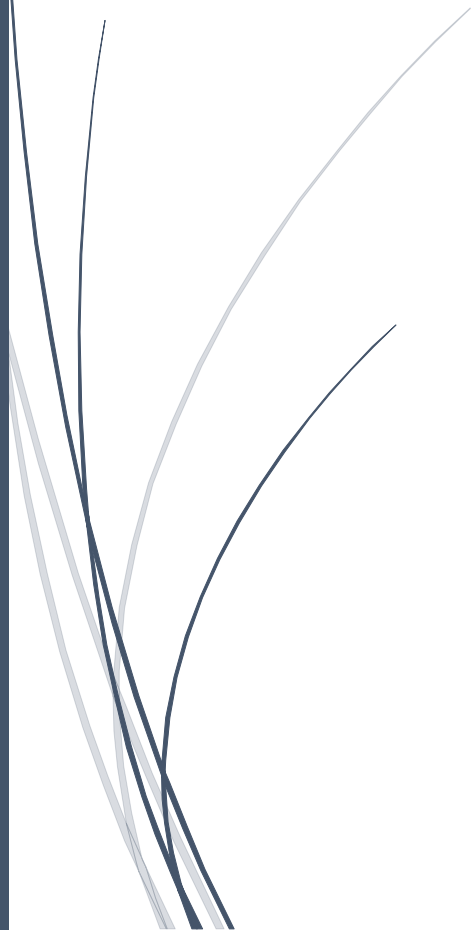
Software Engineering
ENGR_ 2792

CLAP APPLICATION

Phase 1



Authors: Aishwarya Manumari (manu0083), Sean Lee David (davi1032), Daniel Stavrakis (stav0031), Chiemelie Obi-Okonkwo (Obio0005), Kartikey ()

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this design document is to analyse and clarify the appropriate architectural design for the web-based app CLAP. The document will break down the app into a components-based structure to help encapsulate the desired requirements of the client. Abstraction will be utilised where necessary however core functionality will be further investigated to generate a true and clear representation of the clients' needs.

1.2 Scope and Boundary

The scope of the system will be focused on creating an online platform providing clinicians an easy to use interface to interact with medical history stored in a centralised database. Authorised clinicians should be granted access to perform searches, edits and updates to patient's medical history under their care. Further modifications can be made for usage of a patient trying to access and view medical history but at this moment it is not considered in the boundary of this document.

The design is more detailed towards the front-end usability of the app and back end functionality integrated with the app will be discussed in a later document. The document should give a clear high-level understanding of how the clinician interacts with the app and how the app will display the relevant data from the MyHealthRecord system.

1.3 Assumptions and Decisions

Assumptions/interpretations:

- Patients cannot view and edit their profile. This was assumed because the ability for patients to view their profile was not specified in the functional requirements.
- Editing a patient's profile is only permissible by certain clinicians. Clinicians may only have admin duties/ or have adequate training in certain fields and therefore should not be able to view/edit delicate and personnel information without authorisation.
- A patient's profile provides a view for the summary of details. Once a patient has been found on the system it is assumed all related information should be accessible on one central page with the ability to view more detail if necessary.
- The summary of details includes four components. A basic information component (including name, D.O.B, id, his/her clinician name), condition component, treatment component and prescription component. This layout is interpreted for ease of use and readability.
- The condition, treatment and prescription components have a "view full details" button. It is assumed that each component may contain a long history and details should only be displayed if required. When this button is pressed a modal appears with a full description including previous details. There are three different operations the clinician can do while the modal is loaded.

1. They can click outside the modal to return to the main profile view.
2. They can edit the existing description (click edit button).
3. Or they can create a new condition/treatment/prescription section (click add new button).

- The treatment modal has additional functionality giving it the ability to schedule a follow up treatment. In the functional requirements it outlines the ability for a clinician to add a follow up and this function makes sense to position it after a clinician has added/edited the treatment details.

1.4 References

Lists any references, e.g. lecture notes, phase 1 documents, other papers and books, and web sites consulted.

SE2 lecture notes “use cases - review” by Shaowen Qin
Software Engineering by Ian Somerville

2. System Overview

2.1 System Context and Architecture

The table below shows all the use cases the team has gathered from meetings with the client and discussions around the required functionalities. The actors and goals/intentions of each use case has been listed in tabulated format. In section 3 of this document a more detailed explanation of the key functionalities have been explained for a deeper understanding.

Functional Requirement	***Use case***	***Actors***	***Type***	***Description***
2.1.1	Clinician Login	User (initiator), Clinician	primary, and essential	User enters their username and password and clicks the login button. Software Authorises. Checks if a user exists in the database. When authorization is complete the user is logged in.

2.1.2	Clinician Verification	User (initiator), Clinician	primary, and essential	If the clinician is an authorized user (i.e. rightful staff member) their details are verified with the Clinic's database for confirmation. **Pre-condition:** User(Clinician) must verify/authenticate by logging in to CLAP database via *Clinician Login u*se case.
2.1.3	Search for a Patient with ID	User (initiator), Clinician, Patient	primary, and essential	User selects the search box and types in the patient's name. As the user types a drop down box appears with a list of 30 names. The software filters list of patients relating to the letters typed in. User selects the patient in the drop down box when they have found the correct patient. **Post-condition:** Clinician must have completed the *Clinician Login* and *Clinician Verification* use cases.
2.1.4	Advance Patient Search	User (initiator), Clinician, Patient	primary, and essential	The Clinician can filter patients results by most recent/ previous visit/applied medication (if so), type of condition, age, gender, data-of-birth etc. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.5	New Provider	User (initiator), Clinician, Patient	primary, and essential	Is the new provider/clinician appointed for the patient. **Pre-condition:** Must be eligible/authenticate himself/herself for new provider first time access jointly authenticated by the patient and healthcare providers. **Post-condition:** After authentication, the new provider is eligible; therefore can access the below functional requirements. i.e. patient details, summary of previous treatments, add treatments etc. (all the below functional requirements)

2.1.6	View summary of patients details	User (initiator), Clinician, Patient	primary and essential	Once the Clinician is logged in, he/she can view their correct patients details through provided patients id, full name, address and date of birth. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.7	View patient's full previous treatments	User (initiator), Clinician, Patient	primary, and essential	After Patient verification is performed, the clinician has access to the previous medical records of his/her patient. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.8	View patient's full treatment details	User (initiator), Clinician, Patient	primary, and essential	The Clinician can also view patients full treatment details for further clinical intervention if required. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.9	View patient's full prescriptions details	User (initiator), Clinician, Patient	primary, and essential	The Clinician can view patients full prescription details. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.10	View patient's full condition details	User (initiator), Clinician, Patient	primary, and essential	The Clinician can view patients full condition details for further clinical intervention if required. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.11	View list of patients that require follow-up details	User (initiator), Clinician, Patient	primary, and essential	The Clinician has the authority to view a list of patients that require follow-up for further clinical intervention if required. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.12	Add follow up to treatment	User (initiator), Clinician, Patient	primary, and essential	The Clinician can add follow up to treatment for patients that require follow-up for further clinical intervention. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.

2.1.13	Add (new) patient condition	User (initiator), Clinician, Patient	primary, and essential	Upon the completion of the above functional requirements if the clinician has found a new condition in the patient that requires treatment, he/she has the authority for further clinical intervention. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.14	Add prescription by editing a previous prescription	User (initiator), Clinician, Patient	primary, and essential	Clinician has the authority to add prescription by editing previous prescription **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.15	Update to patient's condition	User (initiator), Clinician, Patient	primary, and essential	The Clinician has the authority to update the patient's condition for further clinical intervention if required. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.16	View Patients Current and Past Conditions	User (initiator), Clinician, Patient	primary, and essential	The Clinician can search and previous records of patients who have visited the hospital clinic. They are able to check any current or past conditions that may affect them when another treatment is given. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.17	Add New Prescription	User (initiator), Clinician, Patient	primary, and essential	The Clinician can only assign new prescriptions to the patients by the doctors approval. The Patient will then open the app to view the prescribed medication given to them. The patient is not allowed to add their own prescription. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.

2.1.18	Add New treatment	User (initiator), Clinician, Patient	primary, and essential	The Clinician is allowed to assign a new treatment to the patient if it is required. Treatment scheduling must be done with the Admin to ensure no clashes of treatments take place. **Post-condition:** Clinician completed 2.1.1 and 2.1.2 functional requirements.
2.1.19	Generate Report	Admin (initiator), Healthcare Provider	primary, and essential	Allows system administrators to generate various medical reports on a daily basis for clinic's patients, prescribed by the clinician.

Table 1: Listings of all the use cases.

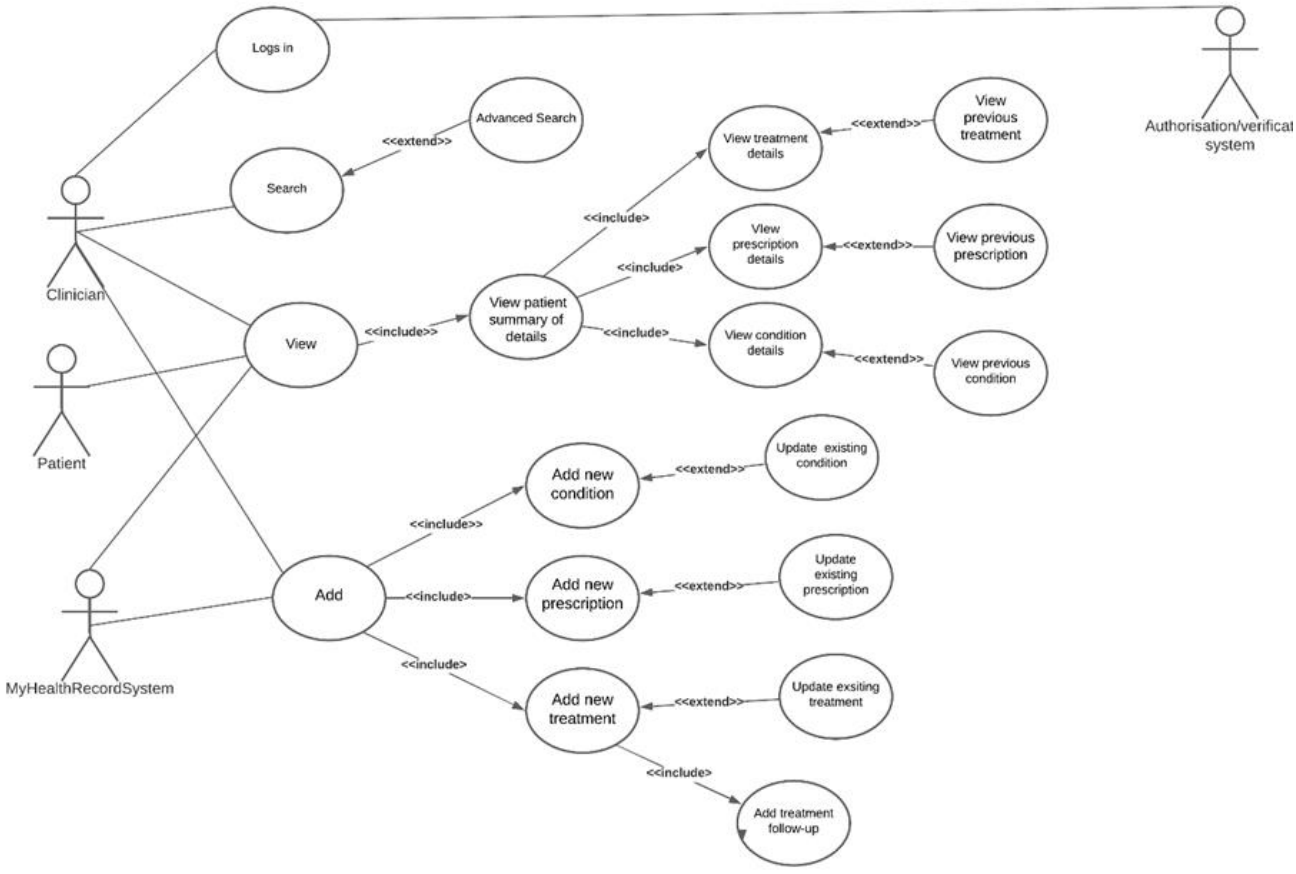


Figure 1: System Context Diagram (use-case diagram)

The use case diagram above is a high level visual which details how the actors interact with the CLAP application as well as the relationships the use cases have with one another. For more detail on the use cases the team has provided more detail by providing activity diagrams

to encapsulate how the user interacts with the application to utilize the data being retrieved from the MyHealthRecord system. This can be viewed in section 3.

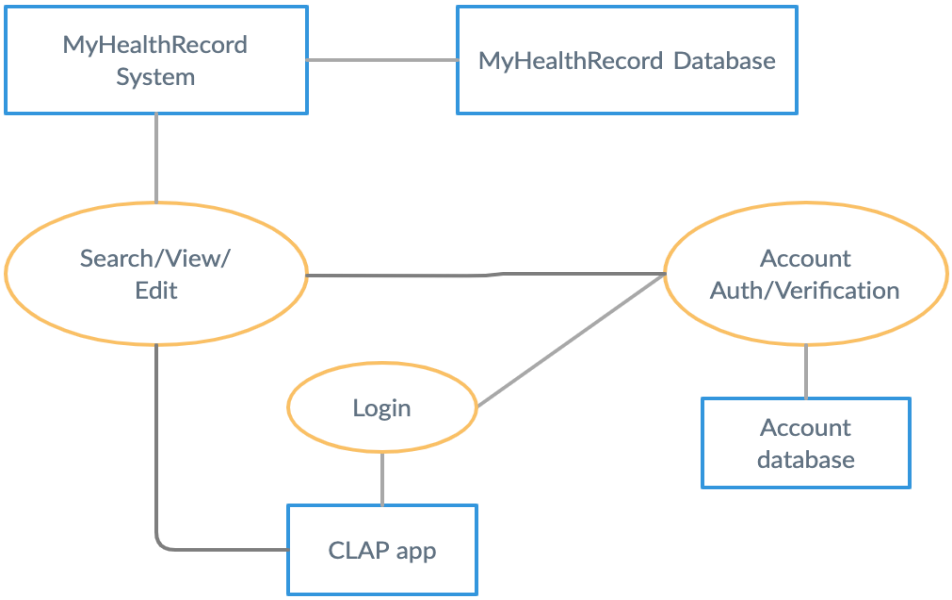


Figure 2: Preliminary System Architecture

The preliminary system architecture provides a high-level design of the architecture of the different systems/software which interact with application. From this diagram the team can discuss the most practical way that the application can develop the appropriate connections to these interfaces.

2.2 System Activities

The activity diagram below represents a high-level view of the CLAP system's functionalities on how each use-case activity conceptually integrate with the CLAP application.

The activity starts by an authorised clinician logging into the CLAP application who is verified by the Healthcare actor. The workflow between each transition is labelled to clearly identify the event. For example, the outgoing transition from the clinician who decides to add a follow-up treatment for the patient is represented in guards to show the condition of the event. The activity diagram also incorporates synchronization of a *fork* and a *join* between use-cases and their activities represented through bars. Lastly, the use cases are categorized with swim-lanes to group each action they are responsible for their particular actor, namely Admin, CLAP Healthcare System, Clinician and Patient.

To capture the procedural possibilities associated with each individual use case, our team has put together ten extended activity diagrams which can be viewed in section 3.

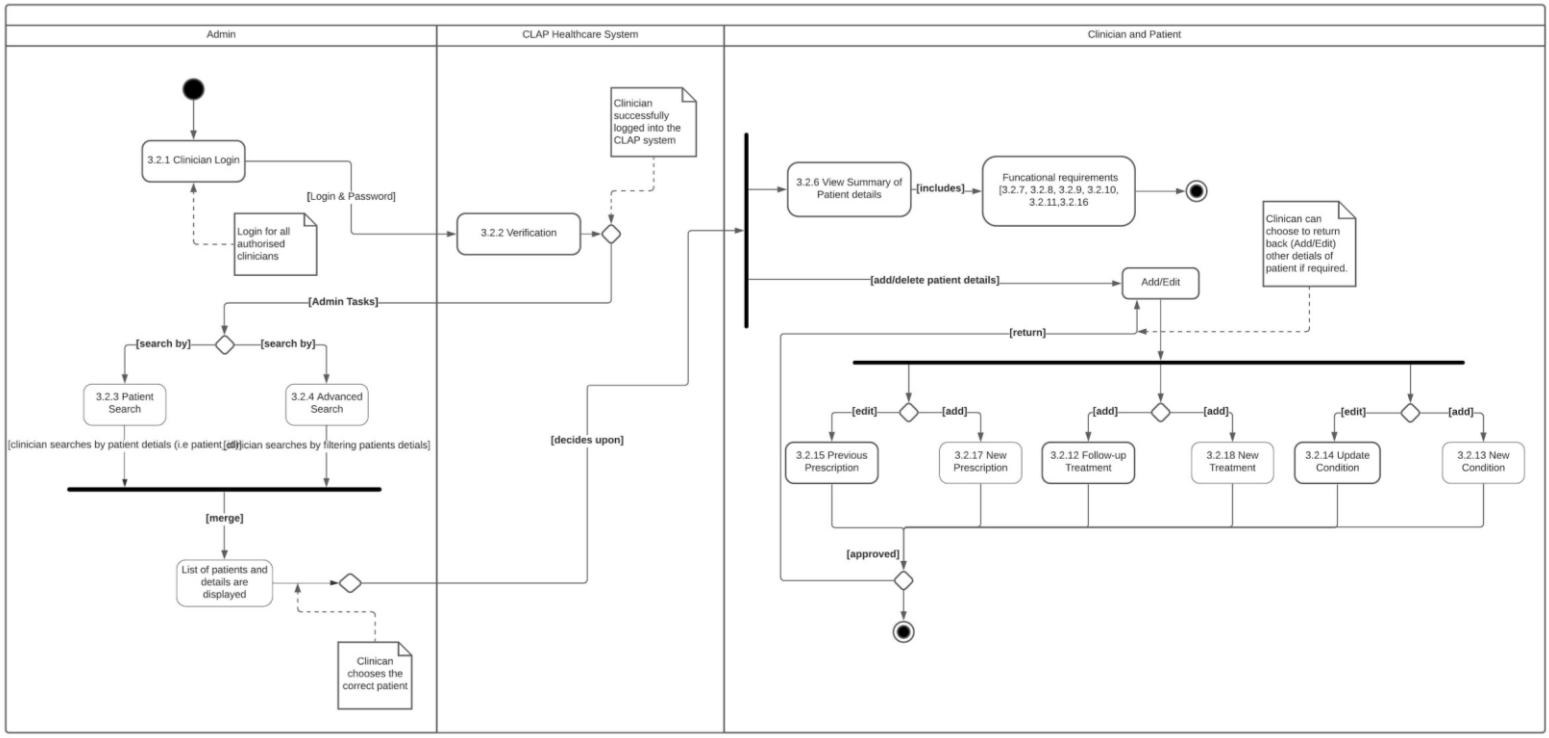


Figure 3: High-Level System Activity Diagram

2.3 Conceptual Model

The main classes that are being included in the final diagram are: clinician, patient, account, condition, treatment and prescription. These classes best represent the system and the use cases derived from the functional requirements.

There were opportunities to represent generalisation and specialisation with the clinician and treatment class however we felt at this initial stage it was not needed and was not specifically outlined in the functional requirements. A separate class for advance search was also excluded because attributes being recorded didn't provide any value to the user at this stage.

There are a couple of occurrences of aggregation and composition which the team could possibly represent at a later stage in the system design. The patient profile is made up of details derived from the condition, treatment and prescription tables. The team decided that the relationship between condition and patient is an aggregation relationship because the condition can exist without the patient and likewise with the treatment being able to exist without the patient profile. However, the nature of a prescription will be best represented as a composition relationship. This is because the prescription (a document which authorises a patient medicine or treatment) does not exist without a patient.

The team saw many evolutions of the conceptual diagrams as seen in below in figures x and y. The first iteration had many classes which then evolved into attributes of classes. Examples of

such classes that are now attributes include follow ups, summary details, previous treatment and previous conditions. These are now represented in the patient class.

Assumptions of Multiplicities

1. A Patient can 0..* treatments and prescription but the patient must have at least 1(..*) condition to be treated and prescribed. These can only belong to 1..1 Patient. (actor is read singular)
2. Admin has 1..* Clinicians while the clinicians can also have 1..* report_ids from Admin.
3. A clinician must only have one authorised login account (1..1) and the account must belong to 1..1 clinician.
4. Clinician does 0..* searches for a patient as he can add any patient details without searching too. However for a search event to happen, it must require 1..* clinicians to action it.
5. In order for the clinician to delete, at least 1(..*) prescription, condition, treatment is required and these belong to only 1..1 Clinician. (actor is read singular)
6. In order for the clinician to add 0..* prescription, condition, treatment can exist and these belong to only 1..1 Clinician. (actor is read singular)

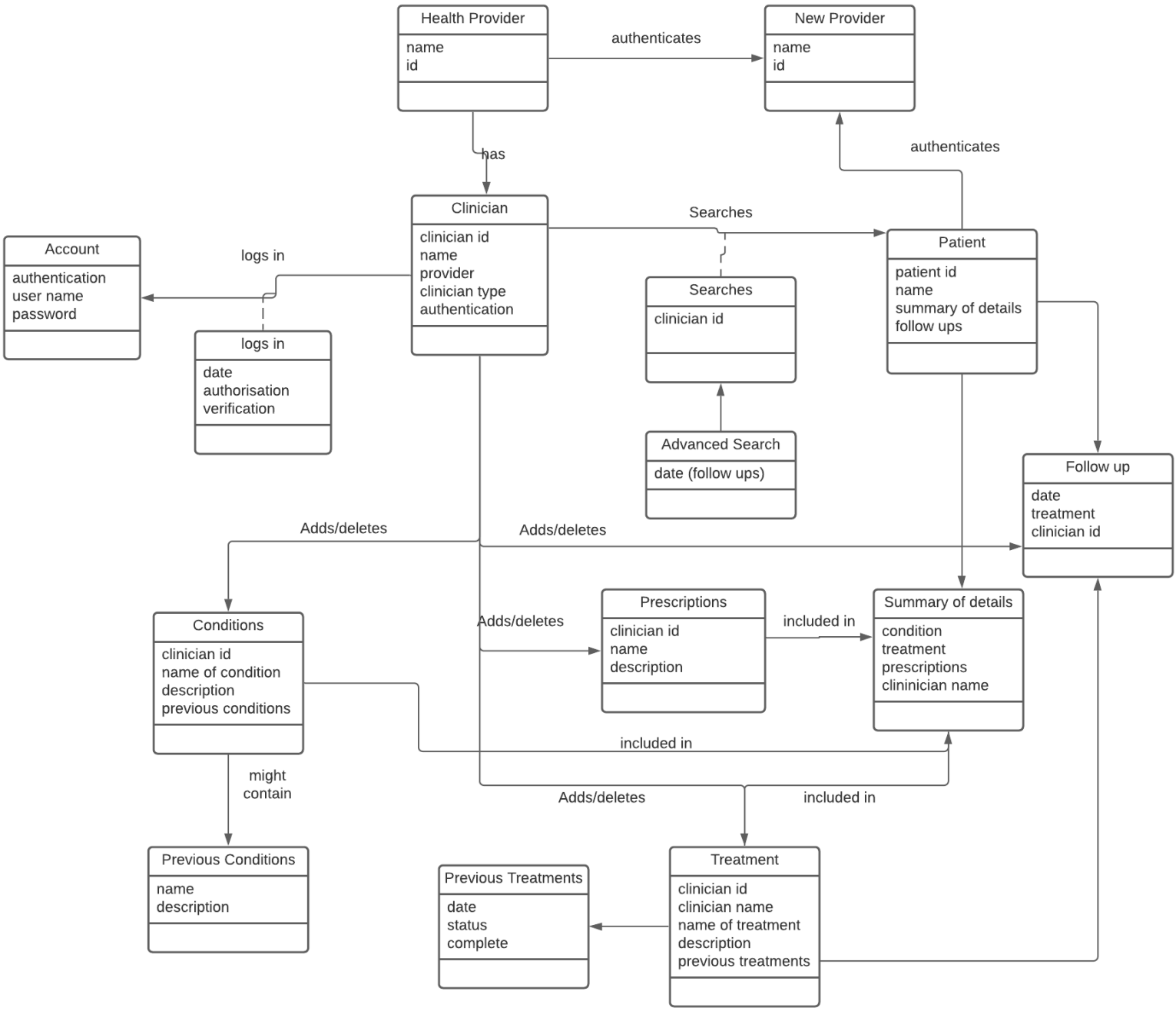


Figure 4: System Conceptual Model Part 1

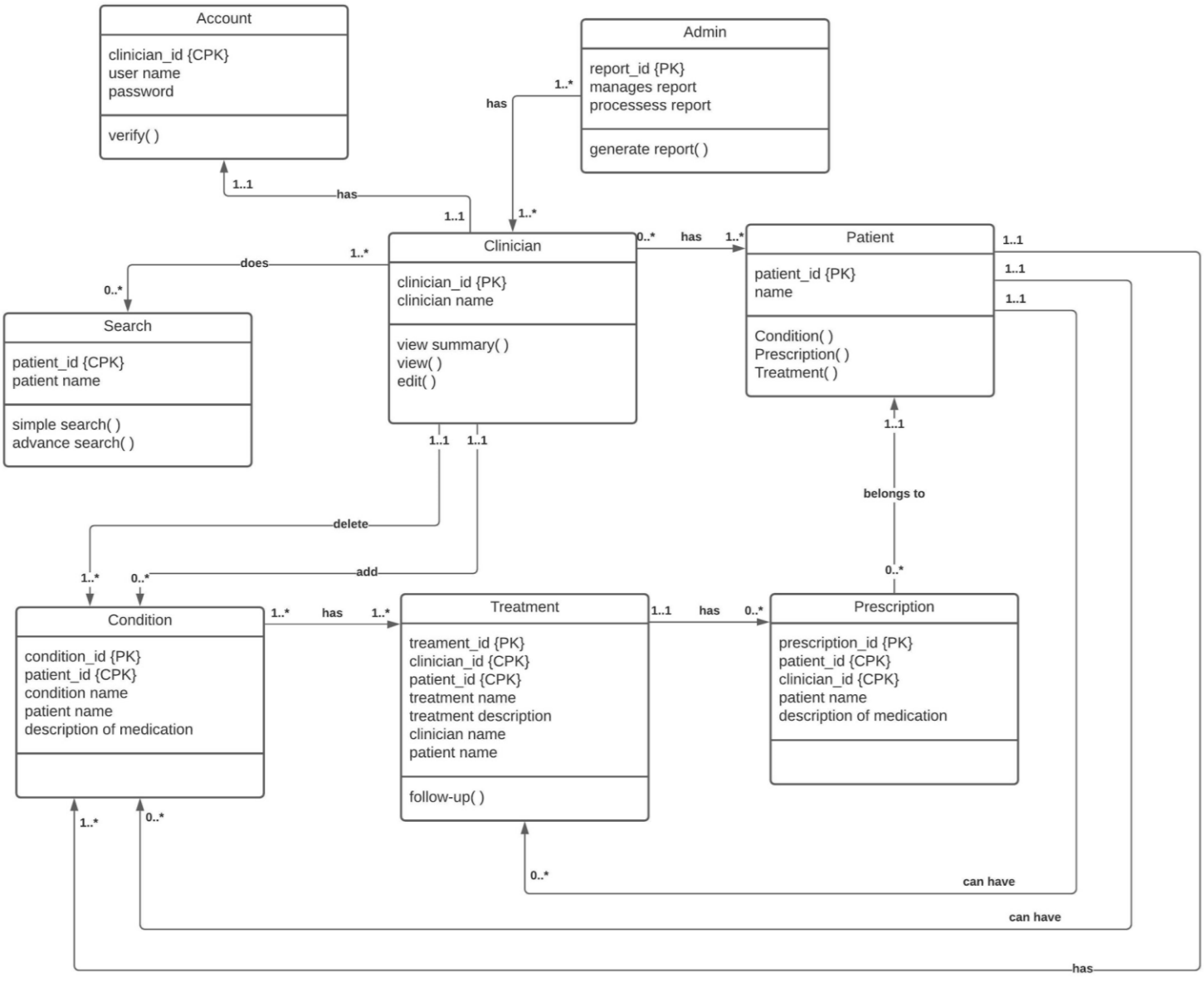


Figure 5: System Conceptual Model Part 2 (Improved)

3 User stories/use-cases

The team has chosen several use cases mentioned from table 2.1 and extended them to gain a better insight to the functionality required. Details have been extended to provide a better understanding of the processes and variations. Conceptual workflow models such as activity diagrams cross reference the high-level system overview provided in section 2.2. From these use cases acceptance tests have been derived so that the testing team can start to test functionality from the very early stages of development.

(Daniel, stav0031)

Advanced Search to filter patients that need follow up

Use case:	Advanced Search to filter patients that need follow up
Actors:	Clinician
Trigger:	clinician choose the filter by follow ups filter and clicks search
Purpose:	the intention is to search all patients that a follow up in the near future
Summary:	A clinician may wish to check-in with a patient that has a scheduled follow-up. This can be helpful to the patient offering them a friendly reminder or an opportunity for them to cancel and reschedule if they must. The ability to filter patients on dates of follow ups makes this procedure more effective and efficient.
Type:	a. primary b. essential
Cross reference:	Search patient use case, SRS functional requirement 3.2.3
Pre-conditions:	Clinicians must first log in and verify/authenticate their account to gain access to the search and advanced search options.
Post-conditions :	if a match exists the flow will continue and provide a list of patients.
Process:	1. The user will then click search to load the search page. 2. To make advanced search options visible the user will click the advanced search icon. 3. Filters will appear just below the nav area of the page. To filter all patients with follow ups they will click the checkbox which will trigger a date picker to be in view. 4. Once a valid date is chosen then the user will click search.
Variations:	If the user selects a date from the past when searching for future follow ups in step 3 an error message will display prompting the user to select a future date. If a clinician attempts to search for a patient's personal information without authorisation their search page will reset.

Special requirements:	Database should be able to sustain up to 1,500 users accessing the database at one time in order to read the requested information.

Test ID	Scenario/Use Case/Variation	Expected Result	Actual Result	Pass/Fail
3.2.4	The user selects a date range between today and the following 7 days and clicks search.	A list of patients containing follow ups within this date range will be listed.		

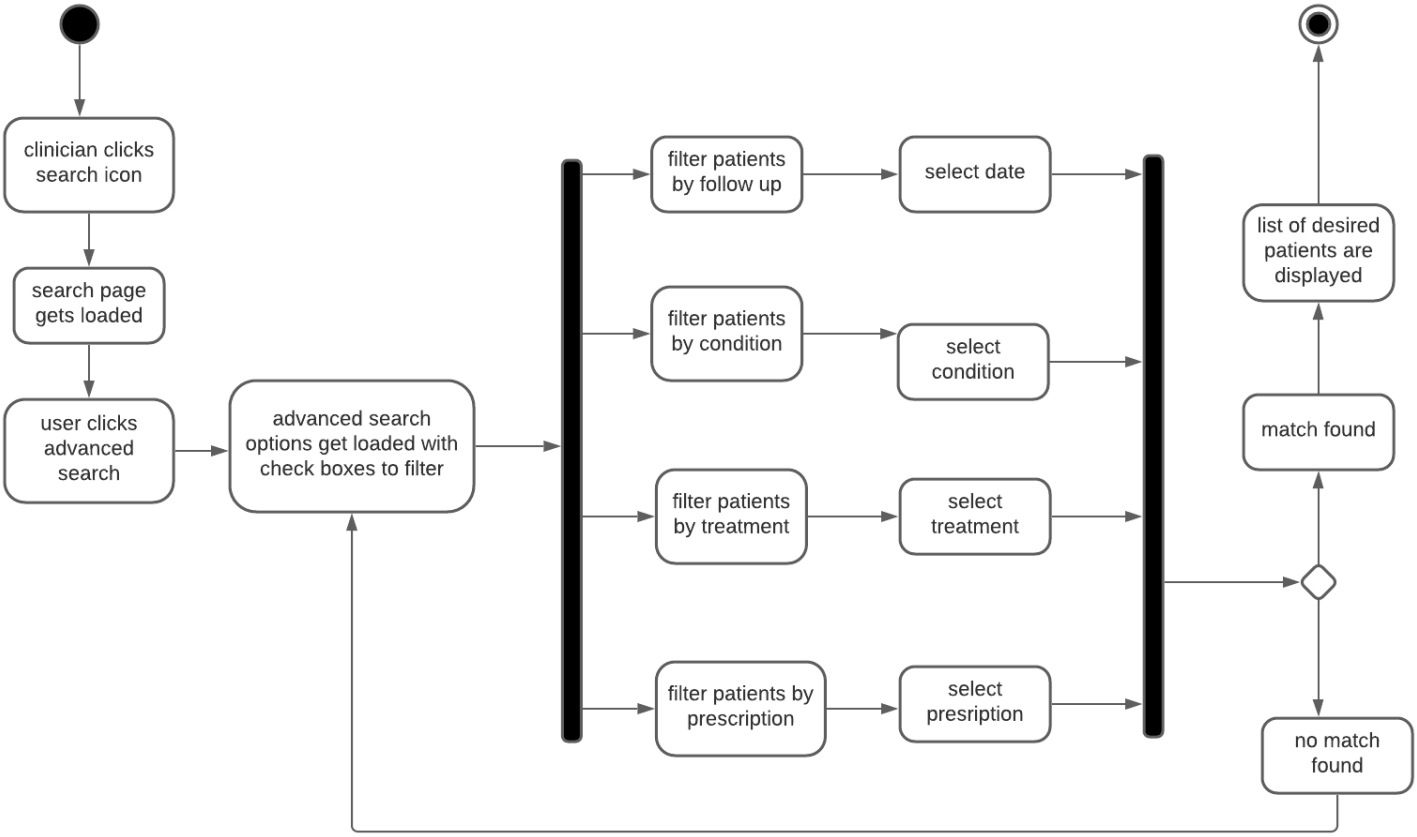


Figure 6: Advance search activity diagram.

Description: The diagram shown in figure 6 above provides a deeper level understanding on the event of an advanced search (3.2.4) which can be cross referenced with the high-level activity diagram above in section 2.2. When the advanced search options are loaded in the form of checkboxes a user will select their desired filter category and then be prompted with a selection from the category. As explained in this diagram if there is no match found they will be redirected to the advanced search before any filters or selections were applied.

(Daniel stav0031)

Add follow up treatment

Use case:	add follow up treatment
Actors:	Clinician
Trigger:	patient needing a follow up treatment
Purpose:	the intention is to schedule a follow up treatment for a review.
Summary:	The Clinician can add follow up to treatment for patients that require follow-up for further clinical intervention. This ensures that the treatment process is being monitored and that the patient will be given a suitable date/time without experiencing delays due to demand.

Type:	a. primary, b. essential
Cross reference:	New treatment use case, SRS functional requirement 3.2.18
Pre-condition:	To add a follow up treatment the clinician must be logged in and authorised. For a follow up treatment to be created the clinician must be on the relevant patient's profile and have navigated to the patient's full description of treatment page.
Post-conditions:	The follow up treatment cannot be set to a past date.
Process:	1. The clinician logs in. 2. Uses search functionality to navigate to the patents profile 3. Clicks on the view full description button located in the treatment component. 4. The treatment page loads. 5. The user then clicks the "add follow up" button. 6. A date picker will come to view with available times for a follow up. 7. The user then saves the information and gets navigated to the patient's profile.
Variations:	If the follow up information fails to save in step 7 an error message will appear prompting the user to try again.

Test ID	**Scenario/Use Case/Variation**	**Expected Result**	**Actual Result**	**Pass/Fail**
3.2.12	Assign a date to a patient for a follow up. Then click save.	Re-enter the treatment page to view that the follow up information has been saved and matches the date that was entered prior.		

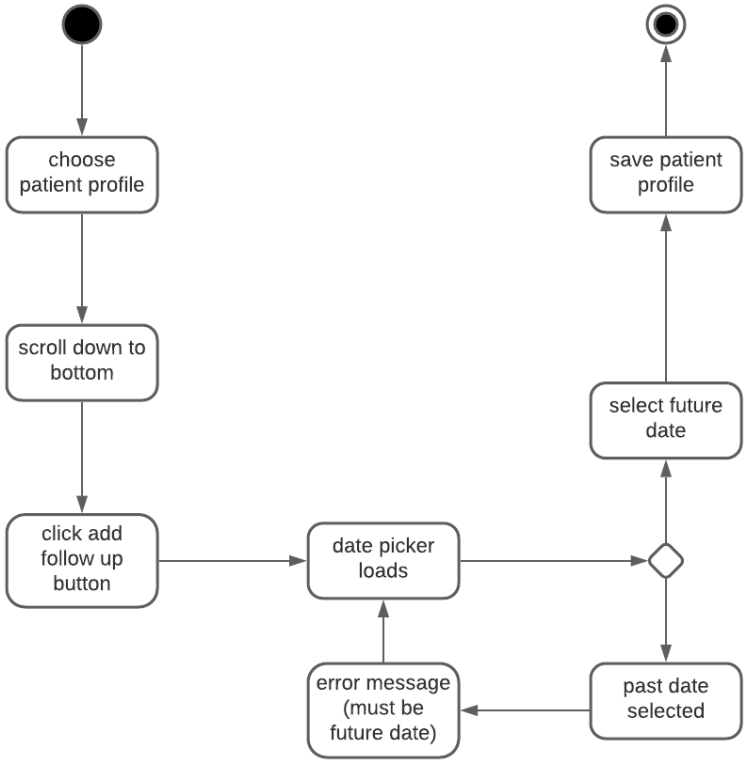


Figure 7: Follow up patient activity diagram

Description: The diagram shown in figure 7 above provides a deeper level understanding in the event of adding a follow up for a patient(...) which can be cross referenced with the high-level activity diagram above in section 2.2. The clinician will find an add follow up button at the bottom of the treatment page. If the date selected is invalid they will be sent back to a fresh date picker.

Sean (davi1032)

View summary of patient's details

Use case:	View summary of patient's details
Actors:	User (Clinician), Patient
Trigger:	Clinician searches for patient using ID or name.
Purpose:	The clinician will be able to view the summary of patient's details.
Summary:	**:** once the clinician is logged in, he/she can view their correct patients details through provided patients id, full name, address and date of birth.
Type:	primary, essential
Cross reference:	Login to the system use case, SRS functional requirement 2.1.1Clinician Verification use case, SRS functional requirement 2.1.2.
Pre-conditions:	Once the Clinician has completed the *Login* use case*,* he/she must verify their uthorisation with the CLAP database via *Clinician Verification* use case.
Post-conditions:	Clinician must have authenticated himself/herself by logging into the CLAP database via *Clinician Login* use case.
Process:	1. Clinician will search for patient using the search bar, with their name or patient ID. 2. A new page gets loaded and displays relevant results based on the search item. 3. The patient's ID and name is displayed for the clinician which patient they would like to view. 4. Clinician chooses the patient they want to view. 5. Results of the patient's details are loaded for the clinician to view.
Variations:	1a: Invalid search 1. System detects an error. 2. Clinician re-enters new patient's name or patient's ID 3. If the details are valid, standard processing resumes. i. incorrect details entered again. ii. Clinician repeats the same process again.
Special requirements:	Good Internet Connection. Device that can load the APP
Acceptance Test:	User searches for patient and they exist on the database. The CLAP APP will search through the database and find an existing patient with given search details. The CLAP APP finds the patient and displays all relevant information to the clinician.

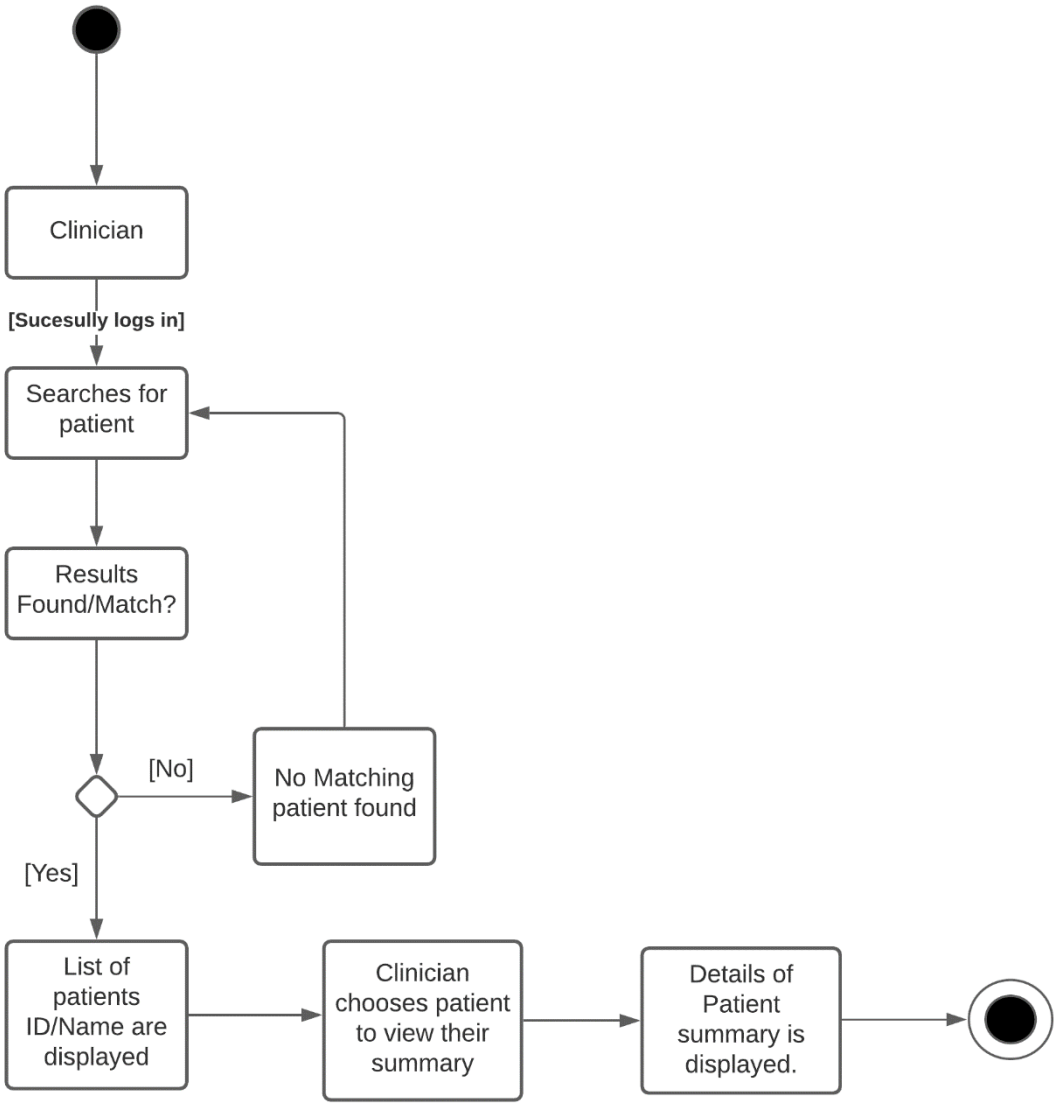


Figure 8: Summary of details Activity Diagram

Sean (davi1032)

Add prescription by editing a previous prescription

Use case:	Add prescription by editing a previous prescription
Actors:	User (Clinician), Patient
Trigger:	When a clinician wants to add a prescription or edit a previous existing prescription.
Purpose:	Clinician will be able to add new prescription
Summary:	The clinician will search for an existing patient using ID or their name. If the patient is found, the clinician can add a new prescription to them or edit an existing prescription. If the client does not find a match, they will have to add the patient to the system for them to assign them a new prescription.
Type:	Primary, essential
Cross reference:	Login to the system use case, SRS functional requirement 2.1.1 Clinician Verification use case, SRS functional requirement 2.1.2.
Pre-conditions:	Once the User(clinician) has completed the *Login* use case, he/she must verify their authorization with CLAP database via *Clinician Verification* use case.
Post-conditions:	User(clinician) must have authenticated himself/herself by logging into the CLAP database via *Clinician Login* use case.
Process:	1. Clinician will search for patient using the search bar, with their name or patient ID. 2. A new page gets loaded and displays relevant results based on the search item. 3. The patient's ID and name is displayed for the clinician which patient they would like to view. 4. Clinician chooses the patient they want to view. 5. Results of the patient's details are loaded for the clinician to view.
Variations:	2a: No results 1.System detects an error. 2.Clinician re-enters new patient's name or patient's ID 3.If the details are valid, standard processing resumes. 1. incorrect details entered again. 2. Clinician repeats the same process again. 3a: No match

	1.If no patient is detected, the clinician will have to add them to the database. 2.Clincian will add patient ID along with noting if the patient is allergic to any type of prescription. 3.Clinican will then add new prescription for the patient. 3b: Existing Patient 1.The clinician does have an existing patient. 2.The clinician can view the existing prescription given to the patient. 3.The clinician can change the current prescription or add a new prescription for the patient.
Special requirements:	Good Internet Connection. Device that can load the APP
Acceptance Test:	

Test ID	**Scenario/Use Case/Variation**	**Expected Result**	**Actual Result**	**Pass/Fail**
2.1.1	Clinician adds new patient into system.	Clinician successfully adds patient	Successful	**Pass**
2.1.1	Clinician adds new patient into system.	Clinician successfully adds patient	Incorrect details was added to the patients details	**FAIL**

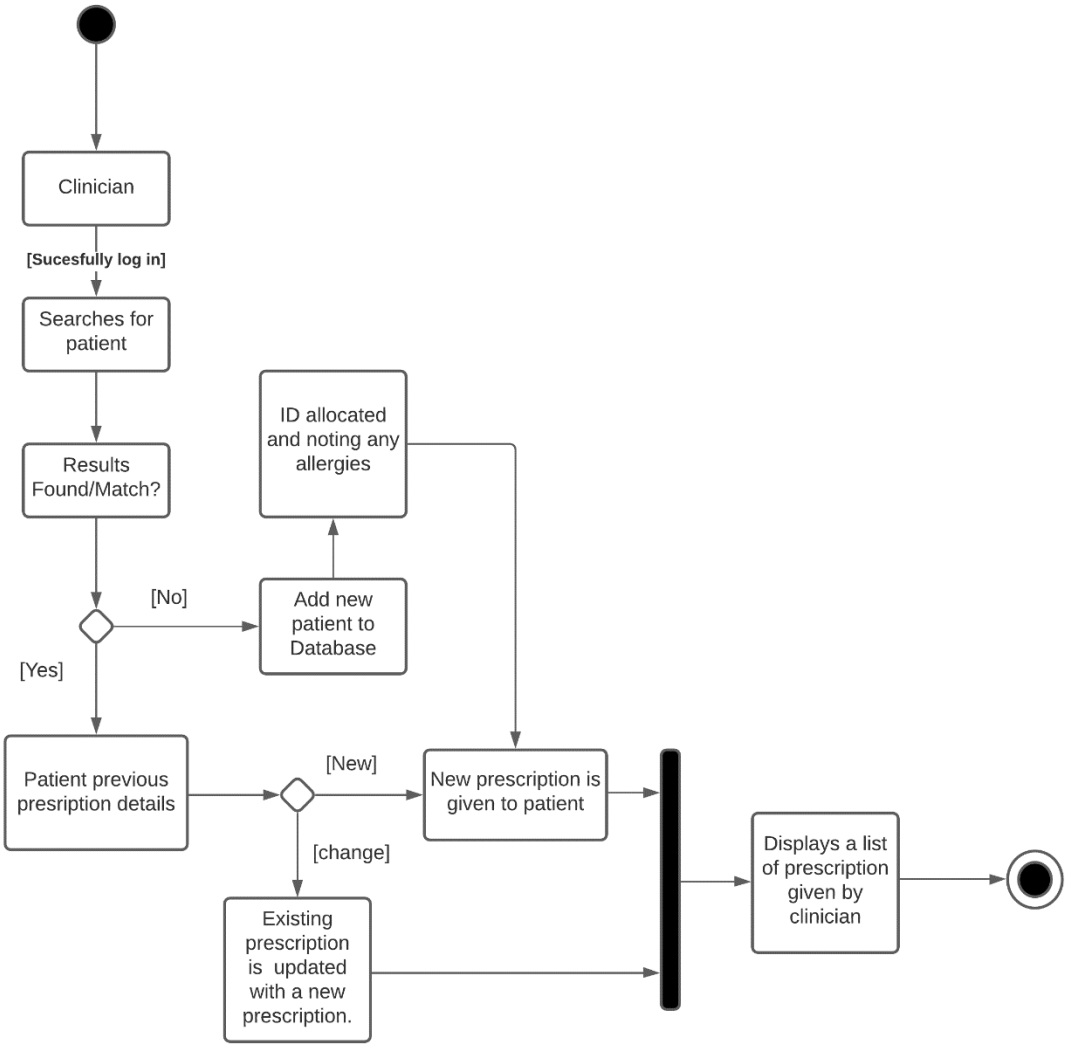


Figure 9: Adding new prescription Activity Diagram

(Aish, manu0083)

Clinician Login

Use case:	Clinician Login
Actors:	Clinician (initiator), User
Trigger:	Clinician chooses to login to the CLAP system by entering his/her username and password.
Purpose:	To perform admin tasks such as search for list of patients or view/edit patient details (i.e condition, treatment, prescription)

Summary:	A clinician may wish to check-in with a patient that has a scheduled follow-up. This can be helpful to the patient offering them a friendly reminder or an opportunity for them to cancel and reschedule if they must. The ability to filter patients on dates of follow ups makes this procedure more effective and efficient.
Type:	*a. primary* *b. essential*
Cross reference:	Clinician Verification use case, SRS functional requirement 2.1.1 and 2.1.2.
Pre-conditions:	Once the User(clinician) has completed the *Login* use case*,* he/she must verify their authorization with CLAP database via *Clinician Verification* use case.
Post-conditions :	User(clinician) must have authenticated himself/herself by logging into the CLAP database via *Clinician Login* use case.
Process:	1. This use case begins when a Clinician wants to access the CLAP system to perform a variety of actions such as admin settings, view list of patients and their details, adding patient conditions, treatments or follow-ups, or prescribing medication. 2. The Clinician begins by entering his username and password to login to the CLAP system. 3. The System determines if the user is a valid account holder from the Verification process. 4. Verification is completed as software authorises only if the user is valid. 5. If invalid the user must re-authorise himself. 6. If valid the clinician is successfully logged in the CLAP system.
Variations:	2a. Invalid username entered. 1. System indicates an identifier error. 2. The Clinician re-enters his username 3. If the username is valid, standard processing resumes. 1a. Incorrect username entered multiple times. 1. System shows a warning message of incorrect credentials. 2b. Invalid password entered 1. System indicates password error. 2. The Clinician re-enters his password 3. If the password is valid, standard processing resumes. 1a. Incorrect password entered for the third time. 1. System locks the account for a certain time period, and an alert message is sent to the account holder.
Special requirements:	Internet Connection Device that supports the CLAP Application

Test ID	**Scenario/Use Case/Variation**	**Expected Result**	**Actual Result**	**Pass/Fail**
2.1.1	Authorised Clinician	Clinician is		

	logs in the CLAP System with their login name and password	successfully logged in		
2.1.1	Incorrect login details provided by the Clinician	Login Error		

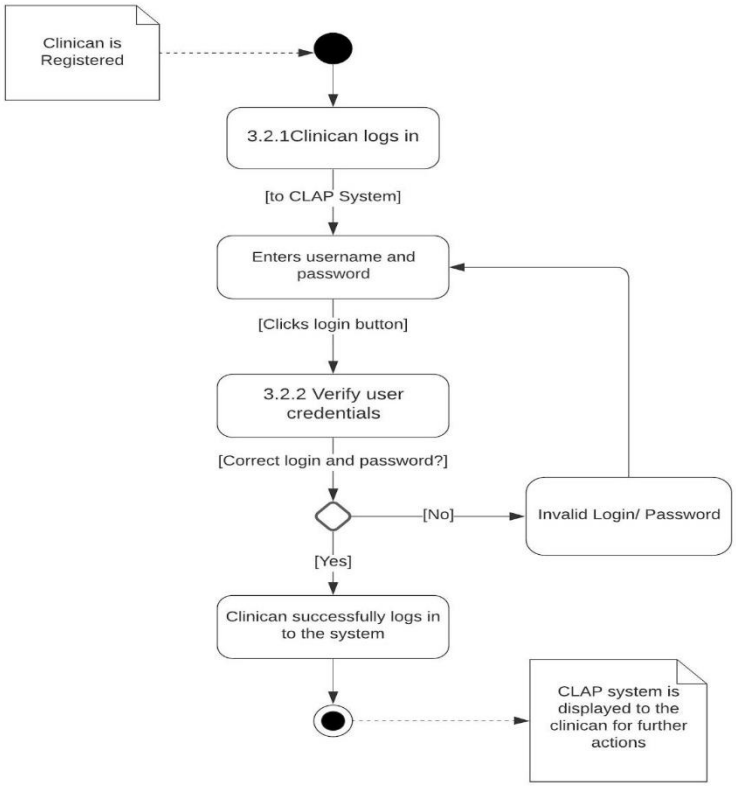


Figure 10: Clinician Login Activity Diagram

Description: The diagram shown in figure 10 above portrays the extended activity diagram for Clinician Login (3.2.1) use case which can be cross referenced with the high-level activity diagram above in section 2.2. The clinician is required to authenticate himself by logging in the CLAP system with his username and password. The possible variations for this diagram are listed in the extended use case above. Hence if invalid credentials the clinician fails to login to the CLAP system. (manu0083)

(Aish, manu0083)

Add New treatment

Use case:	Add New treatment
Actors:	Clinician (initiator), User
Trigger:	Clinician chooses to add a new treatment for the patient
Purpose:	To assign a new treatment for a patient as a medical procedure to treat the patient potentially diagnosed with a condition or illness.
Summary:	The Clinician is allowed to assign a new treatment to the patient if it is required. Treatment scheduling must be done with the Admin to ensure no clashes of treatments take place.
Type:	*a. primary* *b. essential*
Cross reference:	Add Follow-up Treatment use case, SRS functional requirement 2.1.18 and 2.1.12.
Pre-conditions:	Once the Clinician has completed the *Login* use case, he/she must verify their authorization with the CLAP database via *Clinician Verification* use case.
Post-conditions :	Clinician must have authenticated himself/herself by logging into the CLAP database via *Clinician Login* use case.
Process:	1. The use case begins with the Clinician entering the patient name or id in the search field and clicks the button to search. 2. Note that the Clinician must be logged into the CLAP system in order to do so. 3. As a result the list of patient details are displayed to the Clinician on his browser. 4. If for some reason, the Clinician cannot find the correct patient he filters his search results through Advance Search option available on the CLAP system. 5. As a result the correct patient and his details are displayed to the Clinician on his browser. 6. The Clinician then chooses to add a new treatment for the patient 7. He begins by selecting new treatment required for the patient, as a result new treatment is added. 8. The Clinician can also choose to add a new treatment for the patient by updating his previous treatment record. As a result a new treatment record has been updated on the patient further on. 9. The system records all the details entered by the Clinician successfully.
Variations:	2a. Invalid Patient name/id 1. System indicates an identifier error.

	2. The Clinician re-enters patient id or name in the search field 3. If the name is valid, standard processing resumes. 1a. Incorrect name/id entered multiple times. 1. Clinician can advance search by filtering patient details on the system. 2. Process continues until the correct patient match is found. 2b. Successfully Adds New Treatment 1. The Clinician add/updates new treatment for the patient successfully 2. System records all the changes made by the Clinician 3. Is directed back to the Add/Edit page, where the Clinician can add/edit other patients treatments. 2c. Deletes/Withdraws from Adding New Treatment 1. The Clinician chooses to withdraw from adding/editing new treatment for the patient 2. System redirects the clinician back step before; Add/Edit 3. System records all the changes made by the Clinician
Special requirements:	Internet Connection Device that supports the CLAP Application

Test ID	**Scenario/Use Case/Variation**	**Expected Result**	**Actual Result**	**Pass/Fail**
2.1.3	Successful case: Search for Patient	Patient Match found		
2.1.3	Invalid case: Search for Patient	Patient Match not found. Process continues, and the clinician is directed back to the search field.		
2.1.4	Invaild identifier: Advance Search as Clinician fails to find the correct patient multiple times	Patient Match found		
2.1.18	Clinician chooses to add/update new treatment for the patient	Clinician is successfully adds/updates a new treatment		
2.1.18	Clinician decides to withdraw/delete from adding or updating a new treatment	Clinician is directed back to the step before, home page/dashboard.		

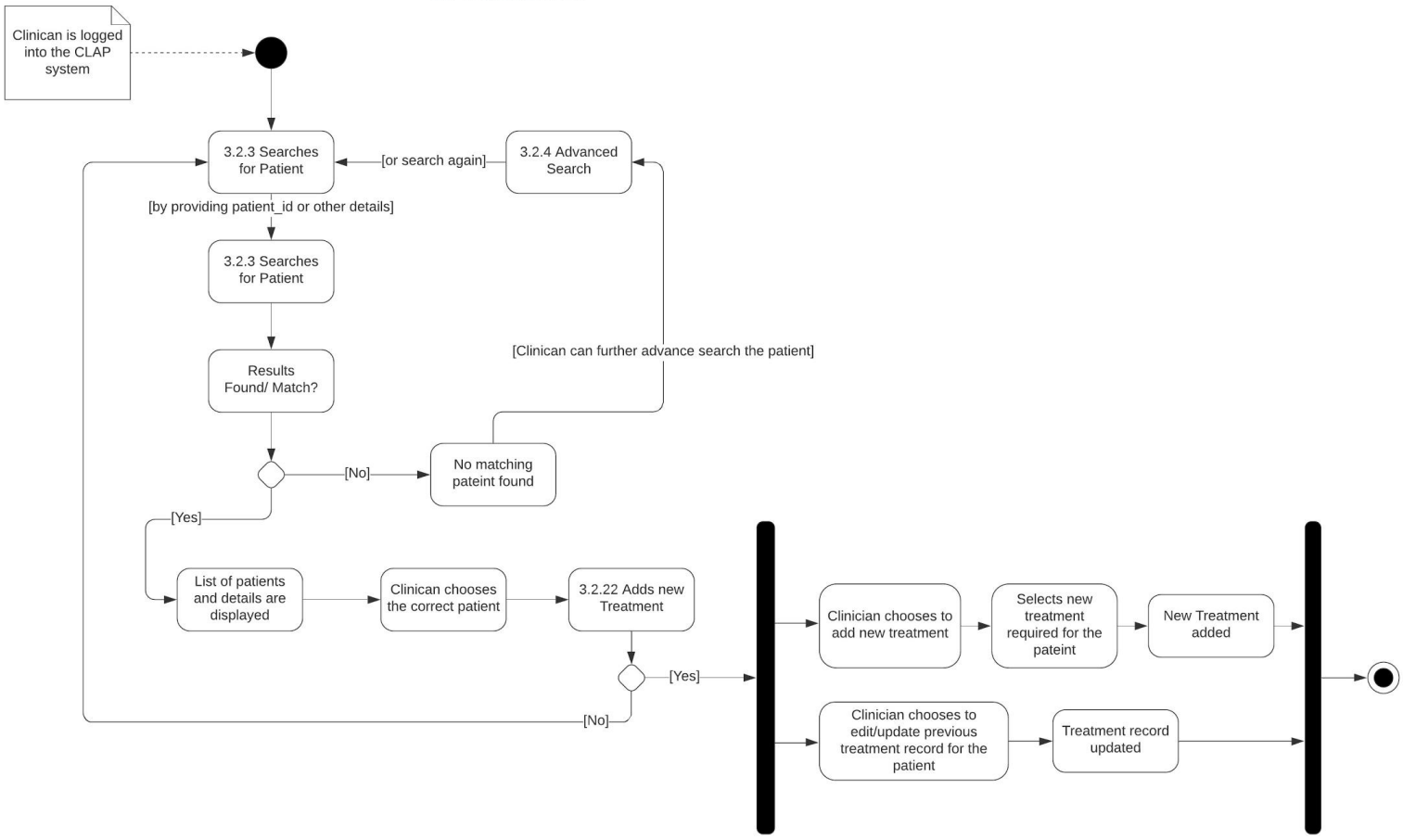


Figure 11: Add New Treatment Activity Diagram

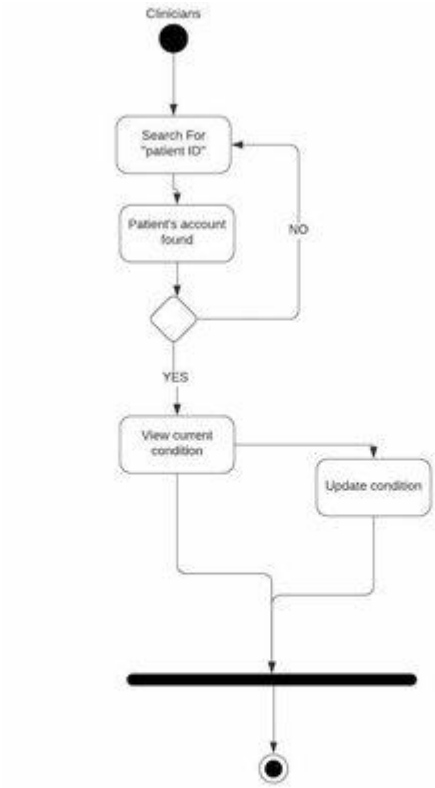
Description: The diagram shown in figure 11 above represents the event of a clinician adding a new treatment for a patient either by adding or editing his/her previous treatment, which can be cross referenced with the high-level activity diagram above in section 2.2. The clinician will click on the add or edit new treatment button provided on the treatment page. The possible variations for this diagram are listed in the extended use case above. Hence if the clinician enters incorrect treatment details for the patient, he can delete the treatment record. Similarly, if he obts to withdraw from entering new treatment details, the clinician is directed back to the home page/dashboard. (manu0083)

Obi (Obio0005)

Update condition

Use case:	Update condition
Actors:	Clinician
Trigger:	When previous conditions need to be updated.
Purpose:	A new condition will be added to the patient's profile.
Summary:	The Clinician is allowed to update new treatment to the patient's record if it is required. Treatment scheduling must be done with the Admin to ensure no clashes of treatments take place.
Type:	f. *primary*
Cross reference:	
Pre-conditions:	Clinicians should log in into the system by verifying his credentials and patients should have an existing condition to update.
Post-conditions :	Clinician must have authenticated himself/herself by logging into the CLAP database via *Clinician Login* use case.
Process:	a) Clinicians log in into the system by adding his details. b) Search for patient by adding patient ID c) clicks on patient profile d) Views existing conditions under patient's name e) Edit the existing conditions by adding new conditions f) Process ends
Variations:	● If the login attempt is failed an error message pops up notifying the log in used details should be viewed. ● If the patient doesn't have id, a new account should be created in the patient name. ● If there's no existing condition, Clinicians should create a new condition and update in next follow ups.
Special requirements:	Internet Connection Device that supports the CLAP Application

Use case:	View previous treatment
Actors:	Staff and myhealthrecordsystem
Trigger:	When previous treatment needs to be updated.
Purpose:	To view the patient's previous treatment
Summary:	The clinician is allowed to view previous treatment to the patient if it is required.
Type:	*a. primary*
Cross reference:	View, search for patient by ID, View summary details.
Pre-conditions:	Once the Clinician has completed the *Login* use case, he/she must verify their authorization with the CLAP database via *Clinician Verification* use case.
Post-conditions :	Clinicians must have authenticated himself/herself by logging into the CLAP database via *Clinician Login* use case.
Process:	1. Clinician logged into the system by adding his/her login details 2. use the search function to navigate to the treatment profile 3. The user also goes to view the summer details of all the treatments he or she has had 4. The user then sees the following: view previous treatments, view previous him prescriptions, view previous conditions 5. click on view previous treatment
Variations:	• If the login attempt is failed an error message pops up notifying the log in used details should be viewed. • If the patient doesn't have id, a new account should be created in the patient name. • If there's no existing condition, Clinicians should create a new condition and update in next follow ups.
Special requirements:	Internet Connection Device that supports the CLAP Application



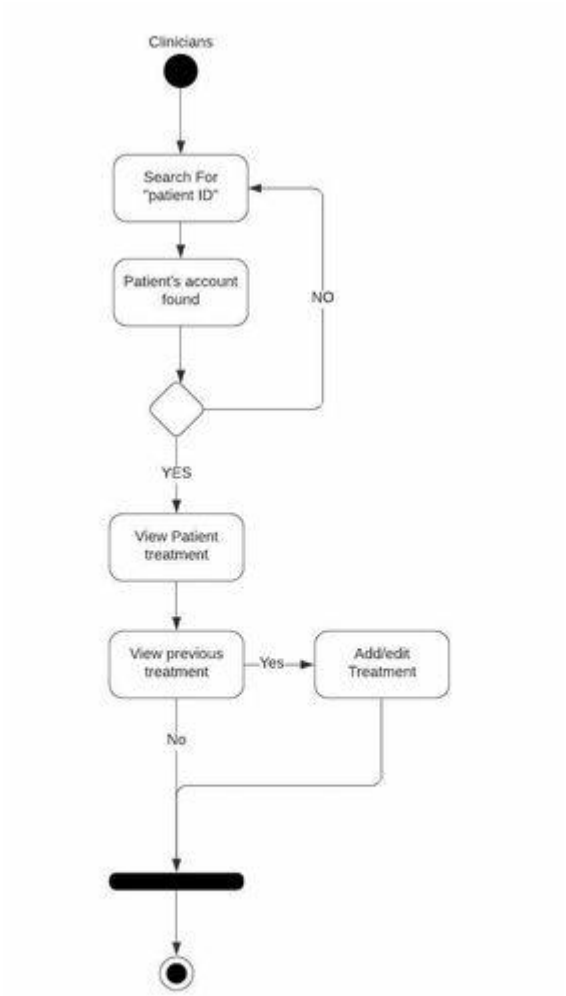


Figure 12: Update Condition Activity Diagram
Figure 13: Add/edit Treatment

Kartikey

View Past Conditions

Use case: View Past Conditions
Actors: User (Clinician), Patient
Type: Primary Essential
Description: The clinician can search and previous records of patients who have visited the hospital clinic. They are able to check any current or past conditions that may affect them when another treatment is given
Process: The clinician will log into the system and open the records of all patients who've been to the hospital. The clinician will search the patient name through the records to view the full record of patient past and current treatment details.
Cross reference: related use cases, SRS functional requirements, other documents
Preconditions:

- The User must verify themselves as a staff by login into the CLAP App via special username and password.
- Once authenticated, the User is allowed to view any existing records of patients who have been admitted.

Use case:	View previous treatment
Actors:	Staff and myhealthrecordsystem
Trigger:	When previous treatment needs to be updated.
Purpose:	To view the patient's previous treatment
Summary:	
Type:	*a. primary*
Cross reference:	View, search for patient by ID, View summary details.
Pre-conditions:	
Post-conditions :	Clinician must have authenticated himself/herself by logging into the CLAP database via *Clinician Login* use case.
Process:	
Variations:	
Special requirements:	Internet Connection Device that supports the CLAP Application

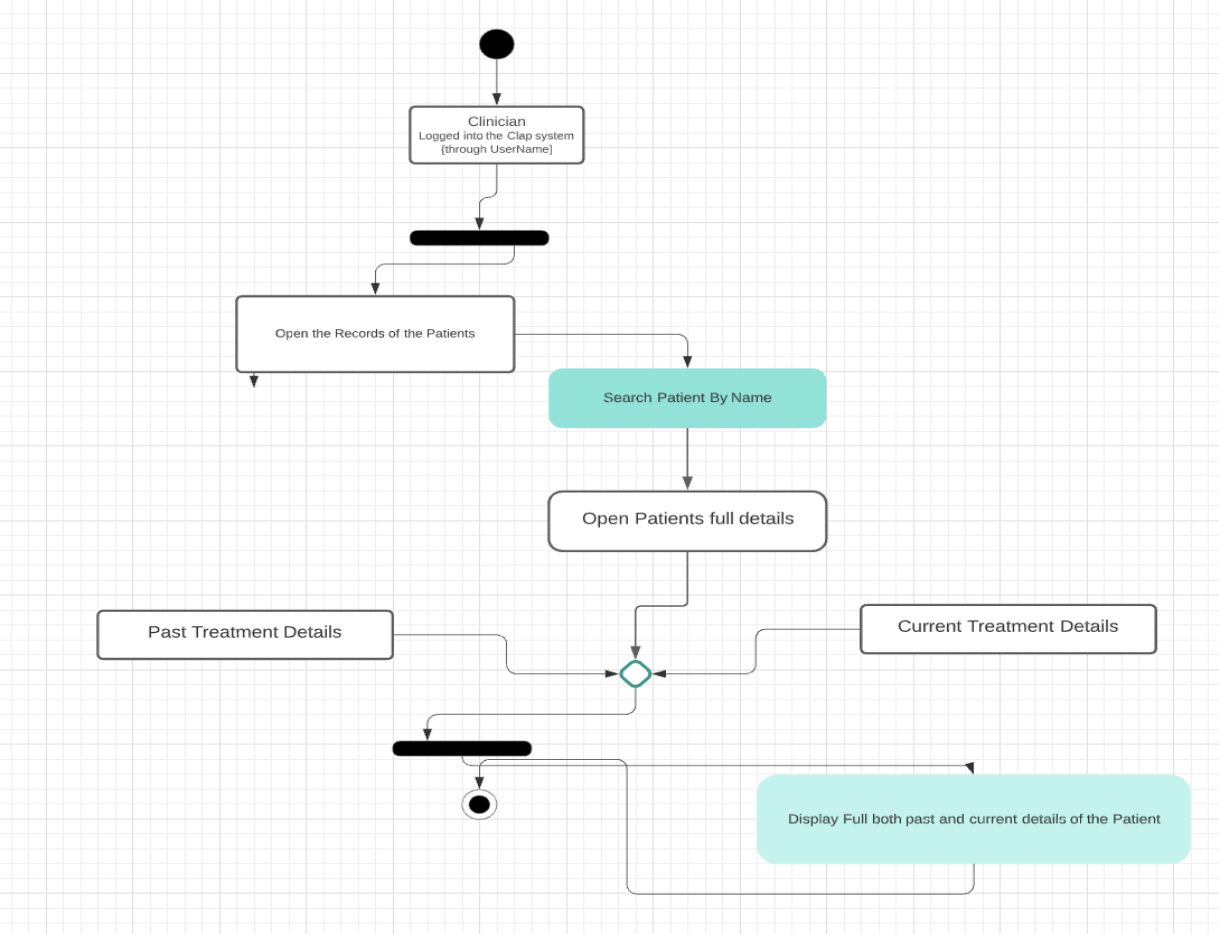


Figure 13: View Previous Treatment Activity Diagram

Appendix A: Glossary

Appendix B: Quality Assurance Process

This is intended to be the group's brief critique of what they have produced in the preceding parts of the document. It should explain what steps the group has taken to ensure the document's correctness, completeness and consistency. A reflection on the benefit of “iteration”, with reference to the evolution of the conceptual class diagram, should be included.

In the initial stages, the group had a meeting minute table to keep track of the weekly progress. It helped us to communicate better as there was a task set each week that should be worked on before the next meet-up. The group also used discord to communicate with members during the week to accommodate different availability times for each individual member. This 2-step process helped the group to get work done and keep track of the entire project as well as individual sections that were assigned to group members.

The assignment required many diagrams to be drawn for each different section. High quality diagrams were produced through the help of a web-application called Lucid-chart. Lucid-chart was an easy tool to learn which made the process of visual representation on the assignment much easier. The main key feature that stood out was the built-in activity diagram and use-case feature to make the diagram process simpler and less time consuming. Lucid-chart also allowed multiple people to work on the same project in real time which helped at times where collaborative work was required.

The overall assignment required good communication skills between the customer and client to ensure that the CLAP APP was moving in the right direction. Weekly meet-ups with the client to ask questions on what can be improved and also receiving feedback on the completed sections helped the group to understand the assignment better which helped to produce high quality work.

The team has created a sufficient list of use cases and recorded them in detail in the above section 2.1. These use cases were derived with attention to detail given to the functional requirements which have been given by the client. The team then followed up with meetings with the client to further investigate any missing use cases.

A diagram was drawn to represent these use cases in a high level format so that it would further clarify any areas of confusion between the client and amongst the team. This was followed by developing a diagram to represent the system architecture outlining the actors and other systems which interact with the application.

To improve the consistency a high level activity diagram was created by cross referencing what was being reflected in the use case table and use case diagram. The team decided that creating the diagrams using the same tool would also improve the consistency and allow easy access for all team members to view and edit the same diagram. Sometimes it was effective for multiple team members to access and edit simultaneously.

Based on the current information the first iteration of our conceptual model was created utilizing an object orientated approach. As the team discovered different ways of the user needing to interact with the data this would trigger changes to be made to preceding diagrams and use cases to maintain consistency. After several discussions a second iteration of the class diagram was made to refine the classes and better represent the relationships between them.

The team used project management software Azure DevOps to delegate the work fairly and help keep record of the documents progress as the deadline approached.

Team Meeting minutes

Date	In class	Online	Agenda items	Action Items	Progress
11/08/2020	Daniel, Obi, Aish	Karthikey, David	Each group member was allocated sections to work. The group also worked on setting up Microsoft azure for better communication.	Section 3: User stories, use case description	Completed
18/08/2020	Daniel, Obi, Aish, David	Karthikey	Section 3.1, writing as many use-cases to meet the requirements for the system.	Section 3: User stories, use case description	Completed
24/08/2020	Daniel David (Absent) Aish(absent)	Obi	The group planned how the activity diagram should match the overall system. We also worked on basic use-case diagram along with the basic system architecture.	Section 2 and 3 activity diagrams	Completed

1/9/2020	Daniel, Aish, David	Obi, Kartikey	Extended use cases and continue working on working activity diagrams,	Section 2 and 3 extended use case and activity diagram	Completed
8/9/2020	Daniel, Aish, David, Obi	Kartikey	Conceptual Model, Use-case diagram, Activity Diagram, Assurance process, Section 1.	Section 1, 2 and 3 being finalised.	Completed
15/9/2020	David, Aish	Daniel, Obi, Kartikey	Second conceptual diagram and finalising each section.	Section 1,2 and 3 being completed.	Completed