Software Requirements Specification

OpenFlexure Microscope

Onat ÖZDEMİR - 2310399

Contents

List of Figures

List of Tables

Revision History

Date	Reason For Changes	Version
15.04.2021	Initial Draft	0.1
22.04.2021	Final Version	1.0

Table 1: Revision History of Software Requirements Specification Document

1 Introduction

This document is the System Requirement Specification for the OpenFlexure Microscope project introduced by the group of researchers from the University of Bath.

1.1 Purpose of the System

The purpose of this project is to design a customizable, low-cost and locally-manufacturable [1] microscope that can be easily built by people from all over the world. Unlike complex and expensive microscopes, OpenFlexure Microscope can be built using 3d machines in low-cost and even an inexperienced person can easily assemble and use it.

1.2 Scope

- System will have a server that runs an internally stored OpenFlexure Web API. Connected users will be able to send requests using this Web API.
- Users will be able to connect to the system both locally and remotely.
- System will have interface for users. Users will be able to navigate microscope stage, view camera livestream, capture images and record videos using this interface. They will also be able to manage and modify the saved media.
- System will have interface for system admins. System admins will be able to manage system settings. Additionally, they will be able to view connected users and manage these connections using this interface.
- System will have interface for maintainers. Maintainers will be able to view detailed system logs which contain description, time and type information. Additionally, they will be able to check software updates and install these updates to the system using this interface.
- System will have interface for maintainers. Maintainers will be able to view detailed system logs which contain description, time and type information. Additionally, they will be able to check software updates and install these updates to the system using this interface.

- The system has many information such as connected user IP addresses, admin and maintainer information, saved media and system logs to save to and retrieve from database. To provide secure and stable connection, the system will also have Database Management Interface.
- Since microscopes are error-prone systems, to hande these errors system will have Back-end Failure Management Interface. When an error occurs, the system will save the detailed log of this error in the protected area of the database and maintainers will be able view these system failure logs.

However,

- Designing the hardware components of the OpenFlexure Microscope
- Designing and implementing the OpenFlexure Connect desktop application
- Designing and implementing the OpenFlexure Web API
- Designing and implementing the OpenFlexure Development Server and Update API

are out of the scope of this project.

1.3 System Overview

This section provides detailed information about the system with its components.

1.3.1 System Perspective

OpenFlexure is not a part of a larger system. However, it interacts OpenFlexure Development Server in order to keep the system software up-to-date. Additionally, the communication between the OpenFlexure and Users is provided by OpenFlexure Web API. Users will be able to send their requests using the Web API and get responds from the server again using this API.

There are three user types in the system: User (can be seen as end-user), Admin and Maintainer. The system will provide different interfaces to each of them so that they can do the functions defined to them. Since admins and maintainers have private permissions, they will need to log in to the system using the keys defined to them in order to use their functions. While Admins and maintainers will be responsible for keeping the system stable, Users will be able to control and use OpenFlexure Microscope.

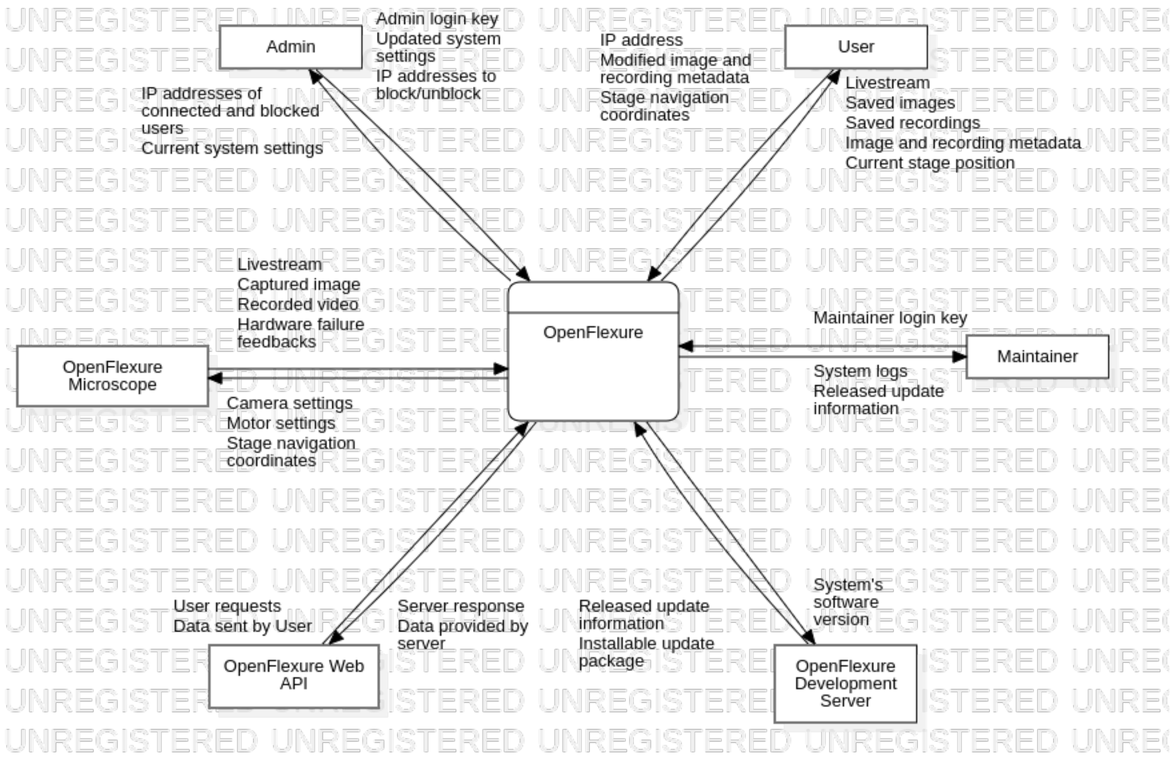


Figure 1: System Context Diagram for OpenFlexure

1.3.1.1 System Interfaces

Database Management Interface: Database Management Interface is one of the primary components of OpenFlexure. It allows storing the IP addresses of users, activities of users, login keys of administrators and maintainers, error logs, etc. When any other interface wants to write to or read from the database, Database Management Interface provides a stable and secure connection to the system database.

Back-end Failure Management Interface: Due to the nature of the microscope components, it is highly possible to have hardware and software failures in microscopes. Managing these failures and acknowledging the responsible staff (maintainers) is a crucial task. This interface is always active in case of handle any system failures. It is responsible for detecting anomalies and failures in the system and keeping detailed logs about the situations. When the interface detects a failure, it communicates with the Database Management Interface to store the error logs in a protected database area that can be reached by maintainers.

1.3.1.2 User Interfaces

User Connection Interface: OpenFlexure desktop application will provide connection interface to the users. Desktop application is not mandatory to connect the system. If user is already know the ip address of the microscope, he/she can directly open User Web Interface by entering the ip address from his/her browser. Desktop application is helpful to detect the microscope servers nearby the user.

User Web Interface: User will be able to control microscope and view the camera feed using the user web interface. Web interface will provide live stream camera feed, navigation menu to change the position of the stage and gallery window to view saved media. User will be able to reach to User Web Interface by just typing the ip address of the microscope's server. The GUI will send the events to Web API, then Web API will transform these events to HTTPS requests.

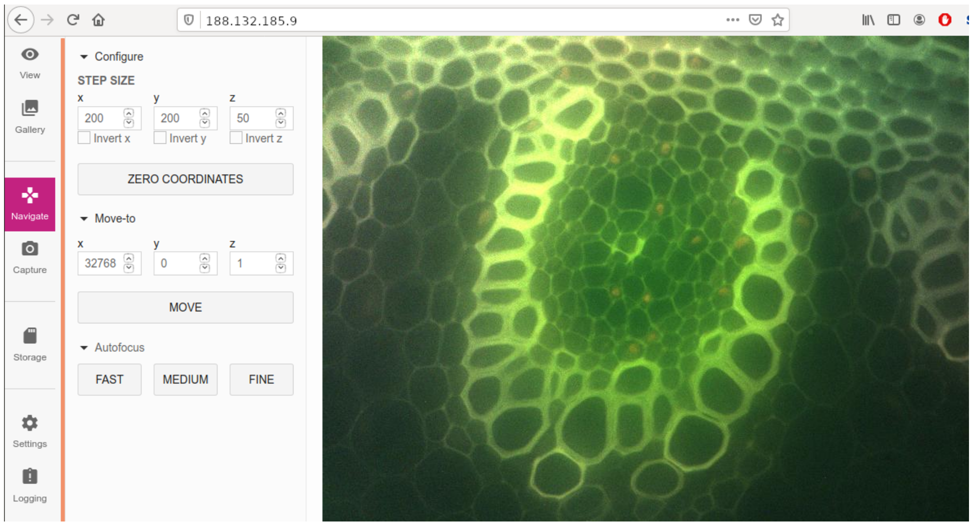
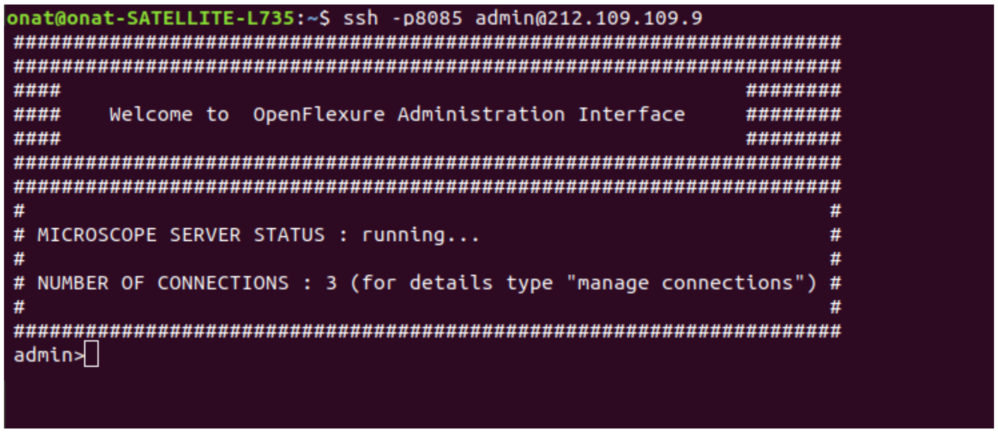


Figure 2: OpenFlexure User Web Interface

System Admin Management Interface: System administrators will use a command-line interface to manage the system. The reason to use a command-line interface rather than GUI is related to speed. Administrators should be able to apply their actions immediately. Since the command-line interface is a more quick way to apply actions, it is preferred over GUI. Administrators will be able to list the detailed information about connected/blocked users and system settings. Using the commands provided by this interface, the administrators will be able to manage/update system settings and block/unblock users.



```
onat@onat-SATELLITE-L735:~$ ssh -p8085 admin@212.109.109.9
####################################################################
####################################################################
####                                                        ########
####    Welcome to  OpenFlexure Administration Interface    ########
####                                                        ########
####################################################################
####################################################################
#                                                                  #
# MICROSCOPE SERVER STATUS : running...                            #
#                                                                  #
# NUMBER OF CONNECTIONS : 3 (for details type "manage connections") #
#                                                                  #
####################################################################
admin>
```

Figure 3: OpenFlexure Administration Command-line Interface

Maintainer Management Interface: Similar to system administrators, the maintainers will also use a command-line interface to manage the system, due to the similar reasons. Since the permissions of the maintainers are limited to system update and retrieving the system logs, commands available to maintainers will also be limited. Maintainer will be able to list saved error logs, system details such as software versions and the information about the latest software updates. Furthermore, maintainers will be able to update the system softwares using the command provided by this interface.

1.3.1.3 Software Interfaces

- Python 3.8 is used for back-end operations
- Flask 1.1.2 is used to implement web application
- PostgreSQL 13.2 is used as database management system

1.3.1.4 Communications Interfaces

Desktop application and web application uses HTTPS protocol to communicate with the back-end system.

1.3.1.5 Hardware Interfaces

OpenFlexure microscopes have two main hardware components: Raspberry Pi camera system to retrieve the livestream, captured image-video data, and motor system to control the position of the stage. To manage and communicate with these components, the system shall have two hardware interfaces:

Camera System Interface: The Camera System Interface is a medium between a Raspberry Pi Camera and the server. The server retrieves the livestream and captured image-video data using the Camera System Interface. Additionally, camera configurations such as frame rate, resolution, and commands such as capture image, record video are also transferred to the Raspberry Pi Camera using this interface.

Motor System Interface: The microscope shall have three motors to control the stage in the x, y, and z axes. These motors will be controlled by an Arduino Motor Controller. Motor System Interface provides a stable connection between the server and the Arduino Motor Controller. The system gives the navigation coordinates of the stage to Arduino Motor Controller using this interface.

Both the Camera System Interface and Motor System Interface also communicate with the Back-end Failure Management Interface to report any hardware-related failures.

1.3.1.6 Memory Constraints

System engineering is already handled for the OpenFlexure Microscope system as a whole.

1.3.1.7 Operations

The operations of the OpenFlexure system can be represented as:
User:

1. Connect to Microscope
2. View Livestream
3. Record Video
4. Capture Image
5. Scan

6. Navigate Stage
7. Open Gallery
8. Modify Media

Admin:

1. Log in as Admin
2. Manage System Settings
3. Manage Connections
4. Block User
5. Unblock User

Maintainer:

1. Log in as Maintainer
2. View System Logs
3. Check Updates
4. Update Software

1.3.2 System Functions

Function	**Summary**
Connect to Microscope	The user connects to the microscope's server
View Livestream	The user views the live camera feed
Record Video	The user starts recording the livestream
Capture Image	The user captures the image on the screen
Scan	The system generates a scan of the area bounded by the coordinates provided by the user
Navigate Stage	The user adjusts the position of the stage
Open Gallery	The user displays the saved images and recorded videos
Modify Media	The user modifies the saved media
Log in as Admin	The admin connects to server with his/her login key
Manage System Settings	The admin manages and updates the system settings
Manage Connections	The admin views the list of connected and blocked users
Block User	The admin blocks a user with given IP address
Unblock User	The admin unblocks a user with given IP address
Log in as Maintainer	The maintainer connects to server with his/her login key
View System Logs	The maintainer can list and view the system logs sorted by their dates
Check Updates	The maintainer checks for the software updates
Update Software	The maintainer updates the system software

Table 2: System Functions

1.3.3 User Characteristics

The users of the OpenFlexure can be classified under three categories: users (end-users), system admins, and maintainers.

Users (End-users): are the people who connect to microscopes for regular purposes such as viewing livestream, capturing images etc. They are not authorized, and their permissions are limited. They do not need to have technical skills to use the system. In educational applications, they can be students.

Admins: are authorized people who have permission to manage connected users and manage system-wide settings. They connect to the microscope server with their private keys and use "Admin Command-line Interface" to manage the system. They are managers rather than technical staff; therefore, system configurations and operations that require technical skills are in the scope of maintainers rather than admins. Although they do not need to have advanced technical skills, they need to know how to use ssh connection and command-line interface. In educational applications, they can be lab assistants or teachers.

Maintainers: are authorized people who have permission to view system logs such as crash reports and suspicious login attempts. They are also responsible for maintaining system hardware and updating the software used. They connect to the microscope server with their private keys and use "Maintainer Command-line Interface" to handle technical operations. Maintainers are expected to have good technical skills and knowledge. In educational applications, they can be IT staff of the universities/high schools.

1.3.4 Limitations

- **Regulatory Policies:** The system holds user IP addresses and media captured by them. Ownerships of these media belong to the users saved them. Additionally, maintainer and admin keys are also stored in the system. These data shouldn't be leaked and should be preserved encrypted.

- **Hardware Limitations:** The servers that process the livestream camera feed should be powerful enough to handle that. For users, internet connected computer is enough to use the system.

- **Interfaces to other applications:** The system should be compatible with the OpenFlexure Development Server to install software updates. Additionally, system should be compatible with camera and motor module drivers to communicate with them.

- **Parallel Operation:** The system should allow multiple users to use the system at the same time. Furthermore, users should be able to perform additional operations while performing long-running operations such as tile scan. Hence, it should allow multiple threads running at the same time.

- **Audit Functions:** System should generate logs when any failures or suspicious login attempts happen. It is maintainers' responsibility to control these logs and fix the system.
- **Control Functions:** Control functions should be reachable for only maintainers and admins, and it requires necessary key verifications. Additionally, permissions of maintainers and admins are separate and maintainers cannot access admin control functions.
- **Higher-order Language Requirements:** Bash will be used for Command-line Interface, Python will be use for server-side applications since Flask is used as a web application framework.
- **Signal Handshake Protocols:** System should use HTTPS protocol to retrieve the requests from users and respond to them.
- **Quality Requirements:** Since system stores captured images and recorded videos, it is important to take backups regularly.
- **Criticality of the application:** Failures may cause research and education practices to be interrupted. Therefore, any system failure shall be solved quickly.
- **Safety and security considerations:** User database records do not contain personal information other than IP addresses. Maintainer and admin keys shall be stored in a protected database and suspicious login attempts shall be investigated by maintainers.
- **Physical/Mental considerations:** Visually and mentally impaired people may find it difficult to use the system.

1.4 Definitions

API	An application programming interface (API) is an interface that defines interactions between multiple software applications
Command-line Interface	A command-line interface (CLI) processes commands to a computer program in the form of lines of text
GUI	The graphical user interface is a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicator
Metadata	Higher-level data that describes or annotates another data
OpenFlexure Development Server	An external server where OpenFlexure updates are served to users and update information is returned according to the incoming system version information
Software Repository	A database where the OpenFlexure software update packages are stored.
SSH	The Secure Shell Protocol is a cryptographic network protocol for operating network services securely over an unsecured network

Table 3: Definitions

2 References

[1] https://openflexure.gitlab.io/microscope-handbook/learn-about-the-project/local-manufacturing.html

[2] https://openflexure.gitlab.io/microscope-handbook/learn-about-the-project/teaching.html

3 Specific Requirements

3.1 External Interfaces

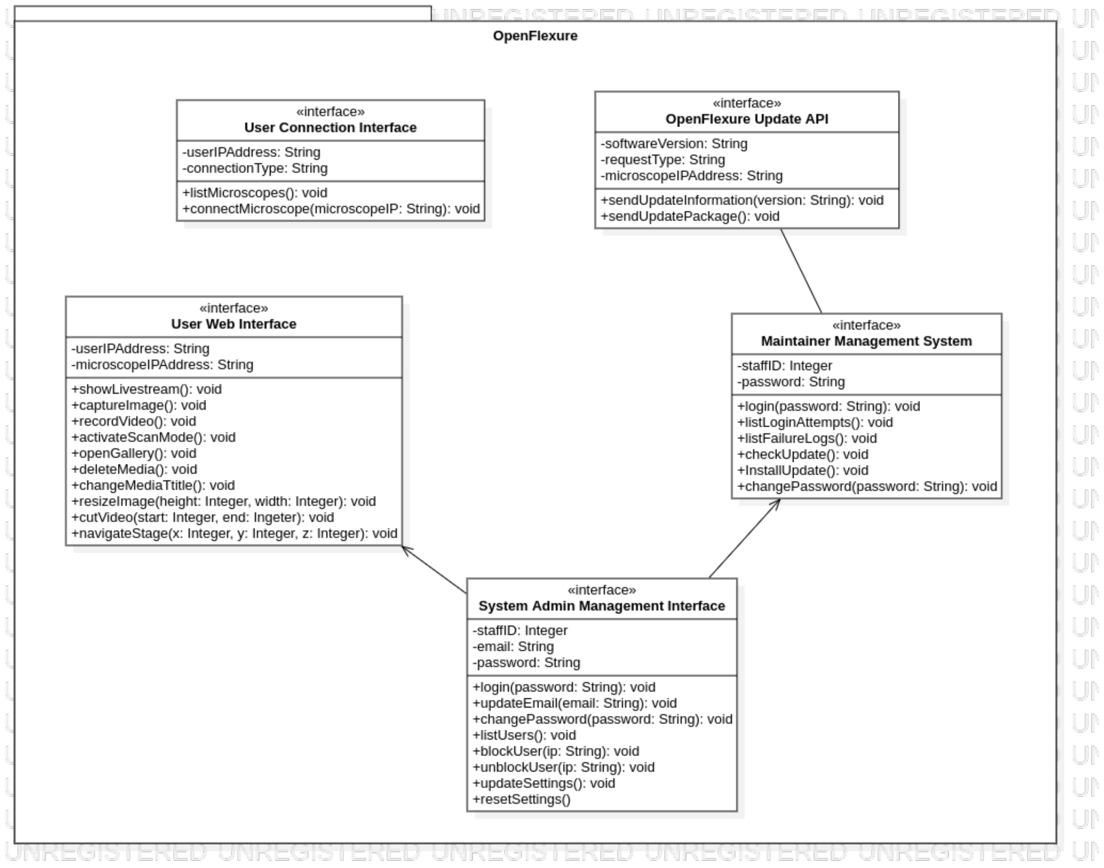

OpenFlexure
«interface»
User Connection Interface
-userIPAddress: String
-connectionType: String
+listMicroscopes(): void
+connectMicroscope(microscopeIP: String): void
«interface»
OpenFlexure Update API
-softwareVersion: String
-requestType: String
-microscopeIPAddress: String
+sendUpdateInformation(version: String): void
+sendUpdatePackage(): void
«interface»
User Web Interface
-userIPAddress: String
-microscopeIPAddress: String
+showLivestream(): void
+captureImage(): void
+recordVideo(): void
+activateScanMode(): void
+openGallery(): void
+deleteMedia(): void
+changeMediaTtitle(): void
+resizeImage(height: Integer, width: Integer): void
+cutVideo(start: Integer, end: Ingeter): void
+navigateStage(x: Integer, y: Integer, z: Integer): void
«interface»
Maintainer Management System
-staffID: Integer
-password: String
+login(password: String): void
+listLoginAttempts(): void
+listFailureLogs(): void
+checkUpdate(): void
+InstallUpdate(): void
+changePassword(password: String): void
«interface»
System Admin Management Interface
-staffID: Integer
-email: String
-password: String
+login(password: String): void
+updateEmail(email: String): void
+changePassword(password: String): void
+listUsers(): void
+blockUser(ip: String): void
+unblockUser(ip: String): void
+updateSettings(): void
+resetSettings()

Figure 4: External Interfaces Class Diagram

3.1.1 User Connection Interface

- If user chooses connect locally option, this interface displays the path of the Web API. If there is no Web API in the user's computer, then it will display error message.

- If user chooses connect remotely, this interface displays the IP addresses and names of the nearby microscopes. There will be option box for each discovered microscope. User will be able to choose the target microscope by clicking on these option boxes.

- If there is no microscopes nearby the user, interface will display an error message.
- Interface will also displays the IP addresses the recently connected microscopes.

3.1.2 User Web Interface

- User will be able to view camera feed by clicking the button with the ”eye” icon from the side pane. If there is no camera connection, error message will be displayed.
- User will be able record the livestream by clicking the ”camera” icon from the side pane. Interface will display the recorded video and user will be able to cut the recorded video by entering the start and end time to the textboxes.
- User will be able record the livestream by clicking the ”camera” icon from the side pane. Interface will display the recorded video and user will be able to cut the recorded video by entering the start and end time to the textboxes.
- User will be able capture image from the livestream by clicking the ”camera lens” icon from the side pane. Interface will display the captured image and user will be able to resize the captured image by entering the height and width to the textboxes. Furthermore, there will be an option box for scan option. Users will be able to activate scan mode by clicking on this option box.
- There will be three textboxes for stage position. User will be able to navigate stage by filling these boxes with coordinates for x, y and z axes.
- User will be able to see media save by him/her in grid style gallery format. He/she will be able to view the metadata information by clicking these gallery entries. Furthermore, user will be able to delete media

by clicking ”delete” button and edit the metadata by clicking ”edit” button and filling the corresponding textboxes.

3.1.3 System Admin Management Interface

- Admin will be able to update his/her mail address using this interface. If the email address is invalid (e.g., doesn't contain @ symbol), an error message will be displayed.
- Admin will be able to change his/her password. Interface will display ”type the new password” text in command-line, then it displays the ”confirm new password” text and compare the inputs for these two messages. If the given inputs are not the same, an error message will be displayed.
- Admin will be able view current system settings. Interface will receive the setting records from the database and display it in the table format. Then, admin will be able to change the values of the settings by giving the defined commands. If command is invalid, an error message will be displayed. If admin gives ”reset settings” command, the system settings in the database will be reset to the default settings.
- Admin will be able view connected and blocked users in a detailed list format. By giving the defined commands, he/she will be able to block/unblock users.

3.1.4 Maintainer Management Interface

- Maintainer will be able to change his/her password. Interface will display ”type the new password” text in command-line, then it displays the ”confirm new password” text and compare the inputs for these two messages. If the given inputs are not the same, an error message will be displayed.
- Maintainer will be able to list failed login attempts in a table where each row contains the IP address of the user, attempt time and attempt type (i.e., admin or maintainer attempt).

- Maintainer will be able to list failure logs in a table where each row contains the the failure date, failure description, failed system component and severity level of the fail. If the maintainer marks the log as solved, then the log will not be displayed next time.

- Maintainer will be able to check whether there is software update or not. OpenFlexure Update API and Maintainer Management Interface will communicate for this task. Maintainer will be able to view the detailed information of the latest software update in command-line interface. Furthermore, he/she will be able to update the system after the checking operation.

3.1.5 OpenFlexure Update API

- When maintainer gives check updates command, Update API shall get the software version and send this information to the OpenFlexure Development Server.

- Update API shall get the respond that contains latest update information from the OpenFlexure Development Server.

- When maintainer gives update command, Update API shall get the installable update package from the OpenFlexure Development Server.

3.2 Functions

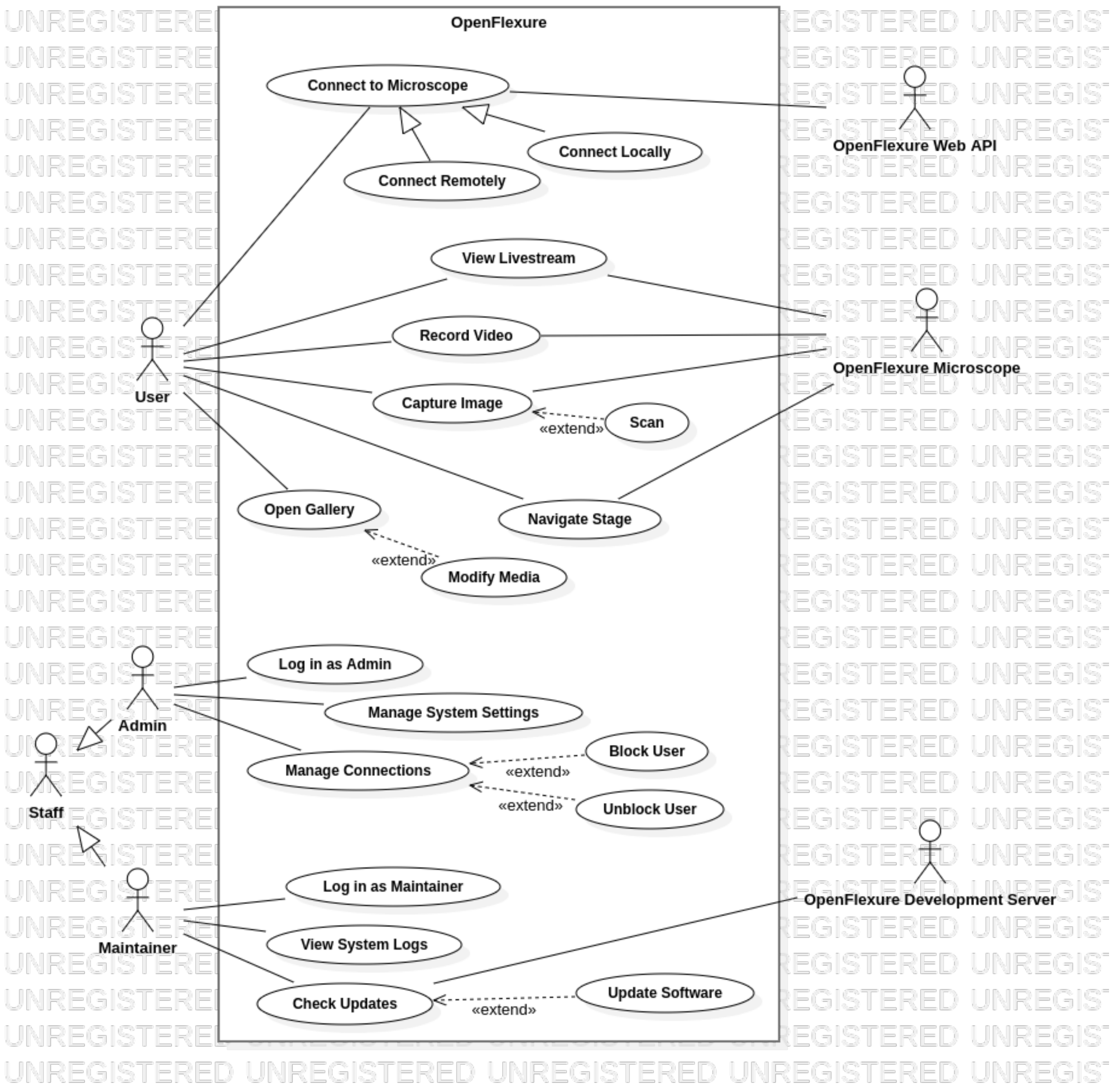


Figure 5: Use Case Diagram for OpenFlexure

Use-Case Name	Connect to Microscope
Actors	User
Description	The user connects to the microscope's server.
Data	IP address of the user's computer
Preconditions	Desktop application must be running.
Stimulus	The user clicks on "Connect to server" button placed in the home screen of the desktop application.
Basic Flow	Step 1 – The popup window contains the connection screen is opened. Step 2 – The user chooses "Connect remotely" option from the option boxes. Step 3 – Desktop application searches for the nearby microscope servers and lists the found ones. Step 4 – The user selects one of the microscopes listed in. Step 5 – The user clicks on "Connect" button.
Alternative Flow#1	Step 2 – The user chooses "Connect locally" option. Step 3 – The desktop application searches for the web api internally stored in the Raspberry Pi, and when it finds, "Connect" button becomes available. Step 4 – The user clicks on "Connect" button.
Alternative Flow#2	-
Exception Flow	If desktop application cannot found any nearby servers or the web api, the system displays an error message saying that "No device were found". If the IP address of the user is blocked, the system displays an error message saying that "Your IP address is blocked".
Post Conditions	The user connects to the microscope.

Table 4: Connect to Microscope

Use-Case Name	View Livestream
Actors	User, OpenFlexure Microscope
Description	The user views the live camera feed.
Data	Camera feed
Preconditions	User must be connected to the system. Camera connection must be provided.
Stimulus	The user clicks on ”View” button.
Basic Flow	Step 1 – The user clicks on ”View” button from the side pine. Step 2 – Server reaches Raspberry Pi camera. Step 3 – Camera livestream is displayed in graphical user interface.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If the camera connection is not established, the system displays an error message saying that ”Check the camera connection”.
Post Conditions	The livestream is displayed.

Table 5: View Livestream

Use-Case Name	Record Video
Actors	User, OpenFlexure Microscope
Description	The user starts recording the livestream.
Data	Recorded video
Preconditions	The user must be connected to the system. Camera connection must be provided.
Stimulus	The user clicks on ”Record” button.
Basic Flow	Step 1 – The user clicks on ”Record” button from the side pane. Step 2 – The tab that contains live camera feed and menu for recording settings opens. Step 3 – When user wants to start recording the livestream, he/she clicks on ”Start Recording” button placed in the menu. Step 4 – The system sends ”record” command to the ”Camera System”. Step 5 – The system displays a blinking led to indicate the recording. Step 6 – The user clicks on ”Stop Recording” button to stop the recording. Step 7 – The system receives the recording from the ”Camera System” and displays it in a popup window. Step 8 – The user gives a name to the recording by filling the textbox placed in the popup window and specifies the time interval if he/she wants to cut the recording before saving it. Step 9 – The user clicks on ”Save” button to save the recording. Step 10 – The system saves the video to the location specified in system settings and create database record for the video using the Database Management Interface.
Alternative Flow#1	Step 8 – The user clicks on the ”Exit” button placed in the popup window to exit without saving the recording.
Alternative Flow#2	-
Exception Flow	If the user tries to save the recording without naming it, the system displays an error message. If there is not enough available storage to store the recording, the system displays an error message.
Post Conditions	The recording is saved to the location specified in the settings and the popup window is closed.

Table 6: Record Video

Use-Case Name	Capture Image
Actors	User, OpenFlexure Microscope
Description	The user captures the image on the screen.
Data	Captured image
Preconditions	The user must be connected to the system. Camera connection must be provided.
Stimulus	The user clicks on ”Capture” button.
Basic Flow	Step 1 – The user clicks on ”Capture” button from the side pane. Step 2 – The tab that contains live camera feed and menu for image settings opens. Step 3 – When user wants to capture the image on the screen, he/she clicks on ”Capture image” button placed in the menu. Step 4 – The system sends ”capture” command to the ”Camera System”. Step 5 – The system receives the captured image from the ”Camera System” and displays it in a popup window. Step 6 – The user gives a name to the captured image by filling the textbox placed in the popup window. Step 7 – The user clicks on ”Save” button to save the captured image. Step 8 – The system saves the image to the location specified in system settings and create database record for the image using the Database Management Interface.
Alternative Flow#1	Step 6 – The user clicks on the ”Exit” button to exit without saving the captured image.
Alternative Flow#2	-
Exception Flow	If the user tries to save the image without naming it, the system displays an error message. If there is not available storage to store the image, the system displays an error message.
Post Conditions	The image is saved to the location specified in the system settings and the popup window is closed.

Table 7: Capture Image

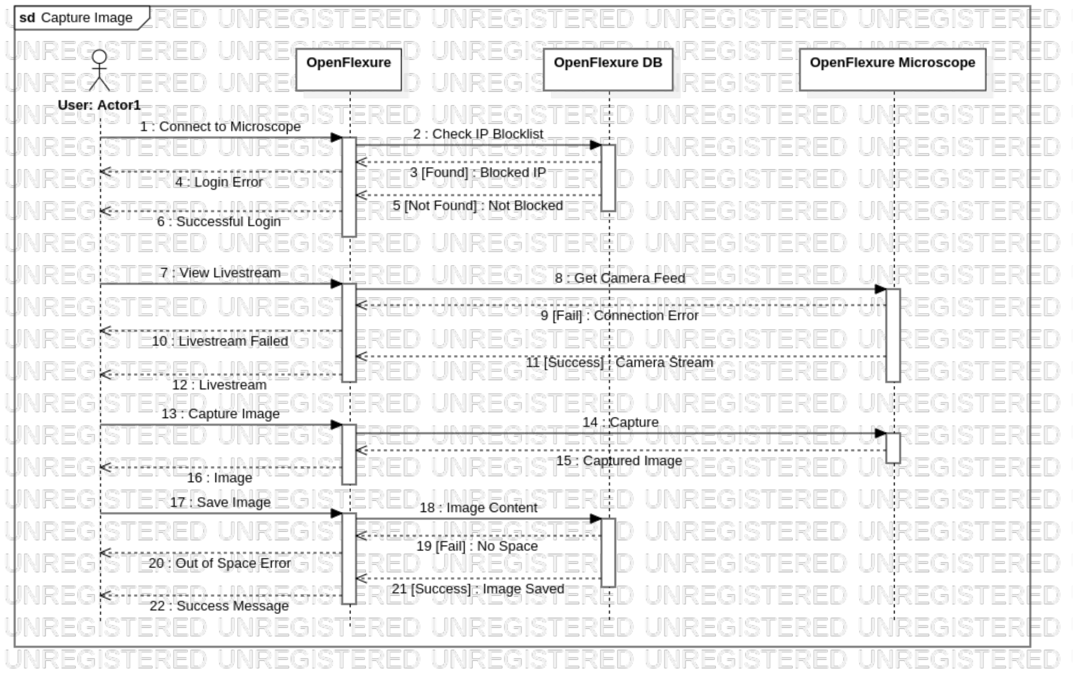


Figure 6: Sequence Diagram of Capture Image Function

Use-Case Name	Scan
Actors	User, OpenFlexure Microscope
Description	The system generates a scan of the area bounded by the coordinates provided by the user.
Data	Generated scan
Preconditions	The user must be connected to the system. Camera connection must be provided. Capture tab must be opened.
Stimulus	The user clicks on "Scan" button.
Basic Flow	Step 1 – The user opens "Capture" tab from the side pane. Step 2 – The user activates "Scan" option by clicking the option box placed in image settings menu. Step 3 – The textboxes showing the coordinates that define the boundary of the area to be scanned becomes available. Step 4 – The user types the coordinates to the textboxes. Step 5 – Until the entire area has been scanned, at each step, the system sends a command to the "Motor Control System" to move by the step size specified in system settings and to the "Camera System" to capture the current image. Step 6 – The system merges all captured images to generate a bigger image of the area and displays it in a popup window. Step 7 – The user gives a name to the generated scan. Step 8 – The user clicks on "Save" button to save the generated scan. Step 9 – The system saves the scan image to the location specified in system settings and create database record for the image using the Database Management Interface.
Alternative Flow#1	Step 7 – The user clicks on the "Exit" button to exit without saving the scan.
Alternative Flow#2	-
Exception Flow	If the user tries to save the scan without naming it, the system displays an error message. If there is no available storage, the system displays an error message.
Post Conditions	The scan is saved to the location specified in the system settings.

Table 8: Scan

Use-Case Name	Navigate Stage
Actors	User, OpenFlexure Microscope
Description	The user adjusts the position of the stage.
Data	Coordinates of the stage in x, y and z axes
Preconditions	User must be connected to the system.
Stimulus	The user clicks on ”Navigate” button.
Basic Flow	Step 1 – The user clicks on ”Navigate” button from the side pane. Step 2 – Tab that contains live camera feed and navigation menu opens. Step 3 – The user chooses the new x, y and z coordinates by filling the three textboxes placed in navigation menu. Step 4 – The user clicks on ”Apply” button. Step 5 – The system sends the given coordinates to the ”Motor Controller”
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If the motor connection is not established, the system displays an error message saying that ”Check the motor connection”.
Post Conditions	The position of the stage is physically changed.

Table 9: Navigate Stage

Use-Case Name	Open Gallery
Actors	User
Description	The user displays the saved images and recorded videos.
Data	Images, recordings and metadata
Preconditions	User must be connected to the system.
Stimulus	The user clicks on "Gallery" button
Basic Flow	Step 1 – The user clicks on "Gallery" button from the side pane. Step 2 – The server reaches to the database and retrieves the media records with the user's IP address in the authorIPAddress field. Step 3 – The system displays a list of images and recordings in a new tab as a grid style gallery where each entry has thumbnail and metadata of the corresponding media.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If there is no saved media associated with the user, the system displays a message saying that "No saved media were found."
Post Conditions	The user views the media saved by him/her.

Table 10: Open Gallery

Use-Case Name	Modify Media
Actors	User
Description	The user modifies the saved media.
Data	Metadata of the saved images and recordings
Preconditions	User must be connected to the system. Gallery tab must be active
Stimulus	The user clicks on one of the media entries in the gallery.
Basic Flow	Step 1 – The user clicks the media entry he/she wants to modify in the gallery. Step 2 – The popup window contains metadata information of the clicked media opens. Step 3 – The user changes the metadata of the media. Step 4 – The user clicks on "Save" button to save the updated metadata. Step 5 – The system updates the corresponding database records using the Database Management Interface.
Alternative Flow#1	Step 3 – The user clicks on "Delete" button. Step 4 – The media is deleted from the storage location and corresponding database records is deleted from the database.
Alternative Flow#2	-
Exception Flow	-
Post Conditions	The media is either deleted or its metadata is updated.

Table 11: Modify Media

Use-Case Name	Log in as Admin
Actors	Admin
Description	The admin connects to server with his/her login key.
Data	Admin login key
Preconditions	-
Stimulus	The admin sends login request to server using the terminal.
Basic Flow	Step 1 – The server receives the login request. Step 2 – The server asks for the admin login key. Step 3 – Maintainer types his/her login key to the terminal. Step 4 – The system reaches to the local database and verifies the given login key. Step 5 – The system signs in the admin.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If the given key is invalid, the system displays an error message saying that ”Given key is invalid”. The system saves the report of the failed login attempt and the IP address of the user to the system logs. Then, the flow starts again from Step 2.
Post Conditions	The admin command line interface opens.

Table 12: Log in as Admin

Use-Case Name	Manage System Settings
Actors	Admin
Description	The admin manages and updates the system settings.
Data	Current settings, updated setting
Preconditions	The admin must be logged in.
Stimulus	The admin gives ”manage settings” command using the admin command line interface.
Basic Flow	Step 1 – The system reaches to the local database and retrieve the system settings record. Step 2 – The system displays the current settings as a table where each row consists of the setting name and current value of the setting. Step 3 – The admin gives ”set {setting name} {new value}” command to change the value of the specified setting. Step 4 – The system reaches to database and changes the value of the given settings field.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	-
Post Conditions	The system settings are updated.

Table 13: Manage System Settings

Use-Case Name	Manage Connections
Actors	Admin
Description	The admin views the list of connected and blocked users.
Data	IP addresses, types (user, maintainer or admin) and the statuses (connected, blocked) of the users.
Preconditions	The admin must be logged in.
Stimulus	The admin gives ”manage connections” command using the admin command-line interface.
Basic Flow	Step 1 – The system reaches to the database and gathers records of the connected and blocked users. Step 2 – The system displays the list of IP addresses, connection times and the statuses (connected, blocked) of the users.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If there is no connected or blocked user, the system displays an error message saying that ”No data were found”.
Post Conditions	The admin views the connected and blocked users.

Table 14: Manage Connections

Use-Case Name	Block User
Actors	Admin
Description	The admin blocks a user with given IP address.
Data	IP address of the user
Preconditions	The admin must be logged in. The admin must have given the ”manage connections” command beforehand.
Stimulus	The admin types ”block user” command in the admin command-line interface.
Basic Flow	Step 1 – The system asks for the IP address of the user to block. Step 2 – The admin gives the IP address of the user obtained from the list displayed after ”manage connections” command. Step 3 – Server reaches to the database and sets the blockStatus field of the user record with the given IP Address as ”blocked”
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If there is no connected user with the given IP address, the system displays an error message saying that ”Invalid Ip address”. If the user with the given IP address has already been blocked, the system displays an error message saying that ”The given IP address has already been blocked”.
Post Conditions	The user with the given IP address will not be able to connect to the server until it is unblocked.

Table 15: Block User

Use-Case Name	Unblock User
Actors	Admin
Description	The admin unblocks a user with given IP address.
Data	IP address of the user
Preconditions	The admin must be logged in. The admin must have given the ”manage connections” command beforehand.
Stimulus	The admin types ”unblock user” command in the admin command-line interface.
Basic Flow	Step 1 – The system asks for the IP address of the user to unblock. Step 2 – The admin gives the IP address of the user obtained from the list displayed after ”manage connections” command. Step 3 – Server reaches to the database and sets the blockStatus field of the user record with the given IP Address as ”unblocked”
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If there is no blocked user with the given IP address, the system displays an error message saying that ”No blocked users were found”.
Post Conditions	The user with the given IP address will be able to connect to the server again.

Table 16: Unblock User

Use-Case Name	Log in as Maintainer
Actors	Maintainer
Description	The maintainer connects to server with his/her login key.
Data	Maintainer login key
Preconditions	-
Stimulus	The maintainer sends login request to server from the terminal.
Basic Flow	Step 1 – The server receives the login request. Step 2 – The server asks for the maintainer login key. Step 3 – Maintainer types his/her login key to the terminal. Step 4 – The system reaches to the local database and verifies the given login key. Step 5 – The system signs in the maintainer.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If the given key is invalid, the system displays an error message saying that ”Given key is invalid”. The system saves the log of the failed login attempt and the IP address of the user to the database. Then, the flow starts again from Step 2.
Post Conditions	The maintainer command-line interface opens.

Table 17: Log in as Maintainer

Use-Case Name	View system logs
Actors	Maintainer
Description	The maintainer can list and view the system logs sorted by their dates.
Data	System logs
Preconditions	Maintainer must be logged in.
Stimulus	The maintainer gives ”view logs” command using the command-line interface.
Basic Flow	Step 1 – The server takes the request. Step 2 – The server gathers the system logs from the database. Step 3 – The system displays the system logs in command-line interface.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If there is no system logs in database, the system displays a message saying that ”No system logs were found”.
Post Conditions	The system logs are displayed in the maintainer’s interface.

Table 18: View System Logs

Use-Case Name	Check Updates
Actors	Maintainer, OpenFlexure Development Server
Description	The maintainer checks for the software updates.
Data	System's current software version, released update information
Preconditions	Maintainer must be logged in.
Stimulus	The maintainer gives "check updates" command using the command line interface.
Basic Flow	Step 1 – The server retrieves the version number of the software from the database. Step 2 – The server sends the current version number of the software to the OpenFlexure Development Server. Step 3 – The server receives the information about the latest update returned by the OpenFlexure Development Server. Step 4 – The server displays the version number, size, release date and changelog of the update in command-line interface.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If the system is already using the latest version, an error message saying that "No updates were found" is displayed.
Post Conditions	Information about the latest update is displayed in the command-line interface.

Table 19: Check Updates

Use-Case Name	Update Software
Actors	Maintainer, OpenFlexure Development Server
Description	The maintainer updates the system software.
Data	Installable update package
Preconditions	Maintainer must be logged in. Maintainer must have given the ”check updates” command beforehand.
Stimulus	The maintainer gives ”update” command using the command line interface.
Basic Flow	Step 1 – The server requests the update package from the ”OpenFlexure Development Server”. Step 2 – The package is downloaded to the Raspberry Pi. Step 3 – The server installs the downloaded package. Step 4 – The server saves the version number and the release date of the update to the corresponding database table.
Alternative Flow#1	-
Alternative Flow#2	-
Exception Flow	If an error is encountered during the installation, for example, if the downloaded package is corrupted, the system saves the detailed error log into the database.
Post Conditions	The system is updated to the latest version.

Table 20: Update Software

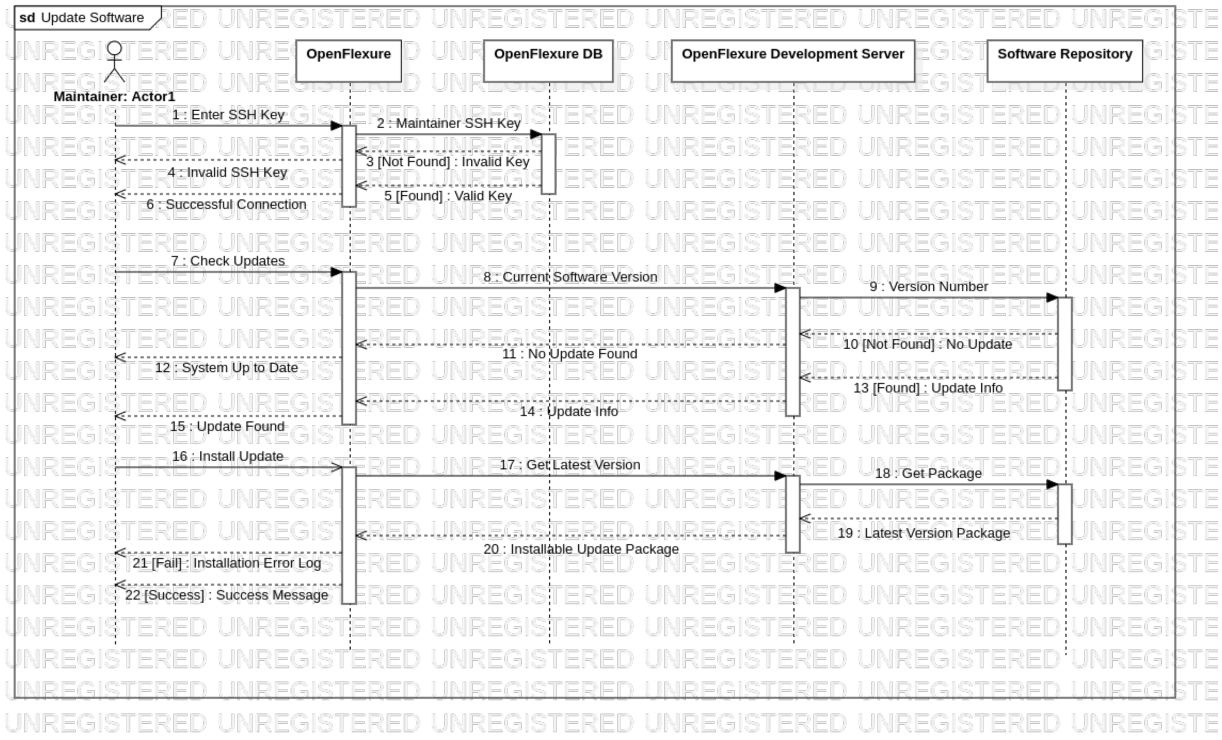


Figure 7: Sequence Diagram of Update Software Function

3.3 Usability Requirements

req1: The users shall be able to connect the microscope server both remotely and locally.
req2: The user web interface design shall be simple and user shall be able to find the function he/she wants at most 5 clicks.
req3: There should be an help button in user web interface and users should be able to understand the point she does not understand within 2 minutes by reading the information written there.
req4: The user will be able to choose dark mode design for the web interface.
req5: The user will be able to navigate camera by using the arrow keys in his/her keyboard in addition to the textboxes.

3.4 Performance Requirements

req6: The latency time of the livestream shall not exceed 40 miliseconds.
req7: The time between users' navigation orders and the motors's reaction time shall not exceed 30 milliseconds.
req8: The microscope servers shall be able to handle up to 30 connections.
req9: The system shall support HD video and image capture.
req10: The response time of the database for any kind of operations shall not exceed 2 seconds.

3.5 Logical Database Requirements

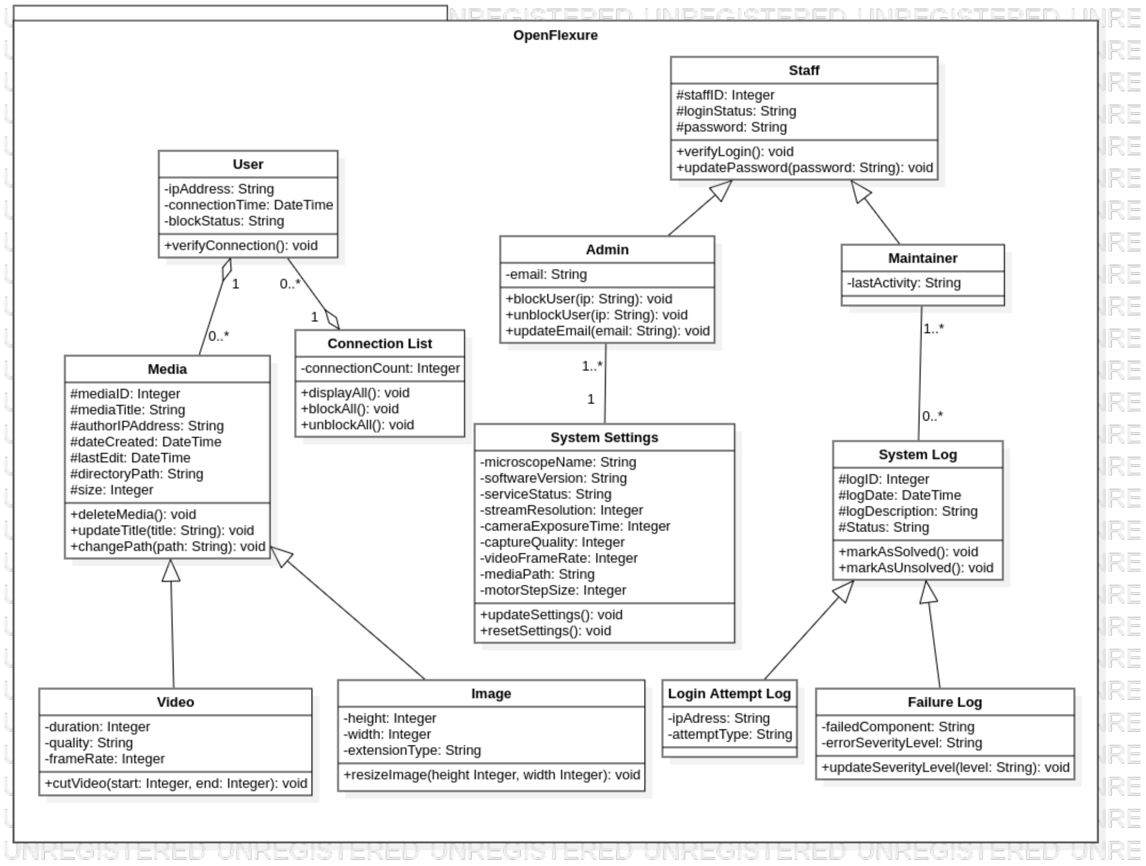


Figure 8: Logical Database Requirements Class Diagram

- When a user connects to the microscope, a record that contains user's IP address, the time he/she connected to system and block status will be created in database. Initially, block status will be None. However, system admin can change the value of this attribute later by blocking/unblocking the user.
- Users cannot change any of these attributes.
- User records should be seen as connection records rather than profiles.

Users do not require passwords or personal informations to connect the microscope. They are identified by their ip addresses.

- Connection list contains the list of users and number of connections.
- When user record is created, it is automatically added to the connection list.
- Connection list can be empty if there is no connection. It is not destroyed if there is no connection.
- If user disconnects, his/her record is automatically deleted from the database and connection list.
- When a user saves media, media records that contains metadata of the media will be created in the database. And authorIPAddress attribute of the media record will be initialized as the IP address of the user who saved the media.
- User may not have any media record associated with him/her. His/her record is not destroyed if there is no media record associated with him/her.
- When user record is deleted, the record of the media he/she saved is not destroyed. Furthermore, authorIPAddress attribute continues to keep IP address without pointing to user record.
- When user record is deleted, the record of the media he/she saved is not destroyed. Furthermore, authorIPAddress attribute continues to keep IP address without pointing to user record.
- Media class is generalization for Video and Image classes.
- mediaID is automatically created by the system and it is unique.
- User can only view the media created by him/her. Hence, he/she can only edit/delete the media associated with him/her.
- User cannot change the directory path. However, admins can change the path from the system settings.

- User can resize the image files by giving height and width parameter and cut the video files by giving starting time and ending time parameters. Again these files have to be associated by the user.
- Staff class is generalization for Admin and Maintainer classes.
- Each staff has unique staffID and passwords for signing in the system. Additionally, staff records contain loginStatus that indicates whether the staff is currently logged in or not.
- Staff can change their passwords. They cannot change their loginStatus and staffID.
- Admins have email addresses in addition to the attributes defined in Staff class. And they can change these email addresses.
- System settings record contains attributes for system wide settings. There is only one system settings record and it is created automatically when the system is established and cannot be deleted from the database.
- Only admins can update and reset system settings.
- Maintainer records have lastActivity field in addition to Staff attributes. It shows the last operation done by the maintainer. (e.g., viewed system log, checked updates etc.)
- System Log class is generalization for Login Attempt Log and Failure Log classes.
- System Log records contain unique logID, logDate, logDescription and status attributes. Status indicates whether the problem defined in the log is solved or not. Status field is initialized as unsolved.
- Maintainers can view system logs and update their status by marking them as solved or unsolved.
- Login Attempt Log contains IP address of the user attempted to login and attemptType that indicates whether user attempted to login as admin or maintainer.

- Failure Log contains failedComponent and errorSeverityLevel fields. errorSeverityLevel indicates the importance of the problem and it can be updated by the maintainers.

3.6 Design Constraints

req11: IP addresses, activity histories and device information of the connected users shall only be accessible by system administrators.
req12: The login keys, IP addresses and activity histories of the system administrators and maintainers shall be encrypted with SHA-256 algorithm in database.
req13: The system logs shall only be accessible by the system administrators and maintainers.
req14: All personal data and saved media shall be stored in accordance with the General Data Protection Regulation.

3.7 Software System Attributes

3.7.1 Reliability

req15: The probability of database record corruptions shall not exceed 0.001.
req16: The mean time between failures shall not be less than 2 weeks.

3.7.2 Availability

req17: The hardware maintainance and system updates shall be done when no users are connected to microscopes.
req18: The technical issues shall be solved by the system maintainers at most 30 minutes.

3.7.3 Security

req19: The database records shall be encrypted with SHA-256 algorithm.
req20: The system administrators and maintainers shall log into the system using passwords of at least 12 digits.

req21: Suspicious login attempts shall be recorded in the system together with the IP addresses of people trying to log in.
req22: The system administrators shall have the permission to block the ip addresses of users.

3.7.4 Maintainability

req23: The maintainers shall update the system software using OpenFlexure Update API.
req24: Broken hardware parts shall be replaced within 2 days at the most, and weekly maintenance of microscope parts shall be done.

3.7.5 Portability

req25: The OpenFlexure desktop application shall be runnable on different operating systems such as MacOS, Windows and Linux.
req26: The system shall not be dependent on a single type of camera and motor product, it shall work in harmony when the camera or motor model used is desired to be changed.

3.8 Supporting Information

- One of the application areas of the system is education [2]. Therefore it is reasonable to allow the connections of multiple users.
- Quick connection and practical usage is important for microscopes. Requiring users to register to use the microscope severely damages the practical use principle. Therefore, it is reasonable to see users as connections rather than detailed profiles. As a result, a user can easily and quickly connect to the microscope without any registration procedure.