CS2212 - Group Project - Team 12

“Western Maps”

Requirements Documentation

Version	Date (DD-MM-YY)	Author (author@uwo.ca)	Summary of Changes
0.3	01-02-23	aboulos4	Creation of Doc - Organization of Sections - Main Labels
0.4	02-02-23	aboulos4	Functionality requirements - Actors + 5 Use cases + 5 Activity Diagrams done
0.5	02-02-23	kkhalil5	6 Use cases + 6 Activity Diagrams done + Main Introduction
0.6	04-02-23	ssagar26-gmenon3	Introduction+Domain Analysis Complete
0.7	04-02-23	ssagar26	Non-Functional Requirements Done
0.8	05-02-23	ssaraf	Completed Use Cases + Activity Diagrams
0.9	06-02-23	kkhalil5 - aboulos4	Main Use Case Diagram Completed+Summary
1.0	06-02-23	All Team	Final Checks Done, Grammatical + Appearances fixed + Added to confluence.

Introduction

Overview

The goal of this project is to create navigation software for some of Western University's campus buildings, to help the public more efficiently navigate them. Currently, only PDF format versions of the campus's internal structure are available. Users of this program will be able to look up rooms, filter certain points of interest (POIs for short), select certain layers to be viewed, as well as simply browse through the map more effectively as well as select custom POIs. The main target of this project is to simplify and improve users' efficiency and capabilities at moving through these complicated university interior spaces. The program will overall be user-friendly and come with a guide to help new users pick it up quickly. Furthermore, an accompanying editing tool, put in by developers for developers, to facilitate the creation/editing of the map metadata is also available to make sure any future changes/updates to improve usability and functionality can be implemented.

Objectives

- Applying the principles of software engineering towards a solving real-world problem
- Working with, interpreting, and following an established project specification document
- Creating models of requirements and design using such tight specifications
- Dealing with decisions made earlier in the design process and implementing designs in Javas
- Creating graphical, user-facing content and applications
- Writing robust and efficient code
- Write good, clean, well-documented Java code that adheres to best practices
- Reflecting on good/bad design decisions made over the course of the project

References

- CS2212 Group Project Specification document
- Wufloorplans
- https://www.roam.ai/blog/5-common-problems-with-location-data

Domain Analysis

This project's software domain is location-based services. This falls under the subdomain of indoor mapping and navigation, with a focus on academic buildings.

In this domain, precise and efficient building mapping is required to assist users in navigating and exploring internal spaces. This includes searching for rooms, locating areas of interest, and creating and saving personal points of interest.

Some of the most prevalent issues found in this domain are:

1. User accessibility: In this field, it is essential to make sure that the user interface is clear and simple to use for all users, regardless of ability.
2. Effective navigation: It might be difficult to provide techniques for finding and navigating interior settings, such as searching by building, room name, or point of interest.
3. Real-time updates: It's critical in this field to make sure that maps and data on indoor places are correct and reflect changes or revisions in real-time.
4. Privacy and security: In this sector, it is important to ensure the privacy and security of user data, including their location and search history.

Some solutions to the problems listed above:

1. When creating the user interface, standards like the Web Content Accessibility Guidelines (WCAG) can be followed to ensure user accessibility. This can be accomplished by using straightforward language, making sure that the text and background contrast effectively, and offering alternative text for images.
2. To provide effective navigation, a mix of search and filtering options—such as searching by building, room name, or point of interest—can be used. Additionally, real-time updates can be offered to improve navigation by giving the user their present location and potential routes to their goal.
3. A Building Information Modeling (BIM) system, which is a digital representation of a building and its components, can be coupled with the system to ensure real-time updates and instantly reflect any modifications or upgrades.
4. Sensitive data, including user location and search history, can be encrypted and kept securely to maintain privacy and security. In order to prevent unwanted access, the system can also employ secure authentication and authorisation mechanisms.

How can we use this domain understanding to improve or accelerate development of this project?

1. User-centred design: The project team will design and create the program with the user in mind, so that it is accessible, simple to use, and satisfies the user's needs, by knowing the user's expectations and wants in this domain.
2. Address frequent problems: The project team can proactively deal with these problems in the creation of the project by being aware of the common problems in this field, ensuring that it is effective, accurate, and current.
3. Additionally, this understanding can guide the development team in making informed decisions about the technologies and tools used to build the application, such as using Java 19, JavaFX, JSON and Javadoc

Functional Requirements

Functionality to be delivered:

1. Browsing between maps.
2. Scrolling the map if it is not fully visible.
3. Display layers (POIs [Point of Interest] that fall under the same category).
4. Hide Layers.
5. Ability to search the map for particular POIs with text.
6. Clicking on a POI from a list of them should display it on the current map, highlight it, scroll to its location, and display a short description of the POI.
7. Maps should have built in POIs such as classrooms, stairwells and elevators, washrooms, and entry/exit points.
8. Clicking a POI should highlight it and display information about it
9. Users should be able to mark POIs as favourites which can be quickly accessed from a favourites menu
10. Users should be able to create their own custom POIs which will have their own layer and can be accessed from a menu.(minimum 5)
11. All data must stay saved between sessions.
12. Cleanly exit the application from any window. If the user is in the middle of a task, a warning should be displayed.
13. Users should have access to a help page which covers how to perform all tasks in the application.
14. There should also be an about page that provides information on the application and its team.
15. There should be an editing mode that allows developers to edit the data for the built-in POIs.
16. The application will display the current weather by means of a weather API.

Scenario Model:

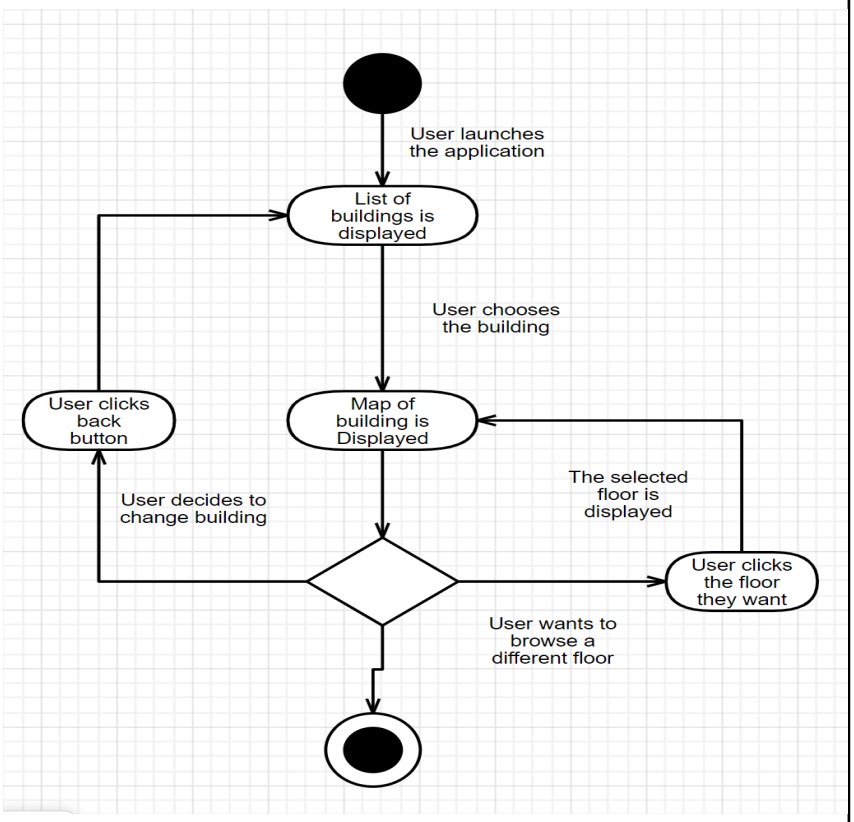
Actors

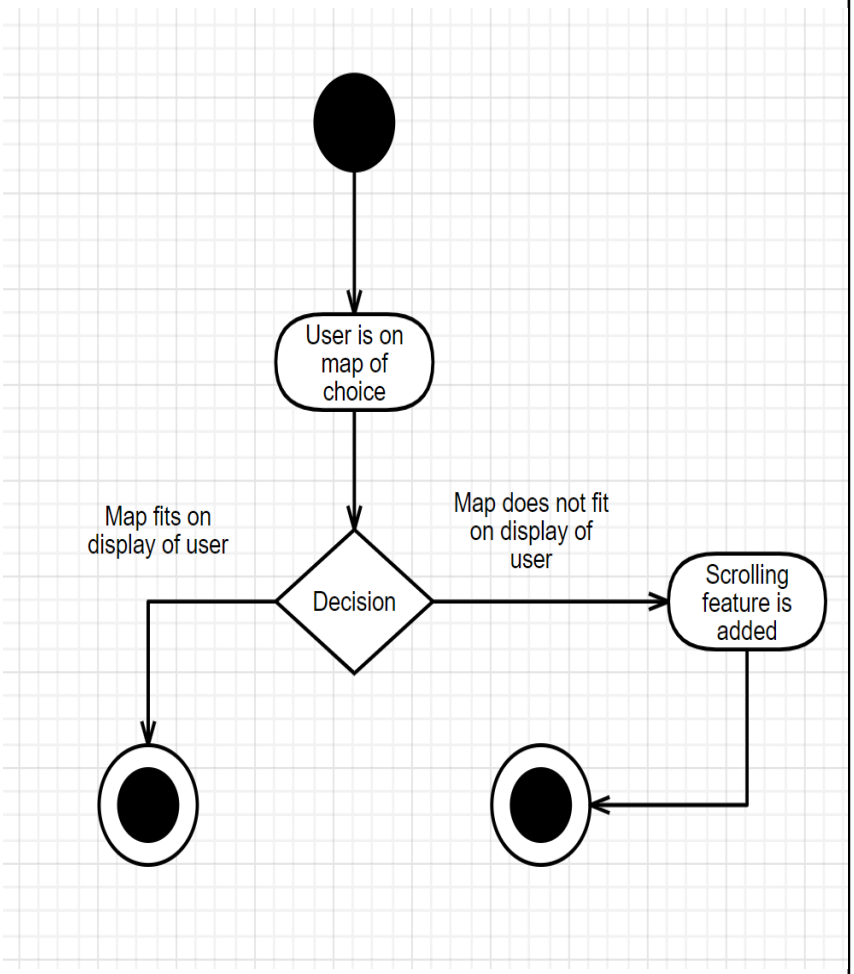
Actor	**People of Western**
Description	A person who works or studies at Western (Student or Staff) and wishes to navigate the buildings' interior spaces.
Aliases	User
Inherits	None
Actor Type	Person
Active/Passive	Active

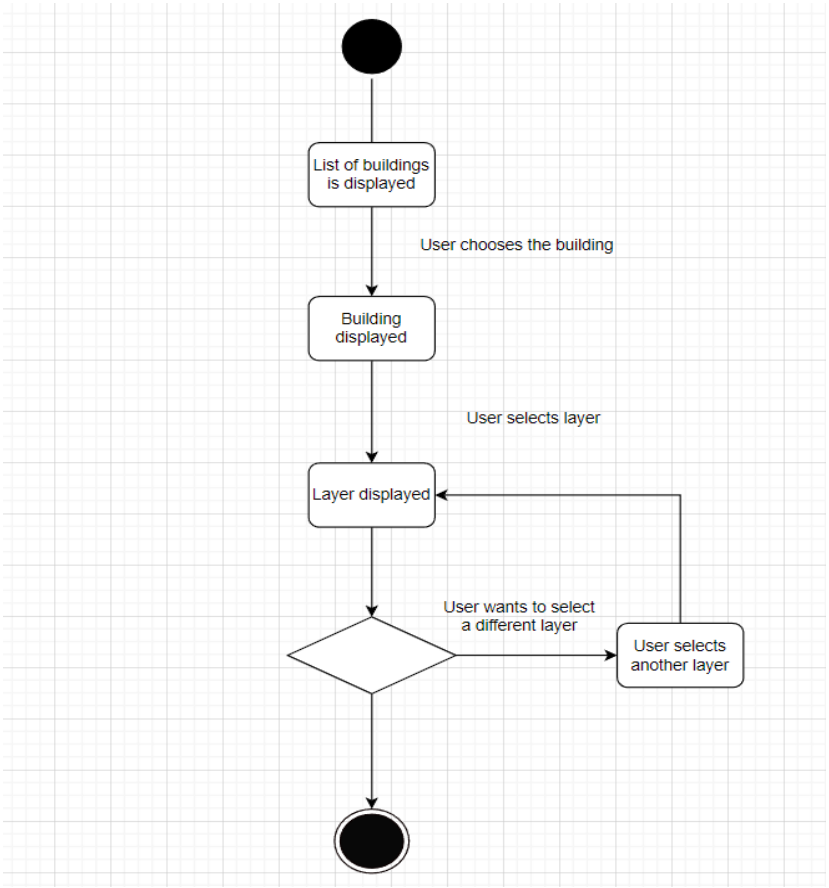
Actor	**Developer**
Description	A person who manages the application and ensures everything is running smoothly and can make changes to the application.
Aliases	Team || Team member
Inherits	User
Actor Type	Person
Active/Passive	Active

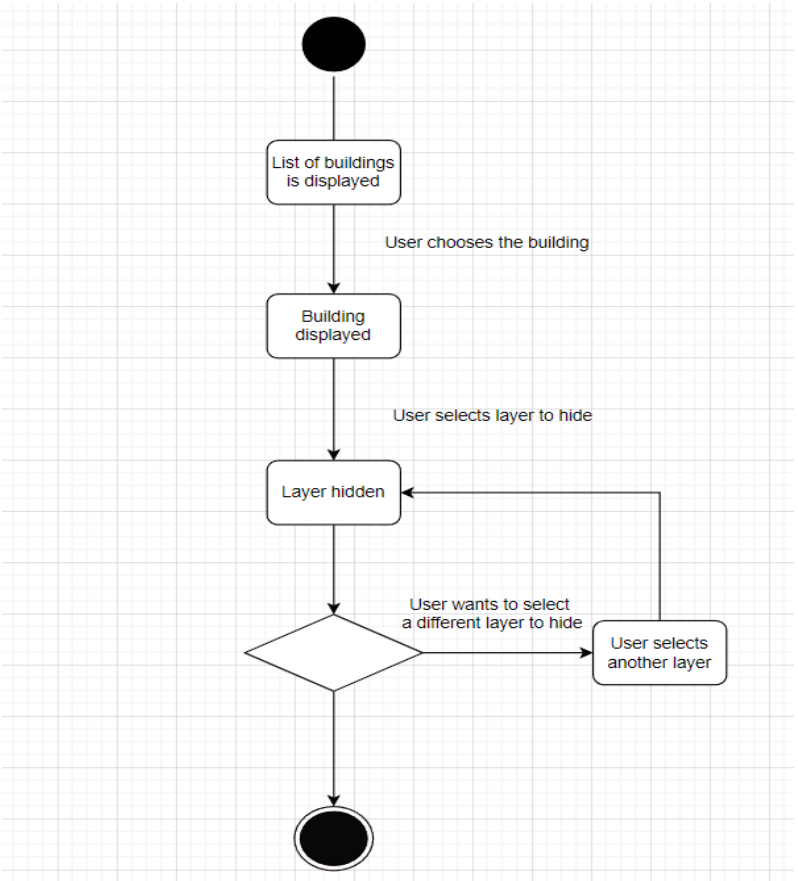
Actor	**Weather API**
Description	Used to pull current weather data to be displayed in the application
Aliases	None
Inherits	None
Actor Type	External system
Active/Passive	Passive

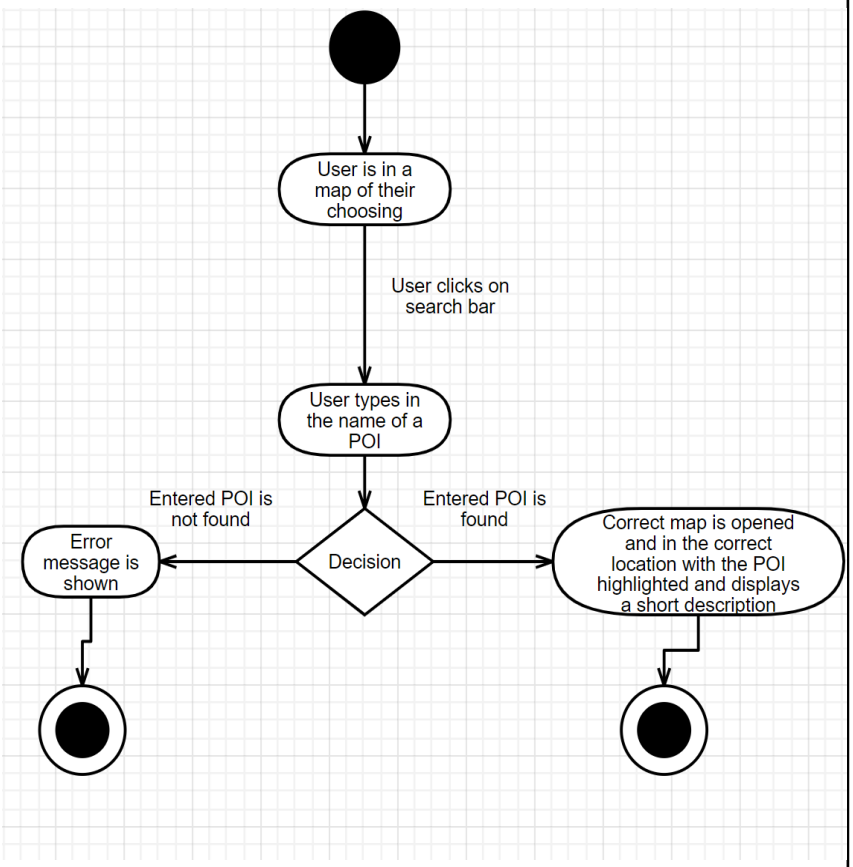
Use Cases

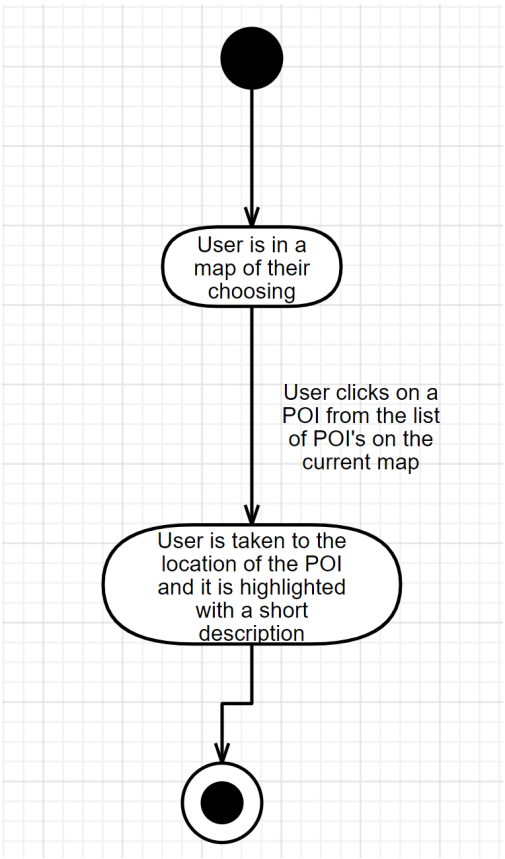
Name	**Browsing the maps**
Primary actor	User
Secondary actor	None
Goal in context	To allow the user to browse all the maps of a building as well as easily switch between buildings
Preconditions	The application has been opened
Trigger	The user chooses a building from the list of buildings
Scenario	1. Users launches the application 2. User chooses from the list of buildings which building they want to view 3. User can browse through the map of each floor by selecting the desired level. 4. The user can switch buildings by going back to step 2 and picking a different building from the list
Alternatives	None
Exceptions	None
Priority	Highest
Activity Diagram	

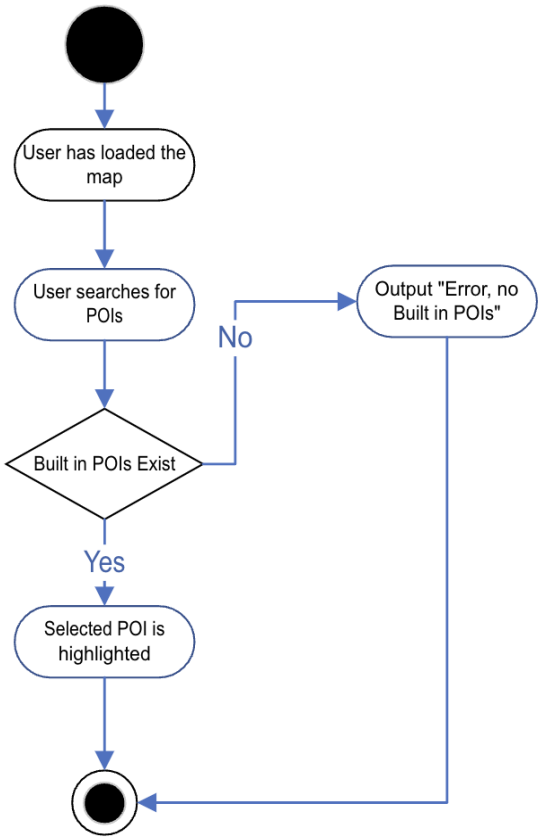
Name	**Scrolling the maps**
Primary actor	User
Secondary actor	None
Goal in context	The user must be able to scroll around the map.
Preconditions	The user is viewing a map
Trigger	The user is not able to view the map in its entirety.
Scenario	1. User launches the application 2. User chooses a building from the list 3. Map is not visible in its entirety on the screen 4. User can scroll around the map to see the rest of it
Alternatives	Map fits on the display and does not need scrolling.
Exceptions	none
Priority	High
Activity Diagram	

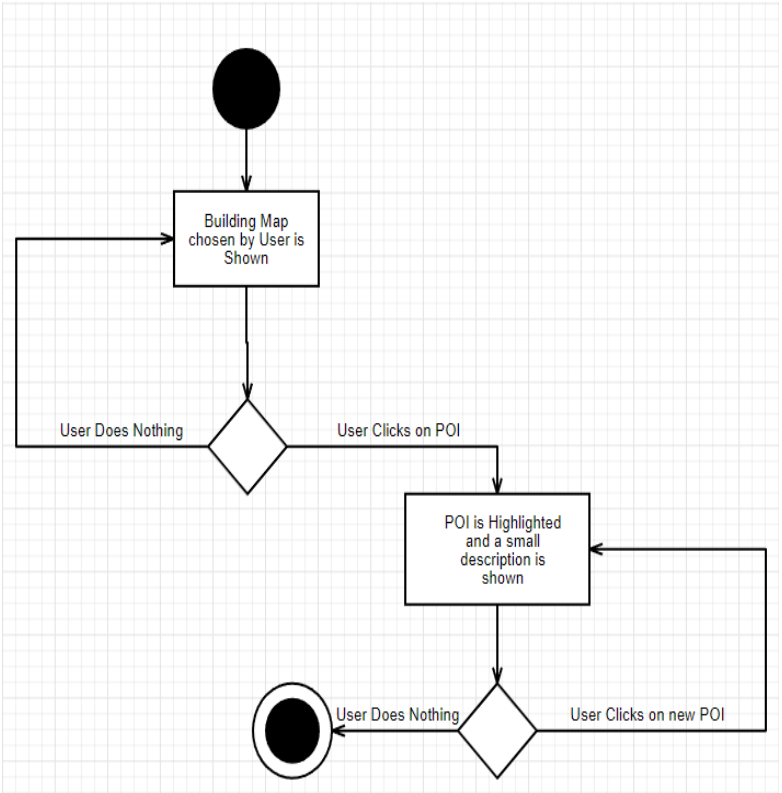
Name	**Displaying the layers**
Primary actor	User
Secondary actor	None
Goal in context	The user must be able to show the different layers of the building (POIs)
Preconditions	The user has already opened the application and selected a building they would like to view
Trigger	User has clicked on the show button for the different layer(s)
Scenario	1. User is in the application 2. User chooses a building from the list 3. User clicks the layer they want to show 4. User is now able to see that layer
Alternatives	None
Exceptions	none
Priority	Medium
Activity Diagram	

Name	**Hiding the layers**
Primary actor	User
Secondary actor	None
Goal in context	The user must be able to hide the different layers of the building (POIs)
Preconditions	The user has already opened the application and selected a building they would like to view
Trigger	User has clicked on the hide button for the different layer(s)
Scenario	1. User is in the application 2. User chooses a building from the list 3. User clicks the layer they want to hide 4. User no longer sees that layer
Alternatives	None
Exceptions	None
Priority	Medium
Activity Diagram	

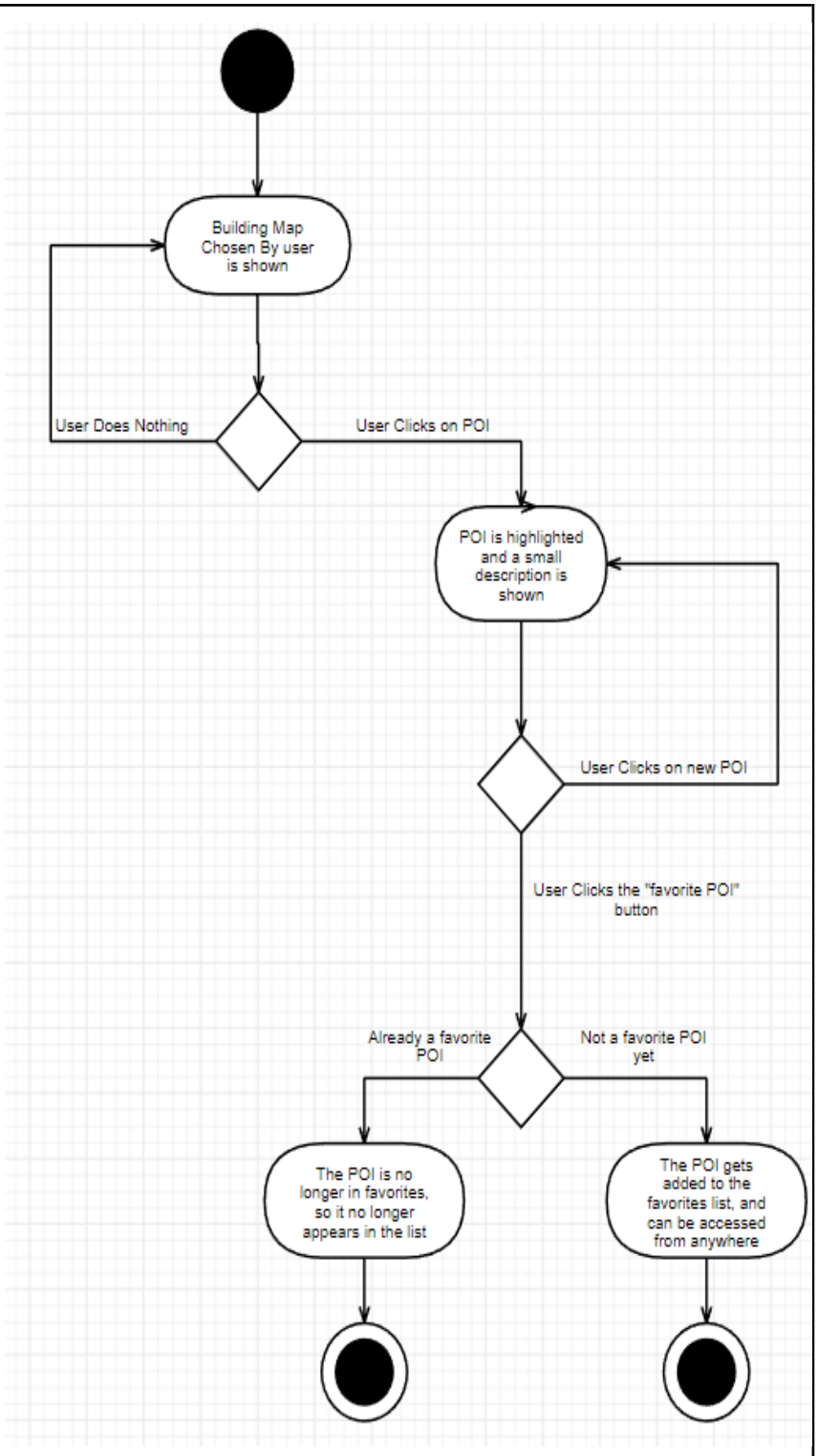
Name	**Searching the maps**
Primary actor	User
Secondary actor	None
Goal in context	User can search for a POI and the correct map is displayed, the POI is highlighted, displays a short description, and the map is in the correct location.
Preconditions	User is in the application
Trigger	User types a POI in the search box
Scenario	1. User is in the application 2. User clicks on the search box 3. User types in the POI they want to find 4. User is taken to the correct map at the correct location 5. User can see the POI highlighted with a short description
Alternatives	User can just navigate to the POI manually
Exceptions	An error message is shown if the POI is not found.
Priority	Medium
Activity Diagram	

Name	**Discovering Points of Interest (POIs)**
Primary actor	User
Secondary actor	None
Goal in context	User can choose from a list of POI on the current map which will display it, highlight it, scroll to the correct location on the map, and display a short description.
Preconditions	User has chosen a map after opening the application.
Trigger	User has clicked on a POI from the list of POIs on the current map.
Scenario	1. User has opened the application 2. User has chosen a map on a building 3. User has clicked on a POI from the list of POIs 4. POI is displayed on map, highlighted, scrolled to, and a small description is shown
Alternatives	User can just navigate to the POI manually or search for it
Exceptions	There are no built in POIs for the said map
Priority	Medium
Activity Diagram	

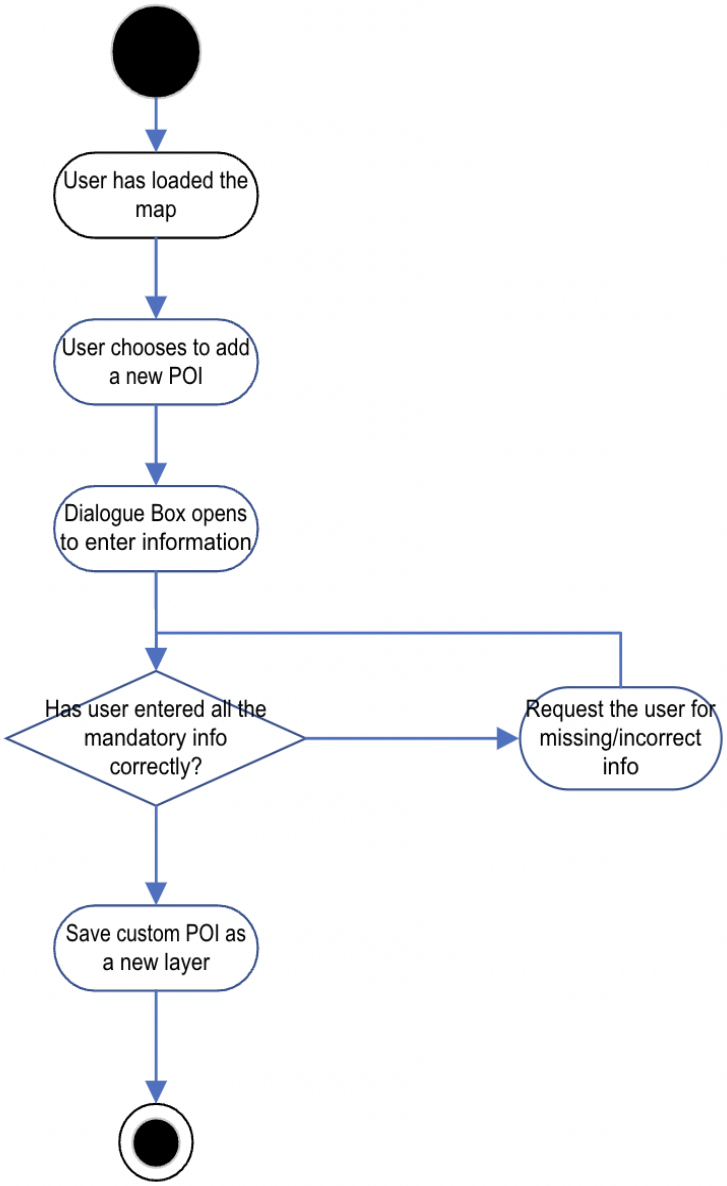
Name	**Built-In POIs**
Primary actor	User
Secondary actor	None
Goal in context	Standard POIs within the building such as classrooms, eateries, which can be searched for faster access
Preconditions	User has loaded a map with POIs built in
Trigger	Searching for a POI or choosing from a list
Scenario	1. User launches the application 2. User chooses a map on a building 3. User has clicked on a POI from the list of POIs 4. POI is displayed on map, highlighted, scrolled to, and a small description is shown
Alternatives	User may also search for the POI, if there is available metadata
Exceptions	There are no built in POIs for the said map
Priority	Medium
Activity Diagram	

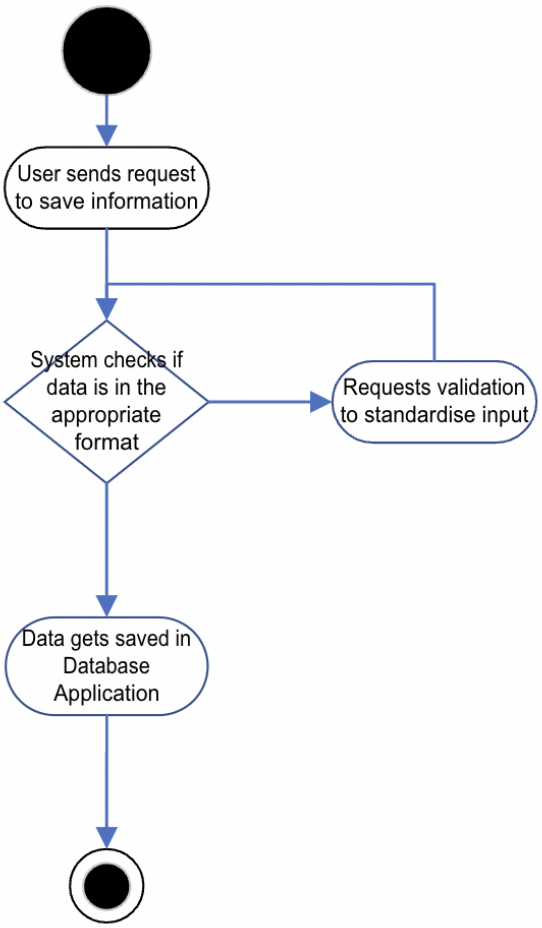
Name	**Clicking on POIs**
Primary actor	User
Secondary actor	None
Goal in context	User must be able to click on POIs. Must result in clear indication of the click and display some minimal information/description about the POI as a result
Preconditions	User has chosen a map after opening the application.
Trigger	User clicks on a POI
Scenario	1. User has opened the application 2. User has chosen a map on a building 3. User has clicked on a POI shown on the map 4. POI is displayed on map, highlighted and a small description is shown
Alternatives	User may also search for the POI, if there is available metadata
Exceptions	None
Priority	High
Activity Diagram	

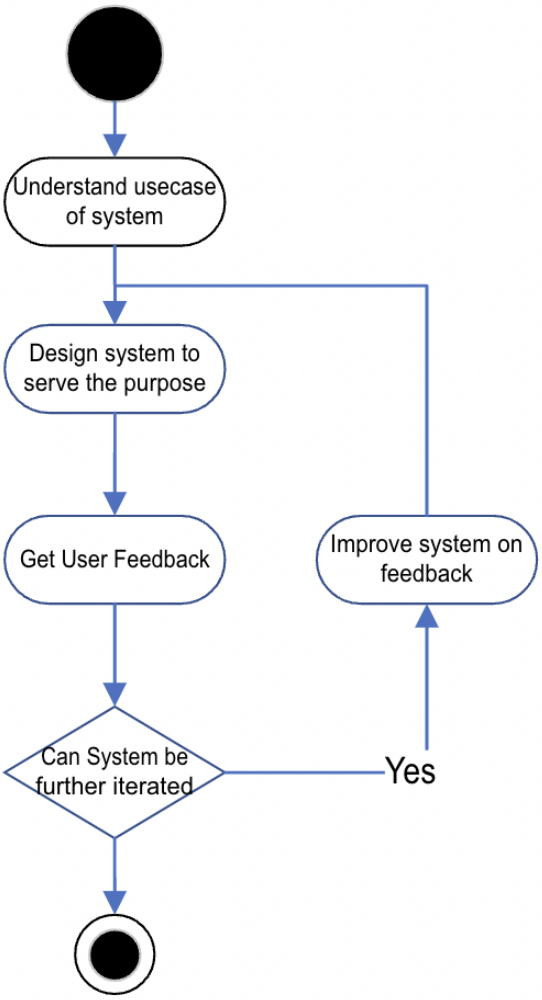
Name	**Choosing POIs as favourites**
Primary actor	User
Secondary actor	None
Goal in context	User must be able to choose certain POIs (custom or other) as a favorite, and it will get added to a favorites list
Preconditions	User has chosen a POI, after choosing a map and opening the application.
Trigger	User clicks on POI “favorite” button
Scenario	1. User has opened the application 2. User has chosen a map on a building 3. User has clicked on a POI 4. User clicks on POI “favorite” button 5. POI get’s added to user’s “favorites” list 6. POIs on this list can be accessed from anywhere, and their location will be highlighted and navigated to when clicked.
Alternatives	POI is already a favorite, which will result in it “unfavoring” and being removed from the list (reverse of above described action)
Exceptions	Favorites list may be full
Priority	Medium

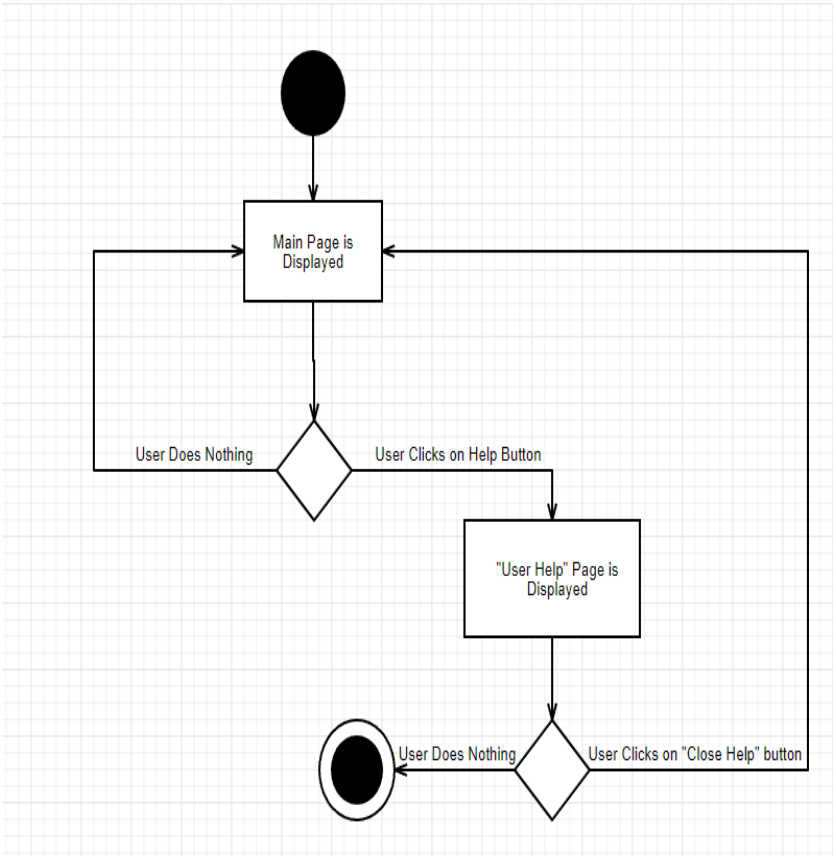
Activity Diagram	

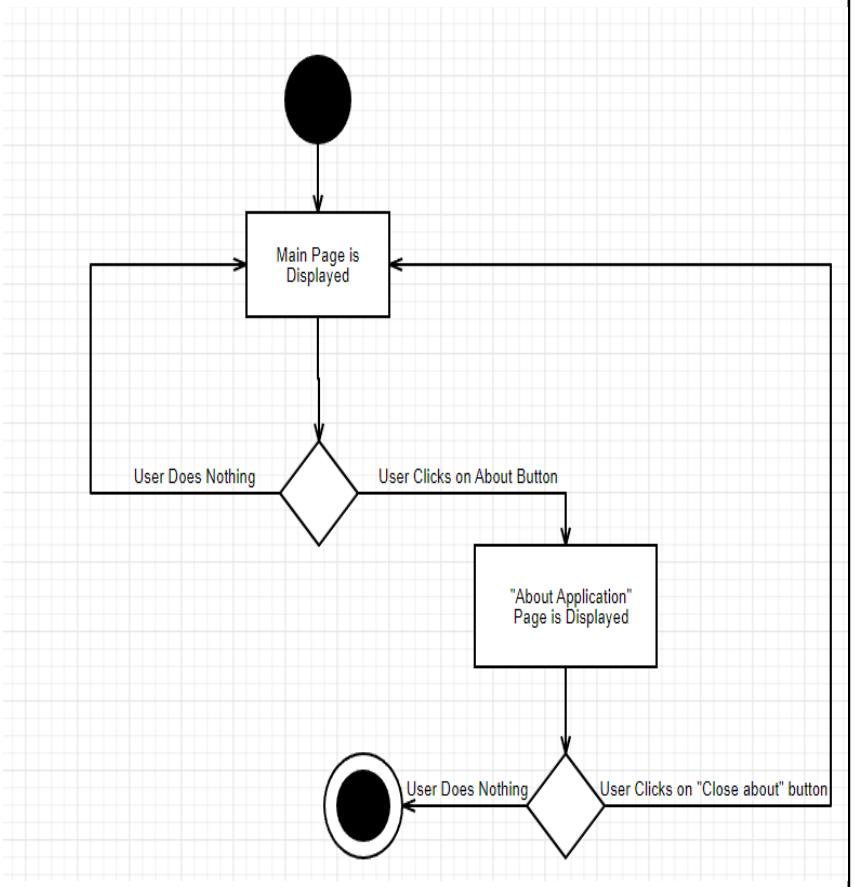
Name	**Custom POIs**
Primary actor	User
Secondary actor	None
Goal in context	User must be able to add their own custom POIs to the map

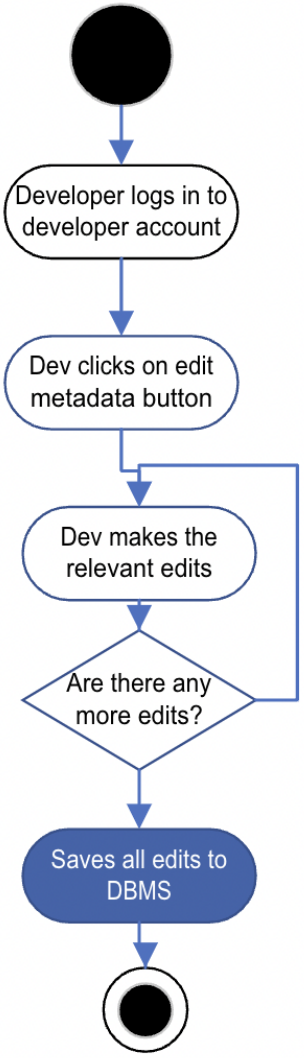
Preconditions	User chooses a point on the map, and adds name, description and room number at a minimum
Trigger	User selects option to add a new POI
Scenario	User may want to add a space not previously marked by built in POI as a custom POI to ease navigation to the space. They do so by selecting the space and clicking, add POI. POI saved on new layer.
Alternatives	User may point to a place manually on the maps
Exceptions	User is trying to reach to a Built in POI, which is already programmed into the system
Priority	Medium
Activity Diagram	

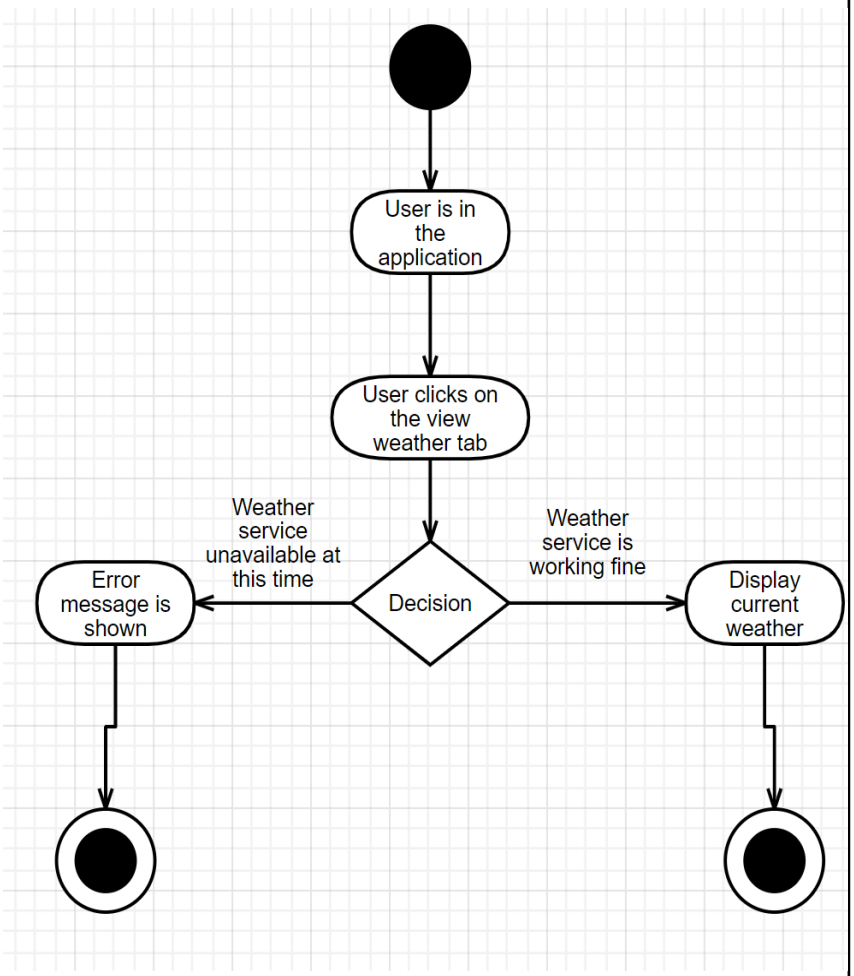
Name	**Data remains saved**
Primary actor	Developer
Secondary actor	none
Goal in context	Save data appropriately to deliver a consistent user experience
Preconditions	Appropriate Database solutions and connections to capture data requests from the application
Trigger	Request from application to save or delete any form of information
Scenario	User requests to save a new Custom POI
Alternatives	None
Exceptions	Some unknown error may cause data to be not saved (unplanned)
Priority	High
Activity Diagram	

Name	**Application usability**
Primary actor	Developer
Secondary actor	User
Goal in context	Ensures the application has an easy interface to reduce a learning curve and finds benefit from the application
Preconditions	None
Trigger	Need of good design principles
Scenario	User finds it difficult to interact with the system, needs easy interface to use the system
Alternatives	None
Exceptions	None
Priority	High
Activity Diagram	

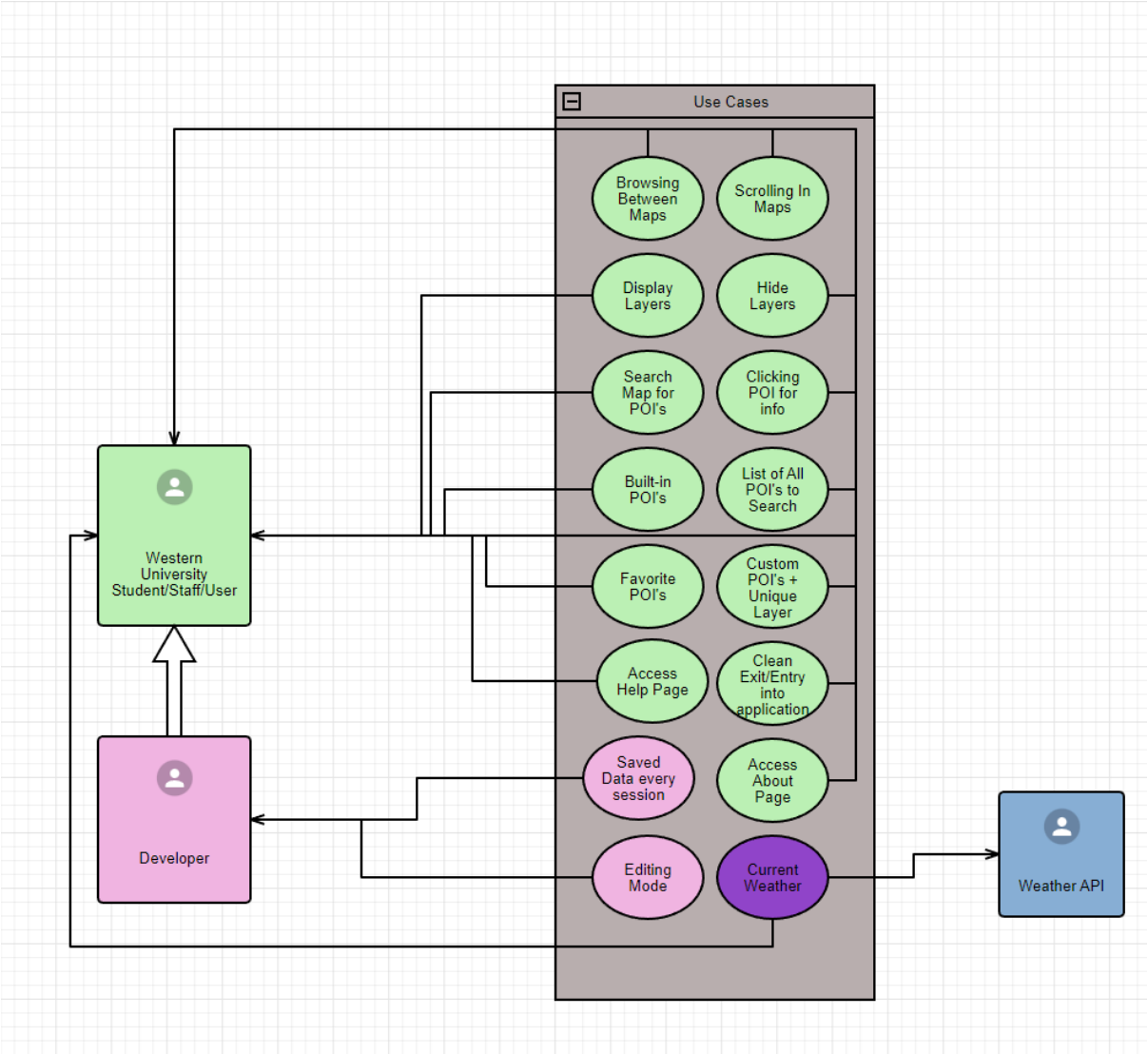
Name	**User help tab**
Primary actor	User
Secondary actor	None
Goal in context	User must be able to get offline help at any time regarding the features of this application, by means of a user guide.
Preconditions	The user has opened the application
Trigger	User navigated to the help button and clicked it
Scenario	1. User opens the application 2. User clicks on the help button 3. Locally stored page gets opened, explaining the application buttons, usage etc
Alternatives	None
Exceptions	None
Priority	High
Activity Diagram	

Name	**About tab**
Primary actor	User
Secondary actor	None
Goal in context	User must be able to visit an “about” screen displaying basic information about the application such as name, version, release date, and team members and their respective contacts
Preconditions	The user has opened the application
Trigger	User navigated to the about button and clicked it
Scenario	1. User opens the application 2. User clicks on the about button 3. Locally stored page gets opened, displaying information about the application such as name, version, release date, and team members and their respective contacts
Alternatives	None
Exceptions	None
Priority	High
Activity Diagram	

Name	**Developer mode**
Primary actor	Developer
Secondary actor	None
Goal in context	Easily graphically edit the metadata about the Built in POIs
Preconditions	Available only to the developers of the system
Trigger	Acceible only through a special account or a similar methodology
Scenario	Developer may want to graphically edit the metadata of Built in POIs
Alternatives	Change the information in the database via a series of queries
Exceptions	None
Priority	High
Activity Diagram	

Name	**Current weather**
Primary actor	User
Secondary actor	Weather API
Goal in context	Display the current weather conditions in the application
Preconditions	The application is connected to a weather API
Trigger	User opens the application and clicks on view current weather
Scenario	1. User launches the application 2. User clicks on view Current weather 3. The current weather is displayed with an icon that shows the weather conditions
Alternatives	None
Exceptions	An error message is shown if the weather service/API is unavailable.
Priority	Low
Activity Diagram	

Use Case Diagram



Non-Functional Requirements

- *Performance*:
    - Maps should load quickly
    - Will be executable on a PC operating windows 10 or above
    - Will be well self-contained and not create, modify, or delete files outside of the directory in which the application is installed.
    - File size will be under 1GB
- *Usability*:
    - Intuitive and user-friendly interface
    - Easily search for and find rooms, points of interest, and maps
    - Colour will be coded carefully in the UI
- *Accessibility*:
    - Accessible and usable with a keyboard or mouse
    - Logical tab order for UI elements
- *Coding styles and conventions:*
    - Use current version of Java (19) and NetBeans IDE
    - Use camelCase for naming variables, methods, and classes.
    - Using images to store maps
    - Code will be tested using Junit5 tests
- *JavaFX*:
    - Keep UI elements and logic separate
    - Use FXML for layout and Java for logic
    - Use CSS for styling
- *JSON*:
    - Use double quotes around property names and values
    - Avoid using special characters in property names

- *Javadoc*:
    - Use Javadoc comments for documenting classes, methods, and variables
    - Include a description of the purpose and behaviour of the class.
    - Use the @param, @return, and @throws tags to describe the purpose and behaviour of methods

The Bitbucket Git repository will house all of the project's code and files. Confluence will be used to store and develop all design work and diagrams for this project. Jira will be used to track all project tasks and issues.

It's also important to maintain consistency throughout the code, so it's recommended to establish and follow a specific coding style guide. Tools such as Checkstyle might be used to enforce these conventions automatically.

Summary

To conclude the project aims to develop navigation software for Western University's campus buildings to help the public navigate them more efficiently. The software will allow users to look up rooms, filter POIs, view selected layers, and browse the map effectively. The target is to simplify and improve navigation in complex university interior spaces. The software will be user-friendly with a guide to help new users and an accompanying editing tool for developers to update the map metadata. Throughout the duration of this project, the team objectives revolve around applying software engineering principles, working with project specifications, creating requirements and design models, implementing designs in Java, creating user-facing content, writing efficient and clean code, and reflecting on design decisions made during the project.

Terms, notations and acronyms used:

1. CamelCase- a method of separating words in a sentence by capitalising the initial letter of each word and not using spaces. Ex- YouTube, iPhone etc.
2. POI (Point of Interest) - Refers to locations on the map that have some use to them. Ex- They are a classroom, or lab, or maybe a custom created point by the user etc.