**FastAPI**

**Features –** High performance, interactive documentation, async-ready

Input data can be validated using Pydantic Models

Tips:

1. Use async def for endpoints with non-blocking I/O operations
2. Use normal functions for endpoints with blocking I/O operations

Starlette for routing , middleware, etc

Pydantic for data validation and serialization

Uvicorn – ASGI server

Docs and redocs are very useful, testing that saves time

**Working**

FastApi built on ASGI – Async Server Gateway Interface and it allows for async requests handling for non blocking I/Os and it uses Starlette for its core web-handling features

Autogenerated docs – Swagger UI and ReDoc

**Example**

http req (POST)

{

‘name’ : ‘ABC’,

‘age’ : 45

}

1. First req is sent to endpoint and the ASGI receives the request and passes it to FastAPI
2. Route matching
3. Parse json to a python dict and may use Pydantic to validate whether it matches the required req format using an object
4. Then the function associated with that route is executed
5. The it convert the return dictionary to json response
6. And then the ASGI server(Uvicorn) sends the response back

**Dependency Injections**

**Receiving pdf data**

Sent with headers are multipart/form-data via http post

Body content has binary data of the pdf and some metadata

Sents as stream of bytes

FastApi classes like UploadFile and File used to get this data

As the docs will be from different networks path is not possible