Software Design Descriptions
OpenFlexure Microscope

Ahmet Burak Baraklı, 2309730
Ozan Akın, 2309599

November 28, 2021

Contents

List of Figures

List of Tables

Revision History

Date	Reason For Changes	Version
28.05.2021	Initial Setup	0.1
11.06.2021	Final Document	1.0

Table 1: Revision History of Software Design Descriptions Document

1 Introduction

The document is a Software Design Descriptions (SDD) of a smart microscope called OpenFlexure, a microscope that is connected to the internet and can be controlled remotely. The OpenFlexure project's website can be found at `https://openflexure.org`.

1.1 Purpose of the System

The purpose of the project is to build a 3D-printed laboratory grade microscope that can be used in environments from Antarctica to Amazon Forests. With the motorized parts and easily printable parts of the microscope, the system can be easily be placed in areas with restricted access to decrease the risks of both patients' and researchers' health. Parts of the microscope can be seen in the figure below. Note that the figure includes both printable and non-printable parts.

Figure 1: Parts of an unassembled OpenFlexure Microscope [2]

For the sake of durability and stability of the inner parts where camera and electronic components, such as Arduino and Raspberry Pi, OFM system has stages that has a wide bottom. Since stages are printed with 3D-printers like most of the parts, the stage part can have various colors. In the following the stage part of several OpenFlexure Microscopes can be seen:

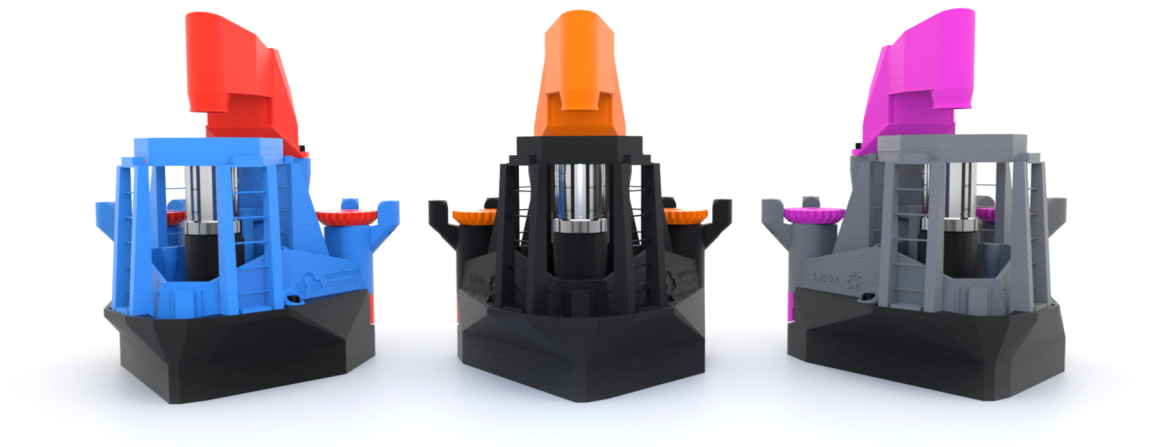

Figure 2: The stage part of OpenFlexure Microscope [3]

1.2 Scope

The project is called OpenFlexure, a 3D printed, remotely controlled microscope. The researchers will be able to watch and record the live stream coming from the microscope. Researchers will be able to change the microscope's location and field of view with motorized parts using two programmable electronic boards, an Arduino to control the motors and a Raspberry Pi to connect to the internet and communicate with the researchers. The camera part of an OpenFlexure Microscope that is connected to motors can be seen in the figure below.

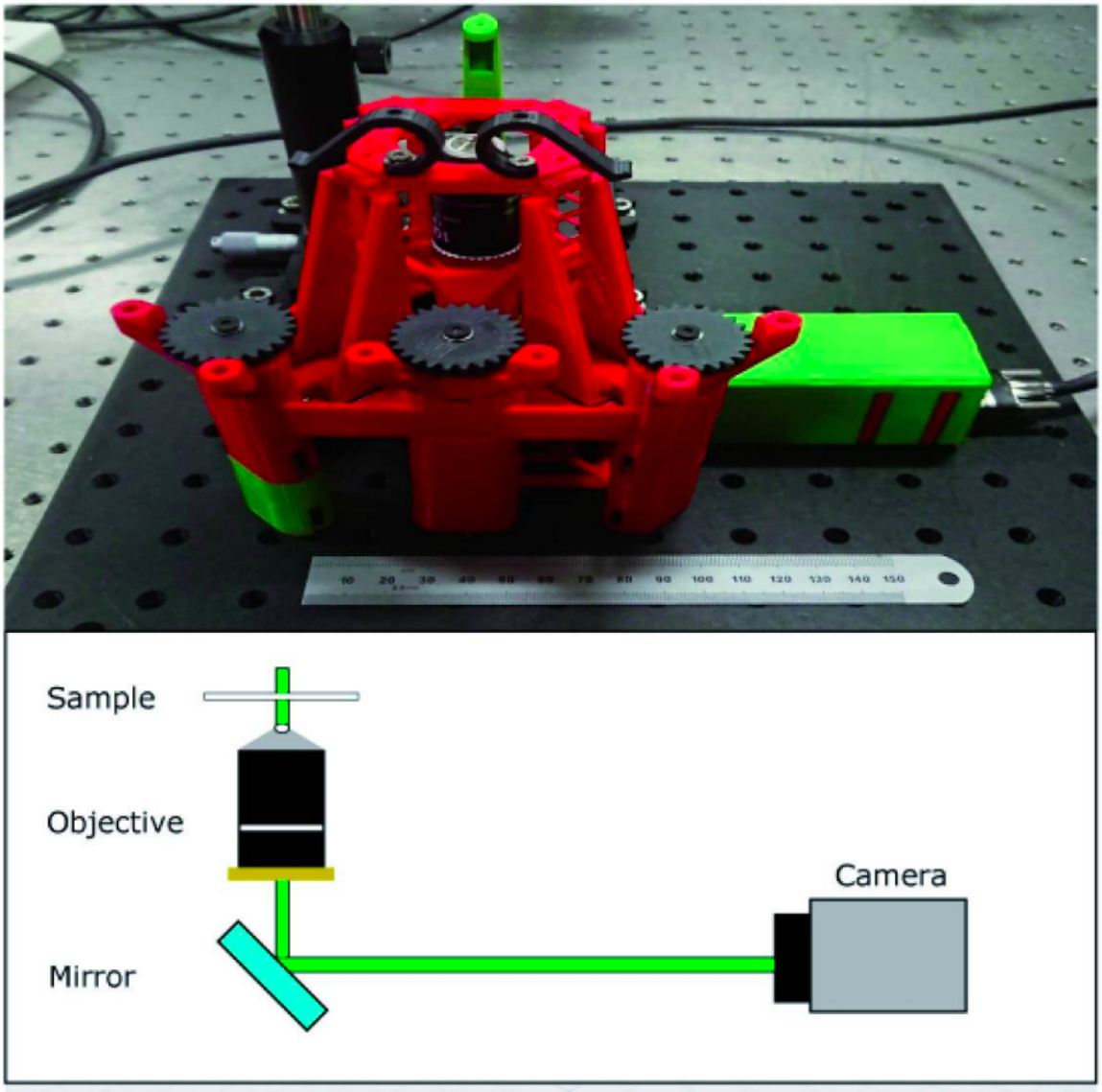

Figure 3: Camera part of an OpenFlexure Microscope [1]

The scope of the project can be listed as follows:

- Providing a remotely controlled microscope environment to help researchers do their research without risking their health.
- Providing a recordable live stream to make it possible to examine the results by more than one person while creating a reproducible environment with the record options of the live stream.
- Providing a cloud-based architecture built-in to the microscope to help researchers combine their results, projects, and visuals.

1.3 Stakeholders and Their Concerns

In OpenFlexure Microscope, the main concern is research since both the developers of the project and users are from the same domain and their purposes are to research, collect and analyze data, and help the environment.

Researchers:
Researchers are end user of this OFM system. They analyze what's in the microscopes, and they use the GUI. Hence, a simple and helpful GUI is required. Moreover, since the data coming from microscopes may be confidential, their another concern is about security of the system. It need to be safe and should not leak any data.

Admins:
Their main concern is the system is available and running without any problem or data loss. Their second concern is security. The system should be safe for all potential researchers.

Developers:
Developers are engineers who are responsible for developing the system. They need to be specialized in several topics: IoT, computer vision and embedded systems. Their main concern is requirements and needs of the system. They require well defined and explanatory requirements.

2 References

This document is prepared with respect to IEEE 1016-1998 standard:

IEEE standard for information technology–systems design–software design descriptions. (2009). New York, NY: Institute of Electrical and Electronics Engineers.

Other sources:

[1] Stephen D Grant et al. "Adapting the 3D-printed Openflexure microscope enables computational super-resolution imaging". In: *F1000Research* 8 (2019).

[2] *https://wikifactory.com/@rwbowman/openflexure-microscope*. `https://wikifactory.com/@rwbowman/openflexure-microscope`. (Accessed on 04/22/2021).

[3] *OpenFlexure Microscope*. `https://openflexure.org/projects/microscope/`. (Accessed on 04/22/2021).

3 Glossary

Researcher	A user who uses the microscope to analysis data.
System admin	The administrators of the system.
Super admin	A system admin that has control over other system admins.
Arduino	Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices.
Raspberry Pi	Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom.
SD card	Secure Digital, officially abbreviated as SD, is a proprietary non-volatile memory card format developed by the SD Association for use in portable devices
GUI	Graphical User Interface
OFM	OpenFlexure Microscope
Electron	Electron is an open-source software framework developed and maintained by GitHub. It allows for the development of desktop GUI applications using web technologies.
Python	Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation.
Flask	Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions
HTTPS	Hypertext Transfer Protocol Secure(HTTPS) is an extension of the Hypertext Transfer Protocol (HTTP). It is used for secure communication over a computer network, and is widely used on the Internet. In HTTPS, the communication protocol is encrypted using Transport Layer Security or, formerly, Secure Sockets Layer.

Table 2: Glossary

4 Architectural Views

4.1 Context View

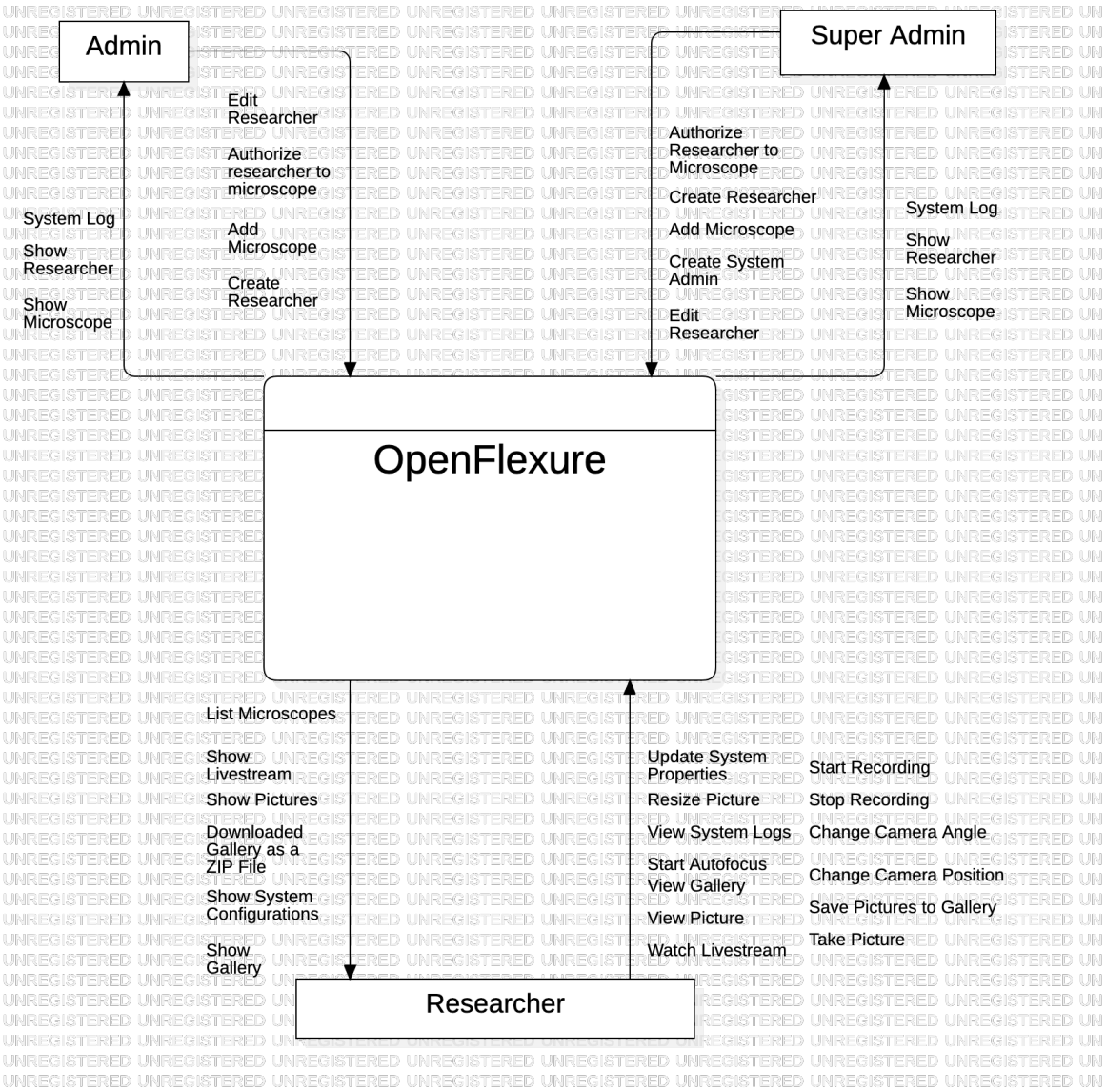

Figure 4: Context Diagram for OpenFlexure Microscope

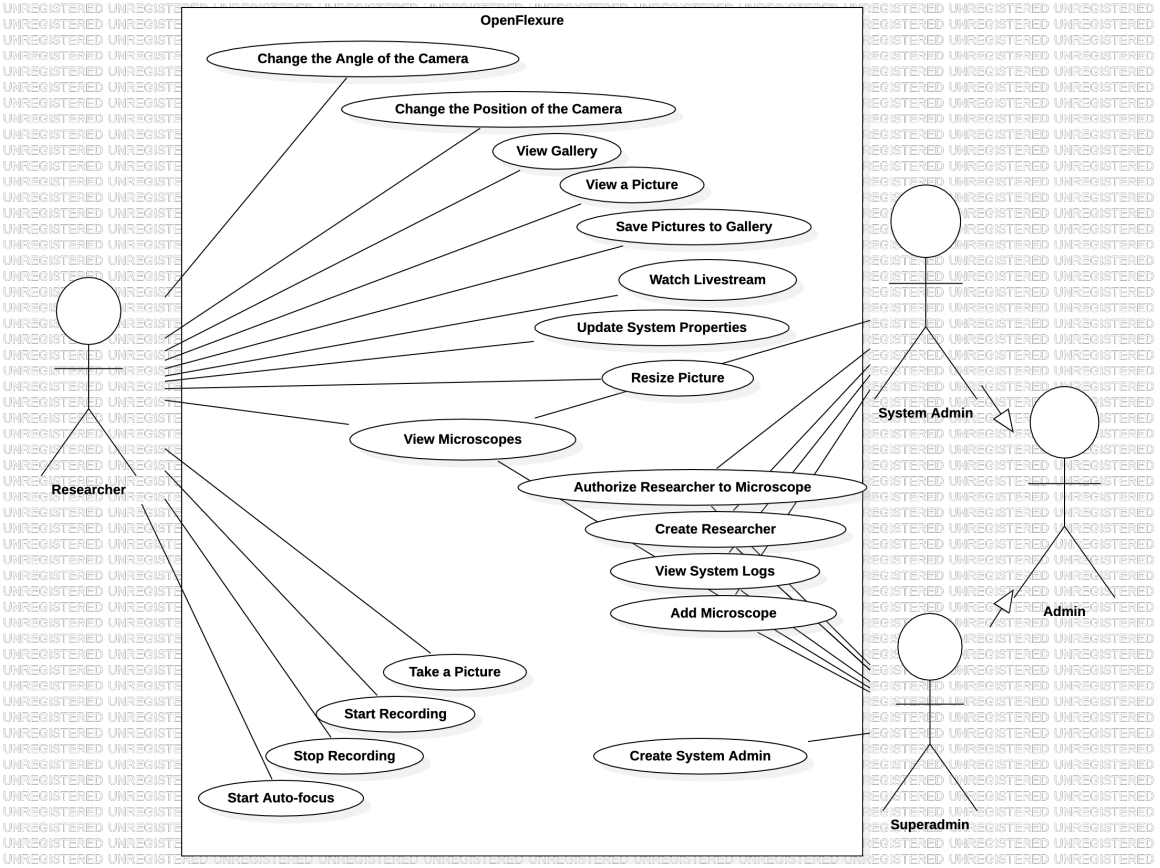

Figure 5: Use-Case Diagram for OpenFlexure Microscope

Use-Case Name	Authorize Researcher to Microscope
Actors	System Admin
Description	When a system admin wants to authorize a researcher to see a microscope, they can add an authorization.
Data	The researcher to be authorized and the microscope will be seen by the researcher.
Preconditions	The user must be authenticated. Also, the user must be an admin.
Stimulus	The System Admin presses the "Authorize Researcher to Microscope" button.
Basic Flow	Step 1 - The system admin presses the "Authorize Researcher to Microscope" button. Step 2 - A popup is shown to admin. Step 3 - The system admin selects the researcher to be authorized and the microscope will be seen by the researcher. Step 4 - The system saves the new authorization.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	-
Post Conditions	The researcher is authorized to see the given microscope.

Table 3: Authorize Researcher to Microscope

Use-Case Name	Add Microscope
Actors	System Admin
Description	The system admin adds a microscope to the system for researchers to view.
Data	Microscope Data
Preconditions	The user must be authenticated. Also, the user must be an admin
Stimulus	The system admin presses "Add Microscope" button.
Basic Flow	Step 1 - The system admin presses add microscope button. Step 2 - The system connects to given microscope. Step 3 - The system admin is informed about success of adding a microscope.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	If given microscope data is invalid or can not be connected, an error is shown instead.
Post Conditions	The newly added microscope is listed in the list of microscopes.

Table 4: Add Microscope

Use-Case Name	Create Researcher
Actors	System Admin
Description	The system admin creates a researcher to the system.
Data	Researcher's Data
Preconditions	The user must be authenticated. Also, The user that creates researcher is system admin
Stimulus	The system admin presses "Add Researcher" button.
Basic Flow	Step 1 - The system admin presses add researcher button. Step 2 - The system creates a new researcher with given data. Step 3 - The system admin is informed about success of creating an user.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The newly created user can use the system.

Table 5: Create Researcher

Use-Case Name	Create System Admin
Actors	Super Admin
Description	The super admin creates a system admin for the system.
Data	System Admin's Data
Preconditions	The system admin must be authenticated user. The system admin must be a super admin.
Stimulus	The user that creates another system admin presses "Add System Admin" button.
Basic Flow	Step 1 - The superadmin pressed the "Add System Admin" button. Step 2 - A popup is shown to the super admin. Step 3 - The user enters the information for the new system admin. Step 4 - The system creates a new system admin with the given information. Step 5 - The super admin is informed with the new system admin's information.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The new system admin can change and configure the system.

Table 6: Create System Admin

Use-Case Name	Change Angle of the Camera
Actors	Researcher
Description	When a researcher wants to change the angle of the camera to see from other perspectives, they can change the angle.
Data	The degree of the angle for the camera, the time that the request will be applied.
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "Change Angle" button.
Basic Flow	Step 1 - A researcher wants to change the angle to see from other perspective. Step 2 - The system receives the request about changing the angle. Step 3 - The system controls motors accordingly to apply the new angle. Step 4 - The user is informed about the change has been applied successfully.
Alternative Flow#1	Step 1 - A researcher wants to change the angle to see from other perspective in a future time. Step 2 - The system receives the request about changing the angle. Step 3 - The system sets a timer to the given timer. Step 4 - When given time is reached, the system controls motors accordingly to apply the new angle. Step 5 - The user is informed about the change has been applied successfully.
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The angle of the camera now changed.

Table 7: Change the Angle of the Camera

Use-Case Name	Change Position of the Camera
Actors	Researcher
Description	When a researcher wants to change the position of the camera to see from other perspectives, they can change the position.
Data	The x, y, z coordinates for the new camera location, the time that the request will be applied.
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "Change Position" button.
Basic Flow	Step 1 - A researcher wants to change the position to see from other perspective. Step 2 - The system receives the request about changing the position. Step 3 - The system controls motors accordingly to apply the new position. Step 4 - The user is informed about the change has been applied successfully.
Alternative Flow#1	Step 1 - A researcher wants to change the position to see from other perspective in a future time. Step 2 - The system receives the request about changing the position. Step 3 - The system sets a timer to the given timer. Step 4 - When given time is reached, the system controls motors accordingly to apply the new position. Step 5 - The user is informed about the change has been applied successfully.
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The position of the camera now changed.

Table 8: Change the Position of the Camera

Use-Case Name	Resize Picture
Actors	Researcher
Description	The researcher can resize any picture.
Data	The picture that will be resized, the width and height of the new picture size.
Preconditions	The researcher must be authenticated and has access to the gallery.
Stimulus	The user presses "Resize" button while viewing a picture.
Basic Flow	Step 1 - The researcher presses the "Resize" button while viewing a picture. Step 2 - A popup is shown to researcher. Step 3 - The researcher enters the width and height of the new picture size. Step 4 - The system resizes the image with the given width and height.
Alternative Flow#1	Step 3 - The researcher selects a pre-defined width and height. Step 4 - The system resized the image with the given width and height.
Alternative Flow#2	–
Exception Flow	-
Post Conditions	The picture is now being resized.

Table 9: Resize Picture

Use-Case Name	Save Pictures to Gallery
Actors	Researcher
Description	A researcher wants to save a picture into the gallery to be able to reach it later.
Data	Filename of the picture to be saved into gallery, gallery which image will be saved.
Preconditions	The user must be authenticated.
Stimulus	When looking at a picture, researcher pressing the "Save Picture to Gallery" button.
Basic Flow	Step 1 - A researcher wants to save a picture into the gallery. Step 2 - The researcher send a request which includes name picture to the system. Step 3 - The system adds the picture into the gallery.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	If there is no picture with that name, then the researcher receives an error.
Post Conditions	The picture must be in the gallery.

Table 10: Save Pictures to Gallery

Use-Case Name	Start Auto-focus.
Actors	Researcher
Description	When a researcher wants to auto-focus camera to make visuals more clear, they can trigger the action.
Data	The auto-focus button action from the web server.
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "Automatic Focus" button.
Basic Flow	Step 1 - A researcher wants to calibrate the camera according to the environment. Step 2 - The researcher presses the "Automatic Focus" button to start automatic focus. Step 3 - The system starts to the auto-focus based on the environment properties. Step 4 - The user is informed about camera has been focused successfully.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The camera is now focused and the visuals are much more clear.

Table 11: Start Auto-focus

Use-Case Name	Start Recording.
Actors	Researcher
Description	When a researcher wants to record a video from live stream and save to SD Card.
Data	Time that the "Start Recording" button is pressed.
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "Record" button.
Basic Flow	Step 1 - A researcher wants to record a Video and save it to the SD Card. Step 2 - The researcher presses the "Record" button to start recording. Step 3 - The system starts to capture the images coming from the camera and saves them into the SD Card. Step 4 - The user is informed about a recording has started.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The system continues to record the content.

Table 12: Start Recording

Use-Case Name	Stop Recording.
Actors	Researcher
Description	When a researcher wants to stop recording the video from the live stream and encode the already saved SD Card Video.
Data	Time that the "Stop Recording" button is pressed.
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "Stop Recording" button.
Basic Flow	Step 1 - A researcher wants to stop recording of a video. Step 2 - The researcher presses the "Stop Recording" button to stop recording. Step 3 - The system receives the record button's action. Step 4 - The system stops the recording. Step 5 - The system starts the encoding of the saved video to convert the record format to something more common such as MP4. Step 6 - The system saves the new encoded video to the SD Card.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The SD Card contains the recorded video.

Table 13: Stop Recording

Use-Case Name	Take a Picture
Actors	Researcher
Description	When a researcher sees an important image at microscope, they can send request to microscope to take picture of it.
Data	Time the button was pressed. Desired size of the Picture.
Preconditions	The user must be authenticated. The microscope and camera must be working.
Stimulus	Researcher pressing the "Take Picture" button.
Basic Flow	Step 1 - A researcher sees an important image at camera. Step 2 - The researcher send request to take picture. Step 3 - The system saves the image when the button was pressed into an image file in SD Card. Step 4 - The user sees the picture.
Alternative Flow#1	Step 4 - The picture file is saved into USB instead.
Alternative Flow#2	Step 3 - The system saves the image when the button was pressed and resize it into desired size an save it into an image file in SD Card.
Alternative Flow#3	–
Exception Flow	–
Post Conditions	The picture which contains the data when researcher pressed the button must be saved.

Table 14: Take a Picture

Use-Case Name	Update a Property of the System
Actors	Researcher
Description	When a researcher wants to update some properties or configuration of the system, they change these values remotely.
Data	The new value for the property.
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "Change Property" button.
Basic Flow	Step 1 - A researcher wants to change a property of the system. Step 2 - The researcher send request to web server to change property. Step 3 - The system updates the property accordingly. Step 4 - The user is informed about the change has been applied successfully.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The property of the system is now changed.

Table 15: Update a Property of the System

Use-Case Name	View Gallery
Actors	Researcher
Description	A researcher wants to view pictures in a gallery.
Data	The gallery which will be viewed.
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "View Gallery" button.
Basic Flow	Step 1 - A researcher wants to see pictures in a gallery. Step 2 - The system returns the gallery. Step 3 - The gallery is sent the researcher.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	If there is no previously saved pictures, then the researcher receives gallery is empty warning.
Post Conditions	The researcher receives the gallery.

Table 16: View Gallery

Use-Case Name	View a Picture
Actors	Researcher
Description	A researcher wants to view previously saved picture.
Data	Filename of the picture
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "View Picture" button.
Basic Flow	Step 1 - A researcher wants to see previously captured picture. Step 2 - The researcher send a request which includes name picture to the system. Step 3 - The system returns the picture. Step 4 - The picture is sent the researcher.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	If there is no picture with that name, then the researcher receives an error.
Post Conditions	The researcher receives the picture.

Table 17: View a Picture

Use-Case Name	Watch Livestream
Actors	Researcher
Description	When a researcher wants to see what is microscope is seeing, they can send request to microscope to stream the camera outputs.
Data	–
Preconditions	The user must be authenticated.
Stimulus	Researcher pressing the "Watch Livestream" button.
Basic Flow	Step 1 - A researcher wants to see what microscope is seeing. Step 2 - The researcher send request to watch livestream. Step 3 - The system sends the camera output with web server until it receives a stop request.
Alternative Flow#1	–
Alternative Flow#2	–
Exception Flow	–
Post Conditions	The researcher receives the livestream of camera output until they send a stop request.

Table 18: Watch Livestream

Use-Case Name	View System Logs
Actors	System Admin
Description	The authorized system admin views the logs of the system.
Data	System Logs
Preconditions	The user must be an authenticated user. Also, the user is an system admin or super admin.
Stimulus	The user presses the "View Logs" button from the left panel.
Basic Flow	Step 1 - The user presses the "View Logs" button. Step 2 - The system logs retrieved from the database. Step 3 - The user sees the logs in the chronological order.
Alternative Flow#1	Step 4 - The user downloads logs from the user interface.
Alternative Flow#2	Step 3 - The user filters the logs with their log type. Step 4 - The user sees the logs in the chronological order.
Exception Flow	Step 2 - The system logs cannot be retrieved from the database. Step 3 - A popup is shown in the user interface with the title "The logs cannot be retrieved from the database".
Post Conditions	The authorized admin has the logs of the system.

Table 19: View System Logs

4.2 Composition View

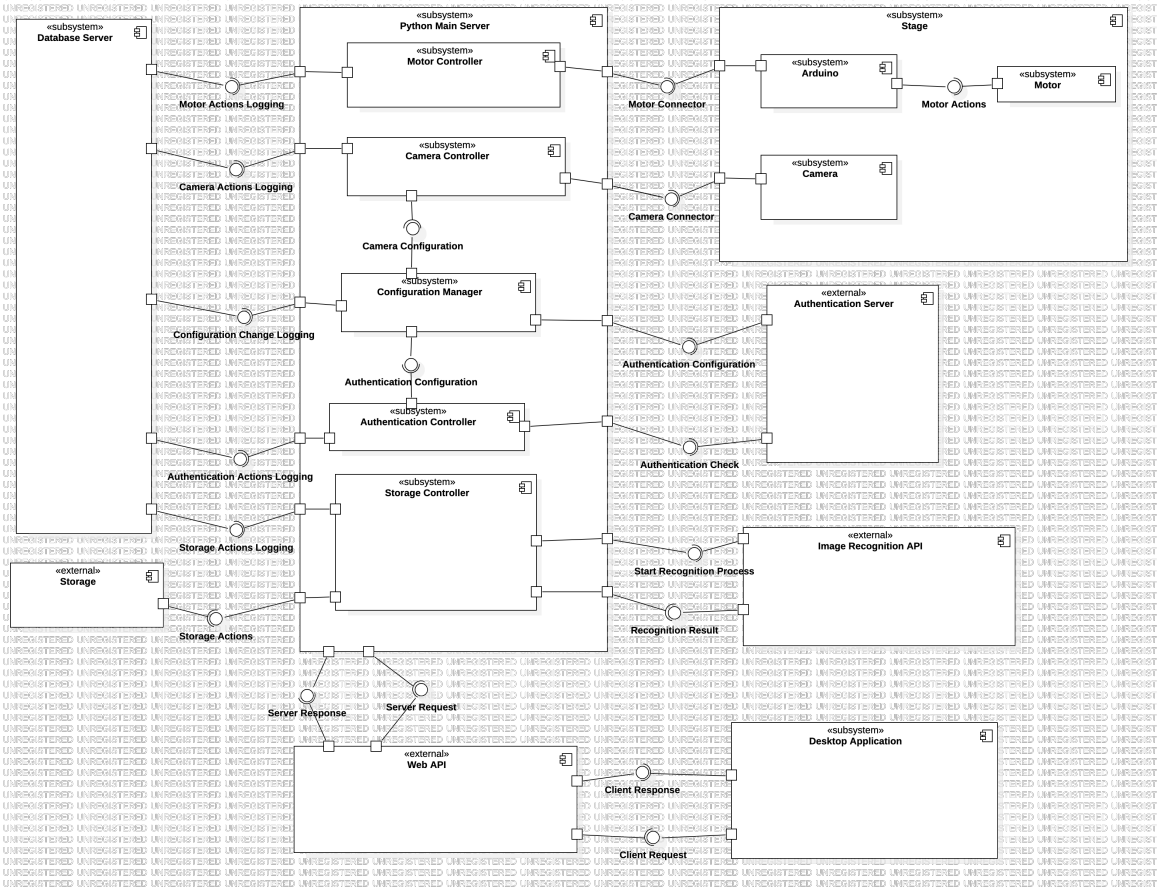

Figure 6: Component Diagram

Design Rationale

- Python main server controls the main activities of the system. It interacts with the outside world. The main server connects storage, stage, authentication server and image recognition API, and it exposes the web API to clients. The main server is also responsible for health checks of the internal services, applications and databases.

- Python main server keeps the state data of the internal controllers. The server is also responsible for relaying some data from outside world to internal components. The server also waits requests in a queue if the internal services do not have any active status.

- Database server is the main database server that keeps the metada information related with the pictures, recordings, microscopes and cameras in the system. The database takes backups in every day.

- Configuration manager is the internal component that reads configurations from database and other components, and provides configurations to other components. The configuration manager also contacts with the authentication server to make sure that the server has been configured correctly.

- Authentication controller is the part that connects an authentication server to the main system. The controller interacts with the configuration manager to configure itself. It provides SSO, OAuth2 or similar protocol to our system.

- Motor controller interacts with the stage, and the Arduino inside the stage to move and change the speed of the motors in the stage. The Raspberry Pi Python main server and Arduino interacts with a serial communication protocol such as UART or I2C and sends the necessary information between each other.

- Camera controller controls the camera in the stage and its configuration. It also reads the configuration at setup and configures properties such as color balance, resolution and white balance. Camera control interacts with the configuration manager to read the initial configuration data from the system.

- Storage controller interacts with the SD card and provides the necessary interfaces to other components to make the storage accessible. Storage controller is also responsible for sending the updated images to image recognition API.

- The image recognition API is an external, subscription-based API that detects the type of micro-organism or environment in the pictures taken from the microscope. It is a machine learning API that works in a asynchronous way. The API queues the images which are request from the storage controller. After the operation finishes, the image recognition API itself requests to Python main server's storage controller and informs with the result data. The operation in the image recognition API can take from 5 seconds to 6 hours.

- Web API is the expose API that helps the desktop application and python web server to talk with each other. Web API is a REST API that helps both the python main server and desktop application to talk with each other in a request-response cycle way.

- Desktop application is the main control and management UI that helps researchers and admins to control the software. The desktop application uses the Web API to access the configurations of the microscopes and cameras, and the saved pictures, recordings and the system logs.

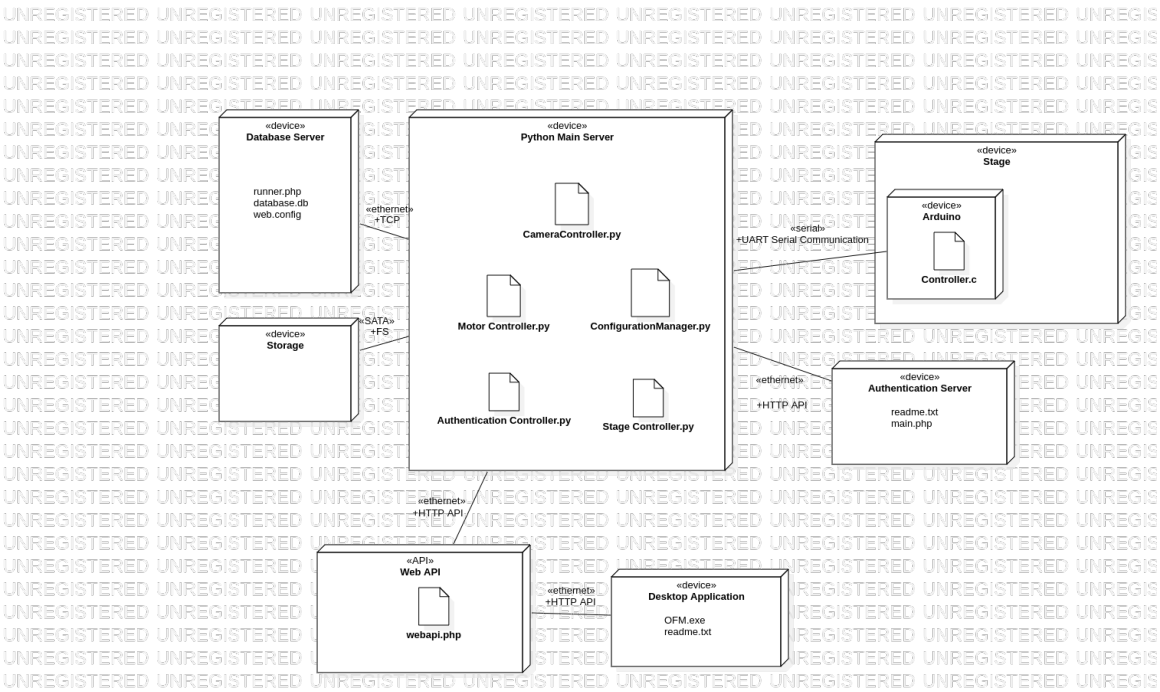

Figure 7: Deployment Diagram

Design Rationale

- The internal components of the system are written in Python. The system uses the `Flash` Web framework to handle the requests and response them.
- The system uses the version 3 of the Python to operate, since other versions of the Python, namely 1 and 2, are quite old.
- The Python main server uses a `WSGI` runner called `gunicorn` to run. Gunicorn automatically load balances the requests and waits if there is no capacity at the current moment.
- The connection between Python main server and the Arduino is UART serial connection. `UART` serial connection is a type of serial communication that uses only two wires to operate.
- The system provides a resource based RESTful API to operate. The RESTful API uses `GET`, `POST`, `PUT`, `DELETE`, `HEAD`, and `OPTIONS` HTTP request types to talk with the outside world.
- The web API acts like a relay server between the desktop application and the main python server. The web API is the RESTful API that the system provides.
- Database server is RDBMS database, namely PostgreSQL. The database server is backup-ed every day to make sure that the system does not loose any data.
- The system uses SSL/TLS powered HTTPS connections between the components to make sure that the connections are secure. SSL/TLS certificates are not self-signed certificates and they are signed by trusted authorities.

- Researchers and admins connect to the system via the desktop application. The desktop application is an `Electron` based application that connects the web API.
- The system uses a Raspberry Pi 4 4GB to operate.
- The Raspberry Pi connected Arduino is the UNO model of the Arduino series.
- The motors that the system uses are step motors that can be controlled using an Arduino. The Arduino uses a motor-driver card such as `L298N` to drive the motors.
- The system is connected to a power supply to protect itself any damage from voltage corruptions. The Raspberry Pi is powered with a 5V 2.5A adapter, and the Arduino is powered by 9V 1A adapter. Thus, two different adapters are needed to power the system. Also, the motors must operate in 12V 2A environment. This power can be directly fetched from the power supply, or a third adapter can be used.

4.3 Information View

4.3.1 Interfaces

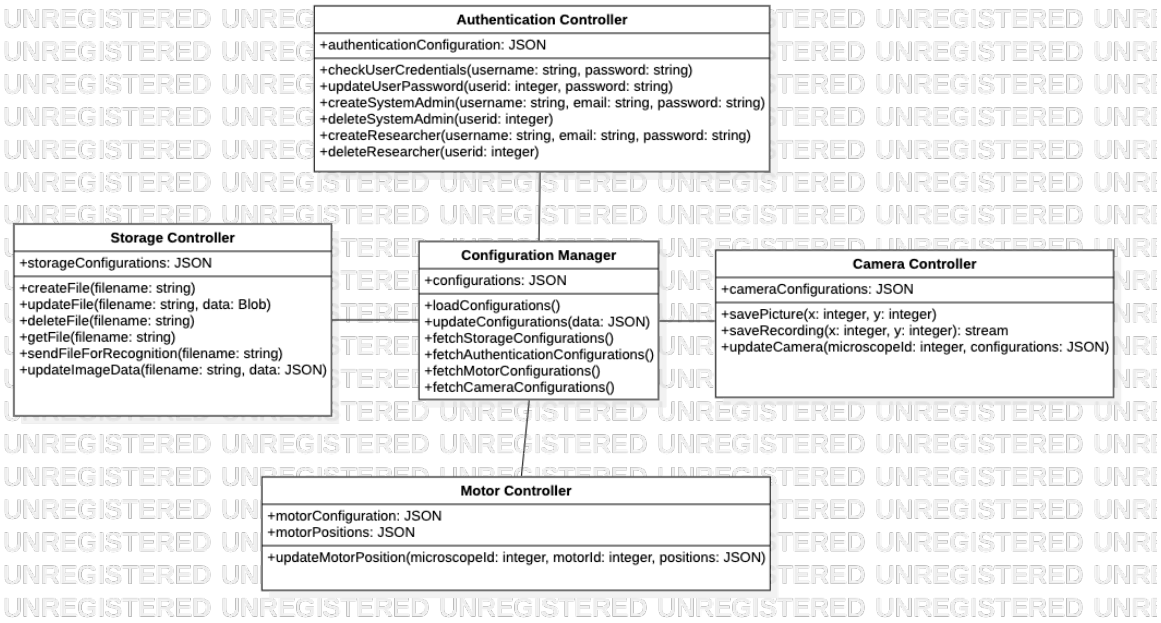

Figure 8: Interface Class Diagram

Operation	Description
checkUserCredentials	Checks the user credentials and confirms that the user password is the same as stored in the database.
updateUserPassword	Updates a researcher's or admin's password.
createSystemAdmin	Creates a system admin user in the system.
deleteSystemAdmin	Deletes a system admin declared with a user id.
createResearcher	Creates a researcher user with the given credentials.
deleteResearcher	Deletes a researcher user declared with a user id.
loadConfigurations	Loads configurations related with the other components in the system from the environment.
updateConfigurations	Updates the system configurations using the system environment.
fetchStorageConfigurations	Fetches the storage configurations using the system environment.
fetchAuthenticationConfigurations	Fetches the authentication configurations using the system environment.
fetchMotorConfigurations	Fetches the motor configurations using the system environment.
fetchCameraConfigurations	Fetches the camera configurations using the system environment.
createFile	Creates a file in the storage declared with a filename.
updateFile	Updates a file in the storage declared with a filename.
deleteFile	Deletes a file from the storage declared with a filename.
getFile	Fetches a file from the storage declared with a filename.
sendFileForRecognition	Sends the file declared with a filename to image recognition API.
updateImageData	Updates recognition data of the file declared with a filename using the image recognition API.
updateMotorPosition	Updates a position of motor position.
savePicture	Saves picture fetched from a camera.
saveRecoding	Saves the current recording.
updateCamera	Updates camera configurations such as color balance, white balance, resolution etc.

Table 20: Operation Descriptions

Design Rationale

- A microscope consists of one camera and one stage. A microscope configuration consists both the components.
- The configuration model `SystemConfiguration` different types of configurations. The type of configuration is determined by its `key`. The configuration manager is responsible for filtering these configurations.

- Every stage has more than one motors. The Arduino creates motor identifiers and maps the motor identifiers to motors.
- Cameras are responsible for both taking a picture and streaming a livestream. The CameraController can access both these properties.
- The Storage Controller is responsible for keeping the file changes and sending them to the image recognition API.

4.3.2 Database Operations

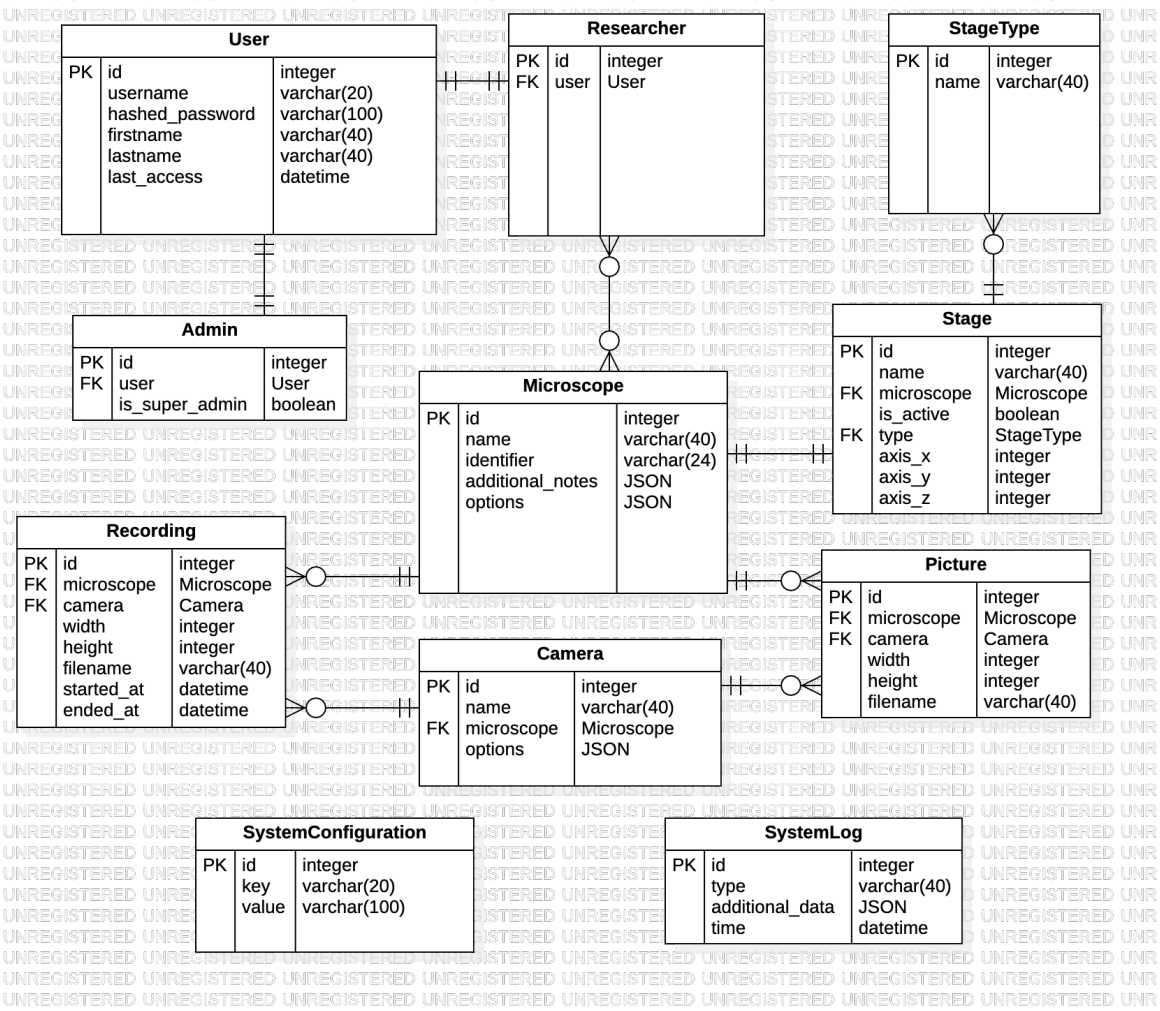

Figure 9: Database Class Diagram

Operation	Crud Operations
checkUserCredentials	Read: User Create: - Update: - Delete: -
updateUserPassword	Read: - Create: - Update: User Delete: -

createSystemAdmin	Read: - Create: User, Admin Update: - Delete: -
deleteSystemAdmin	Read: - Create: - Update: - Delete: User, Admin
createResearcher	Read: - Create: User, Researcher Update: - Delete: -
deleteResearcher	Read: - Create: - Update: - Delete: User, Researcher
loadConfigurations	Read: SystemConfiguration Create: - Update: - Delete: -
updateConfigurations	Read: User Create: - Update: SystemConfiguration Delete: -
fetchStorageConfigurations	Read: SystemConfiguration Create: - Update: - Delete: -
fetchAuthenticationConfigurations	Read: SystemConfiguration Create: - Update: - Delete: -
fetchMotorConfigurations	Read: SystemConfiguration Create: - Update: - Delete: -
fetchCameraConfigurations	Read: SystemConfiguration Create: - Update: - Delete: -
createFile	Read: Picture, Recording Create: - Update: - Delete: -

updateFile	Read: - Create: - Update: Picture, Recording Delete: -
deleteFile	Read: - Create: - Update: - Delete: Picture, Recording
getFile	Read: Picture, Recording Create: - Update: - Delete: -
sendFileForRecognition	Read: Picture, Recording Create: - Update: - Delete: -
updateImageData	Read: - Create: - Update: Picture, Recording Delete: -
updateMotorPosition	Read: Stage Create: - Update: Stage Delete: -
savePicture	Read: - Create: Picture, Recording Update: - Delete: -
saveRecoding	Read: - Create: Picture, Recording Update: - Delete: -
updateCamera	Read: - Create: - Update: Microscope, Camera Delete: -

Table 21: CRUD Operations

Design Rationale

- The database is a RDBMS database, namely PostgreSQL.
- The database itself at first must be only accessible by the Super Admin.
- The System Admin can define new users to access to database for maintenance purposes.
- Only defined users can access to the system.

- All System Admins can define new users to access to the system.
- Every user has a password to access the system. This password can be changed by the owner. In case of a password lost, the system admins can reset the user password.
- The database should be backed up every week in a different storage in case of a failure.

4.4 Interface View

4.4.1 Internal Interfaces

Interface between Camera Controller and Configuration Manager

Configuration manager sends new camera configurations to camera controller in order to change the configurations. If new configurations are the same as the current one, then camera controller ignores the incoming new configurations, since there is no point to bother the camera in this case.

Design Rationale

- Configuration Manager provides list of configurations in order to fulfill camera controller's request.
- Whenever new configurations are non-valid, camera controller eliminates this configurations log of this failure.

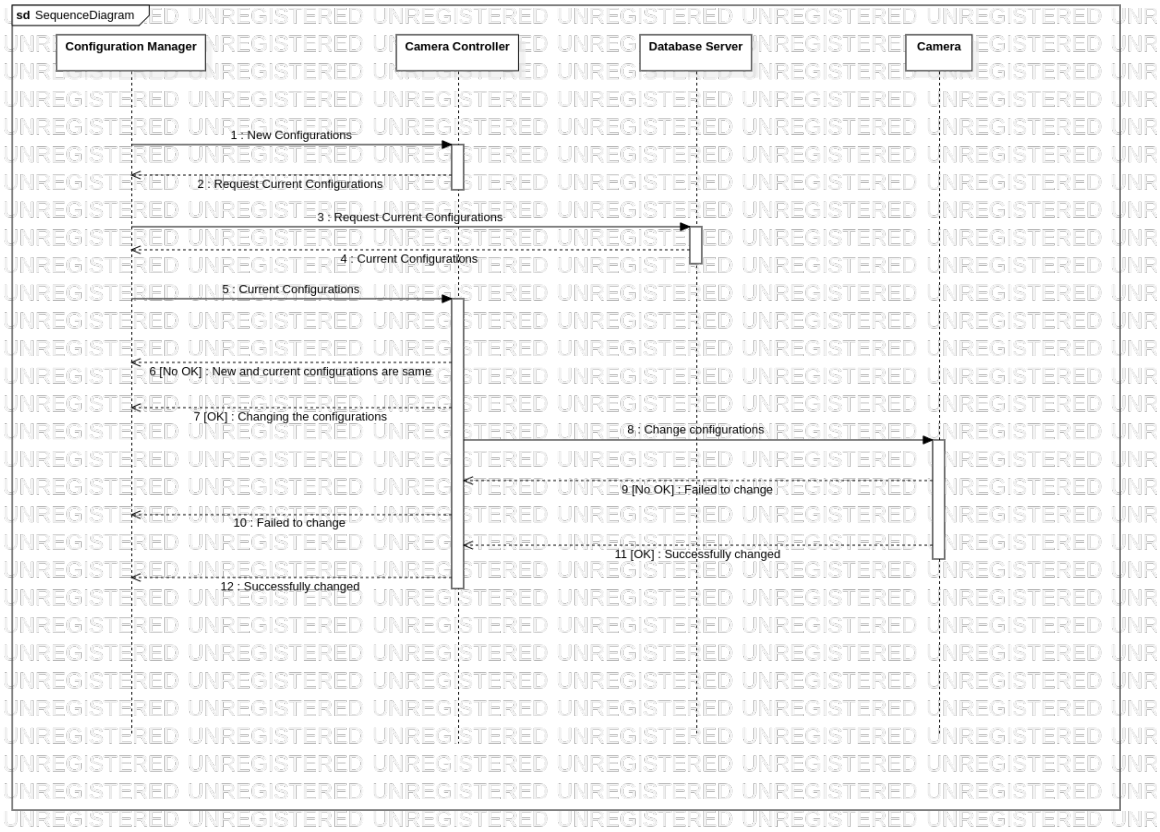

Figure 10: Sequence Diagram of Interface between Camera Controller and Configuration Manager

Interface between Arduino and Motor

Arduino sends new motor configurations to motor to change the configurations. Motor receives the new configurations and applies them. Then, Arduino is informed about whether applying the new configurations have been successful or not. Arduino uses a motor-driver such as L298N to drive motors since the power of Arduino is not sufficient enough.

Design Rationale

- Motor applies the configurations as soon as configurations arrives.
- Arduino is sent a message that whether or not applying the new configurations have been successful.

Interface between Camera Controller and Camera

Camera controller sends new camera configurations to camera to change the configurations. Camera receives the new configurations and applies them. The new configurations could include color balance, white balance etc. These configurations can be adjusted dynamically.

Design Rationale

- Camera applies the configurations as soon as configurations arrives.

Interface between Configuration Manager and Database Server

Configuration manager may want to get or update the configurations in the database. These configurations may include but not limited to: camera configurations, motor configurations, storage configurations. Whenever configuration manager wants to get or update, it create query. It sends created query to database. Database returns if query is successfully executed or not.

Design Rationale

- Configuration manager has priority in database since it may have high priority queries.
- Query results are stored with cause of failure in database if it fails.

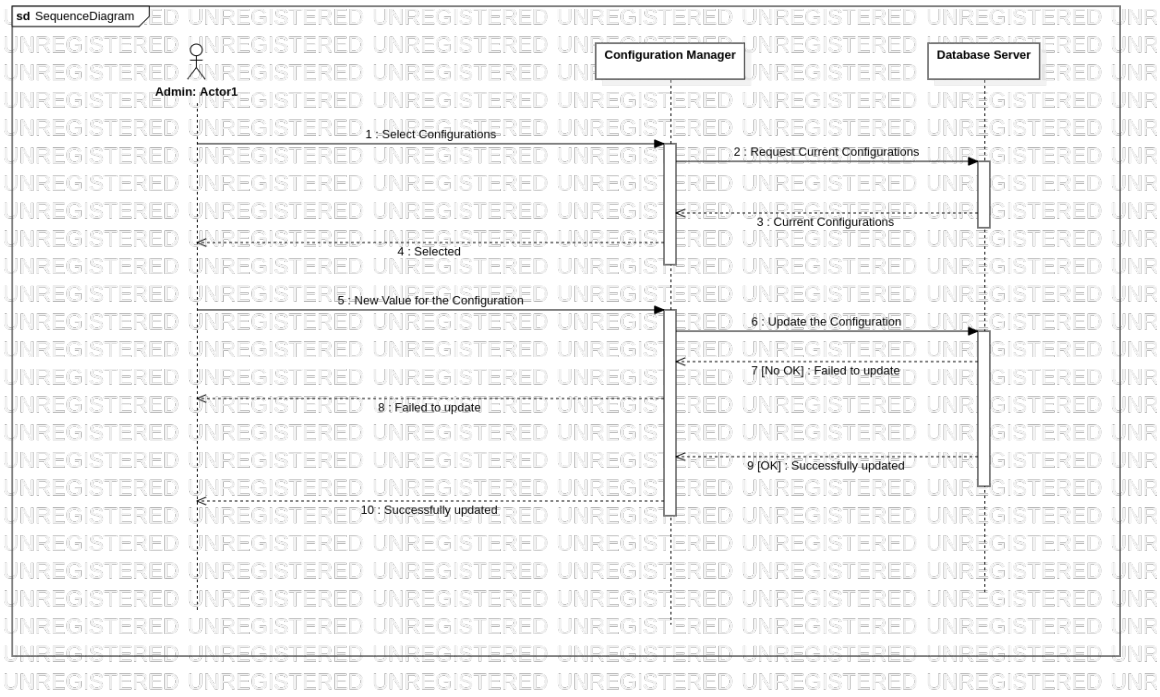

Figure 11: Sequence Diagram of Interface between Configuration Manager and Database Server

Interface between Motor Controller and Configuration Manager

Configuration manager sends new motor configurations to motor controller in order to change the configurations. If new configurations are the same as the current one, then motor controller ignores the incoming new configurations, since there is no point to bother the motor in this case.

Design Rationale

- Configuration Manager provides list of configurations in order to fulfill motor controller's request.
- Whenever new configurations are non-valid, motor controller eliminates this configurations log of this failure.

4.4.2 External Interfaces

User Interfaces:

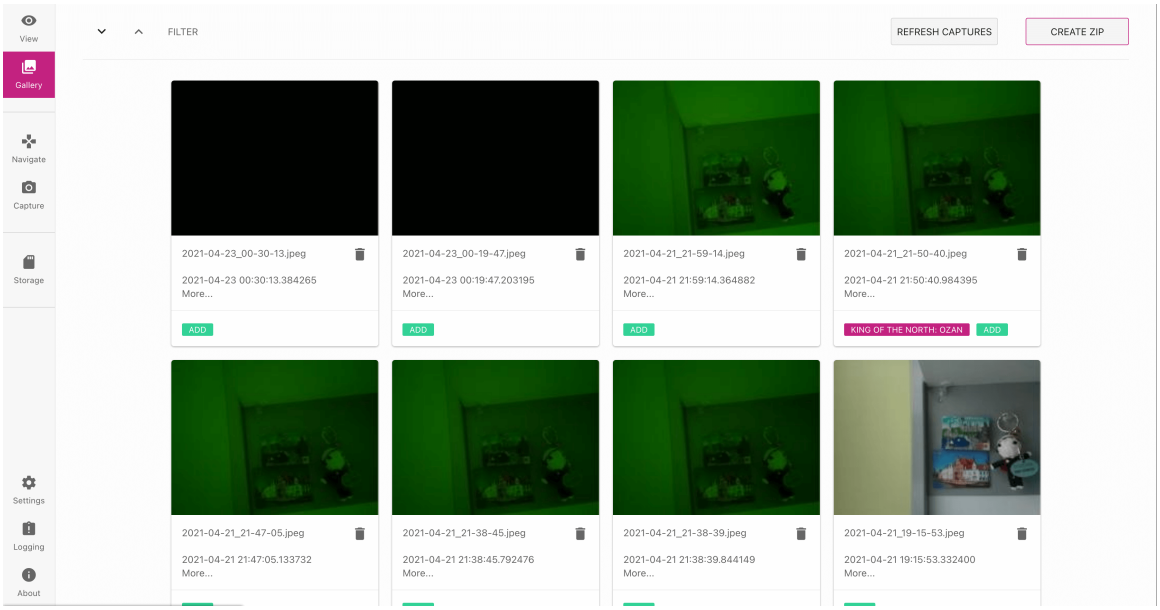

Figure 12: Gallery Page of Open Flexure Microscope

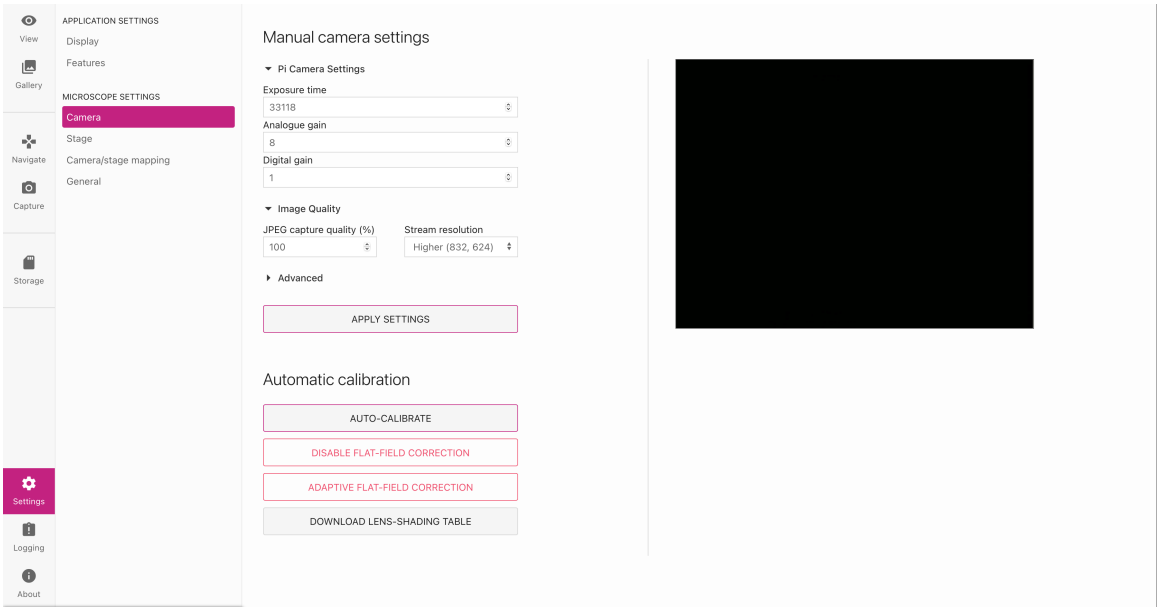

Figure 13: Settings Page of Open Flexure Microscope

Open Flexure Microscope has interfaces for researchers, admins, super admins. Their concerns and permissions varies depending on their position in the system. Detailed explanations are given for each user interface below.

Researcher Interface

Researchers can login to researcher interface with their login credentials. After the login procedure, researchers see an interface contains two tabs: Microscopes tab

and Gallery tab. In the Microscopes tab, researchers can access the microscopes they are authorized. Researchers choose a microscope from these microscopes and get the stream from it, update the microscopes configurations, save the picture. In the Gallery tab, researchers can see previously saved pictures or videos by himself/herself. Moreover, he/she can operate several operation to gallery items such as deletion, renaming, grouping, downloading etc. Researcher can also search for specific gallery item by its name.

Design Rationale

- All operation to gallery items must be recoverable to prevent data loss by mistake.
- For the sake of being user friendly, each selected operation can be handled not exceeding 4 button operation.

Admin Interface

Admin interface will be provide access to all data stored and functionalities in the system except some admin related functionalities. After the login procedure, admins see an interface contains three tabs: Microscopes tab, Users tab and Database tab. In the Microscopes tab, admins can access all the microscopes in the system. Admins can choose a microscope from these microscopes and get the stream from it, update the microscopes configurations. In the Users tab, admins can access all researchers in the system. Admins can authorize a researcher to microscope, add new researcher, update researchers' properties. In the Database tab, admins can view and edit the database.

Design Rationale

- Admin are capable and allowed to perform manipulative operations on the database. Therefore, admin interface provides read and write database access.

Super Admin Interface

Super admin interface will be provide access to all functionalities and data stored in the system. After the login procedure, admins see an interface contains three tabs: Microscopes tab, Users tab and Database tab. In the Microscopes tab, super admins can access all the microscopes in the system. Super admins can choose a microscope from these microscopes and get the stream from it, update the microscopes configurations. In the Users tab, admins can access all users in the system. Admins can authorize a researcher to microscope, add new researcher, add new admin, update users' properties. In the Database tab, super admins can view and edit the database.

Design Rationale

- Super admins are capable and allowed to perform manipulative operations on the database. Therefore, super admin interface provides read and write database access.

System Interfaces:
Interface between Storage Manager and Storage

Storage is awaken with interrupt comes from storage manager. Storage manager may request to store some video or image file in the storage. In that case storage stores the necessary files. Storage manager also may request to get some video or image file in the storage. in that case storage return the necessary files.

Design Rationale

- Storage waits for an interrupt from storage manager. Whenever storage manager request action, it sends interrupt to storage.
- In case any network error, storage stores result of operation (successfully completed/failed) of request action for one day.

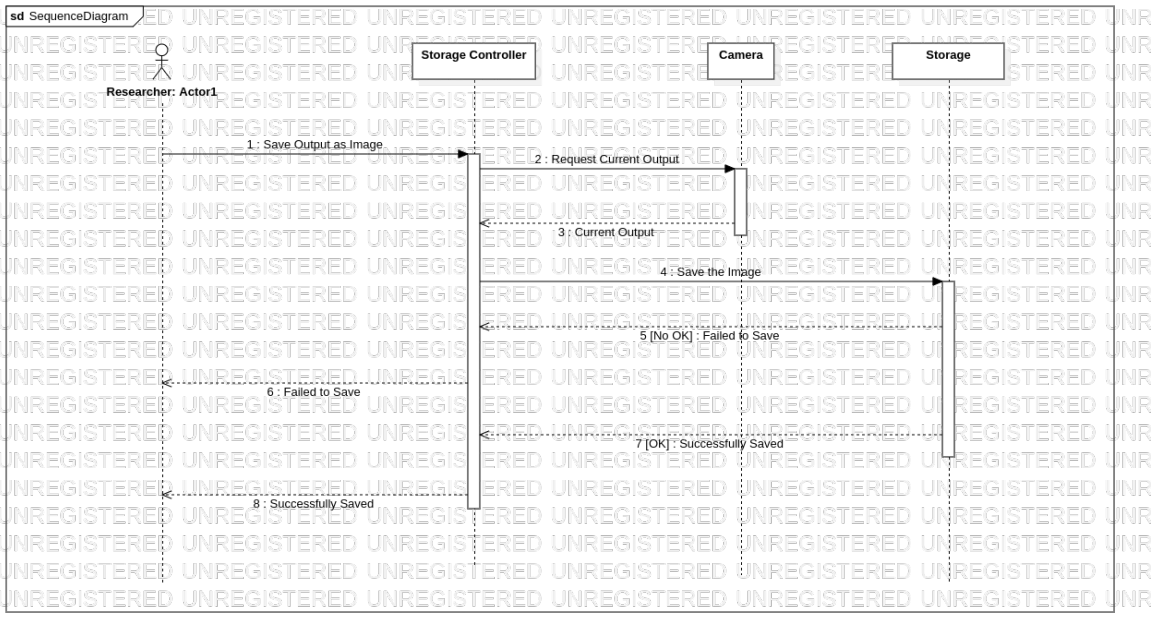

Figure 14: Sequence Diagram of Interface between Storage Manager and Storage

Interface between Authentication Server and Authentication Controller

Authentication server sends authentication credentials to authentication Controller. If user is already logged in, authentication system warned.

Design Rationale

- Whenever an user is detected already logged in, log of this failure is stored in database.

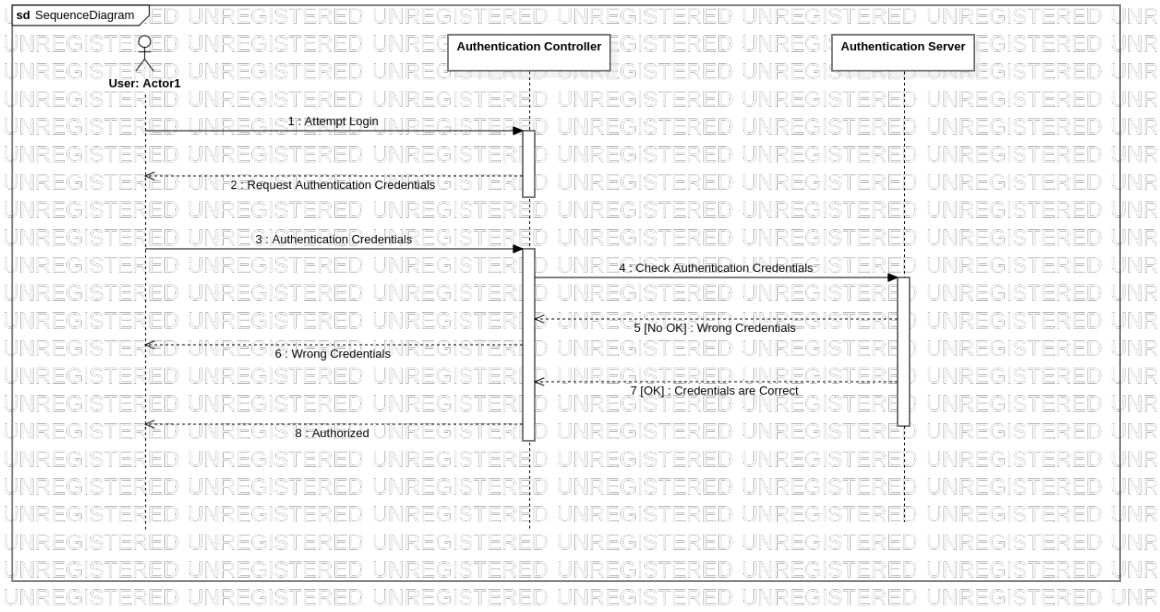

Figure 15: Sequence Diagram of Interface between Authentication Server and Authentication Controller

Interface between Storage Controller and Image Recognition API

External image recognition API takes images and/or videos from the storage controller and applies several image recognition algorithms to desired objects. Since image recognition API is an subscription based that is made by an external team, deals and payments are between researchers and the supplier of that corresponding API. OFM only provides the connection with this API.

Design Rationale

- Image recognition API must be in a confidentiality agreement with OFM system.
- The researchers must provide sample images to the API.

Interface between Desktop Application and Web API

Desktop application will communicate the OFM though the web API. The web API is provided by OFM itself. When desktop application is opened by a researcher or admin or super admin, the connection will be started with the web API. The connection between web API and the desktop application must remain open until desktop application is closed.

Design Rationale

- As information transferred between Desktop Application and Web API may be confidential, the connection must be secure.

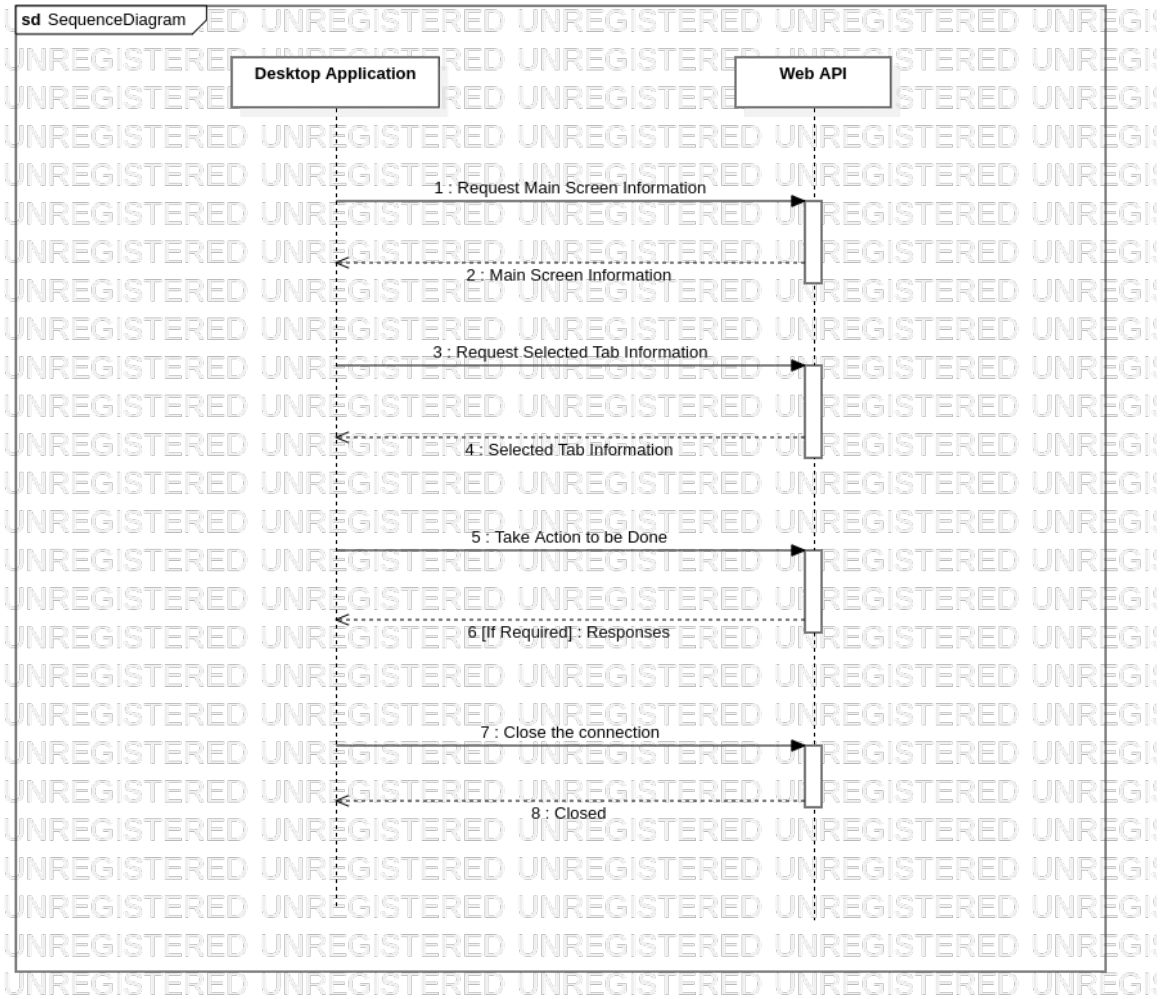

Figure 16: Sequence Diagram of Interface between Desktop Application and Web API

Interface between Web API and Python Main Server

As desktop application will communicate the OFM though the web API, Python Main Server will be notified through the web API that is provided by OFM itself. When a request comes to python main server, it will execute the commands and return if there is something to return.

Design Rationale

- As information transferred between Desktop Application and Web API may be confidential, the connection must be secure.

Interface between Authorization Server and Configuration Manager

Authorization server can run in different configurations in terms of how users will log in, the necessity of the login credentials etc. Therefore configuration manager will keep and handle how authorization server behaves.

Design Rationale

- Configurations of authorization server should be valid as it will effect the behaviour of the authorization server.
- Due to potential hacking attempts, the connection must be protected.

Interface between Image Recognition API and Storage Controller

Since image recognition API works asynchronous, it will send request a connection for completed images. It will send the process data to storage controller in order to store in storage. Since image recognition API is an subscription based that is made by an external team, deals and payments are between researchers and the supplier of that corresponding API. OFM only provides the connection with this API.

Design Rationale

- Image recognition API must be in a confidentiality agreement with OFM system.
- The researchers must provide sample images to the API.