FastAPI Interview Agenda

Basic Level

1. Introduction to FastAPI:
   - What is FastAPI?
   - How does it differ from other Python frameworks like Flask or Django?
   - Key features of FastAPI (ASGI, async support, data validation, etc.).

2. Routes and Path Parameters:
   - How to create a basic route in FastAPI.
   - Path and query parameters.
   - Path parameter validation (e.g., using Path and Query).

3. Request Body:
   - How to send and receive data using Body.
   - Using Pydantic models for request body validation.

4. Response Handling:
   - How to return JSON responses.
   - Customizing HTTP status codes.
   - Using Response and JSONResponse.

5. Dependency Injection:
   - What is dependency injection in FastAPI?
   - Simple examples of using dependencies in routes.

6. Error Handling:
   - Handling errors with HTTPException.
   - Custom error messages and status codes.

Intermediate Level

1. Asynchronous Programming:

   - How to write asynchronous routes in FastAPI.

   - Difference between synchronous (def) and asynchronous (async def) endpoints.

2. Path Operations and Type Annotations:

   - Using Python type hints for parameter validation.

   - Default values in query parameters and body fields.

   - Path operation decorators (@app.get(), @app.post(), etc.).

3. Authentication & Authorization:

   - How to implement basic authentication (e.g., OAuth2 with Password flow).

   - Securing endpoints with Depends and Security.

4. JWT token generation and verification.Database Integration:

   - Integrating FastAPI with databases (e.g., using SQLAlchemy or MongoDB).

   - How to handle database sessions with FastAPI.

   - Dependency injection for database connections.

5. Pydantic and Data Validation:

   - Advanced usage of Pydantic for request/response validation.

   - How to create custom validators.

   - Enforcing field constraints and using Field.

6. Middleware:

   - What is middleware in FastAPI?

   - Writing custom middleware for logging, CORS, etc.

7. Testing FastAPI Applications:

   - Writing tests for FastAPI using pytest and TestClient.

   - Mocking external services and database interactions.

   - Test database setup and cleanup.

Advanced Level

1. WebSockets:

   - How to implement WebSocket endpoints in FastAPI.

   - Use cases for WebSockets (e.g., real-time notifications, chat applications).

2. Background Tasks:
   - Running background tasks in FastAPI.

   - Use cases for background tasks (e.g., sending emails, data processing).

3. Dependency Injection (Advanced):

   - Managing complex dependencies and sub-dependencies.

   - Using classes as dependencies.

   - Caching dependencies with Depends.

4. Event Handling:

   - How to handle startup and shutdown events in FastAPI.

   - Use cases (e.g., initializing database connections, cleaning up resources).

5. API Versioning:

   - Strategies for versioning APIs in FastAPI.

   - Maintaining backward compatibility.

6. Rate Limiting and Throttling:

   - Implementing rate limiting using middleware or third-party libraries.

7. Performance Optimization:

   - FastAPI's performance features (e.g., ASGI, async IO).

   - Profiling and benchmarking FastAPI applications.

   - Using caching mechanisms.

8. Security Best Practices:

- Secure API development (e.g., CSRF, CORS, HTTPS).
- OAuth2 integration with third-party services (e.g., Google, GitHub).
- Role-based access control (RBAC) and multi-tenancy.

9. Production Deployment:

- Deploying FastAPI applications using Docker, Kubernetes, or cloud services.
- Load balancing, scaling, and managing FastAPI applications in production.
- Logging and monitoring in production.

10. API Documentation:

- Customizing FastAPI's auto-generated documentation (Swagger and ReDoc).
- Adding examples to API documentation.
- Using OpenAPI standards for API design.