*SSDCS\_PCOM7E March 2024* - *Development Team Project: Design Document*

**Bulwark Systems © SciTec App for the International Space Station (ISS)**

*Word Count: 912*

**Authors**

Gareth Williams

Mario Butorac

Miguel Maide Bezares

Patricia Santos

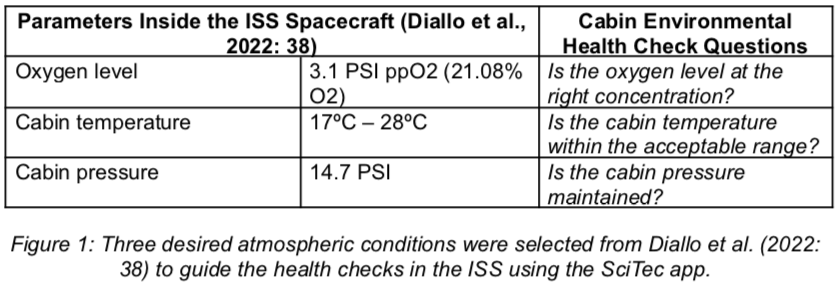
**Introduction**

**International Space Station (ISS)** is a crewed space programme spearheaded by The National Aeronautics and Space Administration (NASA) of the United States in 1984, alongside the European Space Agency, Japan, and Canada, with Russia joining in 1993 (The National Aeronautics and Space Administration, 2023). The programme was established for aerospace research and progressing various fields such as technology (Public Broadcasting Station, N.D.).

Attacks on space systems can **impede worldwide security and economies** since many aspects of everyday human life, like transmission and travel, count on them (King & Goguichvilli, 2020). Wells (2020) argues that cyber assaults on space, such as manipulating and pilfering findings, can **tarnish the reputation of both NASA and ISS**, and negatively affect their financing and trust ratings. Hence, it is important for the ISS to invest in updated cyber security systems that adhere to worldwide standards.

**Creating a Secure Application**

**Bulwark Systems** will create a secure application called **SciTec.** It will enable scientists to perform environmental health checks from inside the spacecraft cabin and simultaneously relay data to the space stations on Earth. Using the desired ISS atmospheric conditions from Diallo et al. (2022), questions will be addressed, as seen in Figure 1.

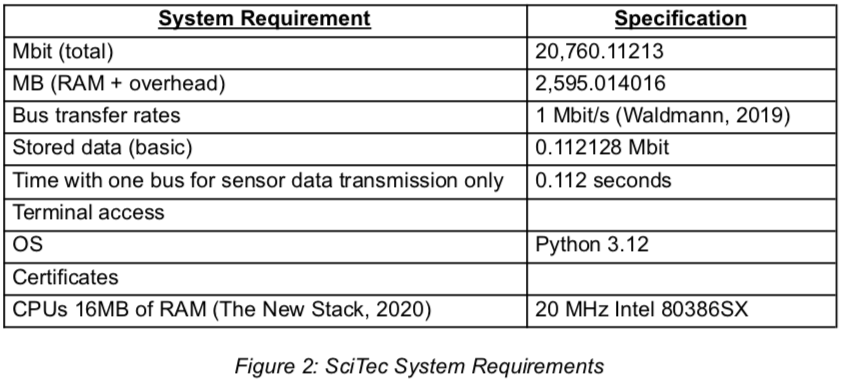


**Rationale**

Pipikaite et al. (2022) posits that space-based services are prone to technological threats, and calls for the strengthening of cybersecurity of celestial facilities. SciTec will be designed to **defend against API Injections, Brute Force Attacks,** and **Denial of Service.** Attackers are possibly fully equipped to avert security protocols once they enter digital space apparatus and execute clandestine cyber attacks that go unnoticed for a long time (Oakley, 2020). Thus, SciTec will limit access to **authorised users–** the **scientists** and **space station administrators.**

**System Requirements**

Memory overhead is rather difficult to predict, but a feasible worst case scenario of 20% as per Rostedt (2022) was selected. The memory profiler library was used to estimate the size of complex functions, yielding 50 Mb each. However, the size of simple data structures were estimated using the *getsizeof()* method from the *sys module* in Python.

**

**Development Approach**

* *Design Document*

During the design development stage, the **agile** methodology that consists of **four Scrum ceremonies** (Radigan, N.D.) will be performed by the group, with some minor changes as the team is meeting virtually:

* ***Sprints*** that last two weeks each.
* ***Instead of daily stand-up meetings, constant communication*** via WhatsApp.
* ***Sprint reviews*** given during meetings and document feedback comments.
* ***Sprint retrospective*** will focus on team dynamics, issues, and how to improve the collaboration.
* *Coding and Testing Output*

**Scrum Solo** will be adapted for the individual coding requirement, where a single developer covers all tasks usually distributed within team members during Scrum (Moyo & Mnkandla. 2020). Brito et al. (2020) adds that Scrum Solo utilises both ‘Personal Software Process’ and ‘Scrum’, and follows the activity of:

* ***Stating requirements***
* ***Sprints and project management throughout the development***
* ***Deployment***

**Design Decisions**

* *Architectural Pattern*

The structure of the application is a **Model View Controller (MVC),** which is organised in the following according to Pillai (2017):

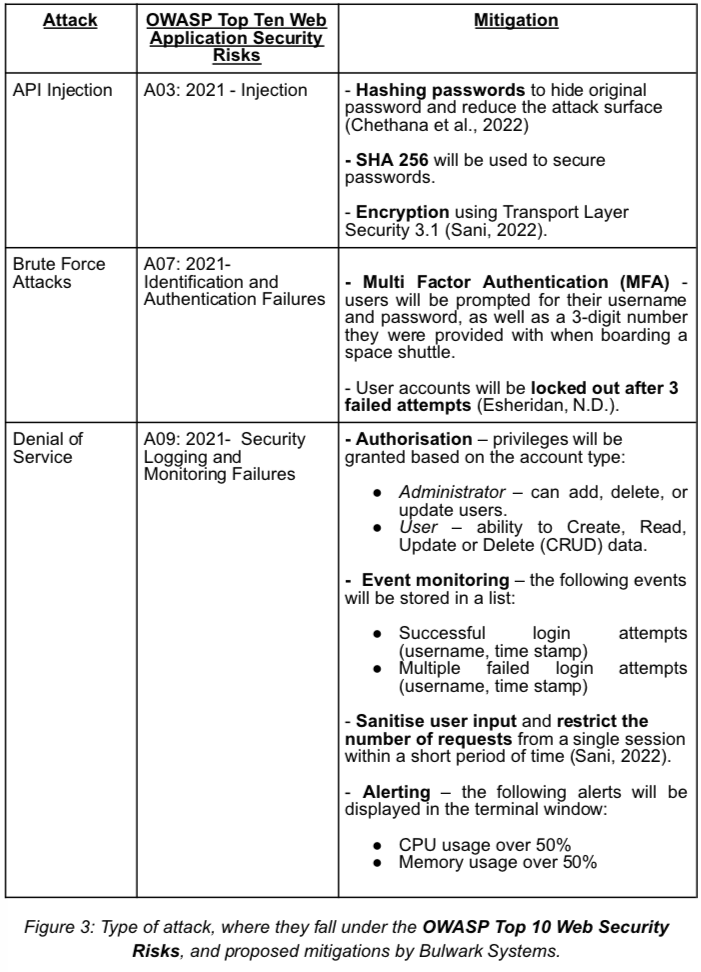
* ***Model*** – the backend which contains data and processing logic.
* ***View*** – the front end that presents information to users, and for this application it will be a Command Line Interface.
* ***Controller*** – processes user input and updates the Model or View. This application will count the number of unsuccessful logins, and after each failed attempt, it will notify both the Model and the View.
* *Data Structures*

All data will be stored within the application in the following:

* ***Dictionary*** – user account details will be stored using *‘key:value’* format.
* ***List*** – all environment health check results will be stored in their respective lists.
* ***Tuple*** – for hashed passwords.

**Security Risks and Vulnerabilities**

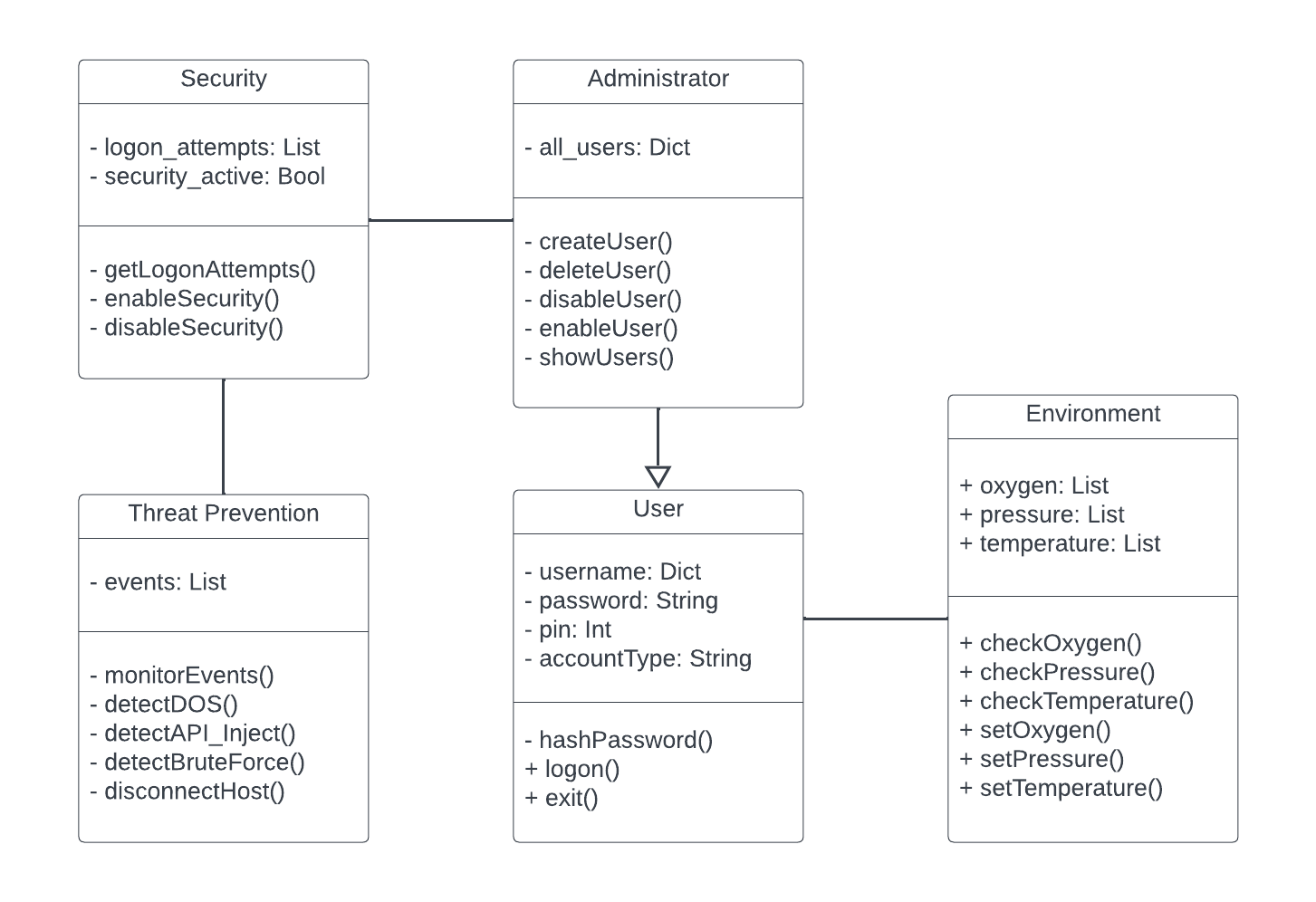
In Figure 7, SciTec will refer to the **protocols provided** by the **Open Web Application Security Project**, **(OWASP)** for the identification and mitigation of threats during cyber security development (Open Web Application Security Project, N.D.)

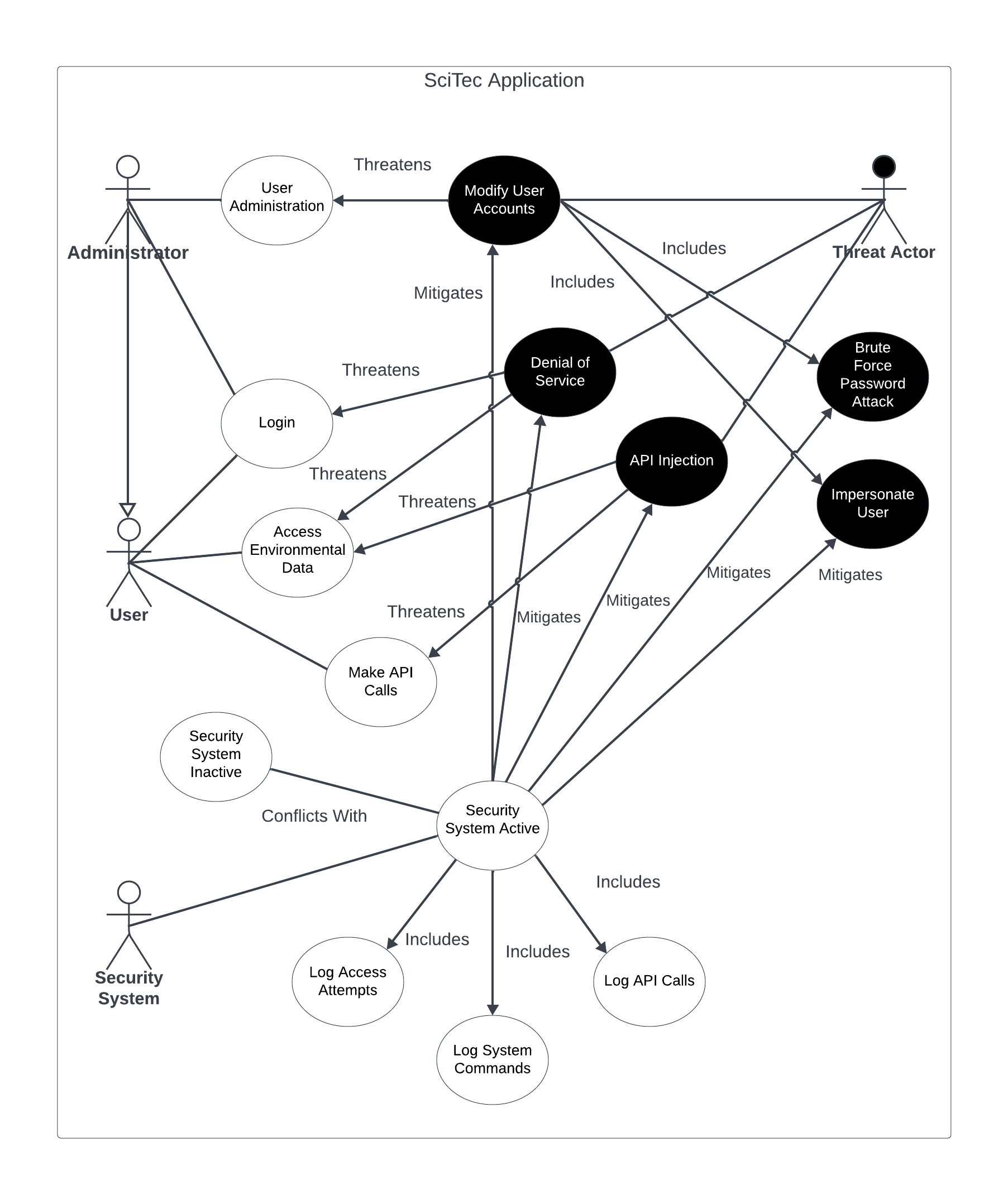
**

To facilitate the requested hacker requirements, the application will provide an option to be run with security features enabled or disabled.

**UML Designs**

The following class and misuse case diagrams describe the data structures and functions of the application, and the security threats and mitigations, respectively.

*Figure 4: Class diagram*

*Figure 5: Misuse case diagram*

**Development Tools**

* ***PyCharm***- integrated development environment (IDE) that will be used to code the software.
* ***Python 3.12***- an object oriented programming language that will be used to develop the application (Python Software Foundation, N.D.b).
* ***hashlib-*** for importing SHA 256.

**Application Tests**

* ***Unittest*** – tests pieces of code and reports on any violations. All tests must pass (Python Software Foundation, N.D.a).
* ***Pylint*** – checks for code errors, best practices, and suggests how to align with those (The Python Package Index, 2024).
* ***Pydocstyle*** – ensures code compliance with PEP257 standards (Rachum & Kothari, N.D.).
* ***Pylama –*** the pycodestyle element will check for PEP8 violations and McCabe will check for Cyclomatic Complexity.

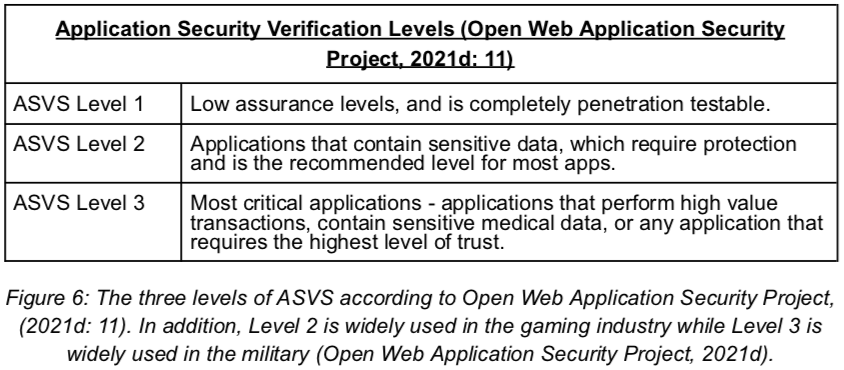
**Data Privacy Regulations**

SciTec will store a limited amount of personal data. To ensure compliance with the relevant articles from the **General Data Protection Regulation (GDPR)** chapters 2 and 3 (GDPR, N.D.), the following measures will be enforced:

* ***Right of access*** – users must be provided with their stored information within one month of request.
* ***Right to be forgotten*** – all data relating to an individual user will be removed upon request, following proof of identification.
* ***User data will not be provided to third parties.***

**OWASP Application Security Verification Standards**

OWASP identifies 3 levels of **Application Security Verification Standards (ASVS),** as seen in Figure 6.



SciTec lands in the **ASVS Level 2**, as it is a system that will be developed in a theoretical and controlled environment. Please refer to the [Appendix](#bookmark) section for a more comprehensive review of the methodology used.

**References:**

Brito, J.N., Rebelo, C. & Brito, M.A. (2020) ‘Scrum Solo Application in a Project with a Strong Integration Component’, *36th IBIMA Conference.* Granada, Spain, 4-5 November. Available from: <https://hdl.handle.net/1822/71816> [Accessed 3 April 2024].

Chethana, S., Charan, S.S., Srihitha, V., Radha, D. & Kavitha, C.R. (2022) ‘Comparative Analysis of Password Storage Security using Double Secure Hash Algorithm’, *2022 IEEE Karnataka Subsection Flagship International Conference (NKCon).* Vijaypur, India, 20-21 November. 1-5. DOI: <https://doi.org/10.1109/nkcon56289.2022.10127057>.

‌Diallo, O., Lowe, L., Neiss, K., Orosco, H., Redell, B., Schlesinger, T. & Williams, J.S. (2022) A Researcher’s Guide to International Space Station Technology Demonstration. Available from: <https://www.nasa.gov/wp-content/uploads/2023/08/np-2022-06-009-jsc-iss-tech-demo-mini-book-proof-3.pdf> [Accessed 2 April 2024].

Esheridan. (N.D.) Blocking Brute Force Attacks | OWASP Foundation. Available from: <https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks> [Accessed 30 March 2024].

General Data Protection Regulation. (N.D.) General Data Protection Regulation (GDPR). Available from: <https://gdpr.eu/tag/gdpr/>. [Accessed 3 April 2024].

King, M. & Goguichvili, S. (2020) Cybersecurity Threats in Space: A Roadmap for Future Policy | Wilson Center. Available from: <https://www.wilsoncenter.org/blog-post/cybersecurity-threats-space-roadmap-future-policy> [Accessed 5 April 2024].

Moyo, S. & Mnkandla, E. (2020) A Novel Lightweight Solo Software Development Methodology With Optimum Security Practices. *IEEE Access* 8: 33735-33747. DOI: <http://dx.doi.org/10.1109/ACCESS.2020.2971000>. [Accessed 2 April 2024].

Oakley, J.G. (2020) *Cybersecurity for Space: Protecting the Final Frontier.*  California, USA: Apress. Available from: <https://learning.oreilly.com/library/view/cybersecurity-for-space/9781484257326/> [Accessed 31 March 2024].

Open Web Application Security Project. (2021a) A03 Injection - OWASP Top 10:2021. Available from: <https://owasp.org/Top10/A03_2021-Injection/> [Accessed 15 April 2024].

Open Web Application Security Project. (2021b) A07 Identification and Authentication Failures - OWASP Top 10:2021. Available from: <https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/> [Accessed 18 April 2024].

Open Web Application Security Project. (2021c) A09 Security Logging and Monitoring Failures - OWASP Top 10:2021. Available from: <https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/> [Accessed 18 April 2024].

Open Web Application Security Project. (2021d) Application Security Verification Standard 4.0.3. Available from [https://github.com/OWASP/ASVS/raw/v4.0.3/4.0/OWASP%20Application%20Security%20Verification%20Standard%204.0.3-en.pdf](https://github.com/OWASP/ASVS/raw/v4.0.3/4.0/OWASP%25252520Application%25252520Security%25252520Verification%25252520Standard%252525204.0.3-en.pdf) [Accessed 15 April 2024].

Open Web Application Security Project. (2024) ASVS/5.0/en at master · OWASP/ASVS. Available from: <https://github.com/OWASP/ASVS/tree/master/5.0/en> [Accessed 18 April 2024].

Open Web Application Security Project. (N.D.) OWASP Top Ten Proactive Controls 2018 | Introduction. Available from: <https://owasp.org/www-project-proactive-controls/v3/en/0x04-introduction> [Accessed 5 April 2024].

Pillai, A.B. (2017) *Software Architecture with Python: Design and architect highly scalable, robust, clean, and high performance applications in Python.* 1st ed. Birmingham, UK: Packt Publishing. Available from: <https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1513359&site=ehost-liv> [Accessed 30 March 2024].

Pipikaite, A., Holla-Maini, A., Ware, B. & Dickinson, M. (2022) Why we need increased cybersecurity for space-based services | World Economic Forum. Available from: <https://www.weforum.org/agenda/2022/05/increased-cybersecurity-for-space-based-services/> [Accessed 31 March 2024].

Public Broadcasting Station. (N.D.) What is the purpose of the Space Station?. Available from: [https://www.pbs.org/spacestation/station/purpose.htm#:~:text=The%20mission%20of%20the%20International,than%20any%20previous%20space%20station](https://www.pbs.org/spacestation/station/purpose.htm%252523:~:text=The%25252520mission%25252520of%25252520the%25252520International,than%25252520any%25252520previous%25252520space%25252520station). [Accessed 31 March 2024].

Python Software Foundation. (N.D.a) unittest — Unit testing framework — Python 3.12.2 documentation. Available from: <https://docs.python.org/3/library/unittest.html> [Accessed 30 March 2024].

Python Software Foundation. (N.D.b) What is Python? Executive Summary. Available from: <https://www.python.org/doc/essays/blurb/> [Accessed 24 March 2024].

Rachum, A. & Kothari, S. (N.D.) pydocstyle’s documentation — pydocstyle 0.0.0.dev0 documentation. Available from: <https://www.pydocstyle.org/en/stable/> [Accessed 30 March 2024].

Radigan, D. (N.D.) Four agile ceremonies, demystified | Atlassian. Available from: <https://www.atlassian.com/agile/scrum/ceremonies> [Accessed 30 March 2024].

Rostedt, S. (2022) Virtual Machine Overhead Compared to Bare Metal with Tracing | PPT. Available from: <https://www.slideshare.net/ScyllaDB/analyze-virtual-machine-overhead-compared-to-bare-metal-with-tracing> [Accessed 2 April 2024].

Sani, M.N. (2022) API Injection Attacks: What They Are and How to Prevent Them. Available from: <https://www.computer.org/publications/tech-news/trends/api-injection-attacks-prevention>[Accessed 3 April 2024].

The National Aeronautics and Space Administration. (2023) International Space Station - NASA. Available from: <https://www.nasa.gov/reference/international-space-station/> [Accessed 31 March 2024].

The New Stack. (2020) The Hardware and Software Used in Space - The New Stack. Available from: <https://thenewstack.io/the-hardware-and-software-used-in-space/> [Accessed 2 April 2024].

The Python Package Index. (2024) Available from: <https://pypi.org/project/pylint> [Accessed 30 March 2024].

Waldmann, B. (2019) MIL-STD-1553 Tutorial. Available from: <https://www.aim-online.com/wp-content/uploads/2019/07/aim-tutorial_ovw-mil-std-1553_190719-u.pdf> [Accessed 2 April 2024].

Wells, S. (2020) Protecting ISS - Aerospace America. Available from: <https://aerospaceamerica.aiaa.org/features/protecting-iss/> [Accessed 5 April 2024].

Wen, S-F. & Katt, B. (2023) A quantitative security evaluation and analysis model for web applications based on OWASP application security verification standard. *Computers & Security* (135): 1-13. DOI: <https://doi.org/10.1016/j.cose.2023.103532>. [Accessed 15 April 2024].

**Bibliography:**

Mohan, M. (2017) Mutable vs Immutable Objects in Python | by megha mohan | Medium. Available from: <https://medium.com/@meghamohan/mutable-and-immutable-side-of-python-c2145cf72747> [Accessed 1 April 2024].

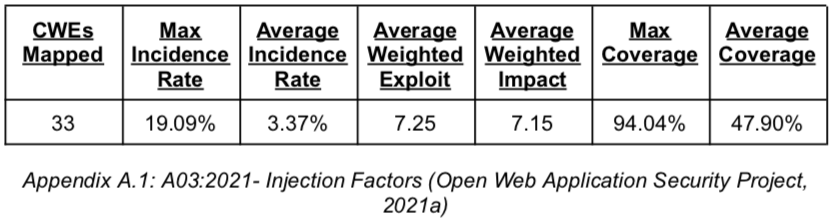
Pauli, J.J. & Xu, D. (2005) ‘Misuse case-based design and analysis of secure software architecture’, *International Conference on Information Technology: Coding and Computing (ITCC’05) - Volume II.* Las Vegas, Nevada, USA, 04-06 April. 398-403. DOI: <https://doi-org.uniessexlib.idm.oclc.org/10.1109/ITCC.2005.199>.

Rocholl, J.C., Xicluna, F. & Lee, I. (N.D.) pycodestyle’s documentation. Available from: <https://pycodestyle.pycqa.org/en/latest/> [Accessed 30 March 2024].

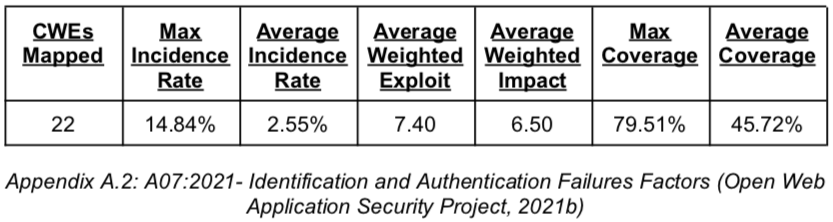
# APPENDIX A - CRITICALITY

To assess the security level of SciTec relative to the OWASP ASVS classification, a method proposed by Wen & Katt (2023) was used. They define criticality as a numerical value that we give to a security risk that communicates how serious it is, hence determining what risks must be mitigated first.

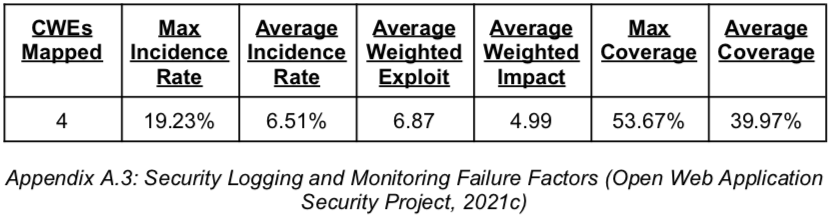
From OWASP (2021a), for **A03:2021 – Injection**, the criticality is the average weighted impact times the average incidence rate = 0.24.



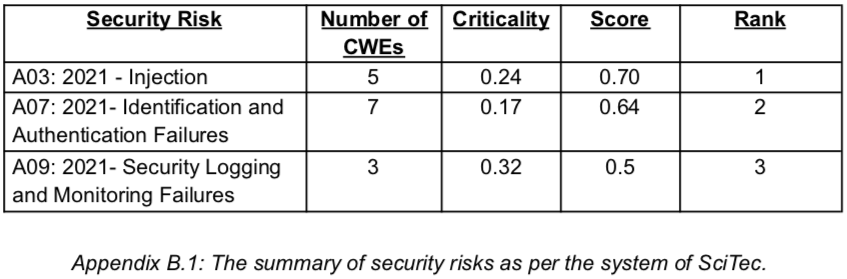
From OWASP (2021b), for **A07:2021 – Identification and Authentication Failures**, the criticality is the average weighted impact times the average incidence rate = 0.17.



From OWASP (2021c), for **A09:2021 – Security Logging and Monitoring Failures,** the criticality is the average weighted impact times the average incidence rate = 0.32.



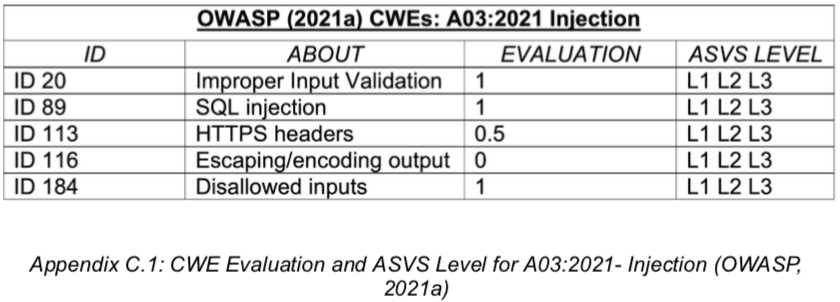
**APPENDIX B - SCORE AND RANKING**

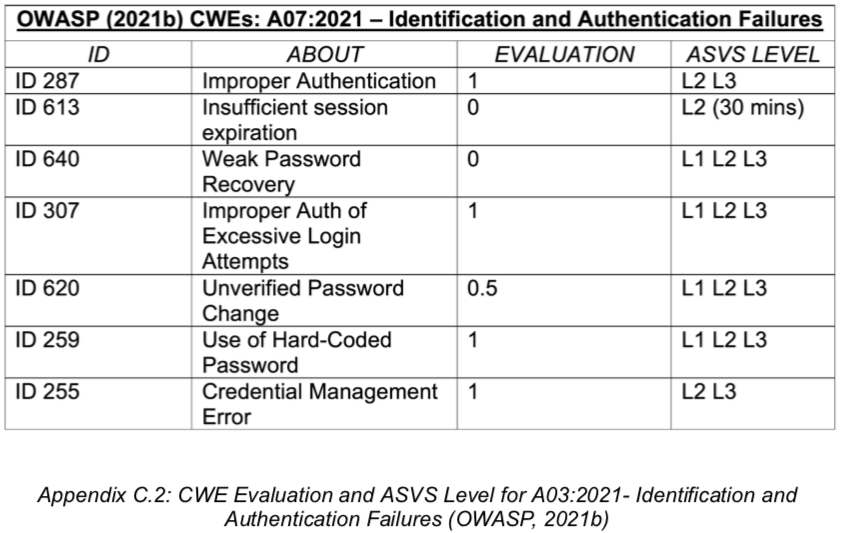


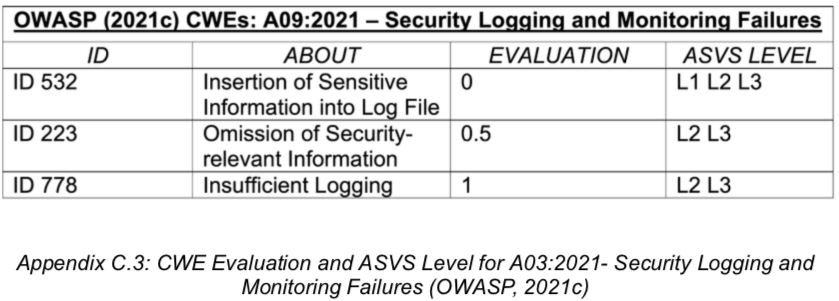
In Appendix B.1, the security scores are rated as 0 for not fulfilled, 1 for fulfilled and 0.5 for partial fulfillment. The evaluation is used to compute the average score for each category (column 4), and the higher the score, the higher the security and ranking as well. From Open Web Application Security Project (2024), these **Common Weakness Enumeration (CWEs)** are mapped according to the Application Security Verification Standards (ASVS) in the three different levels. Level 2 is the most applicable overall, which confirms the system being ranked in level 2 for the ASVS by OWASP (2021d).

Ranks on Appendix B.1 are rather consistent to the ASVS level allocations for each of the CWEs, with injection being the most basic threat to mitigate, as it is mitigated on levels 1, 2 and 3. Bulwark Systems successfully put contingencies in place to minimise injection risks, thus ranking in position 1. Authentication failures show the next highest concentration of level 1 mitigations, the score shows a correlation to this, ranking in position 2. Lastly, logging and monitoring failures follow the trend with the least concentration of level 1 mitigations, at the same time least prepared for the system to mitigate against.

**APPENDIX C - CWE SELECTION**







In summary, Bulwark Systems could handle Security Logging and Monitoring Failures more effectively, as their criticality, according to OWASP, is the highest of the three risks presented in this document. However, Bulwark’s SciTec employs the right mitigation job for the other more basic levels, scoring and ranking higher security-wise.