580 Words

Team Strategy

Shota Kameyama  
Mathew Van Beek  
Nils Linhoff  
Mahamad Ibrahim  
Muhammad Nasim Akbary

SECURE MAINTENANCE REPOSITORY CERN

Design Document

**Contents**

[Problem Description / Brief 1](#_heading=h.gjdgxs)

[Assumptions 2](#_heading=h.tyjcwt)

[System Requirements 2](#_heading=h.1t3h5sf)

[Design Decisions 3](#_heading=h.4d34og8)

[Approach to the Solution 3](#_heading=h.2s8eyo1)

[Security Risks & Vulnerabilities 3](#_heading=h.17dp8vu)

[Tools & Methods 4](#_heading=h.3rdcrjn)

[Appendix: 5](#_heading=h.2jxsxqh)

[Additional tools and libraries: 6](#_heading=h.1y810tw)

[UML Design of A Secure Maintenance Report Repository System 6](#_heading=h.4i7ojhp)

[References: 9](#_heading=h.2bn6wsx)

Table 1: Requirements 2

[Figure 1: Topology of the system 5](#_heading=h.z337ya)

[Figure 2: Draft tool landscape 5](#_heading=h.3j2qqm3)

[Figure 3: Sequence Diagram of A Secure Maintenance Report Repository System 6](#_heading=h.2xcytpi)

[Figure 4: Class Diagram of A Secure Maintenance Report Repository System 7](#_heading=h.1ci93xb)

[Figure 5: Activity Diagram of A Secure Maintenance Report Repository System 8](#_heading=h.3whwml4)

## **Problem Description**

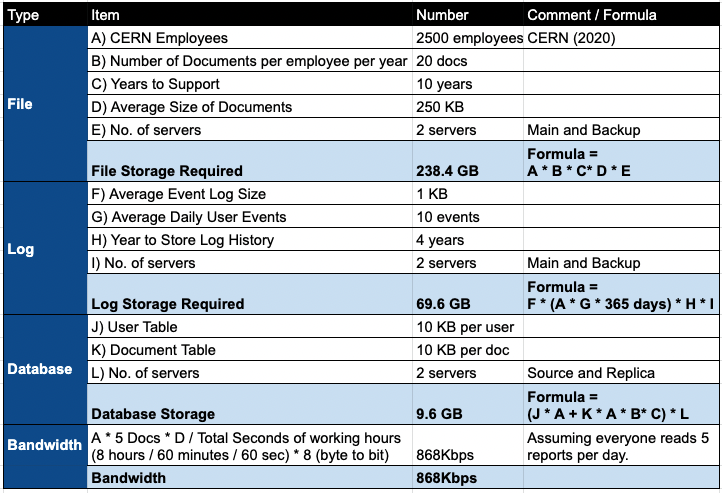
CERN (The European Organization for Nuclear Research) is the world's largest particle physics laboratory with over 2500 employees (CERN, 2020). Numerous experiments are conducted in its laboratories, and it has produced over 19 petabytes of scientific data in 2016. Data necessary for the operation of CERN needs to be stored in secured repositories. This includes maintenance reports of the facility.

## **Assumptions**

* CERN requires continuous operations and maintenance over the year (Serio et al., 2015), considering the necessity of secure storage when storing maintenance reports to relevant CERN employees.

## **System Requirements**

* Figure 1 illustrates the system topology. The system should be open web system to access with browsers with any devices from different locations of CERN's cryogenic installations located at eight different locations over the Swiss-French country border (Serio et al., 2015). The datacenter should be either in France or Switzerland.
* A minimum of 2 CPUs per microservice for concurring events and 4 CPUs for source and replica Databases for more frequent events expected.
* 4GB RAM for each server is sufficient as no complicated action is required.
* Data Storage can store in one document management microservice server, potentially extending to cloud storage if the storage capacity chokes.
* Table 1 illustrates the storage capacities required and their calculation formula to support 2,500 employees (CERN, 2020), which required 250GB for document servers, 100GB for log servers, 10GB for Databases, 1Mbps for average bandwidth.



*Table 1: Requirements*

## **Design Decisions**

* Flask, a quick result-oriented microframework designed for small to large web applications (Ghimire, 2020; Lokhande et al., 2015).
* Argon2id for password hashing that OWASP (2021) recommends most and TLS 1.3 for network encryption.
* MySQL databases.
* RESTful API to communicate between microservices.

## **Approach to the Solution**

* Agile approach to enhance development speed, flexibility and efficiency (Arachchi & Perera, 2018; Fowler and Highsmith, 2001)

## **Security Risks & Vulnerabilities**

Our software encompasses a front-facing website and back-end SQL database, we’ve used combinations of STRIDE and OWASP frameworks to determine risks and vulnerabilities.

* SQL Injection – Allows attackers to spoof identity and tamper with existing data causing it potentially disclosed, destroyed, or made otherwise unavailable (OWASP top ten, 2021).

Mitigations: prepare statements, review code, test for vulnerabilities (OWASP, 2021).

* Content Spoofing – Allows attackers to inject malicious payload, later misrepresented as a legitimate web application (OWASP, 2021).

Mitigations: validate data input, use temporary sessions, encode any user input presenting output by the application (OWASP, 2021).

* Denial of Service – Renders a system unreachable by overloading it with requests (OWASP, 2021).

Mitigations: location and source blocking, display CAPTCHA, behaviour and pattern blocking, traffic scrubbing (CERTNZ 2019).

* Brute Force – audacious guessing attempts to gain user accounts access (OWASP, 2021).  
    
  Mitigations: change default passwords, implement password management process, regular change intervals (Fortinet, 2018).
* Privilege Escalation – Unwarranted high-level access given to legitimate users (OWASP, 2021).  
    
  Mitigations: zero trust policy, grant user access rights when needed (Wadhwa, 2019).
* Unpatched Database –Software that is not successfully patched leaves opportunity for hackers to infiltrate (OWASP, 2021).

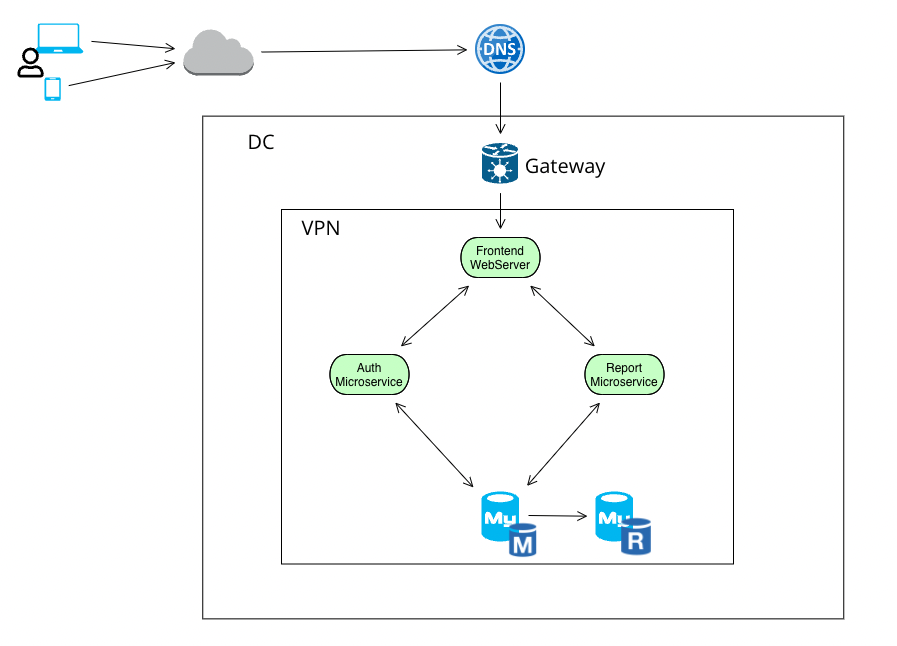
Mitigations: implement vulnerability database management schedule, follow current vulnerabilities (Wadhwa, 2019).

## 

## **Tools & Methods**

* Libraries:
  + - * Database:PyMySQL, a MySQL client library
      * Authentication and security:
        + Flask-Login, session management
        + Flask-Principal, identity management
        + Flask-WTF, form validation
        + Flask-RESTful, API handling for building microservices
        + Itsdangerous, key assignment management
        + Argon2id for password hashing and TLS version 1.3 for network.
* Test:
  + - * Pytest, a framework to write tests
      * Pyre and Bandit as SAST
      * flake8 as Code Guide Enforcement

Appendix:



*Figure 1: Topology of the system*



*Figure 2: Draft tool landscape*

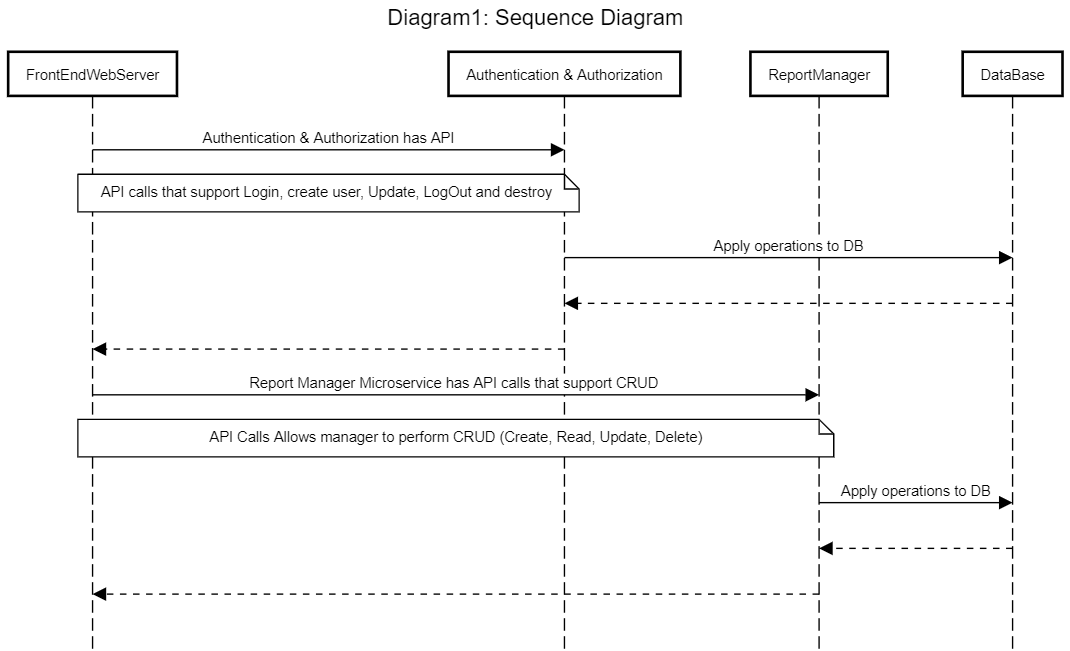
### **Additional tools and libraries:**

These tools will be used if the need arises during development:

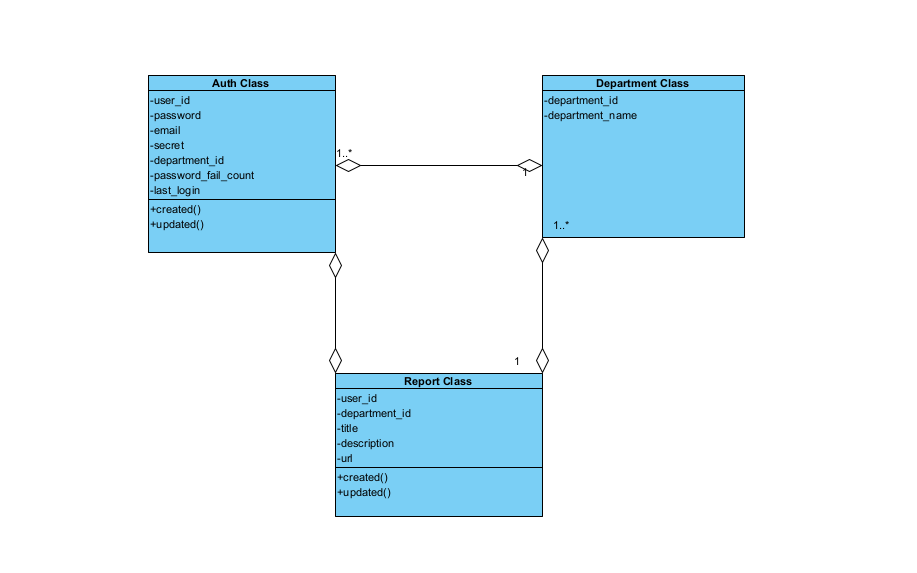
* Pony ORM
* mccabe
* flask security:
  + - cryptography
    - pyqrcode
* PyYAML
* defusedxml
* PeeWee

### **UML Design of A Secure Maintenance Report Repository System**

To depict the behaviour and the interactions between the components of our Repository System we use sequence diagrams. Sequence diagram is a dynamic diagram that is essentially designed to display the interaction between objects of a system in a sequential manner (Bill, 2004), in our case we show the interaction between the front-end web server, Authentication and authorization microservice, report manager microservice as database. (Figure 3)

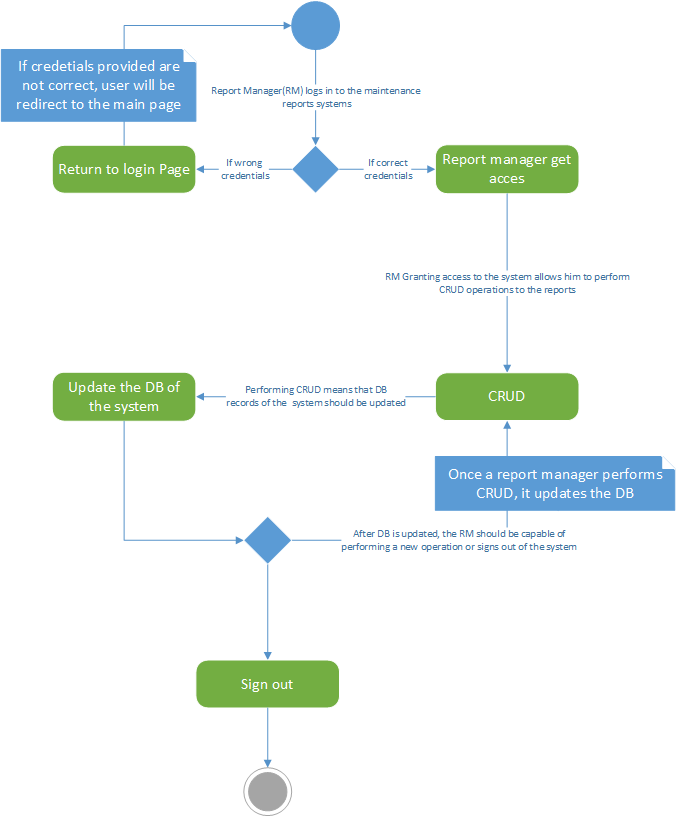
****

*Figure 3: Sequence Diagram of A Secure Maintenance Report Repository System*

The Maintenance Report Repository System consists of three classes for each microservice it has. An authentication class, department class and a report class.Figure 4 displays The class diagram of our system shows how the classes are structured to form the whole architecture of the system. All classes are connected to each other using association relationships which form the communication among classes (Ali et al. 2007).

*Figure 4: Class Diagram of A Secure Maintenance Report Repository System*

According to (Bell, 2004) activity diagram is a less technical diagram when it is compared to the other UML diagrams, Figure 5 shows a high-level scenario of performing CRUD operations by a report manager of our system.



*Figure 5: Activity Diagram of A Secure Maintenance Report Repository System*

## **References:**

1. Ali, N. Shukur, Z. Idris, S.(2007) Assessment system for UML class diagram using notations extraction. *International Journal on Computer Science Network Security* 7(8):181-187. Available from: <https://www.researchgate.net/profile/Zarina-Shukur/publication/253243639_Assessment_System_For_UML_Class_Diagram_Using_Notations_Extraction/links/55487af30cf2b0cf7acec2e4/Assessment-System-For-UML-Class-Diagram-Using-Notations-Extraction.pdf> [Accessed 8 February 2022 ]
2. Arachchi, S. A. I. B. S., & Perera, I. (2018) 'Continuous integration and continuous delivery pipeline automation for agile software project management', *2018 Moratuwa Engineering Research Conference (MERCon)*. Moratuwa, Sri Lanka, 30 May-1 June. IEEE. 156-161. DOI: 10.1109/MERCon.2018.8421965
3. Bell, D. (2004) *UML basics: The sequence diagram.* Available From: <http://www.csun.edu/~twang/380/Slides/SequenceDiagram.pdf> [Accessed 8 February 2022 ]
4. Bell, D. (2004) *UML basics : An introduction to the Unified Modeling Language .* Available From: <https://shortest.link/34TZ> [Accessed 8 Feb 2022 ]
5. CERN (2020) Careers at CERN. Available from: https://careers.cern/ [Accessed 6 February 2022].
6. CERTNZ (2019) Mitigating denial-of-service attacks Available from: https://www.cert.govt.nz/it-specialists/guides/mitigating-denial-of-service-attacks/ [Accessed February 10 2022]
7. Dobing, D. Parsosn, J. (2006) How UML is used. *COMMUNICATIONS OF THE ACM* 49(5):109-111. DOI:<https://doi.org/10.1145/1125944.1125949>
8. Fortinet (2018) What is a Brute Force Attack? Available from: https://www.fortinet.com/resources/cyberglossary/brute-force-attack [Accessed February 8 2022]
9. Fowler, M. & Highsmith, J. (2001) ‘The agile manifesto’, *Software development* 9(8).
10. Ghimire, D. (2020) Comparative study on Python web frameworks: Flask and Django. Available from: https://www.theseus.fi/bitstream/handle/10024/339796/Ghimire\_Devndra.pdf?sequence=2. [Accessed 6 February 2020].
11. Lokhande, P. S., Aslam, F., Hawa, N., Munir, J., & Gulamgaus, M. (2015) Efficient way of web development using python and flask. *International Journal of Advanced Research in Computer Science* 6(2): 54-57.
12. OWASP (2021) Password Storage Cheat Sheet. Available from: https://cheatsheetseries.owasp.org/cheatsheets/Password\_Storage\_Cheat\_Sheet.html [Accessed 6 February 2022].
13. OWASP (2021) Top ten web application security risks. Available from: https://owasp.org/www-project-top-ten/ [Accessed February 1 2022]
14. OWASP (2021) A04:2021 – Insecure Design. Available from https://owasp.org/Top10/A04\_2021-Insecure\_Design/ [Accessed February 8, 2022]
15. Serio, L., Bremer, J., Claudet, S., Delikaris, D., Ferlin, G., Pezzetti, M., Pirotte, O., Tavian, L. & Wagner, U. (2015) 'CERN experience and strategy for the maintenance of cryogenic plants and distribution systems', Cryogenic engineering conference (CEC). Tucson, Arizona, 28 June-2 July. International Atomic Energy Agency (IAEA): IOP Publishing. 1-8. DOI: 10.1088/1757-899X/101/1/012140
16. Wadhwa, M. (2019) How to think about security and threats in your distributed application. Available from: https://www.ockam.io/learn/blog/introduction\_to\_STRIDE\_security\_model [Accessed February 10 2022]