-to-Graph

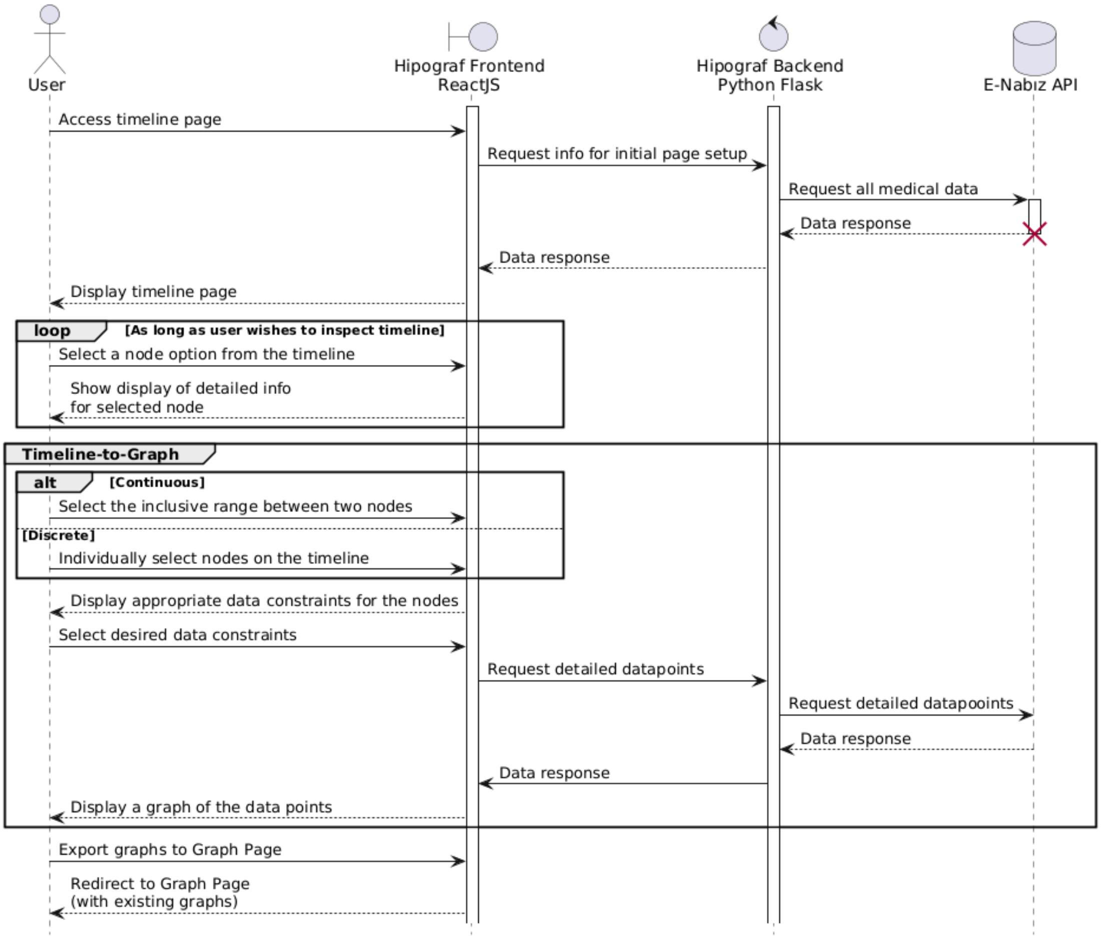
User
Hipograf Frontend
ReactJS
Hipograf Backend
Python Flask
E-Nabız API
Access timeline page
Request info for initial page setup
Request all medical data
Data response
Data response
Display timeline page
loop
[As long as user wishes to inspect timeline]
Select a node option from the timeline
Show display of detailed info
for selected node
Timeline-to-Graph
alt
[Continuous]
Select the inclusive range between two nodes
[Discrete]
Individually select nodes on the timeline
Display appropriate data constraints for the nodes
Select desired data constraints
Request detailed datapoints
Request detailed datapooints
Data response
Data response
Display a graph of the data points
Export graphs to Graph Page
Redirect to Graph Page
(with existing graphs)

Fig. 23. Sequence Diagram for Graph Creation on the Timeline Page.

Timeline Transformations

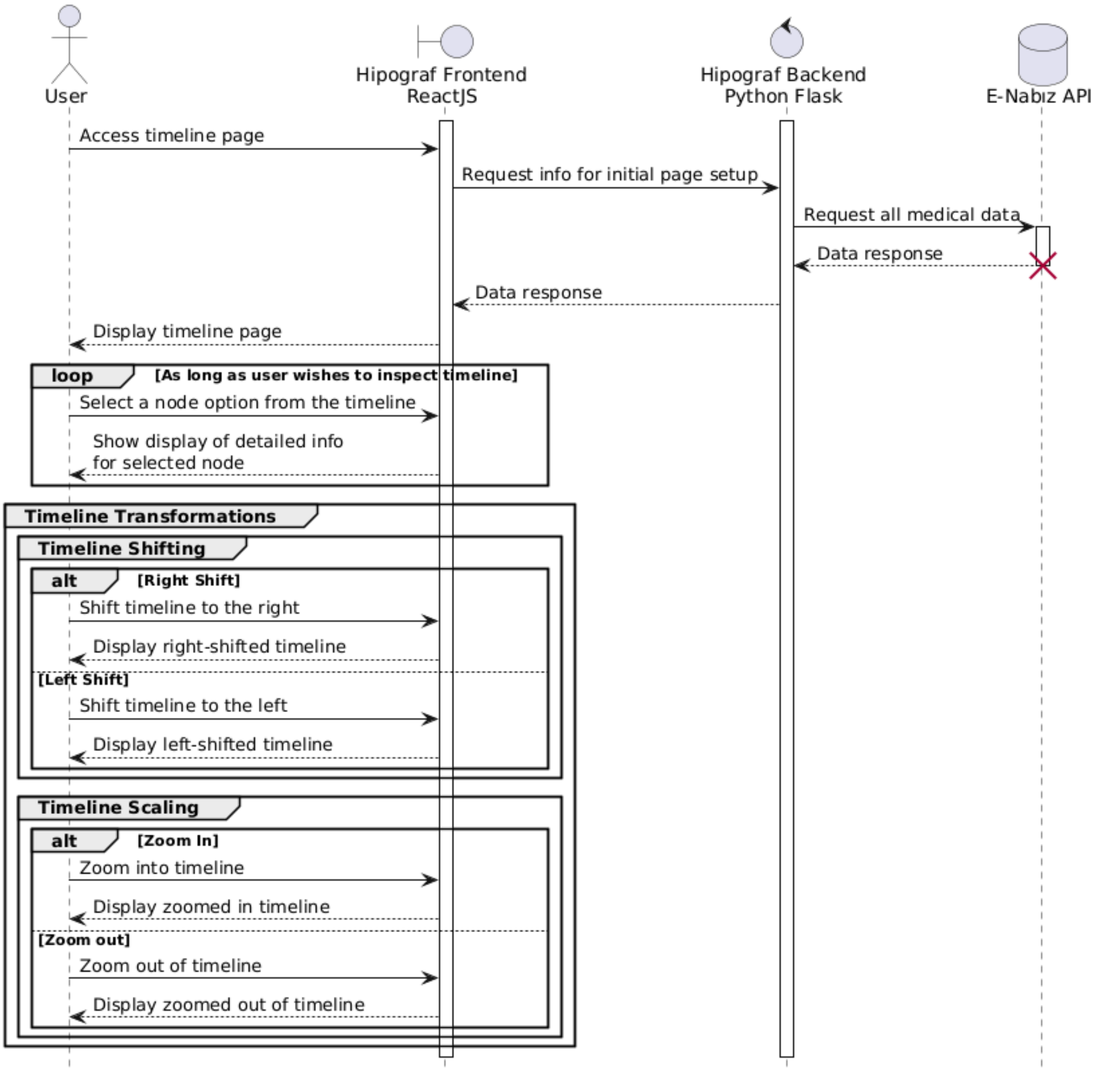

Fig. 24. Sequence Diagram for Transformations on the Timeline Page.

Graph Page Access

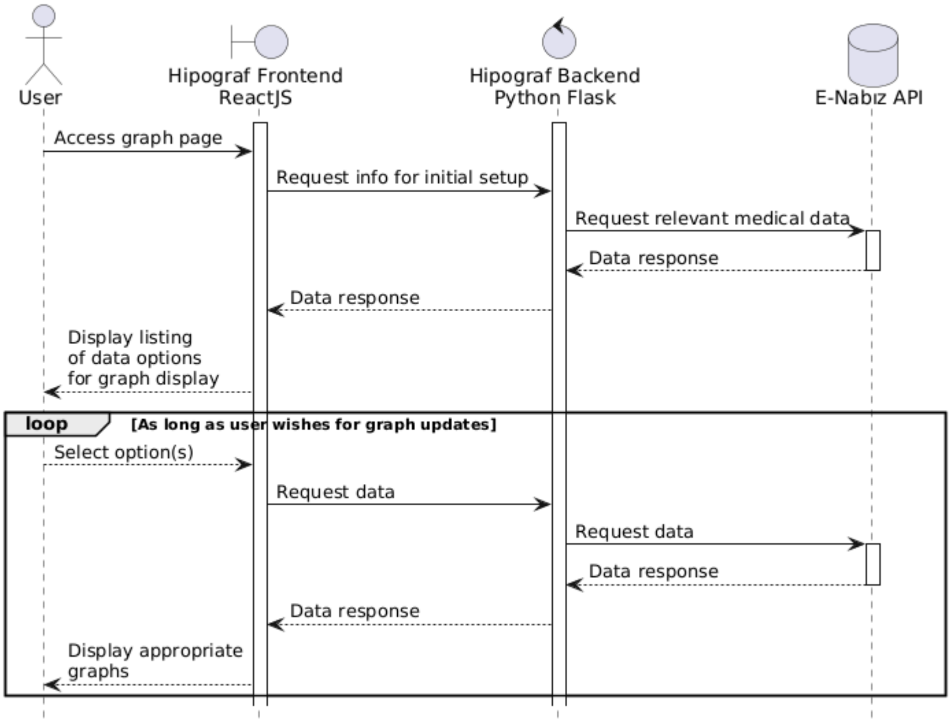
User
Hipograf Frontend ReactJS
Hipograf Backend Python Flask
E-Nabız API
Access graph page
Request info for initial setup
Request relevant medical data
Data response
Data response
Display listing of data options for graph display
loop
[As long as user wishes for graph updates]
Select option(s)
Request data
Request data
Data response
Data response
Display appropriate graphs

Fig. 25. Sequence Diagram for Initial Access of a Graph Page.

Graph Transformations

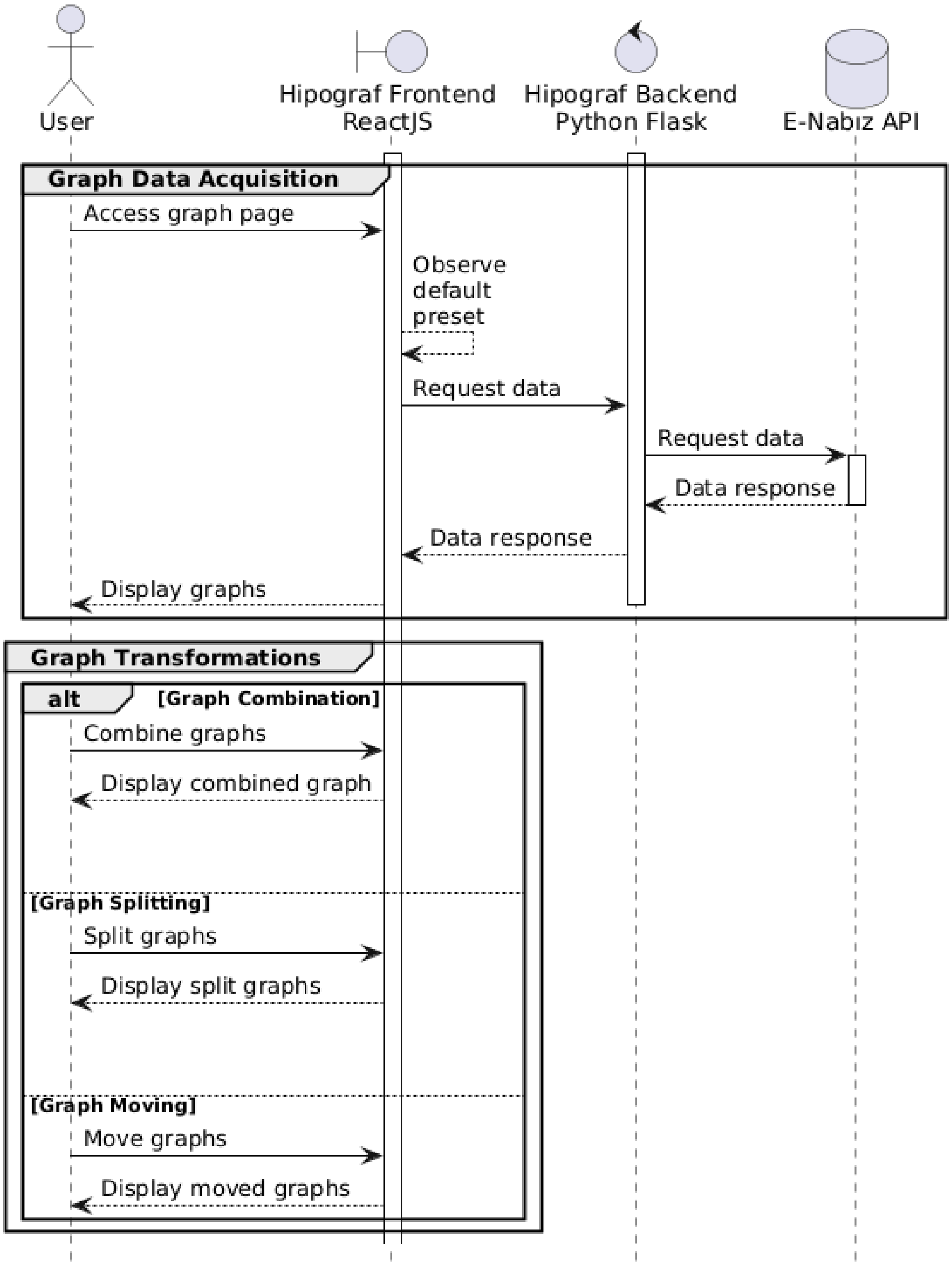

Fig. 26. Sequence Diagram Transformations on the Graph Page.

Graph Preset Saving

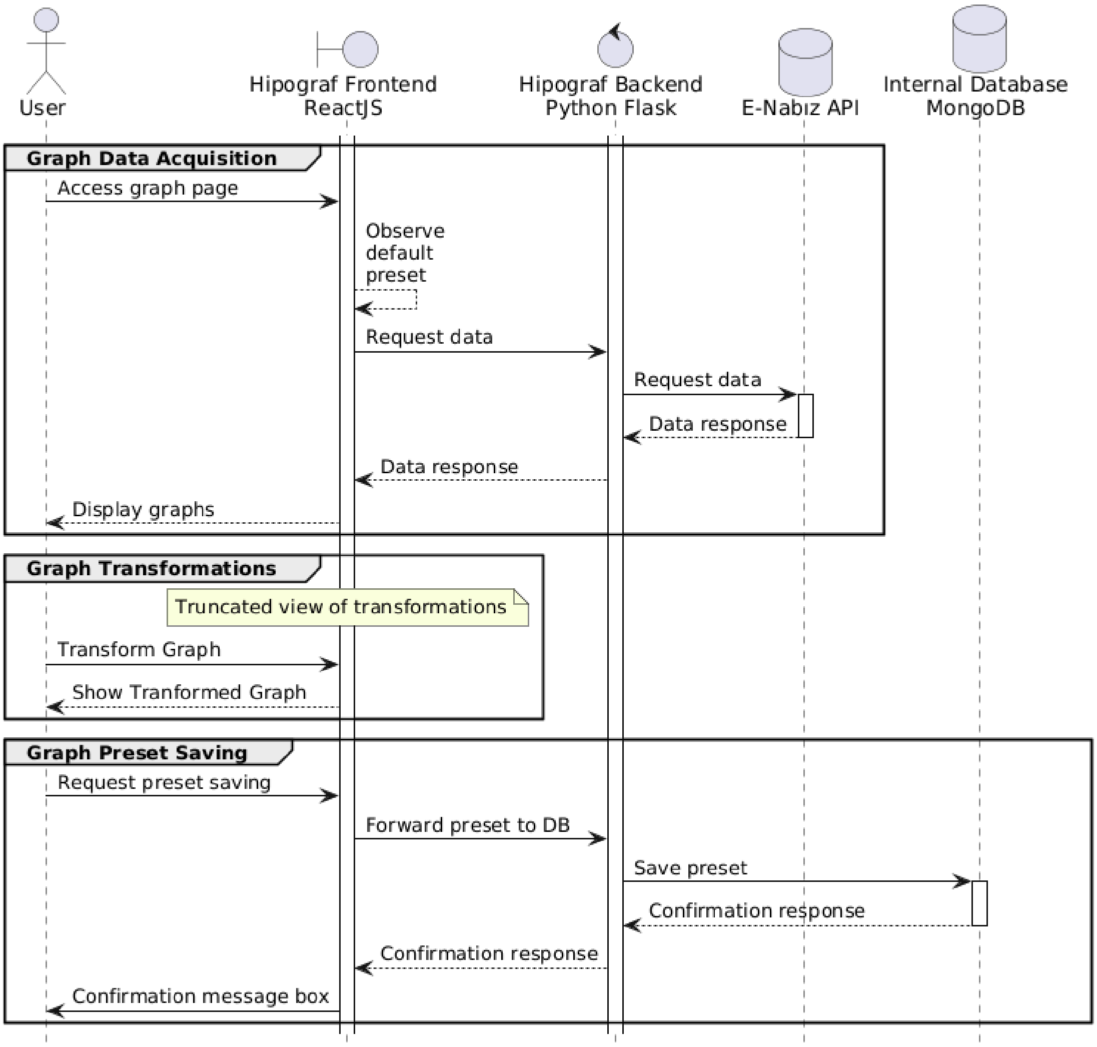

Fig. 27. Sequence Diagram Transformations Graph Preset Saving.

3.5 Structural UML Diagrams

Object Diagram

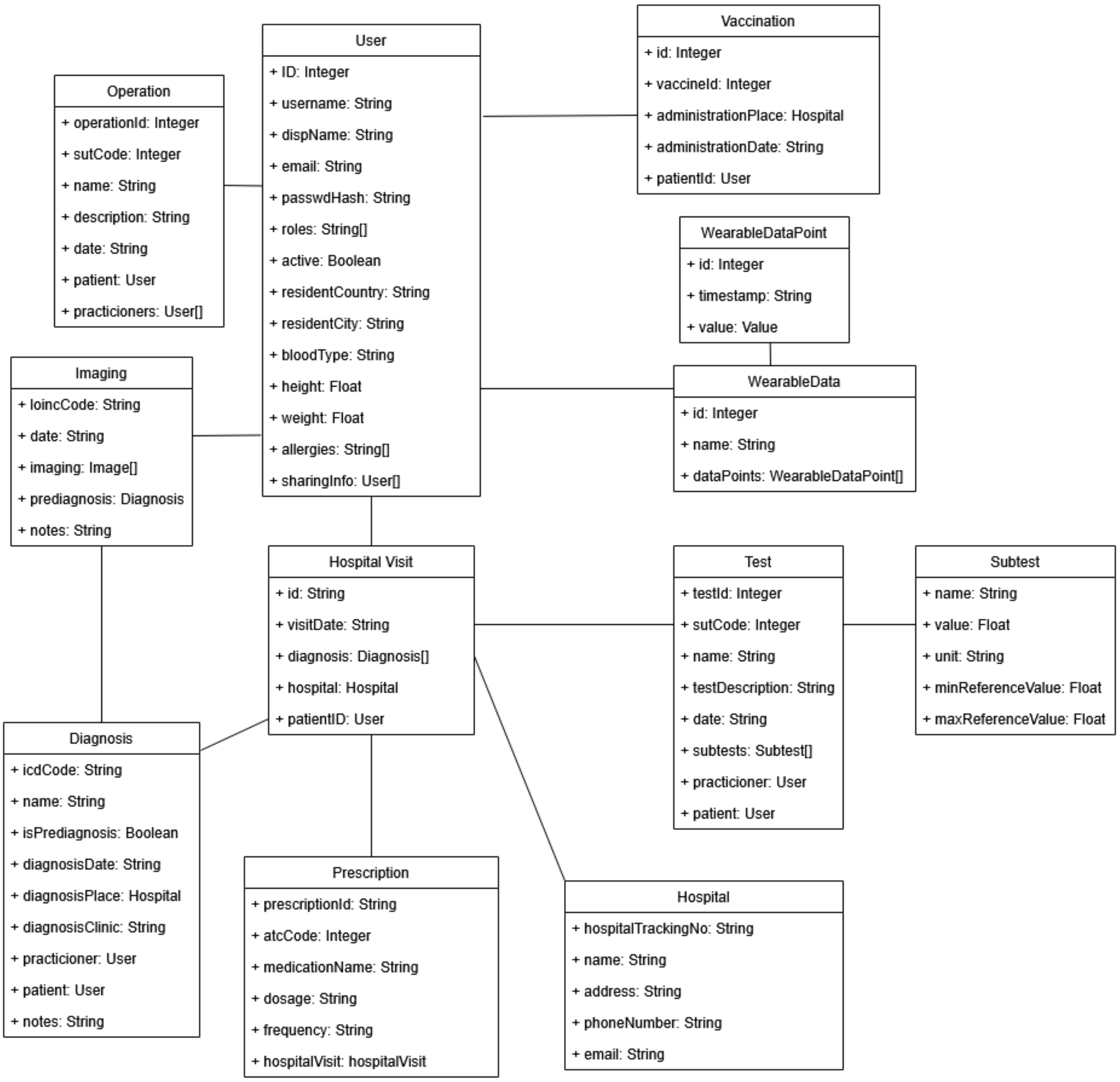

Fig. 28. Object Diagram

Class Diagram

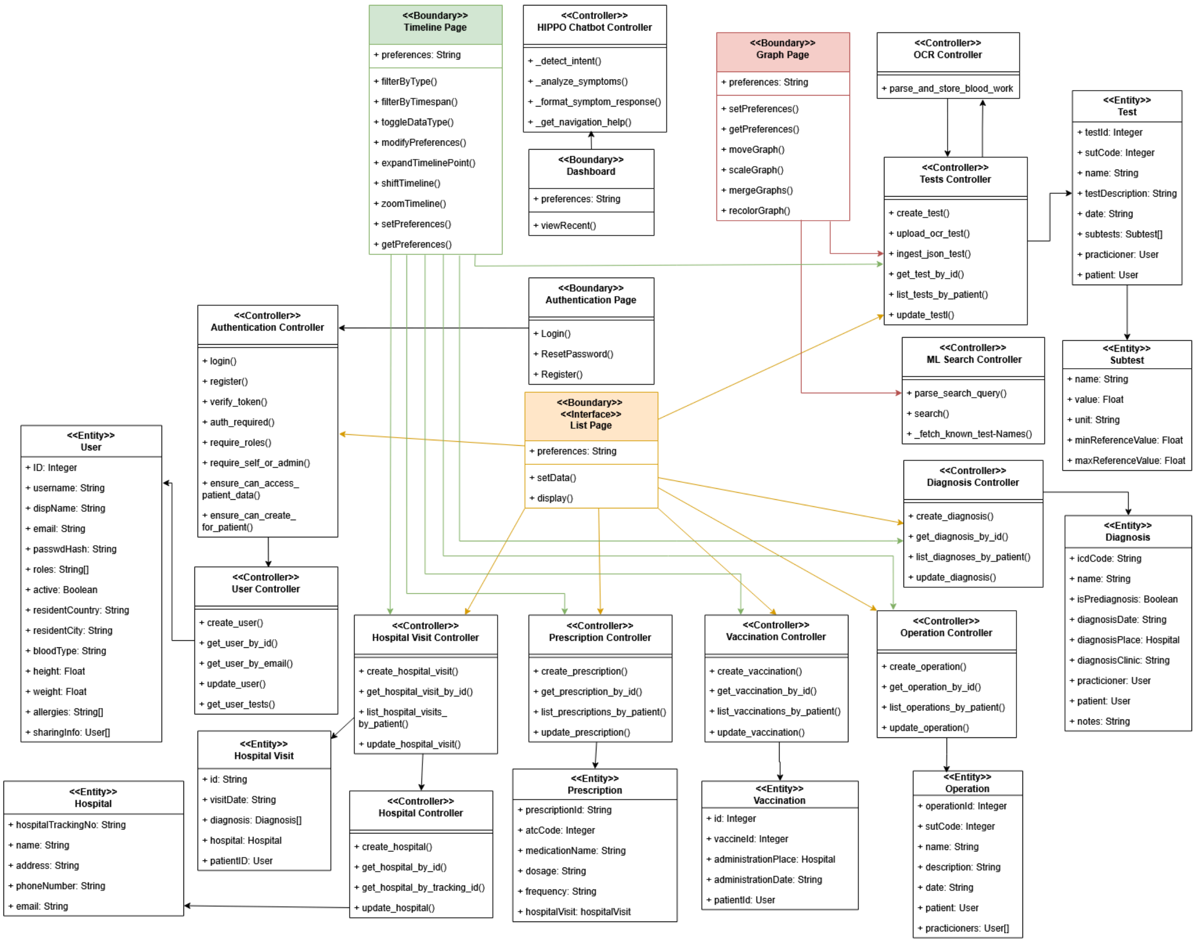

Fig. 29. Class Diagram

3.6 Hardware/Software Mapping

Hipograf will make use of traditional consumer computer hardware available in the medical industry in Türkiye. Its primary focus will be accessibility from any standard computer that is able to connect to the internet. To this end, it will be a web-based application that exists in the standard internet ecosystem.

Hipograf will make use of a variety of software products and packages. First and foremost, Hipograf functions by being connected to external APIs. After the user logs into their E-Nabız account and grants Hipograf access to their medical information, their relevant medical data will be retrieved from the relevant external APIs. As a web application, Hipograf is also reliant on the various software and packages used to build and host the website itself. In particular, it uses React TypeScript and Tailwind CSS. Hipograf will also be connected to a database for storing basic user information such as login credentials, data display preferences and additional user submitted information. This database will be an instance of MongoDB. The backend will link the database to the frontend. The backend of Hipograf will be built using the Flask web framework and will be written in Python.

3.7 Persistent Data Management

As Hipograf is only supposed to be a visualization solution, it is imperative that it does not modify any medical record from any system it is integrated with. Its only role in relation to these systems is to fetch data and safely discard them afterwards. This data impermanence policy does not apply to application-specific data like the data presets of graphs. It also does not apply to anything that the user specifically decides to temporarily upload into the system.

Due to the fact that the application development must be done before integration with external APIs like e-Nabız, the application will have to store some medical data to construct the minimal viable product. This data will come from either the developer team's own medical information per their consent, or be generated data that does not belong to a real patient. After proper integration, the application would stop this storage of medical data.

3.8 Access Control and Security

Access to Hipograf will be limited to logged-in users. All pages (and associated endpoints) in the application apart from the login, account creation and password reset will require standard web authentication. Different kinds of users will be available in the application. User roles include two normal roles, those being practitioners and patients. It also has admins. Practitioners and patients have similar access permissions, with the main difference being that practitioners also have a list of patients that they are responsible for. Admin accounts have elevated permissions, such as being able to edit the account details of another user (when ordinarily a normal user would only be able to edit their own account).

On the security side of things, web-based access to Hipograf will be over an HTTPS connection using Transport Layer Security version 1.3, which itself will use the Advanced Encryption Standard in Galois Counter Mode with 256-bit keys.
Hipograf passwords will have to meet the following requirements:

- At least eight (8) alphanumeric characters
- At least one (1) uppercase letter
- At least one (1) lowercase letter
- At least one (1) numeric digit
- At least one (1) special character

These passwords will be hashed using the Argon2 password hashing algorithm and stored in a database that will not be directly web-facing.

4. Subsystem Services

This section presents a detailed overview of all the subsystems described in the subsystem diagram of Section 3.2.

Presentation Layer

The presentation layer consists of the application web frontend that runs as a React TypeScript application that is built to be served as a static set of HTML, CSS and JavaScript files by the Apache httpd web server.

Page Routing Module

The frontend application uses React Router for the local page routing, which is handled internally and remains fully functional when the application frontend is built for deployment.

Filtering Module

Data fetched from the backend API endpoints largely arrive in their unfiltered form. Most of the data filtration takes place on the frontend, minimizing the amount of data processing that is required from the backend, instead leveraging the user’s own browser (and by extension, computer) to perform these operations.

Application Layer

The application layer of our project consists of a web backend written in Python’s Flask microframework. It is hosted with the Gunicorn HTTP server, with the API calls to the backend being reverse proxied from the frontend’s web server to avoid having a directly web-facing backend server.

Authentication Module

Hipograf’s user accounts use standard and secure web authentication using localStorage based Bearer tokens and endpoints (both frontend and backend) secured behind authentication. The User Controller and Authentication Controller (visible in the Class Diagram in Section 3.5) are closely intertwined, and many backend endpoints are only visible/particular to certain user roles (further elaborated on within Section 3.8).

Endpoint Routing Module

Most of the backend endpoints are served as part of a standard data access API to the frontend. These endpoints, most requiring authentication, serve JSON-formatted data to only the frontend of the Hipograf. It is on the same domain, existing behind the /api path. This is to prevent any potential Cross-Origin Request attacks.

OCR Module

The backend contains a detailed module for performing automated parsing of uploaded PDFs. Existing blood work reports, such as the ones available on e-Nabız, are submitted to the system. The module then proceeds to automatically identify the tested elements and their resulting values and couples it with the Hipograf system, whereupon the user can view this data on the List, Timeline, and Graph pages.

HIPPO Module

Another backend module is the HIPPO LLM-based chatbot component. On the frontend side, HIPPO is designated with a floating button that persists on the bottom left of the screen for authenticated web pages (i.e, everything visible after the user logs into the system). Users can ask HIPPO about the navigational structure of Hipograf, along with instructions on how to use a particular page's mechanisms.

ML Search Module

The final backend module is responsible for the advanced search mechanism present in the Graph page. This search mechanism allows the user to query, in natural language, for test parameters and date ranges and automatically retrieve plots associated with those parameters in the specified date ranges. The module uses an LLM to parse natural language and extract the relevant information from the provided query.

Data Layer

The Data Layer is split up into two main components. These are the external data fetched from external APIs, and also the MongoDB Database used to store certain components of application data (as detailed in Section 3.1).

API Gateway

The application models and data format has been designed with various pre-existing external APIs in mind, including platforms like e-Nabız. This is to ensure that external data can be efficiently fetched from these APIs and used in the Hipograf ecosystem.

MongoDB Database

The MongoDB Database is a MongoDB cluster used to store certain application and user data. The credentials required for authentication, and user customization/medical data used during development of the application are stored in this database. Connection with this database is made via the `pymongo` package, and is done through a secure authenticated channel. The backend includes several repository classes responsible for robustly performing operations on this database for each data model present.

5. Test Cases

We have devised 14 test cases for our application. Of these, the first 10 are integration tests that focus on the functionality of the application by verifying various user stories throughout their procedures. Every user story is covered at least once, and some are covered multiple times. The remaining 4 tests are non-functional tests that measure whether the application design goals have been met.

Test Case 1 - Patient Selection

Test ID	T2506-TC1	Category	Functional
Objective	This test case is to verify that the patient selection system is working as expected, allowing the application to fetch		
Steps	1. A practitioner registers to the Hipograf System. 2. The practitioner accesses the list of patients they have access to. 3. The practitioner selects a patient from the list. 4. The practitioner brings up the patient's prescription page and observes the prescriptions.		
Expected Outcome	The prescription page only displays the prescriptions of the selected patient.		
User Stories Covered	US1 - Registration US10 - List Patients US11 - Select Patient US19 - List Prescriptions		

Test Case 2 - Credential Changing

Test ID	T2506-TC2	Category	Functional
Objective	This test case is to verify elements of Hipograf's data persistence.		

Steps	1. The user logs in to the Hipograf system. 2. The user updates both their username and their password that was registered to their Hipograf account. 3. The user logs out and logs back in with their new password.
Expected Outcome	The username and password have been changed correctly, allowing for the user to access the application after changing them.
User Stories Covered	US2 - Login US6 - Account Information Update US5 - Password Change US3 - Logout

Test Case 3 - Blood Test Uploading

Test ID	T2506-TC3	Category	Functional
Objective	This test case is to verify that the upload blood test feature works as intended.		
Steps	1. Having forgotten their password, the user resets their Hipograf password and logs in to the system. 2. The user uploads several of their official blood tests as a PDF, which is then read and processed by the system. 3. The user checks the blood list page to ensure that the data has been uploaded correctly. 4. Finally, the user navigates to the timeline page to get a time-based perspective of their blood test data.		
Expected Outcome	The blood test data should be correctly displayed on the timeline, along with whatever other relevant data is fetched from the external API.		
User Stories Covered	US2 - Login US13 - Upload Blood Test US12 - List Blood Test US24 - Scroll Timeline		

Test Case 4 - Chatbot Usage

Test ID	T2506-TC4	Category	Functional
Objective	This test case is to verify that a brand new user can use the AI chatbot, HIPPO, to successfully navigate the application.		

Steps	1. A new user registers to Hipograf. 2. The user accesses the HIPPO chatbot on the dashboard and asks it about the application. 3. The user asks HIPPO where it can find information about past radiological images and is directed to the radiological image page.
Expected Outcome	Correct navigational responses by HIPPO should be provided.
User Stories Covered	US1 - Registration US9 - Chat with HIPPO US16 - List Radiological Images

Test Case 5 - Account Deletion

Test ID	T2506-TC5	Category	Functional
Objective	This test case is to verify that account deletion works correctly.		
Steps	1. The user logs into the Hipograf system. 2. The user demands that all of their preference data pertaining to timeline and graph selections be deleted. 3. The user then proceeds to delete their account entirely. 4. Having been logged out automatically, the user attempts to reset their password.		
Expected Outcome	The password reset should fail (it is an expected failure, the ideal outcome).		
User Stories Covered	US2 - Login US8 - Clear Preferences Data US4 - Account Deletion US7 - Reset Password		

Test Case 6 - Graph Preset Saving and Loading

Test ID	T2506-TC6	Category	Functional

Objective	This test case is to verify that previously saved graph presets are able to load correctly.
Steps	1. The user logs into the Hipograf system. 2. The user navigates to the graph page and creates a simple graph setup of particular wearable data that exists in the system. 3. The user recolors one of the graphs and saves their presets, making sure that the 4. The user exits the application and logs back in. 5. The user navigates to the graph page and loads their previously saved graph preferences.
Expected Outcome	The graphs that are loaded should match the previously saved graph.
User Stories Covered	US2 - Login US20 - List Wearable Data US28 - Create Graph US31 - Recolor Graph US34 - Save Preferences US35 - Load Preferences US3 - Logout

Test Case 7 - Timeline Preset Saving and Loading

Test ID	T2506-TC7	Category	Functional
Objective	This test case is to verify that previously saved timeline presets are able to load correctly.		
Steps	1. The user logs into the Hipograf application. 2. The user navigates to the timeline page and filters the visible events until only diagnoses in the past 6 months can be seen. 3. The user clicks on a particular diagnosis visible on the timeline for detail and ensures that they match the ones given on the diagnosis list page. 4. The user saves these timeline filtration preferences. 5. The user exits the application and logs back in. 6. The user navigates to the timeline page and loads their previously saved timeline preferences.		
Expected Outcome	The timeline that is loaded should match the previously saved timeline configuration settings.		

User Stories Covered	US2 - Login US14 - List Diagnosis US21 - Filter Event Category US26 - Save Preferences US27 - Load Preferences US25 - Obtain Event Information

Test Case 8 - List Page Consistency

Test ID	T2506-TC8	Category	Functional
Objective	This test case is to verify that information across various list pages are consistent with one another.		
Steps	1. The user logs into the application. 2. The user checks the hospital visit page for visits in the past year. 3. The user proceeds to check the operations and vaccinations pages in the same timeframe constraint.		
Expected Outcome	The vaccinations and operations listed should only have occurred on dates that have had hospital visits listed.		
User Stories Covered	US2 - Login US15 - List Hospital Visit US17 - List Operations US18 - List Vaccinations		

Test Case 9 - Graph In-Depth Features

Test ID	T2506-TC9	Category	Functional
Objective	This test case is to verify various facets of the extended features of the graph page.		

Steps	1. The user navigates to the graph page. 2. The user creates a graph manually. 3. The user creates a graph using the natural language search functionality. 4. The user resizes both graphs and chooses different timeframes for the data they are presenting. 5. The user combines the two graphs into one. 6. The user splits the graphs again into their original forms.
Expected Outcome	All graph operations conducted should give correct and appropriate results.
User Stories Covered	US28 - Create Graph US29 - Search for Graph US30 - Edit Graph US32 - Combine Graph US33 - Split Graph

Test Case 10 - Timeline In-Depth Features

Test ID	T2506-TC10	Category	Functional
Objective	This test case is to verify various facets of the extended features of the timeline page.		
Steps	1. The user logs into the application and navigates to the timeline page. 2. The user chooses a particular timeframe of the last year for the timeline. 3. The user changes the current timeframe to include only the last 6 months. 4. The user scrolls through the timeline, viewing segments before the selected timeframe. 5. The user filters the timeline so that only hospital visits will remain. 6. The user clicks on one of the hospital visit nodes to obtain a brief description about the event.		
Expected Outcome	All timeline operations conducted should give correct and appropriate results.		
User Stories Covered	US21 - Filter Event Category US22 - Choose Timeframe US23 - Edit Timeframe US24 - Scroll Timeline US25 - Obtain Event Information		

Test Case 11 - Reliability

Test ID	T2506-TC11	Category	Non-Functional
Objective	Verifying the reliability of various components of the Hipograf architecture.		
Steps	1. Monitor the status of the frontend server for a period of three weeks using systemd's journalctl module. 2. Repeat the previous step for the backend server. 3. Check the statistics for the uptime of the MongoDB database from its administrative dashboard.		
Expected Outcome	An uptime of over 95% should be seen across the whole suite of systems tested.		

Test Case 12 - Performance

Test ID	T2506-TC11	Category	Non-Functional
Objective	Verifying application performance using the online functionality provided by tools like Google PageSpeed Insights [15].		
Steps	1. Using Google PageSpeed Insights, measure the time taken to access the login page (after having emptied the browser's page cache) 2. Measure the time it takes for a majority of HTML objects to be loaded into the Dashboard using the same tool 3. Continue Step 2 for major application pages		
Expected Outcome	The planned maximum time to load the main content of Hipograf should be less than 2 seconds, and the input delay between subsequent operations should be less than 200ms.		

Test Case 13 - Security

Test ID	T2506-TC11	Category	Non-Functional
Objective	Verifying that various aspects of Hipograf's application-wide security expectations on all three layers (Presentation /Application/Data) meet existing standards [16].		
Steps	1. Check through the web browser used to connect to Hipograf that the connection is an HTTPS connection 2. On the same medium, check the validity of the TLS certificate. 3. Check the MongoDB database to ensure that user passwords in the User model field 'passwdHash' are hashed correctly, as specified. 4. Ensure that the password length/content requirement checks are in-place in both the backend and the frontend.		
Expected Outcome	All manually conducted baseline security checks should be successful.		

Test Case 14 - Usability

Test ID	T2506-TC11	Category	Non-Functional
Objective	Verifying the WCAG 2.1 accessibility guidelines for the application webpages concerning limited input vectors in scenarios where a mouse is not accessible or unusable [3].		
Steps	1. Navigate to the Login Page and verify that the elements of the page are all accessible by sole use of the keyboard. 2. Navigate to the Dashboard and perform the same verification. 3. Navigate to the Timeline Page and perform the same verification. 4. Navigate to the Graph Page and perform the same verification. 5. Navigate to the List Pages and perform the same verification.		
Expected Outcome	All core functionalities of the application are accessible only by using the keyboard (without the mouse).		

6. Consideration of Various Factors in Engineering Design

6.1 Standards

Throughout the course of the project, we have followed various standards in multiple different categories. These include engineering standards, legal standards and medical standards. Engineering standards specifically concern both the planning/design and the implementation of the project. Legal standards concern data privacy. Medical standards concern industry conventions for the labelling of various medical terms. Relevant definitions and the meaning of abbreviations can be found in Section 1.3.

Requirements Engineering and Documentation

For requirements engineering, we have sought standardization documents jointly published by the ISO, IEC and IEEE. These documents include ISO/IEC/IEEE 29148:2018 and ISO/IEC/IEEE 12207:2017. The first document gives an overview of requirements engineering for multiple different project types, including SRS which is the one that we are concerned with. The second document expands on this, also being cited in the first document, providing more detail on how various fields are expected to be filled out in the software case [2, 17].

Design Modelling

We refer to the UML 2.5.1 modelling standards published by the OMG for the system's design, encapsulated with the suite of structural and behavioral diagrams that are defined with UML including use case diagrams, activity diagrams, state diagrams and class diagrams [4].

Web Accessibility

In order to ensure that Hipograf remains accessible to as many people as possible, we intend to follow the WCAG 2.1, which are detailed web accessibility standards published by the W3C [3].

Privacy of Sensitive Data

In order to ensure that we are complying with national laws when dealing with patient data, we are referring to KVKK guidelines in dealing with sensitive data in a legally acceptable manner [8].

Disease Classification

The ICD is a detailed compendium of medical ailments and afflictions that is maintained by the WHO. Each disease or condition is provided with its own unique alphanumeric identifier code. Similarly classed or otherwise related conditions are given similar identifiers [5].

Disease Classification - Turkish Standard

The Turkish Ministry of Health keeps an online and publically accessible reference database for medical encodings and codes used in Türkiye, known as the ‘Sağlık Kodlama Referans Sunucusu’ (SKRS). One of these includes a translated and reformatted set of the aforementioned ICD-10 codes [6].

Treatment/Medication/Equipment Standards

The SKRS also contains other standards, including the SUT codes. This standard defines similar alphanumeric codes for various treatments, medication and equipment stationed at medical institutions around Türkiye [6].

6.2 Constraints

Rationale For Web Application

Hipograf will be primarily built for use on a standard personal use computer. This constraint exists because the devices accessing the application in a medical facility or similar setting are most likely to be a desktop or laptop computer, which entails comparable screen sizes and identical peripherals. Despite this expectation, we have not opted for a native desktop application. The issue is that a desktop application would make the application impossible to use from a mobile device without a direct port. We desired to keep the option of accessing Hipograf from mobile devices available even if the touchscreen reduces usability.

Security

Hipograf is designed to operate on private information in a state where it is computationally infeasible for an adversarial actor to gain malicious access. This requires that the system has proper authentication and authorization in place, which has been discussed in more detail in Section 3.5.

Public Health

As it is concerned with the health of the public, any changes to the general health of the national public has a great likelihood of affecting Hipograf. Health-related crises are likely to increase the load on Hipograf, requiring more performant servers.

Public Safety

Hipograf is not meaningfully influenced by the existing state of public safety. In the event that public safety is gravely threatened for unrelated reasons, Hipograf usage will be similarly affected.

Public Welfare

If a local populace's buying power was to be affected, it is possible that Hipograf would have to adapt to suit the new user base's computer specifications by further decreasing performance requirements.

Global Factors

Hipograf is only concerned with the E-Nabız system in Türkiye.

Cultural Factors

Hipograf is not influenced by cultural factors. It is designed to be used by medical entities that are themselves tied to state operations. A shift in culture will not change the usage of Hipograf.

Social Factors

Hipograf is not subject to social factors.

Environmental Factors

Hipograf is not subject to environmental factors.

Economic Factors

Hipograf will be a free-to-use software application. Its source code is under a permissive license and thus will be free-to-distribute.

6.3 Relevant Tables

Table I
Factors that can affect analysis and design

	Effect level	Effect
Public Health	High	More robust performance requirements
Public Safety	Very Low	Could reduce usage
Public Welfare	Low	More performant code
Global Factors	N/A	None

Cultural Factors	N/A	None
Social Factors	N/A	None
Environmental Factors	N/A	None
Economic Factors	N/A	None

Table II
Application Level Risk and Alternatives

Risk	Likelihood	Effect on the project	B Plan Summary
Integration Difficulties with E-Nabız	Medium	Patients can not automatically use their medical data stored in E-Nabız.	Release the software as a standalone application, allowing users to import their medical data semi-automatically .
Data Misrepresentation	Low	While a list is plain and unambiguous, visual representation gives a user much stronger intuition for pattern detection. There exists a risk that a mistake in how a graph plots data creates confusion in the representation of this data.	Re-evaluate the way in which the existing data is represented through the visual components and modify these components accordingly.
Data Leak	Low	The preferences of the users can be leaked, leading to reduced trust in the user base.	Create an incident report detailing the events that led to the database leakage, and improve the security based on the results of the report.

7. Teamwork Details

7.1 Contributing and functioning effectively on the team

Team Hipograf follows a hybrid development schedule wherein different team members work on different parts of the project at different times. This has two benefits: Everyone is familiar with every facet of the development process, and as an extension of this fact, members can be assigned to work on more urgent sections without significant delay. In the most recent burst of productivity:

- Salih Furkan Göktaş focused on the planning, writing and diagrammatizing of the report, along with deployment of the application suite on a remote server.
- Orhun Güder focused on the OCR component of the application backend and assisted with the report.
- Şükrü Eren Gökırmak worked on broad areas of the backend, including the ML Search module of the graph pages.
- Artun Berke Gül helped to locate areas of possible discrepancy between the frontend pages and backend models and assisted with the data layer integrations.
- Ramiz Arda Ünal worked on the general frontend design of the application and also worked on the HIPPO module of the application backend.

It should be noted that the amount of work done by each team member is roughly equivalent to every other team member.

7.2 Helping create a collaborative and inclusive environment

Creating a collaborative and inclusive environment is a very delicate process because everybody needs to be willing to do everything they can do to help, as the unwillingness of a single team member has the potential to hamper the efforts of the rest. In other words, everyone has to contribute equally. Thanks to our existing intra-group synergy, strong personal bonds and dedication to nurturing this environment amongst one another, everyone in Team Hipograf contributes to this equally. It wouldn’t be possible without everyone giving it their all. As such, there is not much to say about individual team members in this section.

7.3 Taking lead role and sharing leadership on the team

Team Hipograf follows a somewhat decentralized leadership structure where people take up leadership positions as temporary leaders of the hybrid work compartments described in Section 7.1. In our most recent ventures, the following leadership positions have been established:

- Ramiz Arda Ünal took the lead role in frontend development and design.
- Salih Furkan Göktaş took the lead role in report writing and system modelling.
- Orhun Güder took the lead role in the OCR module backend subsystem.
- Artun Berke Gül took the lead role in the application layer interactions.
- Şükrü Eren Gökırmak took the lead role in general backend development.

8. References

[1] “e-Nabız V.2.1 Kullanım Kılavuzu 2025”. https://enabiz.gov.tr/document/User_Manual.pdf. [Accessed: Nov 27, 2025].

[2] “IEEE/ISO/IEC 29148-2018”. https://standards.ieee.org/ieee/29148/6937/. [Accessed: Nov 27, 2025].

[3] “Web Content Accessibility Guidelines (WCAG) 2.1”. https://www.w3.org/TR/2025/REC-WCAG21-20250506/. [Accessed: Nov 27, 2025].

[4] “Unified Modelling Language, v.2.5.1”. https://www.omg.org/spec/UML/2.5.1/PDF. [Accessed: Nov 27, 2025].

[5] “ICD-10 Version:2019”. https://icd.who.int/browse10/2019/en. [Accessed: Nov 27, 2025].

[6] “Sağlık Kodlama Referans Sunucusu”. https://skrs.saglik.gov.tr/. [Accessed: Nov 27, 2025].

[7] “SUT NEDİR?”. https://old.peduro.org.tr/sut_nedir.php. [Accessed: Nov 27, 2025]

[8] “Mevzuat Bilgi Sistemi”. https://www.mevzuat.gov.tr/mevzuat?MevzuatNo=6698&MevzuatTur=1&MevzuatTertip=5. [Accessed: Nov 27, 2025].

[9] “Power BI - Data Visualization | Microsoft Power Platform” https://www.microsoft.com/en-us/power-platform/products/power-bi. [Accessed: Nov 27, 2025].

[10] “SAS Visual Analytics | SAS” https://www.sas.com/en_us/software/visual-analytics.html. [Accessed: Nov 27, 2025].

[11] “Healthcare Data Visualization: Insights for Better Decision-Making” https://binariks.com/blog/data-visualization-in-healthcare/. [Accessed: Nov 27, 2025].

[12] “Medihis”. https://www.medihis.com.tr/tr-TR/anasayfa. [Accessed: Nov 27, 2025].

[13] “KeepTrackMed - Personal Health Record Tracking App”. https://www.keeptrackmed.com/. [Accessed: Nov 27, 2025].

[14] “Personal Health Tracker: Symptoms, Medications, Chronic Illness | CareClinic App”. https://careclinic.io/. [Accessed: Nov 27, 2025].

[15] “About PageSpeed Insights”. https://developers.google.com/speed/docs/insights. [Accessed: Nov 27, 2025].

[16] ”About Certbot”. https://certbot.eff.org/pages/about. [Accessed: Nov 27, 2025].

[17] “IEEE/ISO/IEC 12207-2017”. https://standards.ieee.org/ieee/12207/5672/. [Accessed: Nov 27, 2025].