# AeroAspire - SDE Intern
## Gokul Krishna S

### Week 3 – Day 1 (October 6)

**Questions/Reflections:**

1. What is the flow when a HTTP GET request comes to Flask → how Flask handles routing → response.

    - HTTP GET Request Flow in Flask
    - When a GET request hits a Flask app, the WSGI server (like Werkzeug in development or Gunicorn/uWSGI in production) receives the HTTP request and passes it to Flask via the WSGI interface.
    - Flask parses the request, matches the URL and HTTP method to a route defined in the app, and calls the associated view function.
    - The view function processes the request, optionally interacts with services or databases, and returns a response (usually JSON for APIs).
    - Flask serializes the response and sends it back through the WSGI server to the client.

2. What is WSGI? How Flask sits on WSGI server (development vs production).

    - WSGI (Web Server Gateway Interface) is a Python standard for web servers and applications to communicate.
    - In development, Flask uses its built-in Werkzeug server (not for production).
    - In production, Flask should run behind a WSGI server like Gunicorn or uWSGI, which handles concurrency, process management, and forwards requests to Flask.

3. How you’d structure a medium size API project (folders, modules).

- project/

```
│
├── app/
│   ├── __init__.py
│   ├── models/
│   ├── routes/
│   ├── schemas/
│   ├── services/
│   ├── utils/
│   └── extensions.py
│
├── tests/
├── migrations/
├── config.py
├── requirements.txt
└── run.py
```

- Use Blueprints to organize routes by resource or feature.
- Place business logic in services/, data validation in schemas/ (e.g., Marshmallow), and database models in models/.
- Use an app factory pattern for flexibility and testing.

4. What is REST: what makes an endpoint RESTful?

- REST (Representational State Transfer) is an architectural style for APIs.

  **RESTful endpoints:**
- Use HTTP methods semantically (GET for read, POST for create, PUT/PATCH for update, DELETE for delete).
- Are stateless (no session stored on server).
- Use resource-based URLs (e.g., /api/v1/users/123).
- Return appropriate status codes and representations (usually JSON).

5. What status codes should be returned and when?

| **Status Code** | **When to Use** |
|---|---|
| • 200 OK | • Successful GET, PUT, or DELETE |
| • 201 Created | • Resource created (POST) |
| • 204 No Content | • Successful request, no body (DELETE, PUT) |
| • 400 Bad Request | • Invalid input, malformed JSON |
| • 401 Unauthorized | • Authentication required/failed |
| • 403 Forbidden | • Authenticated but not allowed |
| • 404 Not Found | • Resource does not exist |
| • 409 Conflict | • Resource conflict (e.g., duplicate) |
| • 422 Unprocessable Entity | • Validation error |
| • 500 Internal Server Error | • Unexpected server error |