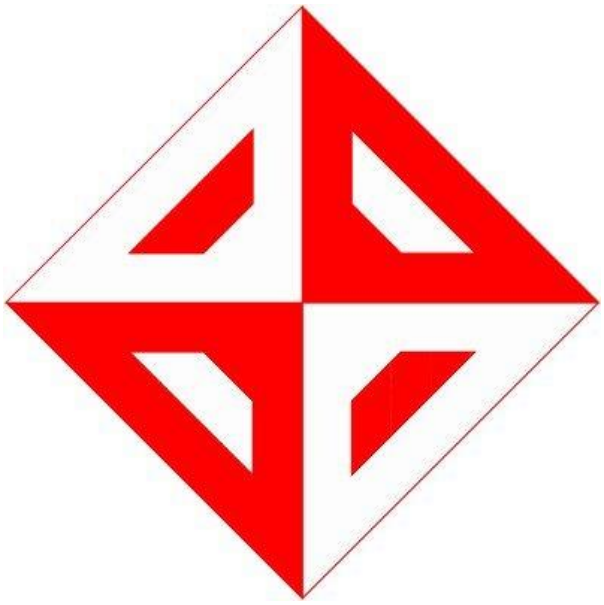

MIDDLE EAST TECHNICAL UNIVERSITY

Spring 2021

Computer Engineering Department

PROJECT:

OPENFLEXURE MICROSCOPE

Software Requirements Specification

Version 1.1

Prepared by:

Zeynepsu Kesim - 2292340

Table of Contents

List of Figures

List of Tables

Version History

Version	Date	Explanation
1.0	16.04.2021	Purpose of the system, scope and system overview is written. Context diagram is drawn. Use cases are determined and use case tables are created.
1.1	23.04.2021	Rest of the document is written with necessary tables and figures.

Table 1: Version History

1 Introduction

This document is the Software Specification Requirements (SRS) of the OpenFlexure Microscope which is co-developed by University of Bath and the Tanzanian engineering company STICLab.

1.1 Purpose of the System

The purpose of this project is to develop a cross-platform controllable microscope that will not rely on outdated hardware, or any equipment and reduce the work of microscopists that need to write complicated experiment codes and bind them together with inconsistent interfaces to run their experiments.

1.2 Scope

In the scope of the system, scientists can run their experiments without depending on any outdated hardware or inconsistent interfaces. To accomplish this, scientists register the system from Web GUI via entering their scientific credentials. When they are approved by the system, they can control the microscope, run experiments, and generate reports.

During their work, system runs number of services that are connected to the Web GUI to fulfil scientist's needs. Once scientists are in the system, they can monitor the microscope with data coming from the streaming service. Also, they can control the camera, microscope position via HTTP calls to IO Controller which is running on a Raspberry Pi. All these features are for making the scientist's work less than before.

The main services of the system are:

- Web GUI
- Microscope IO Controller
- Streaming Service
- Analytics Service
- Experiment Processor Service

1.3 System Overview

This section of the document will provide detailed information about the system including all components.

1.3.1 System Perspective

OpenFlexure Microscope system is a system that enables scientists to conduct their experiments in a much more flexible way without relying on extra equipment, hardware, or software. The microscope's adjustments, microscope camera screening, the experiments of scientists and analysing experiment results can be done and monitored from the platform.

This platform consists of various services including analytics system, streaming system, experiment processor system. Moreover, scientists can use the Web GUI to register the system and be authorised. After that they can access all the functionalities provided by the platform to perform their scientific works.

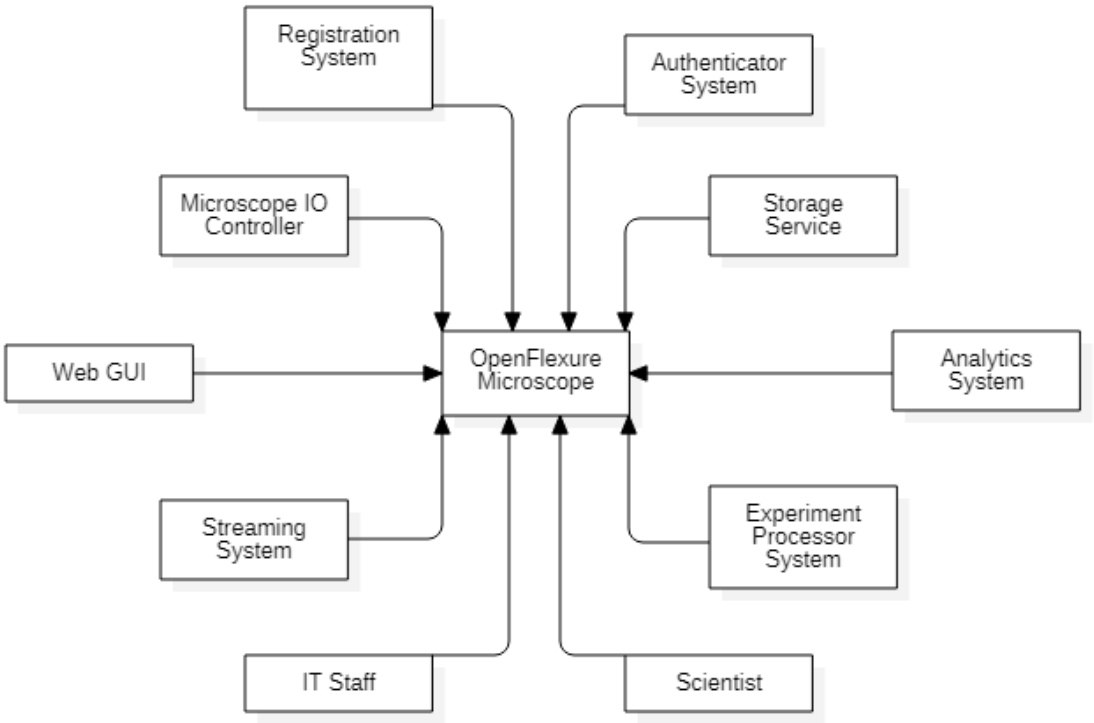

Figure 1: Context Diagram

1.3.1.1 System Interfaces

- **Streaming System:** This interface, running on a Raspberry Pi, is for streaming live camera feed from the microscope to the Web GUI, it provides taking snapshots of a specific time from the camera. Also, this interface gives the information like camera status, camera position, angle etc.
- **Experiment Processor System:** This interface used for running experiments of the scientists and giving experiment results. There are predefined experiments in the platform, but scientists can also create new experiments on the system by the Web GUI.
- **Analytics System:** This interface used for analysing past experiment results and generating weekly/monthly/quarterly statistical reports about them.
- **Web GUI:** This system is used for delivering end-user interfaces, displaying live camera feed, conducting experiments and monitoring analysis results.
- **Registration System:** This system is for verifying the identity credentials of the scientists.

- **Storage System:** This system is used for keeping the persistent data on the platform. For snapshots taken from the microscope's camera, an SD card or an available USB storage device is used, using Raspberry Pi. For other purposes, system uses a relational database.

User Interfaces

- **Registration Interface:** System provides scientists a registration interface that they will need to enter their scientific identity credentials to be verified.
- **User Interface:** Scientists can use the interface to control the microscope to change microscope camera angle and position, conduct their experiments, analysing experiment results and generating analysis reports. There are also interfaces for the account settings, activity history, and the activities of other scientists of the same scientific organisation.
- **IT Staff Interface:** This interface for the use of system admins, and engineers. They can see the error logs and warnings and all the data on the platform. They are responsible with keeping the communication between different systems is up and running.

1.3.1.2 Hardware Interfaces

- **Microscope IO Controller:** This interface, running on a Raspberry Pi and using an Arduino-based motor controller, is used for repositioning the microscope or microscope camera, and it requires different types of initiators to do the action. The types can be a signal from a USB cable, or an HTTP call.

1.3.1.3 Software Interfaces

- **Database:** Logs, past experiments, account information is stored in a PostgreSQL Database Server.
- **Authenticator API:** This is a third-party API to authenticate scientists in the platform.
- **Operating Systems:** The application is built as a cross-platform product. So different operating systems like macOS, Linux/Unix and Windows are all supported.

1.3.1.4 Communications Interfaces

Web GUI uses the HTTPS protocol for communication with the backend of the OpenFlexure system. With using the HTTPS protocol, system ensures the security of the data transferred between client and the server.

Also, all communications are done with TCP except the microscope camera feed streaming. The streaming is using UDP since the video transmission does not require a strict handshake protocol.

1.3.1.5 Memory Constraints

The SD Card memory of the microscope have enough capabilities to store all data until the data is transferred into the main system database. Therefore, it is not a big deal for the system.

1.3.2 System Functions

Functions	Descriptions
Register the system	Users submit their scientific credentials to be checked by the registration system.
Authenticate User	The authentication system verifies that the login credentials provided are correct.
Get error logs and manage data	IT staff examines error logs and data to ensure that all services are operational.
Run Data Analysis on the Experiment Results	When a scientist requests an analysis of previous data, the system performs the required computations based on the parameters supplied by the scientist and generates the reports.
Run Scientific Experiment	Scientists can use the platform to run pre-defined experiments or upload a custom experiment to be executed.
Get activity history	Scientists can view their own activity history, past experiments, and so on, as well as the activity history of other scientists in the same organization.
Manage account	Scientists can make changes to their account information, and they can delete their accounts.
Take snapshots	Snapshots of the microscope can be taken with microscope's camera and can be saved to the scientist's account or locally.
Get Live Camera Feed	The live camera feed is streamed to the Web GUI for scientists by the streaming service.
Control the Microscope	The directives supplied by the scientists in the Web GUI can be used to adjust the direction, location, or angle of the microscope and its camera.
Get Camera Information	Scientist can view camera information such as angle, position, and speed.

Table 2: System Functions

1.3.3 User Characteristics

There are two main users of the OpenFlexure Microscope system which are scientists and IT Staff.

Scientists needs to be registered to the system by giving their identity credentials after entering the Web GUI. If the validation of identity credentials fail, scientist shall not be able to register to the system. After passing the validation, scientists shall control the system. Thus, users need to have basic knowledge in computers and the internet. Since the system specifically designed for controlling and monitoring the microscope, making statistical inferences, conducting experiments, and generating reports including experiments' results, users are expected to have basic skills in microscopes, statistics, and research.

IT staff is in charge of solving technical problems that will occur in the microscope system. They will need to have high computer and programming skills. In addition to that, they are expected to have hardware knowledge, and problem-solving skills which they can use their skills efficiently, and thus, they should own the system and its capabilities and features.

1.3.4 Limitations

- **Regulatory Policies:** Since many scientific experiments is conducting on the platform and the experiment results are storing in the system with scientist's personal information.
- **Hardware Limitations:** Since the system runs on an embedded environment-Raspberry Pi and Arduino-, scientists should have an active OpenFlexure microscope to use when needed and wanted. For scientists, they also need additional device for connecting the system such as a regular laptop, or a desktop.
- **Interfaces to Other Applications:** OpenFlexure microscope system should be able to communicate other services and APIs that it uses.
- **Parallel Operations:** System must be capable of serving multiple scientists contemporaneously in parallel, thus, advanced locking and threading mechanism is a must.
- **Audit Functions:** System does not involve banking, hence there is no audit function.
- **Control Functions:** Controlling the functions are available to IT Staff, and requires necessary privileges.

- **High-Order Language Requirements:** Main microscope server is written in Python with much of the backend code released as a separate library, and this library uses Flask because it includes various utilities to simplify thread-based concurrency, and documentation generation.
- **Signal Handshake Protocol:** System uses HTTP Protocol for sending and receiving information from the server.
- **Quality Requirements:** It is very important to keep experiments' data and personal data safe. Thus, system should back-up regularly.
- **Criticality of the Applications:** System failures are very important, and system needs to be reliable since scientists run experiments on the system.
- **Safety and Security:** All data is protected by the technologies of IT Staff uses. Furthermore, the Web GUI should be tested regularly against attacks such as XSS attacks and so on.
- **Physical/Mental considerations:** There is no physical or mental constraints.

1.4 Definitions

Term	Definitions
IT Staff	Information Technology Staff
API	Application Programming Interface
SD Card	Secure Digital Memory Card
HTTP	Hyper Text Transfer Protocol There can be different types of HTTP requests like POST (update or add a resource), DELETE (delete a resource). Commonly, this protocol returns a status code 200.
Web GUI	Graphical User Interface for scientists to use the system
UDP	User Datagram Protocol is a communication protocol used across the internet for video streaming etc.
IO Controller	Input Output Controller
XSS	Cross Site Scripting
TCP	Transmission Control Protocol
USB	Universal Serial Bus
HDMI	High-Definition Multimedia Interface

Table 3: Definitions

2 References

This document is prepared with respect to IEEE 29148-2018 standard: 29148-2018 - ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes –Requirements Engineering.

Collins, J. T., Knapper, J., Stirling, J., McDermott, S., & Bowman, R. (2021). *Modern Microscopy with the Web of Things: The OpenFlexure Microscope Software Stack*. http://arxiv.org/abs/2101.00933

ISO/IEC/IEEE. (2018). *Systems and Software Engineering. Life cycle processes. Requirements engineering ISO/IEC/IEEE 29148:2018*. https://www.iso.org/standard/72089.html

Stirling, J., Sanga, V. L., Nyakyi, P. T., Mwakajinga, G. A., Collins, J. T., Bumke, K., Knapper, J., Meng, Q., McDermott, S., & Bowman, R. (2020, October 29). The OpenFlexure Project. The technical challenges of Co-Developing a microscope in the UK and Tanzania. *2020 IEEE Global Humanitarian Technology Conference, GHTC 2020*. https://doi.org/10.1109/GHTC46280.2020.9342860

3 Specific Requirements

3.1 External Interfaces

Following class diagram represents the relationship between interfaces and their functionalities.

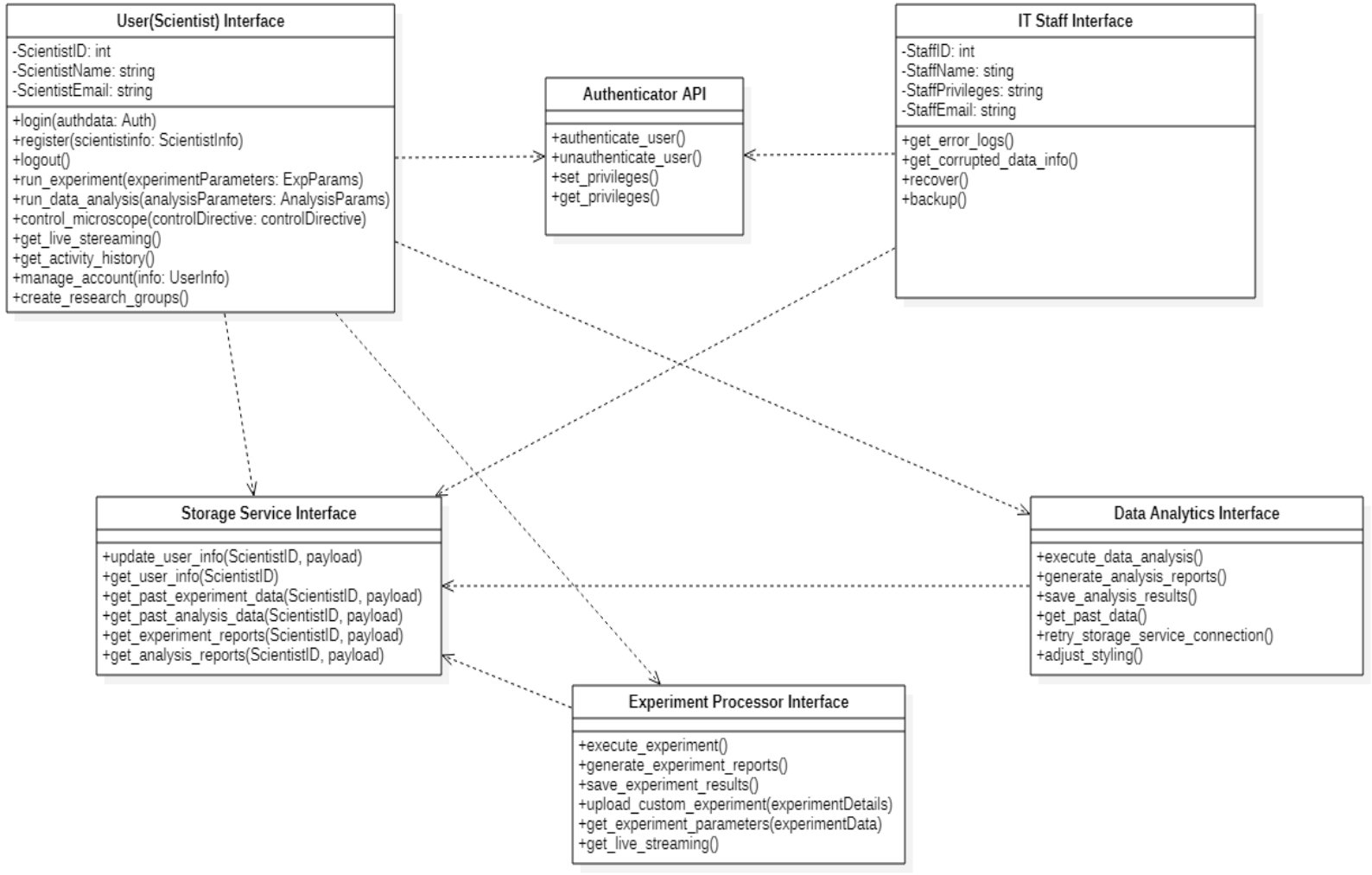

Figure 2:External Interfaces Diagram

3.2 Functions

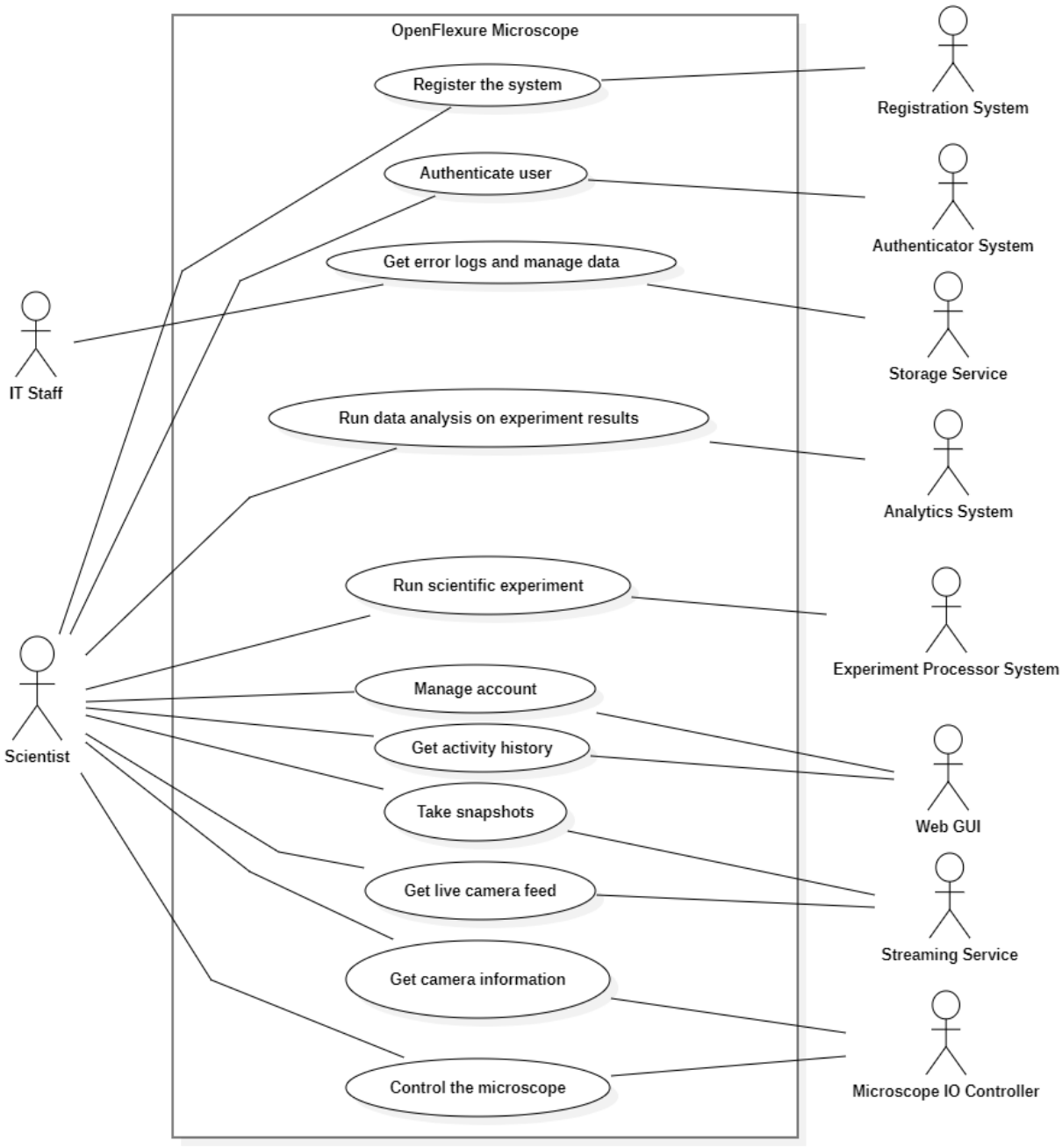

Figure 3: Use Case Diagram

Use Case Name	**Register the system**
Actors	Scientist, Registration system
Description	Users gives their scientific credentials to the registration system to be verified
Data	Information about the scientist
Preconditions	-
Stimulus	Filled information sent from Web GUI
Basic Flow	1. Scientist gives their personal and scientific credentials 2. Registration system checks whether the credentials scientist has supplied are correct. 3. A verification code goes to the email of the scientist. 4. Scientist enters the verification code to the system 5. Registration system verifies the scientist
Alternative Flow #1	2. Given credentials are wrong. Then, scientist is prompted to enter correct information
Alternative Flow #2	5. System fails to verify scientist's verification code due to wrong entrance of the verification code
Exception Flow	3. E-mail did not reach the scientist's email due to error on email service. Then, scientist requests verification code again
Postconditions	Scientist is verified

Table 4: Register the System

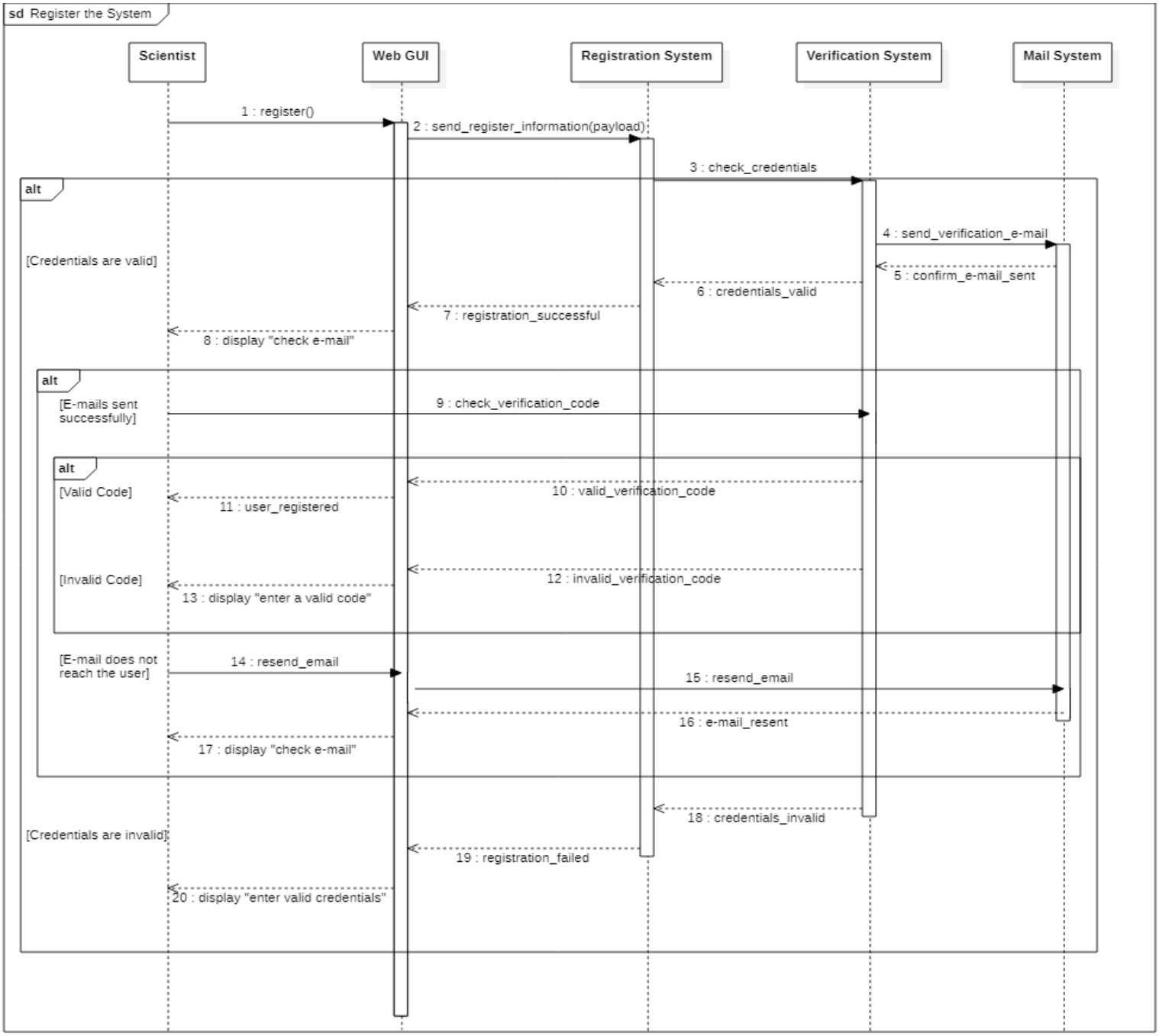

Figure 4:Register the System Sequence Diagram

Use Case Name	**Authenticate User**
Actors	Scientist, Authentication System
Description	Authentication system checks the given login credentials are valid
Data	Login credentials
Preconditions	Scientist should be registered
Stimulus	-
Basic Flow	1. Scientist enters login information. 2. Auth system check if the login credentials are correct and exist. 3. System make user authenticated in the system
Alternative Flow #1	2. User enters wrong credentials. Then system prompts user to give correct credentials
Alternative Flow #2	2. User does not exist in the system. Then system redirects him/her to register page
Exception Flow	-
Postconditions	User is authenticated

Table 5: Authenticate User

Use Case Name	**Get error logs and manage data**
Actors	IT Staff, Storage Service
Description	IT staff checks the error logs and data to make sure all services up and running
Data	-
Preconditions	-
Stimulus	-
Basic Flow	1. IT Staff enters to the Database Management System, and gather the logs 2. IT Staff checks the error logs of the application 3. Staff make sure there are no critical errors
Alternative Flow #1	1. IT Staff enters to the internal storage (or SD Card) to make sure there are no corrupted data
Exception Flow	-
Postconditions	-

Table 6: Get error logs and manage data

Use Case Name	**Run Data Analysis on the Experiment Results**
Actors	Scientist, Analytics Service, Storage System, Web GUI
Description	When scientist wants an analysis of the past data, system makes the wanted computations according to the parameters that scientist has supplied and generates the reports.
Data	Data in the storage system and the parameters that scientist supplied to the system.
Preconditions	-
Stimulus	Entering the parameters to the system by the scientist
Basic Flow	1. Scientist enters the Web GUI. 2. Enters the parameters that being wanted to be computed. 3. System gets the past data from the storage system. 4. System makes the analysis according to the parameters. 5. Generates a report with the findings
Alternative Flow #1	4. Data is insufficient to compute the wanted parameters and make the analysis. Then, system gives a warning to the scientist. 5. System offers 2 options to ask the scientist if s/he wants to continue with the existing insufficient data and make the computations accordingly, or enters new parameters to compute from the beginning.
Exception Flow	3. System may not be able to reach the past data in the storage system due to connection or time-out error. 4. System retries to get data from the storage system (retry limit is 1)
Postconditions	A final report is generated

Table 7: Run Data Analysis on the Experiment Results

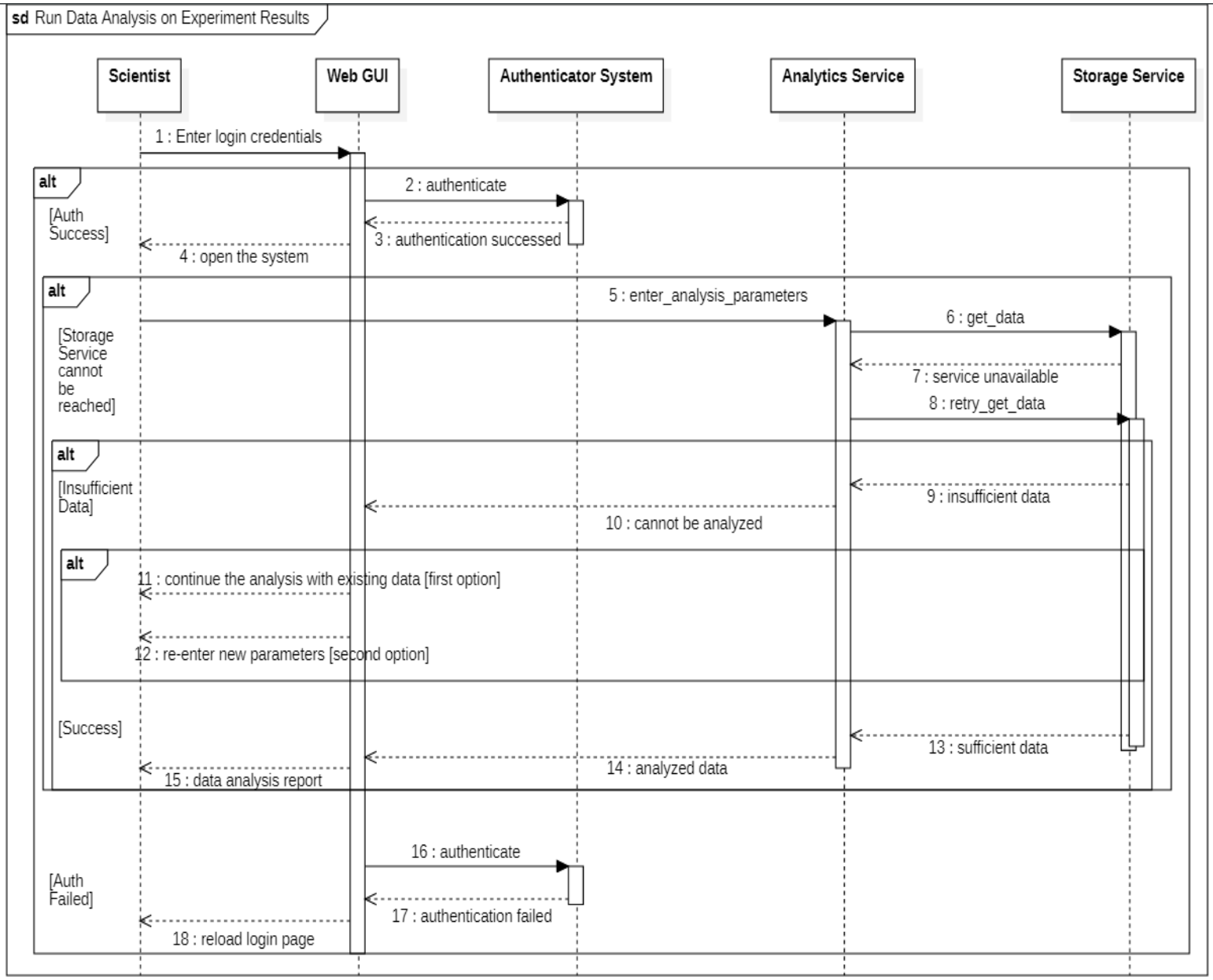

Figure 5:Run Data Analysis on Experiment Results Sequence Diagram

Use Case Name	**Run Scientific Experiment**
Actors	Scientist, Experiment Processor System, Web GUI
Description	Scientists can run pre-defined experiments from the platform or upload a custom experiment to be executed.
Data	Experiment features, parameters supplied to the system by the scientist
Preconditions	There should be a sample to experiment on the microscope
Stimulus	Run an experiment command from the scientist
Basic Flow	1. Scientist selects which experiment to run. 2. Scientist decides parameters to be computed on that experiment. 3. Experiment processor system computes the parameters. 4. Generates a report with the findings of that experiment
Alternative Flow #1	1. Scientist can upload a custom experiment to the platform. 2. Scientist decides parameters to be computed on that experiment. 3. Experiment processor system computes the parameters. 4. Generates a report with the findings of that experiment.
Exception Flow	-
Postconditions	A final report is generated

Table 8: Run Scientific Experiment

Use Case Name	**Manage account**
Actors	Scientist, Web GUI, Storage Service
Description	Scientists can update their account information.
Data	-
Preconditions	Scientist should be authenticated
Stimulus	Update request from Web GUI
Basic Flow	1. Scientist enters the manage account page. 2. Scientist updates the account information. 3. Web GUI posts an HTTP POST request to storage service. 4. Storage service handles the request and updates the account information in the Database
Alternative Flow #1	2. Scientist may want to delete their account. 3. Web GUI posts an HTTP DELETE request to storage service. 4. Storage service handles the request and make user disabled.
Exception Flow	-
Postconditions	Account information is updated

Table 9: Manage Account

Use Case Name	**Get activity history**
Actors	Scientist, Web GUI, Storage Service
Description	Scientist can see their activity history, past experiments etc. or the activity history of the scientists of the same organisation.
Data	-
Preconditions	Scientist should be authenticated
Stimulus	Scientist should go to the activity history page
Basic Flow	1. Scientist enters the activity history page on the Web GUI. 2. Web GUI gathers data from storage service and returns the activity history
Alternative Flow #1	1. Scientist enters the activity history page on the Web GUI, and selects the 'see organisation activity history' option. 2. Web GUI gathers data from storage and returns the activity history.
Exception Flow	-
Postconditions	Activity history displayed to the scientist

Table 10: Get Activity History

Use Case Name	**Take snapshots**
Actors	Scientist, Streaming service, Web GUI
Description	Scientist can take camera snapshots of the microscope and then save it on his/her account or locally
Data	-
Preconditions	Scientist should be in the live streaming feed
Stimulus	Snapshot command from Web GUI
Basic Flow	1. Scientist should go to the streaming page in Web GUI. 2. Scientist should click the snapshot button. 3. Web GUI sends an HTTP request to the streaming service. 4. Streaming service creates another thread instead of live feed thread and sends that specific snapshot in the payload. 5. Scientist takes the snapshot and save it locally.
Alternative Flow #1	5. Scientist can save the snapshot in the platform using Raspberry Pi instead of downloading locally.
Exception Flow	4. Streaming service unable to create new thread due to insufficient processor power. System generates an error log message for the investigations of IT Staff.
Postconditions	Scientist takes the snapshot

Table 11: Take Snapshot

Use Case Name	**Get Live Camera Feed**
Actors	Scientist, Streaming Service
Description	Streaming service streams the live camera feed to the Web GUI for scientists
Data	-
Preconditions	There should be a monitor that scientist can see the stream (laptop, or external monitor)
Stimulus	Scientist should go to the live streaming page.
Basic Flow	1. Streaming service finds all the connected devices (laptops, displays etc.) 2. Streaming service sends all the frames that captured by the camera via UDP. 3. Web GUI gets the frames and displays the camera feed.
Alternative Flow #1	-
Exception Flow	1. Streaming service may not be able to find connected devices or stream. In that case IT staff should be informed.
Postconditions	Scientist can see live camera feed.

Table 12: Get Live Camera Feed

Use Case Name	**Get Camera Information**
Actors	Scientist, Microscope IO Controller
Description	Scientist can see camera information like angle, position, speed etc.
Data	-
Preconditions	There should be a monitor that scientist can see the stream (laptop, or external monitor)
Stimulus	Scientist should go to the live streaming page.
Basic Flow	1. Scientist enters the streaming page. 2. Web GUI sends a request to microscope IO controller to gather camera information. 3. Microscope IO Controller checks the hardware properties and sends the information to the Web GUI. 4. Web GUI displays the camera information on the live streaming page.
Alternative Flow #1	-
Exception Flow	2. Web GUI may not be able to reach the microscope IO controller due to connection or time-out error. 3. System retries to reach the microscope IO controller (retry limit is 1)
Postconditions	Information about the camera is taken and displayed on the live streaming page on Web GUI

Table 13: Get Camera Information

Use Case Name	**Control the Microscope**
Actors	Scientist, Microscope IO Controller System
Description	When scientist wants to change the direction, position, or angle of the microscope, s/he can do it by using the directives in the Web GUI.
Data	New direction, position, or angle of the microscope supplied by the scientist
Preconditions	Scientist should be in live streaming page
Stimulus	Incoming update directive from the scientist
Basic Flow	1. Scientist sets the new values for the microscope properties. 2. GUI sends the update request to the IO Controller system 3. System sets new values for the microscope properties with using Arduino. 4. System returns an HTTP response with status 200.
Alternative Flow #1	-
Exception Flow	1. Microscope IO Controller system may not be reachable due to a hardware failure. In case that happens, Web GUI generates an error log for the IT staff.
Postconditions	New microscope position, direction, or angle value is set.

Table 14: Control the Microscope

3.3 Usability Requirements

- Scientists shall be able to be involved in different groups, with different accounts than the personal accounts, to collaborate. In these accounts, they can see all the experiments, results, analyses, and reports that any of them shared or done.
- A scientist group shall connect to microscope and monitor live camera streaming simultaneously.
- In case of there exists no internet connection, microscope shall be controlled via USB cable and monitored via HDMI connection.
- All features in the system shall be used efficiently and easily when a wireless connection exists.
- IT Staff shall be able to get any error logs or corrupted data in at most 3 steps.
- To emphasize significant points, scientists shall be able to change the colour or font shapes of the writings in the reports.
- A comprehensive demo is shown to teach the system which takes no more than 2 hours when a new scientist registers to the system.

3.4 Performance Requirements

- Microscope shall respond nearly in real time whenever a scientist wants to change microscope's position, angle, or direction from the Web GUI.
- Experiment results shall be executed after the experiment in 30 seconds.
- Data analysis reports shall be generated in 2 minutes.
- System shall support 100 users simultaneously on the live streaming feed.
- Scientists' collaborative group sizes shall be supported up to 258 users.
- Users shall be directed to a requested page in 3 seconds.
- Any database or storage system related operations' latency shall not be more than 3 seconds.
- The system's backend must have at least 32 Gbps of bandwidth to serve live streaming for multiple users efficiently.
- Parallel computing shall be supported by the system's server, and hardware shall be selected correspondingly.

3.5 Logical Database Requirements

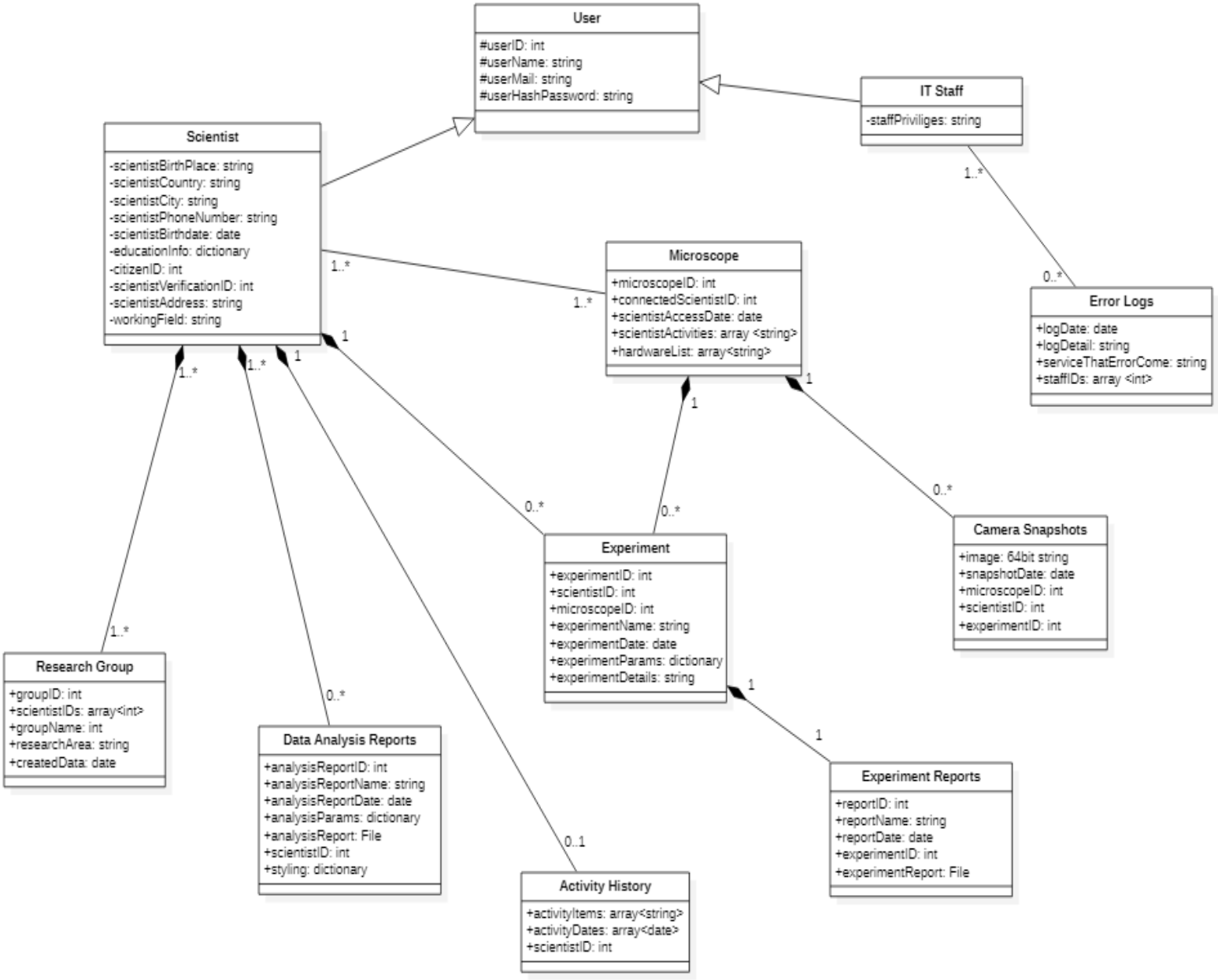

Figure 6:Logical Database Diagram

- When a scientist register the system, a scientist entry shall be created and a unique scientist ID will be given by PostgreSQL automatically.
- An e-mail or/and a personal information like citizen ID cannot be used by more than one user.
- IT Staff may access everything except user passwords and their access to the scientist table shall be read only.
- Scientist can change every information except their scientific credentials.
- A research group can contain more than one scientist and a scientist can belong more than one research group; hence, this relationship is a many to many relationship.
- When a scientist requests a data analysis reports, a new data analysis entry is created.
- If there is no scientist, the data analysis report table cannot exist, so it is a weak entity.
- Analysis params of the data analysis report entity shall not be null since analysis will be done by these parameters.
- Activity history is a weak entity of scientists and there shall be one to one relationship between these two entities because every user has exactly one activity history except scientist is not a new registered user.
- When a scientist requests to execute an experiment, a new experiment entry is created.
- If there is no scientist, the experiment cannot be executed, and thus, it is a weak entity.
- Experiment params of the executed experiment shall not be null.
- Experiment has a one to one relationship with experiment report. Also, experiment report is a weak entity of experiment because an experiment must exist to create a report for it.
- Scientist has a many to many relationship with microscope, so a microscope can be used by multiple scientists and a scientist can use multiple microscopes.
- Experiment can be run on one microscope, but a microscope can run multiple experiments. Therefore, they have a one to many relationship.
- Microscope has one to many relationship with camera snapshot, so a camera snapshot shall have a microscope, but microscope can have multiple camera snapshots.

- Many IT Staff may see many error logs; therefore, these tables have many to many relationship.

3.6 Design Constraints

- Users' private information, such as their identification credentials, passwords, recent activities, and experiment information and results, is protected by the system. No one other than the scientist shall access to these information data.
- Data shall be stored for legal purposes.
- All information that has been sent and received shall be encrypted in the database.

3.7 Software System Attributes

I. Reliability:

- Hardware components' failures shall not exceed one per 2 months.
- Data loss in the live streaming shall be less than 1%.
- Possible errors or system failures shall be logged and solved by IT Staff in less than one day.

II. Availability:

- IT staff shall be responsible for taking immediate actions after fatal errors.
- The system shall be able to tolerate minor failures while keeping the main operations running.
- System shall be available 99% of time besides maintenance time.
- System backup shall be done when no scientist is connected to the microscope system.
- All systems shall be restarted within 5 minutes of failure.

III. Security:

- ❖ All data is kept and transmitted encrypted.
- ❖ The system's components shall be checked on a regular basis.
- ❖ Whenever a new functionality added to the system, a regression test shall be done to test necessary permissions as well as application logic.
- ❖ Database and storage system shall be resistant to any attacks.

IV. Maintainability:

- ❖ Documentation of the system is well written and needs to be updated in any significant change in the codebase.
- ❖ Updating any hardware component shall not affect functionalities of the system.
- ❖ The subsystems of the OpenFlexure Microscope system shall be implemented in a microservice architecture and changes in one service shall not affect other services.

V. Portability:

- ❖ Web application shall not depend on different browsers and their versions as well as operating systems.
- ❖ Programming languages that are used in the development of the system shall not dependent.

3.8 Supporting Information

The OpenFlexure Microscope system is an improved microscope with additional web servers behind it in order to enable scientific organisations and scientist to conduct experiments more easily and track their work and progress more efficiently as well as faster.

4 Verification

5 Appendices

5.1 Assumptions and Dependencies

5.2 Acronyms and Abbreviations