Needed tools for developing frontend and backend:

1. Programming Languages

* Frontend
  + HTML
  + CSS
  + JavaScript
* Backend
* Node.js
* Python
* Java

2. Libraries

* Frontend
* React.js
* Angular
* Vue.js
* Bootstrap
* Backend
* Express.js, NestJS(Node.js)
* Django, Flask, FastAPI(Python)
* Sprint Boot(Java)

3. Database

* Relational: MySQL, Oracle

4. API

* Postman/Insomnia (API Testing)
* Swagger/OpenAPI (API Documentation)
* Unit testing libraries: Jest(JS), PyTest(Python), Junit(Java)

**1. Frontend (Web Interface)**

This is what users interact with.

**Languages/Frameworks:**

* **HTML, CSS, JavaScript** (core building blocks)
* **React.js** (popular, flexible, good for dynamic UI)
* **Vue.js / Angular** (alternatives)
* **Tailwind CSS / Bootstrap** (for styling)

**Tools:**

* VS Code (IDE)
* Browser DevTools (Chrome/Firefox) for debugging

**2. Backend (Server + AI Model Hosting)**

This is where your AI model runs and responds to the frontend.

**Languages/Frameworks:**

* **Python** → most common for AI models
  + Frameworks: Flask, FastAPI, or Django (for serving the API)
* **Node.js** → if you prefer JavaScript on the backend
  + Framework: Express.js

**Why Python:**

* Python has **libraries for AI** (TensorFlow, PyTorch, Scikit-learn, Hugging Face, OpenAI API).

**API Layer:**

* Create a **REST API or GraphQL API** in the backend to communicate with the frontend.

**3. Database**

If your app needs to **store user data, model inputs, or predictions**:

* **Relational:** PostgreSQL, MySQL
* **NoSQL:** MongoDB (for unstructured data like JSON)

**4. AI Model Tools**

**Python Libraries:**

* **TensorFlow / Keras** → Deep Learning models
* **PyTorch** → Deep Learning / Custom AI models
* **Scikit-learn** → Classical ML algorithms
* **Hugging Face Transformers** → NLP models

**Optional:**

* OpenAI API / other pre-trained models if you don’t want to train from scratch

**5. Communication Between Frontend and Backend**

* **REST API** (most common) → Frontend calls endpoints like /predict to get results
* **WebSockets** (optional) → Real-time communication if needed

**6. Deployment / DevOps**

* **Docker** → containerize your backend + AI model
* **Cloud Platforms** → AWS, Azure, GCP for hosting models
  + AI-specific services: AWS SageMaker, GCP Vertex AI, Azure ML
* **NGINX** → serve frontend and reverse-proxy backend

**7. Testing Tools**

* **Postman / Insomnia** → test API endpoints
* **Unit testing:** PyTest (Python), Jest (JS)

JS Asynchronous programming

Lamda function

\*Spread operator

Node, event, loop

Js is single threaded

Plugin