**1. Why did you choose the tools, libraries, and language used for the coding exercise?**

I chose **Python** as the primary language due to its readability, extensive community support, and simplicity for building RESTful APIs. I have previous experience using Python for building APIs, and hence it was the most comfortable language for me.

For the web framework, I selected **Flask** because it is lightweight and suitable for building simple projects. It has a minimalistic nature and makes it easy to rapidly build or add on endpoints. I also used the **Flask-RESTful** extension, since it simplifies writing RESTful services by providing resource classes and request parsing. This helped in organizing my code and keeping the request handling logic clean and manageable.

For storing data, I used **SQLAlchemy**, which allowed me to define my database schema through model classes, handle migrations smoothly, and avoid writing raw SQL queries. Using SQLite as the database provided a zero-configuration solution ideal for development and testing, ensuring the setup remained simple and self-contained.

**2. What are the advantages and disadvantages of the solution?**

**Advantages:**

My resource-based approach makes it easy to understand how endpoints are structured and what data they return or accept. This is especially useful for those that may be unfamiliar with Flask. My code also allows for rapid development or adding new endpoints quite easily. SQLAlchemy’s ORM abstraction also reduces the likelihood of SQL injection vulnerabilities, and makes the code more maintainable and readable.

**Disadvantages:**

While Flask is great for small to medium projects, scaling this application to handle millions of users instead of a single user might require additional steps to ensure performance. Adding additional features like authentication or input validation can also be troublesome since flask does not have built in capabilities for these. Similarly, SQLite is fine for small-scale use, but it’s not suitable for high load production environments. Scaling up may require switching to other options like MYSQL.

**3. What has been a favorite school/personal project thus far? What about it that challenged you?**

My favorite school project so far has been “Mazeball”, where I developed a game in which players have to direct a ball through a maze using their phone’s gyroscope. The most challenging part was that this was the first time I was developing a mobile based application instead of a web-based application. I had to dive deep into working with sensor data, for example the gyroscope, and ensure cross platform compatibility. I also had work with a new platform Unity, which required me to learn the language C#. The result was worth it!