Table of Contents

Total word count (excluding references, but including figures' captions): **1,092 words**

1. Introduction

The Dutch Police Internet Forensics is a police unit that allows investigators to perform specialised, forensically stored searches on the Internet to obtain evidence for possible prosecution (Henseler & van Loenhout, 2018). The team's main objective is to develop a secure application for the Dutch Police unit, which complies with the General Data Protection Regulation (GDPR) (European Data Protection Regulation, 2018) to ensure personal sensitive data are protected (Chopade & Pachghare, 2019), thus allowing authorised stakeholders to manage their data whilst protecting their privacy by design.

2. System's main requirements

As per the ISO/IEC Standard 27000 document (ISO, 2018) and the Open Web Application Security Project's (OWASP, 2021) industry-grade security standards, the system will be implemented as a RESTful API (Patni, 2017) and must:

- implement strict user authentication and authorisation policies, based on which some or all create/read/update/delete (CRUD) permissions are granted, as well as data validation, to preserve confidentiality, integrity, non-repudiation, and reliability (ISO, 2018).
- operate over secure web connections, leverage appropriate encryption and decryption, and enable monitoring of user activities via systems' audit logs, thus preventing 'cryptographic failures' (OWASP, 2018, 2021).

- fulfil a user role-based and hierarchical authentication model by denying by default and adhering to the least privileges principle (Sanders & Yue, 2019; OWASP, 2021).
- Enforce Application programming interface (API) rate limit to prevent distributed denial-of-service and controller access.

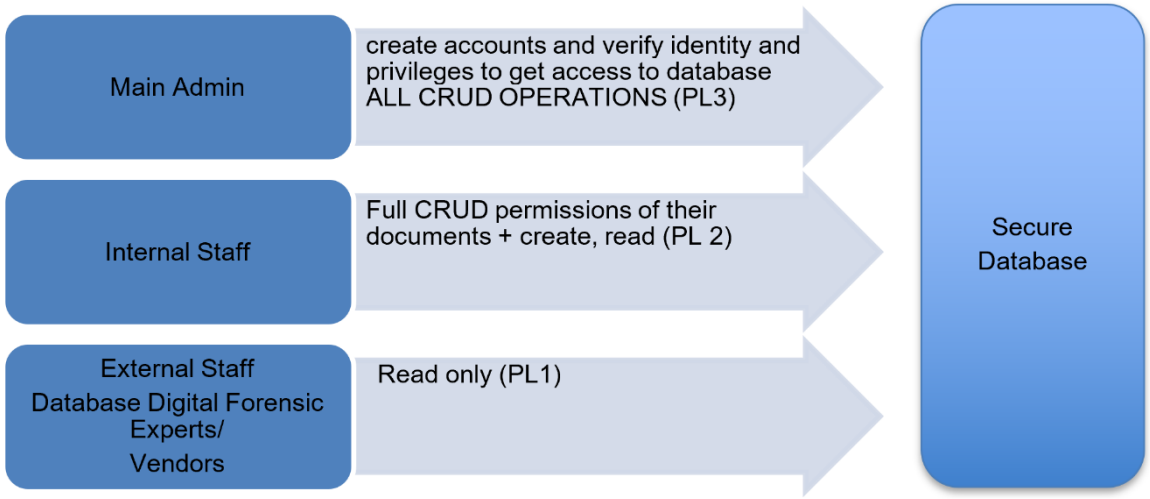
3. Development Methodology

The system will be developed and delivered via the Agile methodology (Srivastava et al, 2017), thus producing a shippable product iteratively and performing quality assurance and user acceptance testing throughout the software development life cycle (Srivastava et al, 2017).

3.1 Assumptions and Users

- Only authorised users can access and securely manage the data in the system.
- The application's users are: the main administrators, internal and external staff, who investigate digital and cybercrimes (hacking, identify thefts, data breaches, etc.).
- As per the least privilege principle (Sanders & Yue, 2019) and as per **Fig. 1**, only administrators have full control of the system, whilst all others need a specific level of access to identify, collect, and analyse potential evidence stored in the application's database (Sindhu & Meshram, 2012). In particular, external staff has limited access to the database (**Fig. 1**).
- The "rights to be forgotten" or the rights to erasure of data upon request and after the end of a contract as per the GDPR (European Data Protection Regulation, 2018) are strictly adhered to.

- Sensitive data on cybercrimes are stored at the Dutch Police Internet Forensics. Users have access to various levels of confidential information based on their permissions.



Note: PL = Permission Level

Figure 1. Users' roles' descriptions.

4. Security

The application-related security challenges were identified leveraging the following paradigms: the STRIDE (Seale et al., 2018) and the DREAD models (Ramachandran, 2016), and the OWASP (2010) security coding principles. As shown in **Fig. 2**, the STRIDE model includes the key threats and the security features to prevent them (Seale et al., 2018); among these threats, spoofing, tampering, information disclosure, and elevation of privilege (Pandi et al., 2020) are particularly important considering the use case of the Dutch Police and potential legal and financial consequences of breaching the European Union (EU) "General Data Protection Regulation" (GDPR) (Tankard, 2016).

Threat	Desired Security Property
Spoofing	Authentication
Tampering	Integrity
Repudiation	Non-repudiation
Information Disclosure	Confidentiality
Denial of Service	Availability
Elevation of Privilege	Authorization

Figure 2. The STRIDE model (Seale et al., 2018).

Moreover, the DREAD model (**Fig. 3**) was leveraged to quantify the security-related risk via rating five main threats, i.e., damage to the business and the stakeholders involved, reproducibility of the attack, exploitability, affected users, discoverability of the vulnerability. In line with the significant impact of security vulnerabilities on the organisation and the affected users (Pandi et al., 2020), considering the potential damage was used to ensure security by architectural design, via appropriate user authentication and GDPR-compliant (Tankard, 2016) authorisation.

Figure 3. The DREAD model (EC-Council, 2022).

As aligned with the STRIDE model (Seale et al., 2018), according to the OWASP's (2010) security principles, appropriate security checks, e.g., verifying roles, permissions, and validating user inputs, will be implemented in the software to preserve the confidentiality, integrity, and availability of the users' data (Pandi et al., 2020) whilst allowing the Dutch Police Internet Forensics unit to perform its operations.

5. Unified Modelling Language

The **activity diagram** of the proposed design flows from a user's perspective, including registration and login, is illustrated in **Fig. 4**, which provides a general view of the application's expected usage whilst complying with security directives, including GDPR (**Fig. 4**).

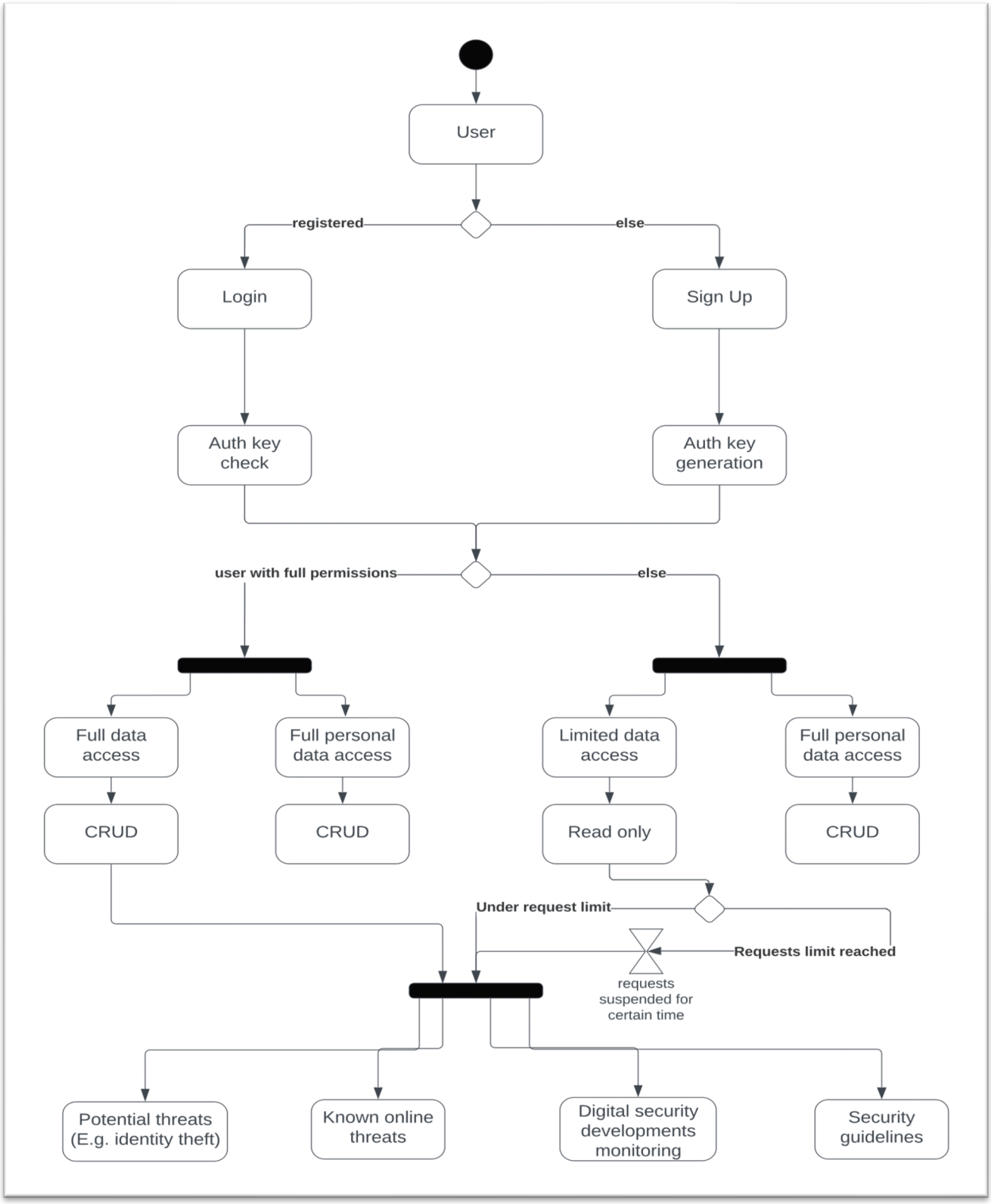


Figure 4. The activity diagram including user's registration, login, permissions-based access and CRUD operations.

The **use case diagram** in **Fig. 5** provides a high-level view of the relationships between the agents, e.g., two authorised users with different permissions, and the system, as aided by the system's main requirements, whilst complying with GDPR.

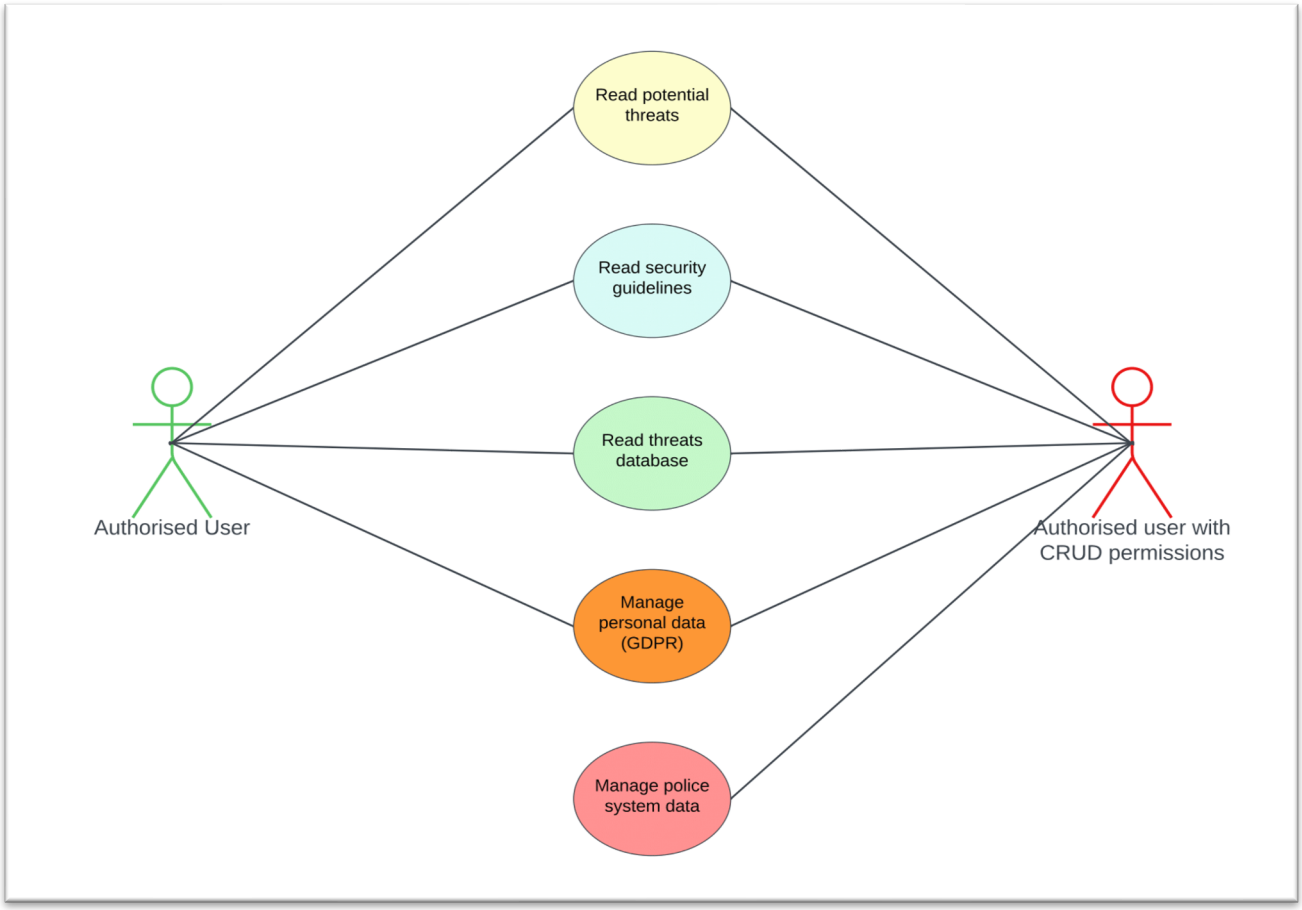


Figure 5. The use case diagram with the relationships between agents and the application.

A **sequence diagram** describes the interactions between various phases in order and is a lower-level diagram with respect to the use case diagram (Bell, 2003, 2004). **Figs. 6** and **7** describe how a regular user and another with internal permissions respectively can log in and interact with the appropriate data as per GDPR, highlighting the security measures in place.

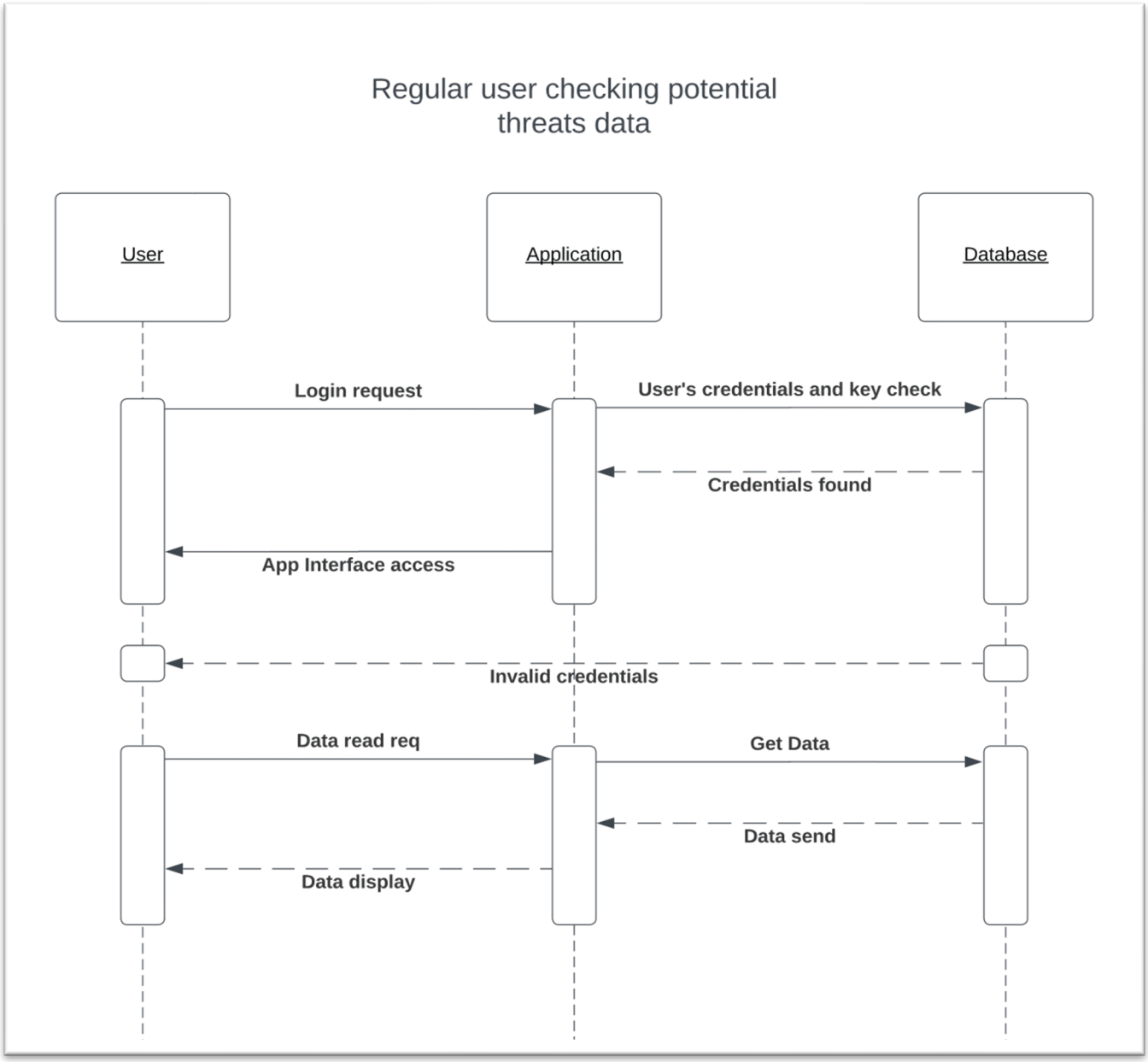


Figure 6. A sequence diagram of the operations of a regular user in the application.

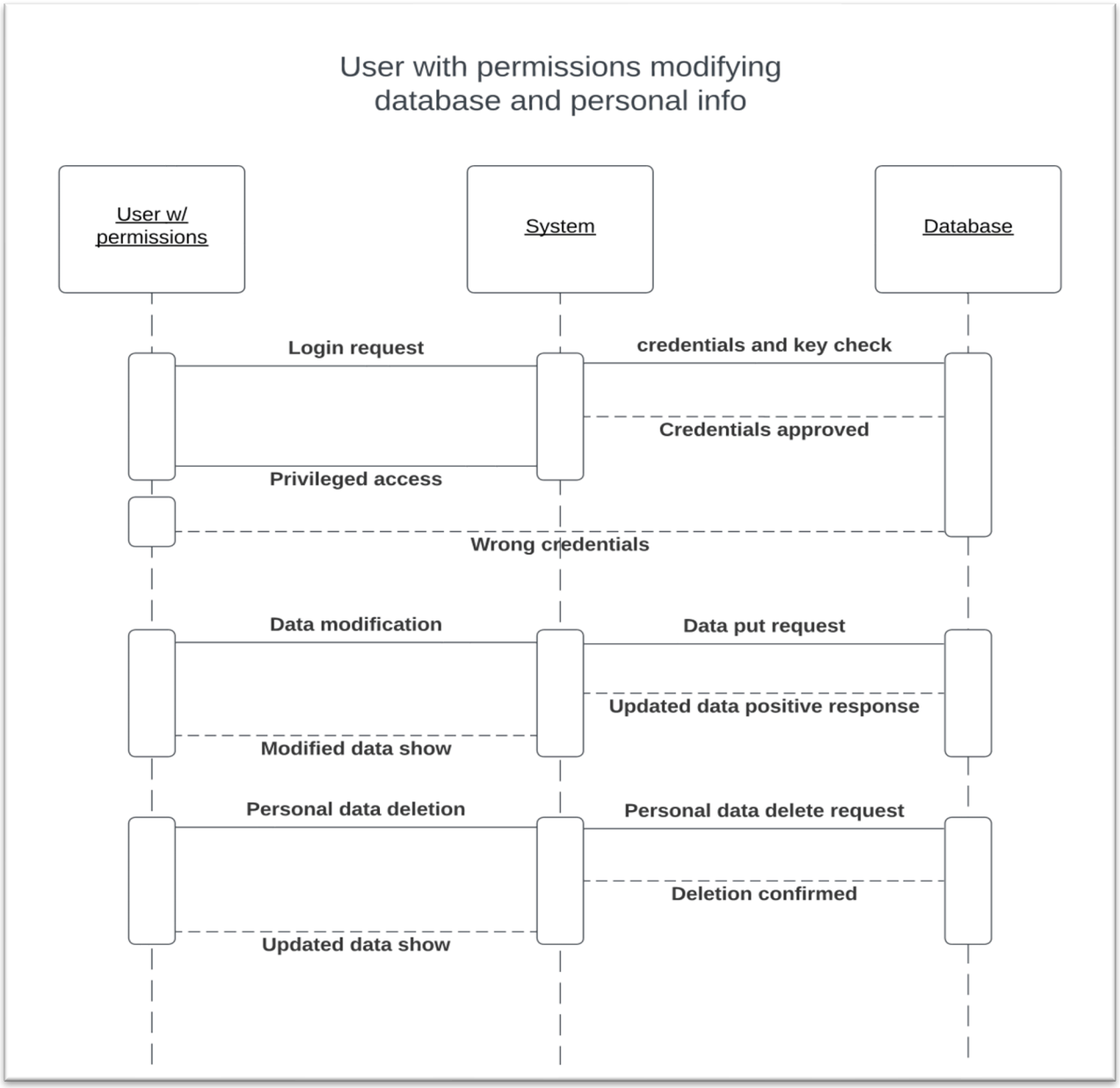


Figure 7. A sequence diagram of the operations of a user with CRUD permissions in the application.

6. Tools and Libraries

The main tools and their purposes are listed below:

- Primary Development Language: Python 3.x.
- Database: SQL.
- Source Control and Versioning: GitHub.
- IDEs (considering all team members' preferences): Atom/VS Code/PyCharm.

The main libraries are described below, along with their purpose and rationale:

- Data Encryption: python-secrets
    - This library can manage groups of secrets, e.g., passwords, encryption keys, etc. (Dittrich, 2022).
- Interface: click
    - This library can help to create a command line interface (CLI)-based application via entry points, also handling authentication via the 'password_option()' decorator and user inputs' validation (Pallets, 2022a).
- Database Implementation: pyodbc
    - This will be used as an Open Database Connectivity (ODBC) connection to and management of the database (Kleehammer, 2022).
- REST API development: flask
    - The 'flask' library can help with creating a RESTful microservice in Python (Pallets, 2022b).
- Security: safety
    - The 'safety' library can scan the dependencies in Python and flag any security vulnerabilities (pyup.io, 2022).
- Quality, Validation and Verification: Pytest, Pylint
    - These libraries can help with creating unit tests to validate the system's functionality (PyCQA, 2022; pytest-dev, 2022).
- Documentation: Swagger
    - This library can create interactive documentation of the main CRUD operations allowed (Swagger, 2022).

References

Bell, D. (2003) An introduction to UML language, IBM. Accessed on July 15th at https://developer.ibm.com/articles/an-introduction-to-uml/.

Bell, D. (2004) Explore the UML sequence diagram, IBM. Accessed on July 15th at https://developer.ibm.com/articles/the-sequence-diagram/.

Chopade, R. & Pachghare V.K. (2019) Ten years of critical review on database forensics research. *Digital Investigation*. Accesed on July 18th, 2022, at https://doi.org/10.1016/j.diin.2019.04.001.

Dittrich, D. (2022) python-secrets: Python CLI for managing secrets and eliminating default passwords in FOSS. *GitHub*. Accessed on July 24th, 2022 at https://github.com/davedittrich/python_secrets.

EC-Council (2022) Cyber Threat Modeling. Accessed on July 6th, 2022, at https://www.eccouncil.org/threat-modeling/.

European Data Protection Regulation (2018) (EU) 2016/679 General Data Protection Regulation (GDPR). Accessed on July 18th, 2022, at https://gdpr-info.eu/.

Henseler, H., & van Loenhout, S. (2018) Educating judges, prosecutors and lawyers in the use of digital forensic experts. *Digital Investigation* 24: S76-S82.

ISO (2018) ISO/IEC 27000:2018: “3 Terms and Definitions”. Accessed on July 18th, 2022, at https://www.iso.org/obp/ui/#iso:std:iso-iec:27000:ed-5:v1:en.

Kleehammer, M. (2022) pyodbc: Python ODBC bridge. *GitHub*. Accessed on July 24th, 2022 at https://github.com/mkleehammer/pyodbc.

OWASP (2010) OWASP Secure Coding Practices. Accessed on July 6th, 2022, at https://owasp.org/www-pdf-archive/OWASP_SCP_Quick_Reference_Guide_v1.pdf.

OWASP (2018) C7: Enforce Access Controls. Accessed on July 18th, 2022, at https://owasp.org/www-project-proactive-controls/v3/en/c7-enforce-access-controls.

OWASP (2021) OWASP Top Ten Web Application Security Risks. Accessed on July 18th, 2022, at https://owasp.org/www-project-top-ten/.

Pallets (2022a) click: Python composable command line interface toolkit. *GitHub*. Accessed on July 24th, 2022 at https://github.com/pallets/click.

Pallets (2022b) flask: The Python micro framework for building web applications. *GitHub*. Accessed on July 24th, 2022 at https://github.com/pallets/flask.

Pandi, G. S., Shah, S., & Wandra, K. H. (2020) Exploration of Vulnerabilities, Threats and Forensic Issues and its impact on the Distributed Environment of Cloud and its mitigation. *Procedia Computer Science* 167: 163-173.

Patni, S. (2017). Pro RESTful APIs. Apress.

PyCQA (2022) pylint: It's not just a linter that annoys you! *GitHub*. Accessed on July 24th, 2022 at https://github.com/PyCQA/pylint.

Pytest-dev (2022) pytest: The pytest framework makes it easy to write small tests, yet scales to support complex functional testing. *GitHub*. Accessed on July 24th, 2022 at https://github.com/pytest-dev/pytest.

pyup.io (2022) safety: Safety checks Python dependencies for known security vulnerabilities and suggests the proper remediations for vulnerabilities detected. *GitHub*. Accessed on July 24th, 2022 at https://github.com/pyupio/safety.

Ramachandran, M. (2016) Software security requirements management as an emerging cloud computing service. *International Journal of Information Management* 36(4): 580-590.

Sanders, M. & Yue, C. (2019) Mining Least Privilege Attribute Based Access Control Policies. In *2019 Annual Computer Security Applications Conference (ACSAC '19)*, December 9–13, San Juan, PR, USA. ACM, New York, NY, USA.

Seale, K., McDonald, J., Glisson, W., Pardue, H., & Jacobs, M. (2018) MedDevRisk: Risk Analysis Methodology for Networked Medical Devices. In *Proceedings of the 51st Hawaii International Conference on System Sciences*.

Sindhu, K. K. & Meshram, B. B. (2012) Digital Forensics and Cyber Crime Datamining. *Journal of Information Security* 3(3).

Srivastava, A., Bhardwaj, S., & Saraswat, S. (2017, May). SCRUM model for agile methodology. In 2017 International Conference on Computing, Communication and Automation (ICCCA) (pp. 864-869). IEEE.

Swagger (2022) Swagger UI: Swagger UI is a collection of HTML, JavaScript, and CSS assets that dynamically generate beautiful documentation from a Swagger-compliant API. *GitHub*. Accessed on July 24th, 2022 at https://github.com/swagger-api/swagger-ui.

Tankard, C. (2016) What the GDPR means for businesses. *Network Security*, 2016(6): 5-8.