INSTRUCTIONS OF THE CASE STUDY:

Your team is expected to prepare and deliver a **design proposal report** of your intended development work for the organisation. Note that the associated grading criteria are highlighted in the requirements below, to be reviewed alongside the criteria grid (Module Resources).

- Your report should detail the **system requirements and assumptions** (such as local or remote access, magnitude of storage required, CPU capacity, etc), design decisions (such as encryption algorithms, approach to data storage, use of databases, use of frameworks) and approaches that you have adopted to create your secure software solution. Build up a rationale where appropriate supported by literature **(Knowledge and Understanding weighted at 25%).**
- It should list security challenges you have identified/ expect to encounter (such as those in the STRIDE model and/or those forming the OWASP principles). Highlight (briefly) what paradigm(s), pattern(s), theories and practices you intend to utilise on this project to address security, technical and business challenges. Justify your approaches supported by literature. **(Application and Understanding weighted at 25%).**
- You should produce a graphical design, based on UML, that illustrates your approach via a number of views which should include, at a minimum, sequence diagrams, class diagrams, and activity diagrams. AND
- You should also state any tools, libraries and models that you will use in your solution, justified by academic research. AND

- Ensure that your system considers the functionality that similar applications provide, such as user registration, user roles, and CRUD functionality: which should you include? You will need to explain your reasoning and justifications for any omissions during your demonstrations **(Criticality weighted at 25%)**.

Presentation and Structure of your work (weighted at 25%) includes spelling, style, evidence of proofreading, correct use (and format) of citations and references.

It is recommended you use tables and bullet-point lists to stay within the word count limit.

Checklist for the assignment:

- Bulleted list of system requirements and assumptions, design decisions and approaches you will use based on background information and additional academic research (ensure you include any references you have used);
- Bulleted list of security risks/ vulnerabilities you have identified, including reference to frameworks used (e.g. STRIDE, OWASP) with potential mitigations and references;
- UML design of solution using multiple diagrams (e.g. class, sequence, activity) with references;
- Bulleted list of tools (including development and test tools), libraries and models you will use;
- Remember to use a spell checker and proofread your work before submission.

- **You should get your design outline reviewed and approved by your tutor BEFORE** you submit the final version in this Unit. You are invited to arrange a meeting between your team and the tutor to get feedback.

My Input:

Design Decisions and Approaches That You Have Adopted to Create a Secure Software Solution, and the Libraries Needed.

1. **Custom-designed Input Validating Code**

   Regular Expressions (RegEx or re) library (to enable the use of 're.search()' method to search and detect a series of strings within a given criteria (Lenka, 2020).

2. **One-Time Pin/Password (OTP) Generator and Password Generator for MFA**

   Importation of Random library (to allow generation of random numbers) and also Math library (to give access to common mathematical functions) (Oliphant, 2007).

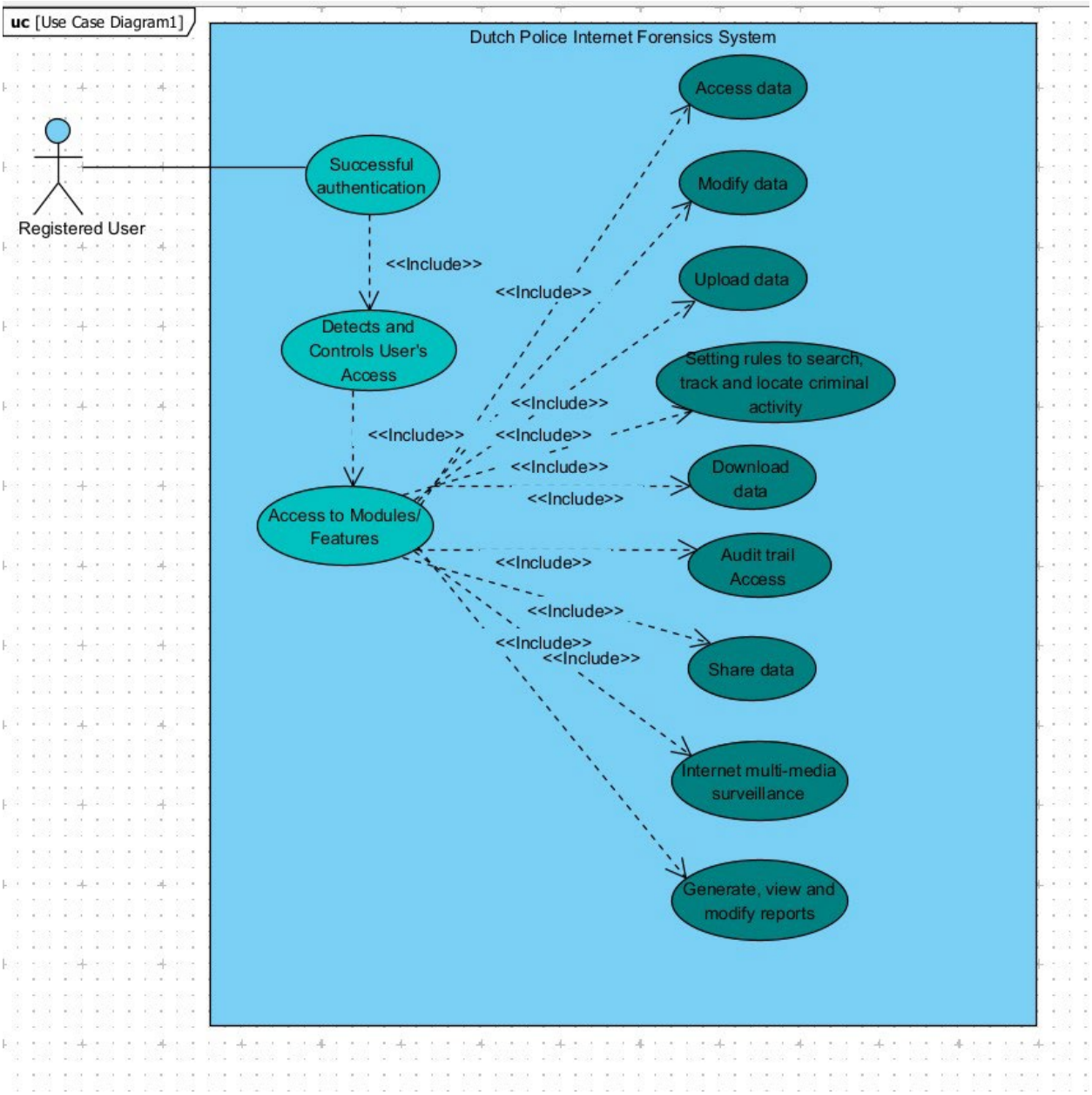
3. **Time based login of 3 attempts then blocks user after been unsuccessful**

   Mitigates cyber-attacks like brute-force, etc.

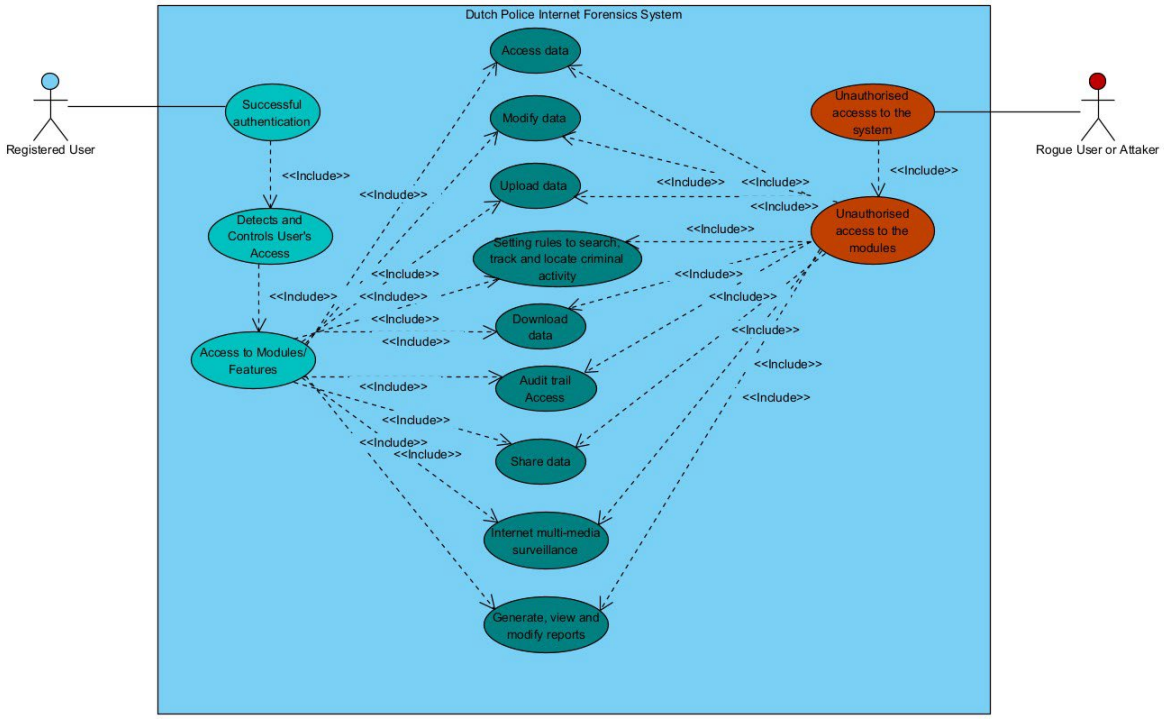
4. **Google translate library**
5. **Use of html, css, javascript and python for developing the web app**
6. **A database application SQL**
7. **Python Imaging Library:** commonly used to handle all types of image manipulation (Dauzon, et al., 2016)

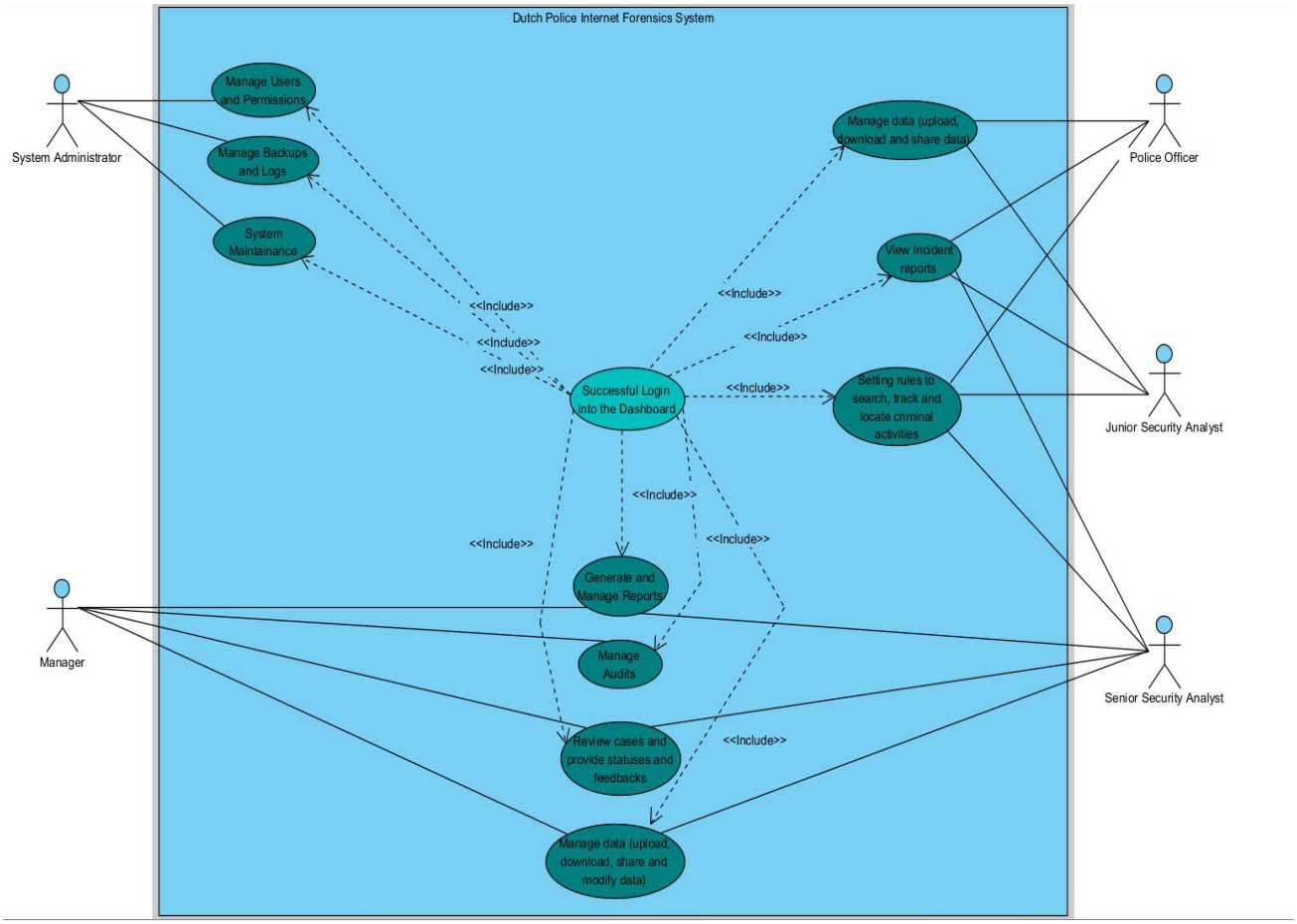
8. **Django forms library:** it joins 3 main components of the framework: database fields, html form tags and ability of the user's input to be validated and display error messages (Dauzon, et al., 2016).
9. **Python library:** to interface with SQLite (Dauzon, et al., 2016).
10. **Pycha library:** is a charting and graphic python library (Dauzon, et al., 2016).
11. **Javascript library:** allows a faster creation of client-side functionality (Dauzon, et al., 2016).
12. **ReportLab library:** allows generation of PDFs with python code (Melé, 2020)
13. **Venv library:** allows creation of lightweight virtual environments (Melé, 2020)
14. **Django-allauth:** used for authentication (local and social).
15. **Django import export:** allows backup of databases.
16. **Requests library:** allows HTTP basic authentication (Melé, 2020)
17. **Pillow library:** for handling images (Melé, 2020)
18.

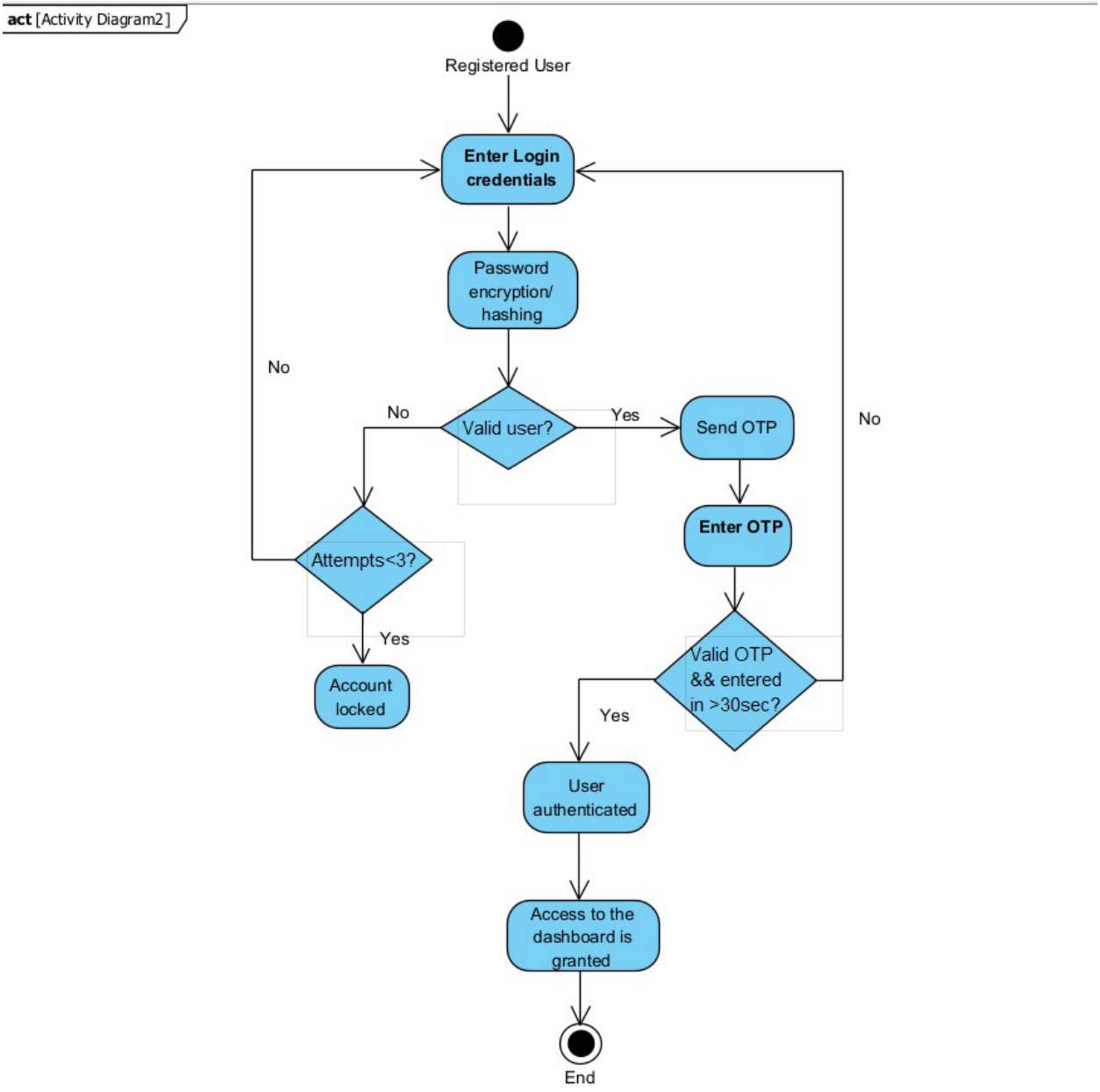
UML Diagrams

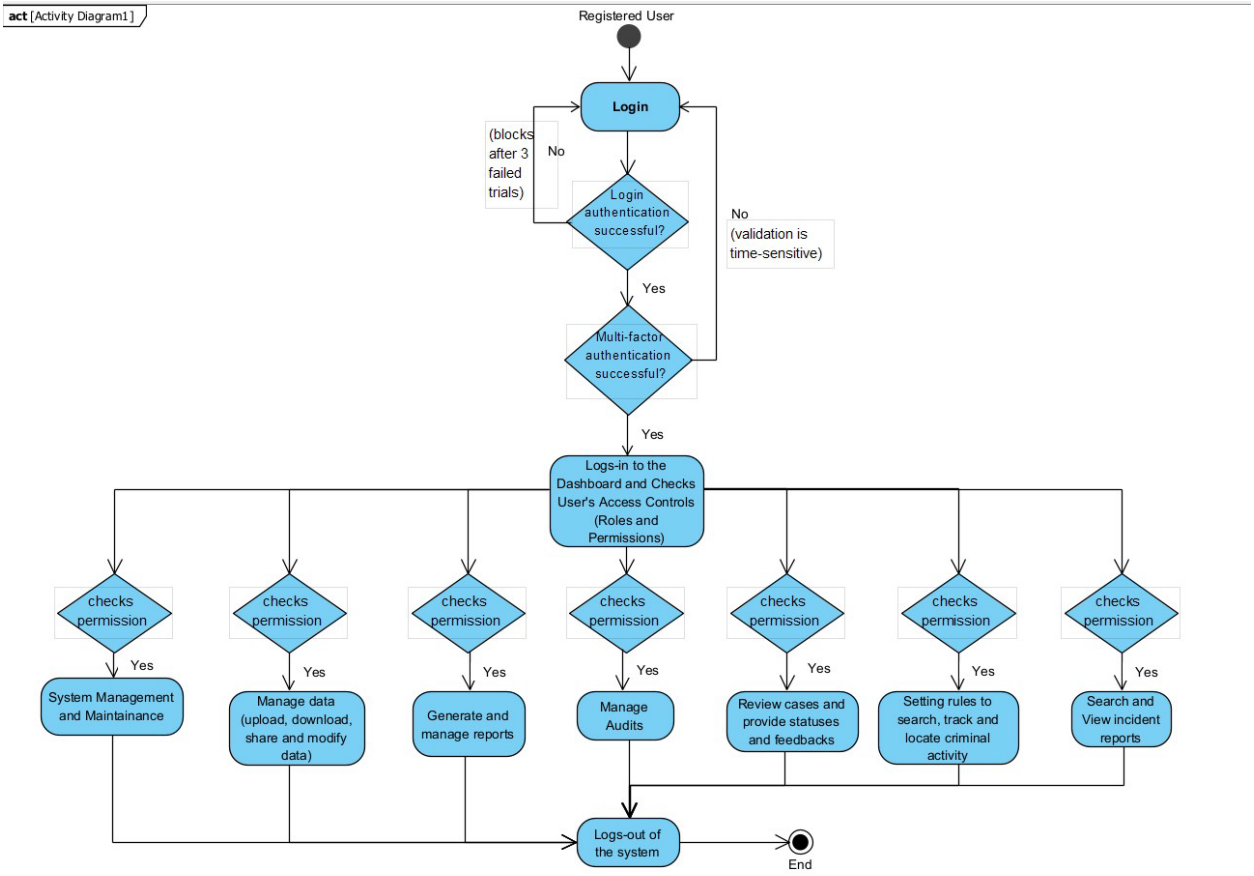


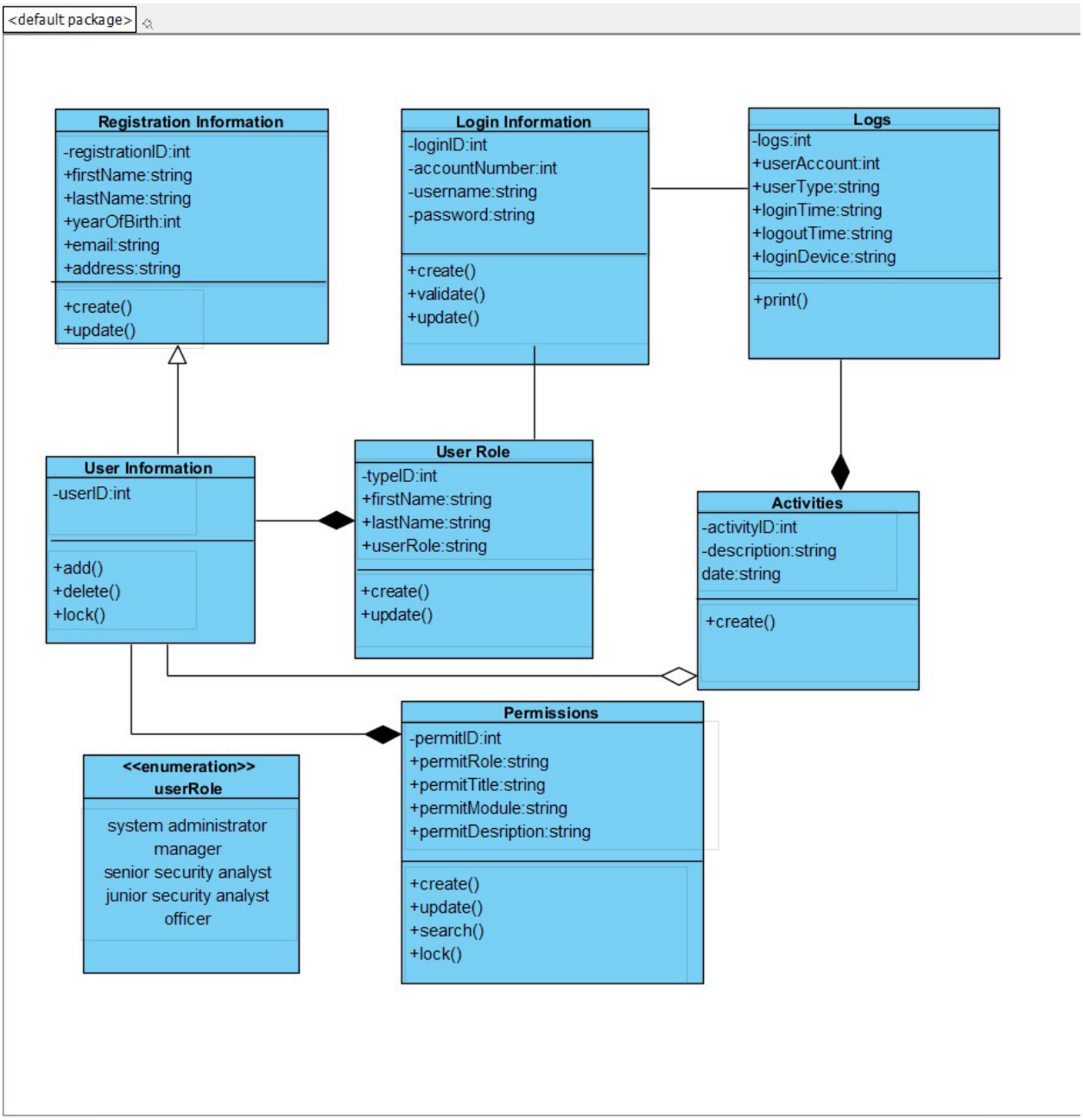
uc [Use Case Diagram2]

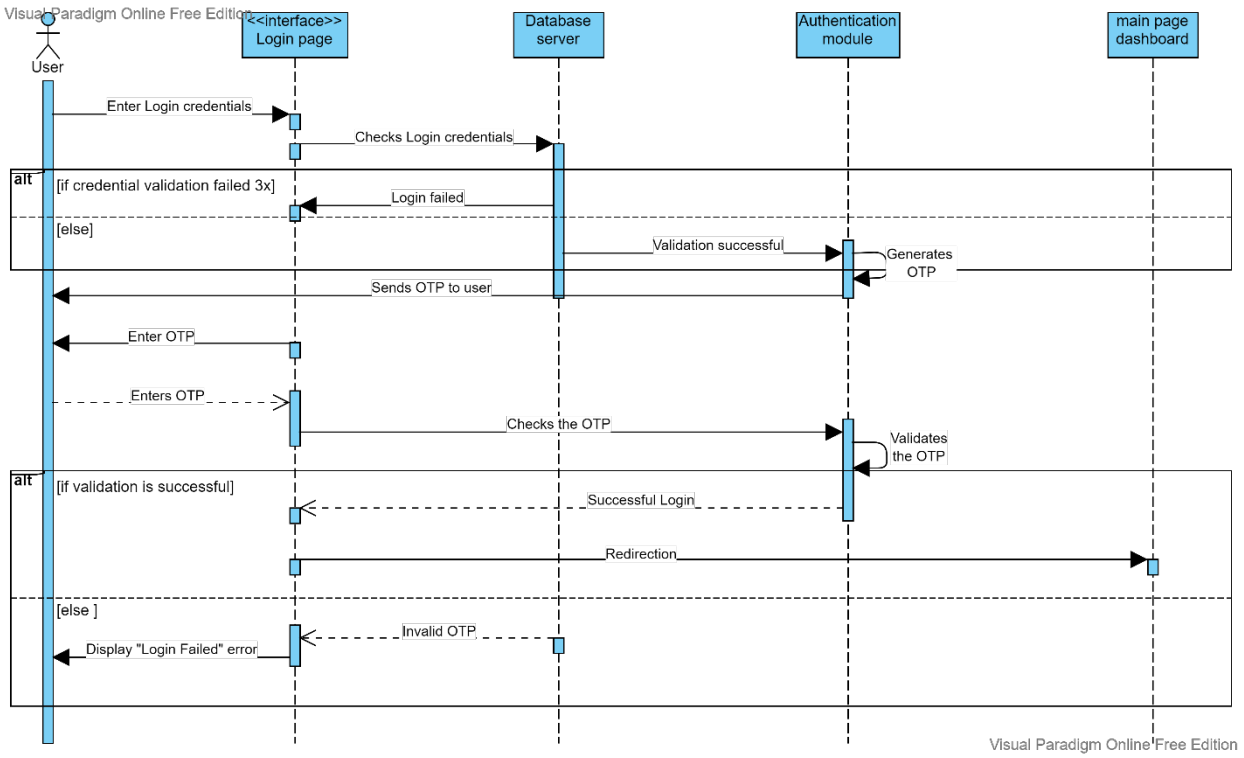



Dutch Police Internet Forensics System
System Administrator
Manage Users and Permissions
Manage Backups and Logs
System Maintainance
Manager
Police Officer
Junior Security Analyst
Senior Security Analyst
Manage data (upload, download and share data)
View Incident reports
Setting rules to search, track and locate criminal activities
Successful Login into the Dashboard
Generate and Manage Reports
Manage Audits
Review cases and provide statuses and feedbacks
Manage data (upload, download, share and modify data)
<<Include>>

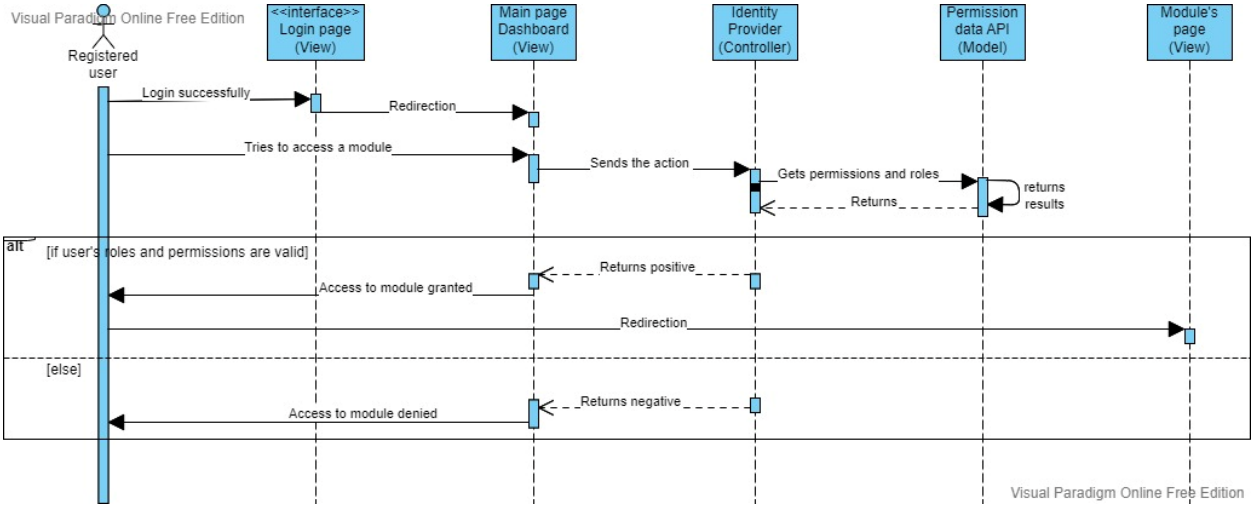

act [Activity Diagram2]
Registered User
Enter Login credentials
Password encryption/ hashing
Valid user?
No
Yes
No
Send OTP
Enter OTP
Valid OTP && entered in >30sec?
No
Yes
Attempts<3?
No
Yes
Account locked
User authenticated
Access to the dashboard is granted
End


act [Activity Diagram1]
Registered User
Login
(blocks after 3 failed trials)
No
Login authentication successful?
No
(validation is time-sensitive)
Yes
Multi-factor authentication successful?
Yes
Logs-in to the Dashboard and Checks User's Access Controls (Roles and Permissions)
checks permission
Yes
System Management and Maintainance
checks permission
Yes
Manage data (upload, download, share and modify data)
checks permission
Yes
Generate and manage reports
checks permission
Yes
Manage Audits
checks permission
Yes
Review cases and provide statuses and feedbacks
checks permission
Yes
Setting rules to search, track and locate criminal activity
checks permission
Yes
Search and View incident reports
Logs-out of the system
End


<default package>
Registration Information
-registrationID:int
+firstName:string
+lastName:string
+yearOfBirth:int
+email:string
+address:string
+create()
+update()
Login Information
-loginID:int
-accountNumber:int
-username:string
-password:string
+create()
+validate()
+update()
Logs
-logs:int
+userAccount:int
+userType:string
+loginTime:string
+logoutTime:string
+loginDevice:string
+print()
User Information
-userID:int
+add()
+delete()
+lock()
User Role
-typeID:int
+firstName:string
+lastName:string
+userRole:string
+create()
+update()
Activities
-activityID:int
-description:string
date:string
+create()
Permissions
-permitID:int
+permitRole:string
+permitTitle:string
+permitModule:string
+permitDesription:string
+create()
+update()
+search()
+lock()
<<enumeration>>
userRole
system administrator
manager
senior security analyst
junior security analyst
officer



Sequence diagram showing a user who is trying to access one of the modules after successfully logging in to the main page dashboard



References

Dauzon, S., Bendoraitis, A. & Ravindran, A., 2016. *Django: Web Development with Python.* Mumbai: Packt Publishing Ltd.

Lenka, C., 2020. *Python program check validity Password.* [Online]
Available at: https://www.geeksforgeeks.org/python-program-check-validity-password/
[Accessed 28 May 2022].

Melé, A., 2020. *Django 3 by example.* 3rd ed. UK: Packt Publishing Ltd.

Oliphant, T. E., 2007. Python for Scientific Computing. *Computing in Science and Engineering,* 9(3), pp. 10-20.