Mindmap to Learn FastAPI

## **Introduction to FastAPI**

* **~~Overview: FastAPI is a modern, fast (high-performance), web framework for building APIs with Python 3.7+ based on standard Python type hints.~~**
* **~~Installation: FastAPI can be installed using pip: pip install fastapi. It is often used with an ASGI server such as Uvicorn: pip install "uvicorn[standard]".~~**

## **First Steps**

* **Creating an App**: Define a FastAPI instance.
  + Example: from fastapi import FastAPI; app = FastAPI().
* **Running the App**: Use Uvicorn to run the app.
  + Example: uvicorn main:app --reload.

**Path Operations**

* **Path Parameters**: Define dynamic URL paths.
  + Example: @app.get("/items/{item\_id}").
* **Query Parameters**: Accept query parameters in URLs.
  + Example: @app.get("/items/") def read\_item(item\_id: int).

**Request Body**

* **Pydantic Models**: Use Pydantic models to define the request body.
  + Example: from pydantic import BaseModel; class Item(BaseModel): name: str, price: float.
* **Parsing Request Body**: Use the Pydantic model in endpoint functions.
  + Example: @app.post("/items/") def create\_item(item: Item).
* **Response Handling**
  + **Response Models**: Define the response structure with Pydantic models.
    - Example: @app.get("/items/", response\_model=Item).
  + **Custom Responses**: Return custom responses using JSONResponse.
    - Example: from fastapi.responses import JSONResponse; return JSONResponse(content={"message": "Hello"}).
* **Dependency Injection**
  + **Dependencies**: Use dependencies to share logic and resources.
    - Example: @app.get("/items/") def read\_items(common: str = Depends(common\_parameters)).
  + **Sub-dependencies**: Define nested dependencies.
    - Example: @app.get("/subitems/") def read\_subitems(subcommon: str = Depends(subcommon\_parameters)).
* **Security**
  + **OAuth2**: Implement OAuth2 with Password (and hashing), Bearer with JWT tokens.
    - Example: from fastapi.security import OAuth2PasswordBearer; oauth2\_scheme = OAuth2PasswordBearer(tokenUrl="token").
  + **Security Scopes**: Define security scopes and permissions.
    - Example: @app.get("/users/me", scopes=["me"]).
* **Middleware**
  + **Custom Middleware**: Add custom middleware for processing requests and responses.
    - Example: @app.middleware("http") async def add\_process\_time\_header(request: Request, call\_next).
  + **Using Built-in Middleware**: Integrate built-in middleware for CORS, GZip, etc.
    - Example: from fastapi.middleware.cors import CORSMiddleware; app.add\_middleware(CORSMiddleware).
* **Background Tasks**
  + **Defining Background Tasks**: Execute code in the background during request processing.
    - Example: from fastapi import BackgroundTasks; @app.post("/send-email/") def send\_email(background\_tasks: BackgroundTasks).
* **WebSockets**
  + **WebSocket Endpoint**: Define WebSocket endpoints for real-time communication.
    - Example: @app.websocket("/ws/{client\_id}") async def websocket\_endpoint(websocket: WebSocket).
* **Testing**
  + **TestClient**: Use the TestClient from FastAPI for testing your endpoints.
    - Example: from fastapi.testclient import TestClient; client = TestClient(app).
* **Database Integration**
  + **SQLAlchemy**: Use SQLAlchemy for ORM.
    - Example: from sqlalchemy import create\_engine; engine = create\_engine(DATABASE\_URL).
  + **Tortoise ORM**: Alternative ORM with Pydantic integration.
    - Example: from tortoise.contrib.fastapi import register\_tortoise; register\_tortoise(app).
* **Asynchronous Programming**
  + **Async Endpoints**: Define endpoints as async functions.
    - Example: @app.get("/items/async") async def read\_items\_async().
  + **Async Libraries**: Use async libraries for I/O-bound operations.
    - Example: import httpx; async with httpx.AsyncClient() as client.
* **Configuration and Environment**
  + **Environment Variables**: Use pydantic.BaseSettings to manage settings.
    - Example: from pydantic import BaseSettings; class Settings(BaseSettings): DATABASE\_URL: str.
  + **Configuration Files**: Load configurations from files.
* **Advanced Features**
  + **GraphQL**: Implement GraphQL endpoints using libraries like graphene.
    - Example: from starlette.graphql import GraphQLApp; app.add\_route("/graphql", GraphQLApp(schema=schema)).
  + **Event Handlers**: Use event handlers for startup and shutdown tasks.
    - Example: @app.on\_event("startup") async def startup\_event().
* **Documentation and OpenAPI**
  + **Auto-generated Docs**: FastAPI automatically generates interactive API docs.
    - Accessible at /docs (Swagger UI) and /redoc (ReDoc).
  + **Customizing Docs**: Customize the OpenAPI schema and docs.
    - Example: app = FastAPI(docs\_url="/documentation", redoc\_url=None).
* **Deployment**
  + **Uvicorn**: Deploy using Uvicorn with production settings.
    - Example: uvicorn main:app --host 0.0.0.0 --port 80 --workers 4.
  + **Docker**: Containerize your FastAPI application with Docker.
    - Example: Create a Dockerfile and build the image.
* **Performance**
  + **Benchmarking**: Use tools like locust or hey to benchmark.
    - Example: hey -z 30s http://localhost:8000/items/.
  + **Optimizations**: Profile and optimize your code.
    - Use async I/O, efficient query handling, and caching.