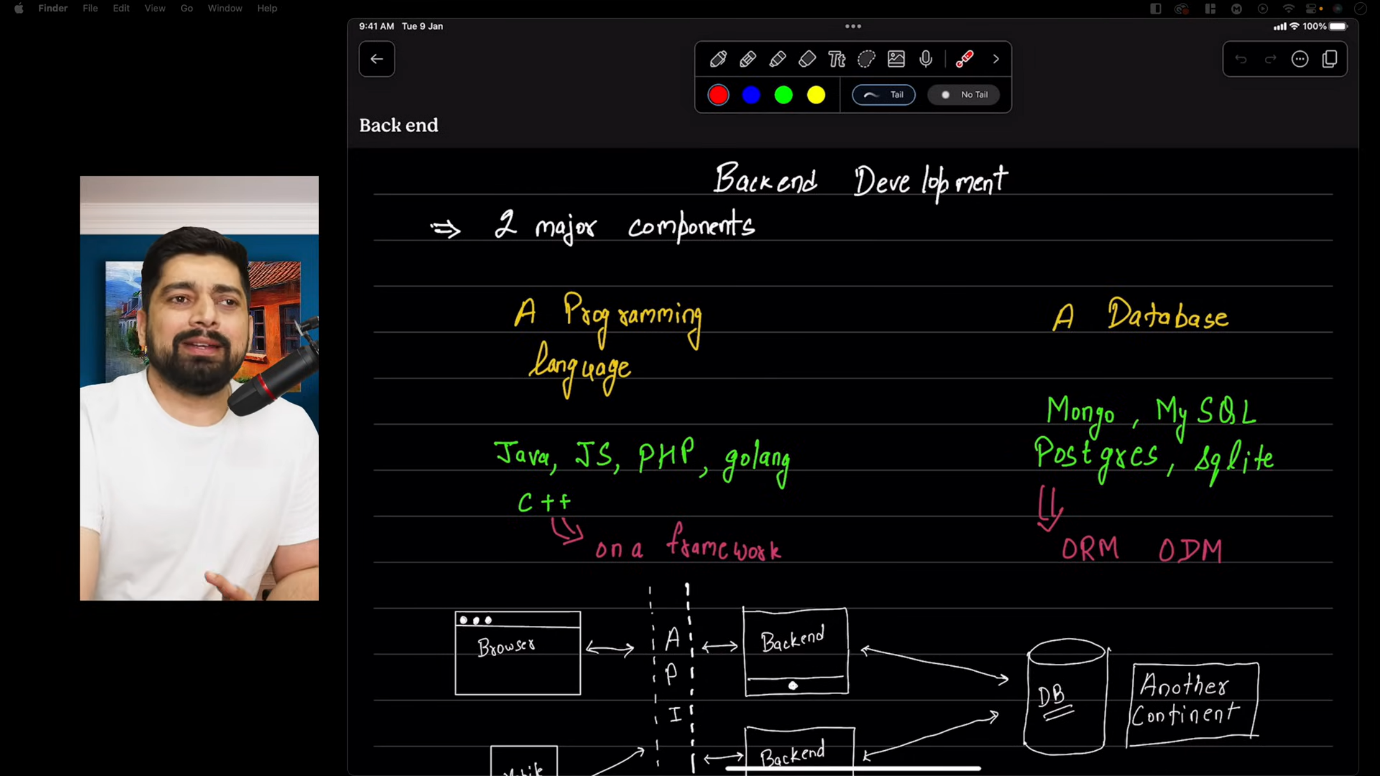
Backend Basics &

Javascript Backend Roadmap



We will need a programming language for writing business logic or data queries for storing or retrieving data from database. We may also need framework of that particular language for ease of writing complex code.

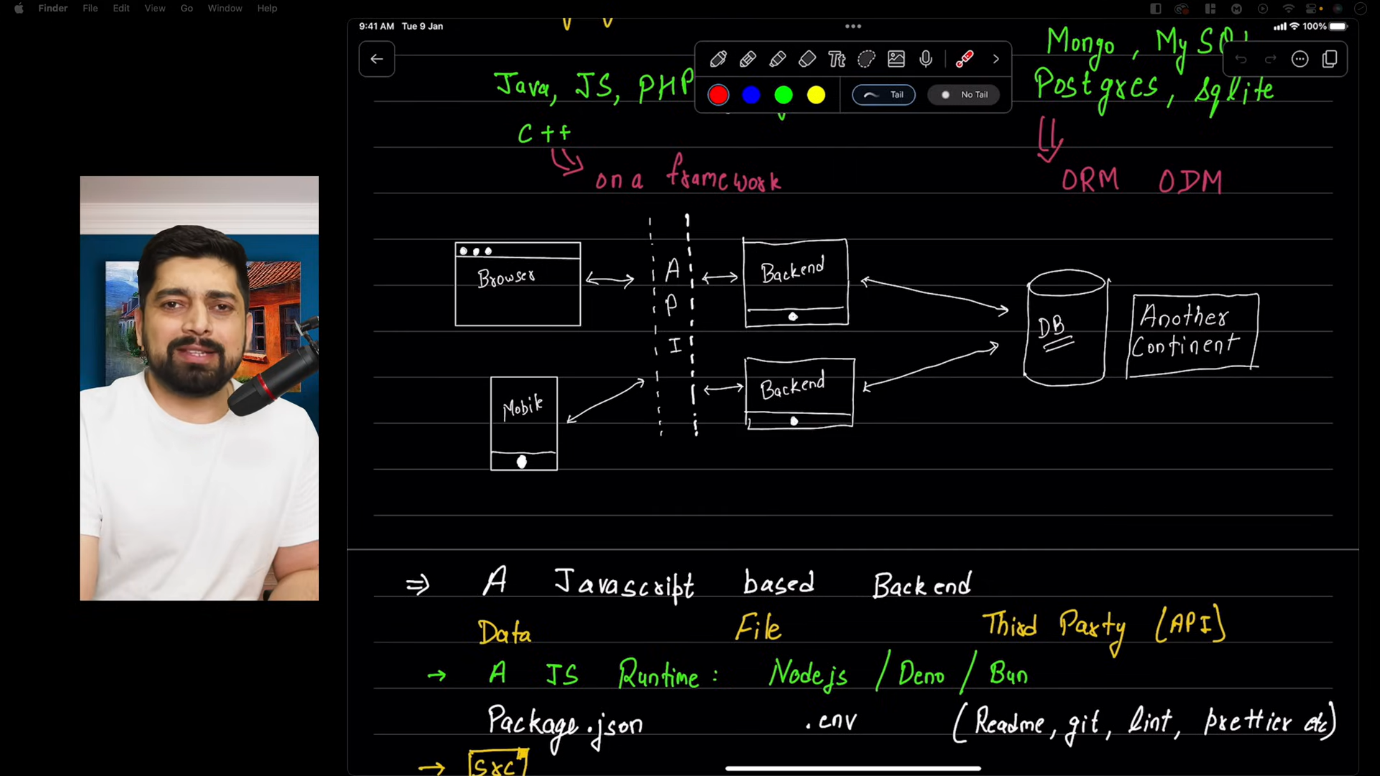
Languages (& their frameworks) for backend development =>

Java(springboot), Js(React,Angular,etc), PHP(Laravel, CakePHP), Golang(), C++(Crow)

Databases (& their frameworks) for backend development =>

Mongo, MySQL, Postgress, Sqlite and frameworks – (ORM, ODM)

Server is nothing but a software that serves.



**Flow of a full stack application =>**

**Frontend**(will do a query, like is 2+2 =4? If true send some response or add something in result, if false send something else **or**

we’ll get some data from frontend for checking, if correct send some response, otherwise send something else like error or any message)

**Database**(always resides in another continent)

**BACKEND** (We’ll write a backend on machine which is continuously running on servers. Suppose we visit a url like /login, /signup,etc. that will be detected by frameworks. And acc to the routes we visited, specific function for that route will be called from backend. **In Backend, we will generally write different functions , and they will interact with database , and we will send the response from database in form of an API**)

**API**(Response from backend who fetched it from database , eg- true, false, object , array,etc.)

NOW, MAYBE THESE CALLS(or queries from frontend) ARISE FROM BROWSER OR MOBILE

**Now, in JAVASCRIPT based backend we will deal with three things-**

**DATA** - Either we have to deal with data directly (username, password, number, strings, objects, arrays)

**FILE –** OR We have to handle file (image, pdf, video, etc)

**THIRD PARTY (API) –** (Dealing with third party APIs. Suppose we have to google login or we have to upload file on AWS or sending email, sms, calls, file upload download from third party APIs)

* **We’ll need a JS Runtime (Nodejs/Deno/Bun)**
* **FILE STRUCTURE :**

**Package.json -** This file will contain all the dependencies, libraries

**.env -** This will store environment variables.

**(Readme, git, lint, prettier, etc) -** Some other basic files.

* **src**

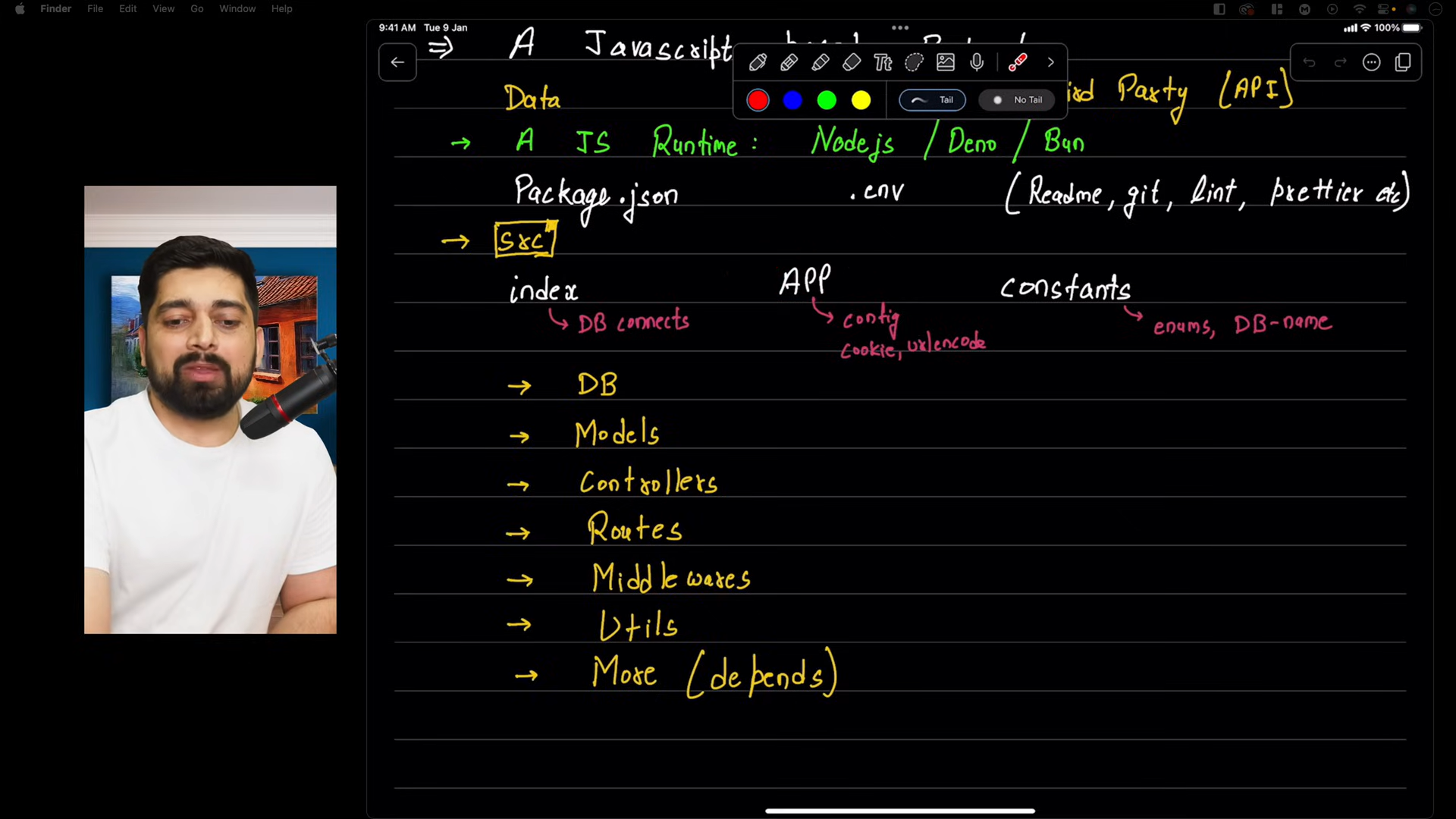
**index –** index file is the entry point of application . We’ll connect db in this file, when app starts

**App –** app file handles configurations(backend cookies, configurations, urlencode)

**constants –** is for storing constant names (enums, DB-name). Suppose we made a airticket booking application, and we have three type of seats aisle, middle, window.

And we’ll want to create enums for these three constants. If we don’t , then people can also select pilot seat

* **DB** **(FOLDER) -** This folder will contain all the code which will connect to the database
* **Models (FOLDER) –** This folder contain data models
* **Controllers (FOLDER) –** This will contain functions like MVC models (C in MVC is controller). These functions will take data and process it.
* **Routes (FOLDER) -** /SIGNUP, /LOGIN, ETC. (all code related to routes & functions which will be called on visiting them)
* **Middlewares (FOLDER) –** will discuss it later
* **Utils (FOLDER) –** utility folder is for utilities. This will handle work like handling functions which will be needed in many places like mailing(which will be needed at many places like in forgot password, congratulate signup, reset password) or file upload (it is also needed in many places in any full-stack app)
* **More(depends) (FOLDER)**



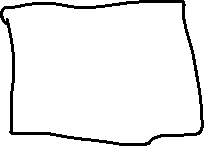
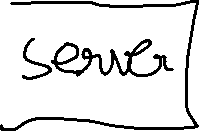
**How to deploy backend code in production**

1. Install nodejs
2. Now, we have to make a server, and if someone send a req from com/mobile(frontend)

And, for sending response to that request we will need two tech or package – express and mongoose.

Firstly, express will handle sending the response from server to comp/mobile for request.

Mongoose will handle database work.



Express

get

**Computer**

**or Mobile**

**Browser**

* Now, suppose if visit a url, a server is listening to the request, which will send something in response.

**SO, there should be something which is listening continuously for requests and express is the package which will handle listening to the requests.**

**/ - Home route**

**/login – login route**

* **There are many type of requests-**

**GET, Deleting data or any file or any entry from database, or many more types of requests.**

* **For now, we will see get request. When we enter anything in url, that is get request**
* **We have to make a server using express for listening to requests.** We have to listen to many routes like /, /login , etc.
* **CODE =>**

1. First, we have to make a empty node application, for that run command

**npm init (means node package manager se 1 application ko initialize krdo)**

This will create a file named **package.json** which contains all the common items like file name,version, desc, scripts, dependencies, etc.

And, we can use **npm install** to install any package.

Steps:

* Package name: firstBackend
* Version
* Desc: a basic app to deploy
* Entry point – index.js
* Test command: not now
* Git repo – later
* Keywords - node chai (for now)
* Author – bharti sahani
* License:

1. Make a file named index.js and write console .log for testing and to run this file, we can use **node index.js**

**In package.json, in scripts we have to add start command –**

**“start” – “node index.js”**

Now, whenever we’ll write npm run start in terminal, node index.js command will run.

1. Now, we have to make backend, and to make backend we will need express.

**npm i express,**  and after installing we can see express in dependencies section in package.json

1. Now, take basic code from express official site –

const express = require('express')

//require module syntax (1st method)

//import express from "express"  //import module syntax (2nd method)

//Now we'll mae a varaible named app using express

const app =  express()

const port = 3000 //virtual ports, server will listen on port 3000

//app will listen on routes eg: / route p listen kro, and send response if you get any request , and to send response it'll be provided with a callback

app.get('/',(req,res)=>{

    res.send('hello express')

})

app.get('/twitter',(req,res)=>{

    res.send('hello twitter')

})

//now app will listen on main port and will give a callback

app.listen(port,()=>{

    console.log(`example app listening on port ${port}`)

})

**HENCE WE’VE MADE A SERVER SUCCESSFULLY, NOW WE WILL DEPLOY IT.**

**BUT BEFORE DEPLOYING WE SHOULD KEEP SOME THINGS IN MIND-**

* We need to be careful with special variables (sensitive info like username, password, db info, etc)
* And, in our computer we are using 3000 port , but it is not necessary that on the other person’s computer 3000 port must be free, or maybe they are forcing their ports on our applications.

So we need a package - **dotenv**

**Npm install dotenv**

* **Now we will make a file .env, and we’ll put our sensitive info variables in it.**

PORT = 3000

* Now, in index.js, write

**require(‘dotenv’).config()** on top of the code

* And, now change port variables in index.js with **process.env.PORT**
* **We’ll push our code to git**

1. git init
2. Now before adding all the files to git, we will push imp and sensitive data to git ignore file
3. Make a file .gitignore and then add node\_modules and .env
4. Now git add .
5. Then, follow all the steps and push the code.

* **Now, got ot your app deploying platform-**

1. Select an app
2. Take code from github
3. Then set plan
4. Then, environment variables need to be set
5. Then follow the steps and build
6. And, after testing destroy the app to save balance/bill

**Connect Backend With Frontend**