**NODE JS**

1.**Node and Express Fundamentals -------------------**

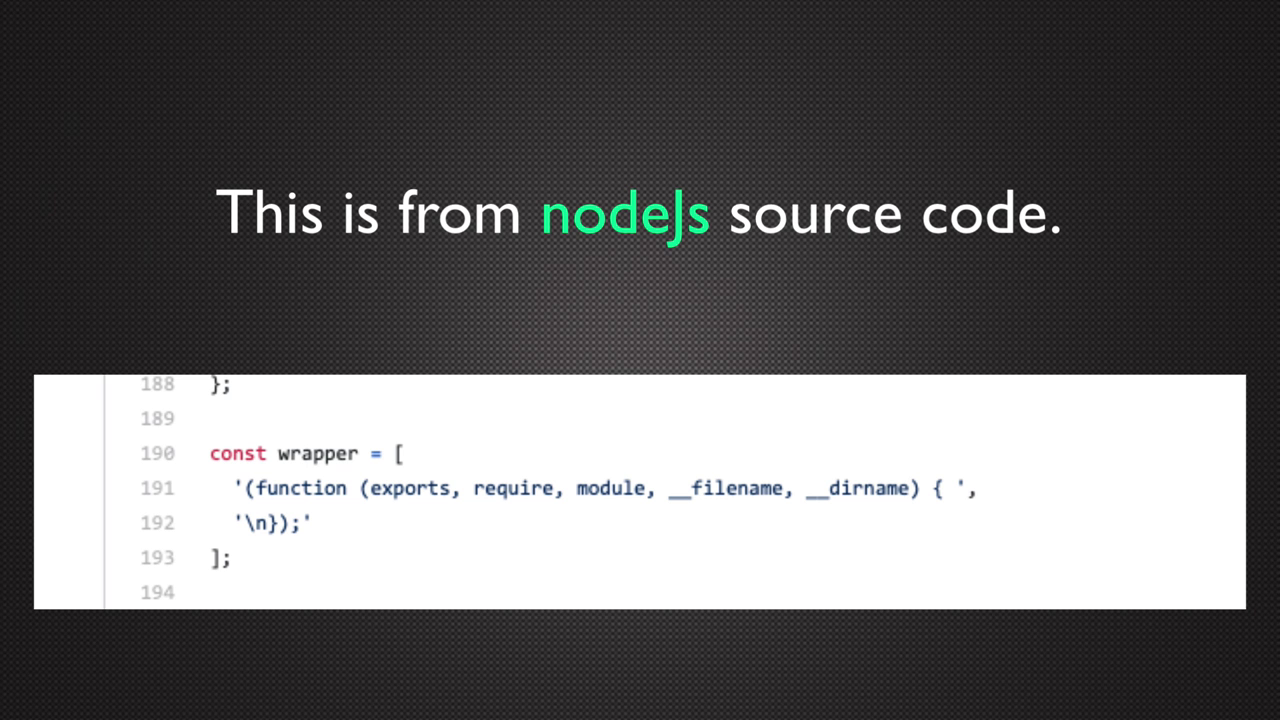
GLOBALS – NO WINDOW !!

Require --- function to use modules

Module --- info about current module

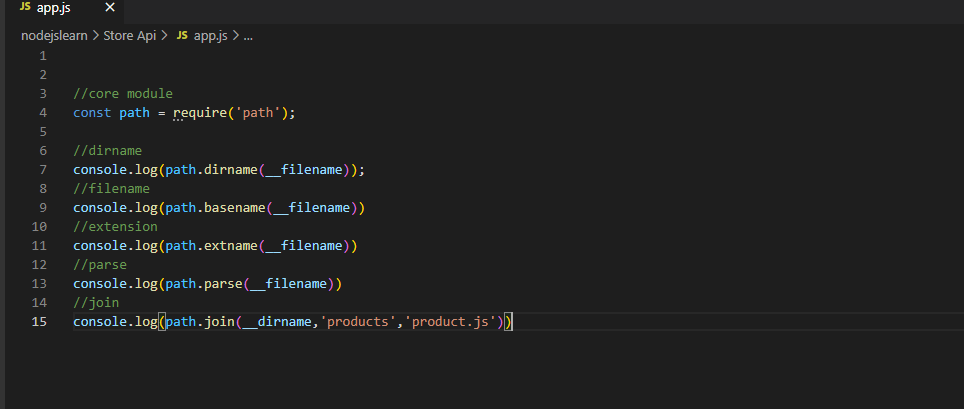
Process --- info about env where the program is begin executed

Every module in node js can be thought of as a immediately invoked function.

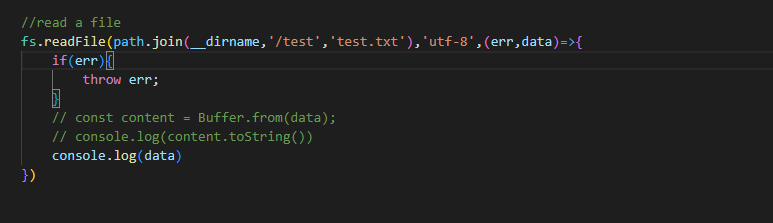


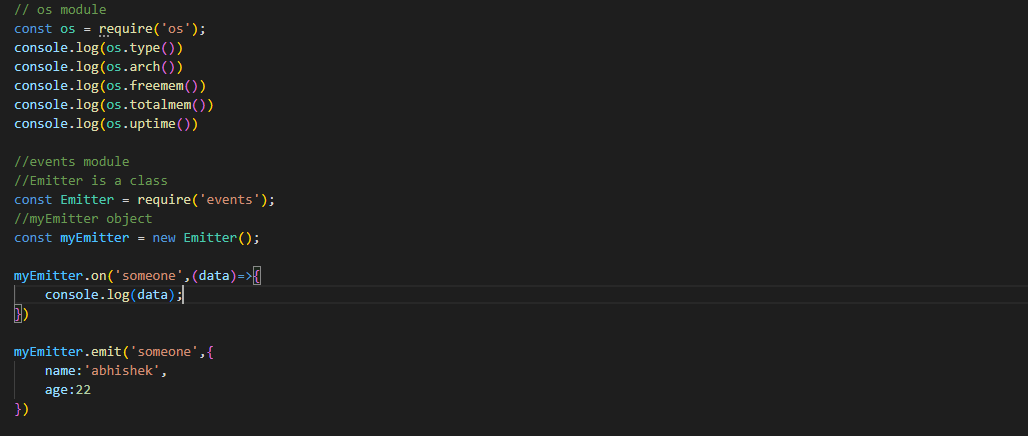
BUILT IN MODULE

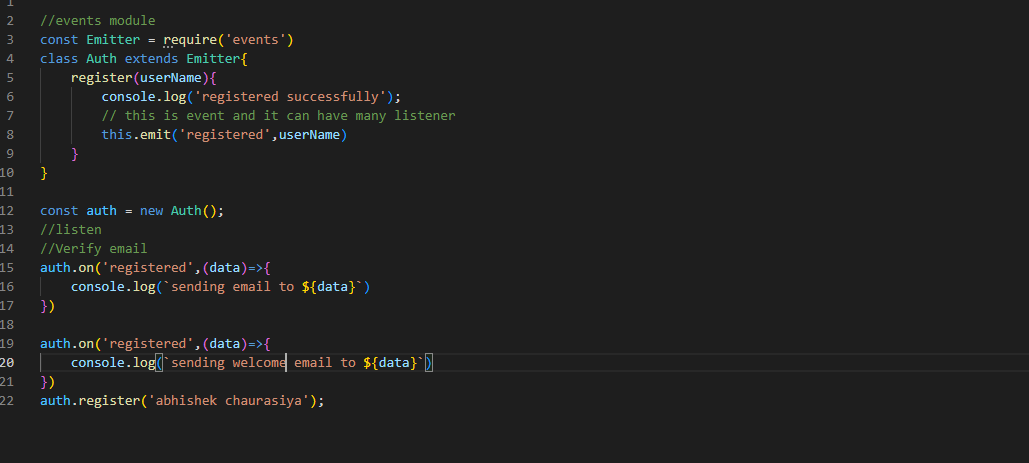
Os , Path , Fs , Http





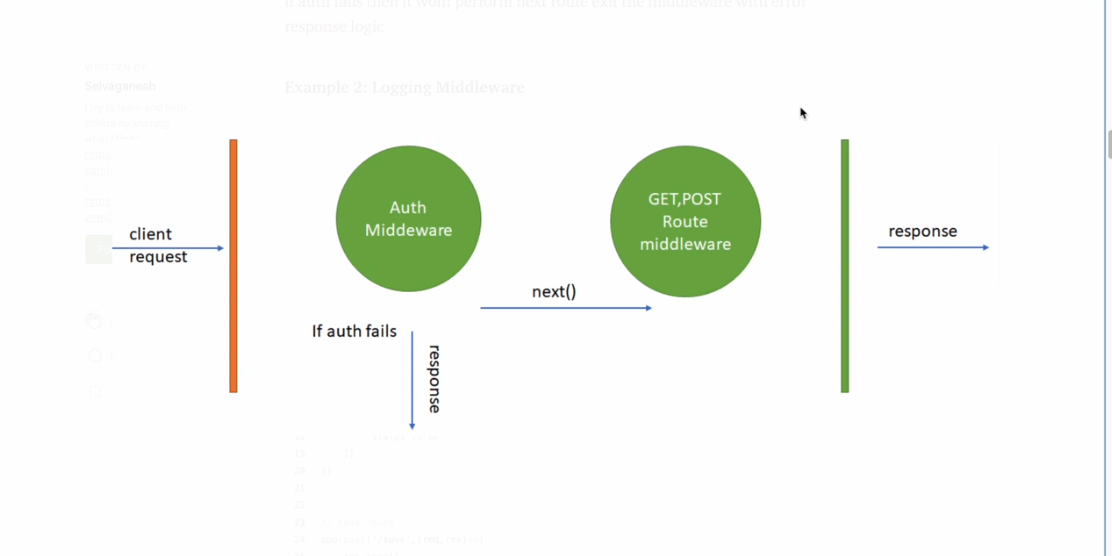






**EXPRESS JS**

Problem – the code will get messy in server.js or app.js and the solution provided is Solution - express router.



MIDDLEWARE---

It is a function that executes after the server receives the request

And before the controller action sends the response

Middleware execute before every single controller action by

Adding it to the application level.

\*\* Controllers actions are just like middleware , they are essentially

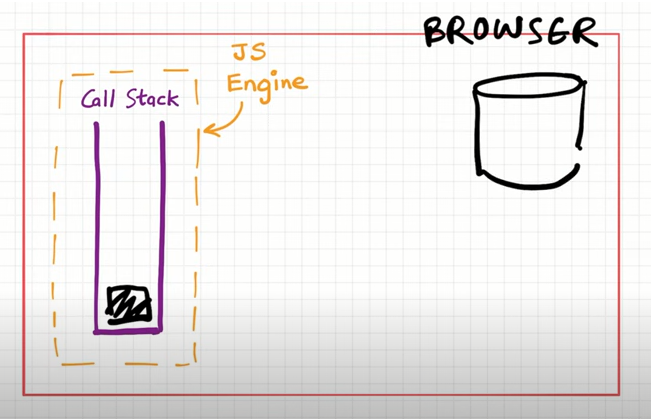
Middleware but with no middleware that comes after them

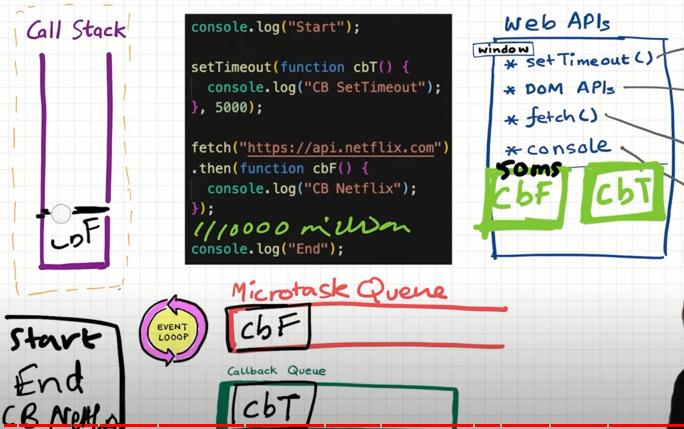
\*middleware will execute in order

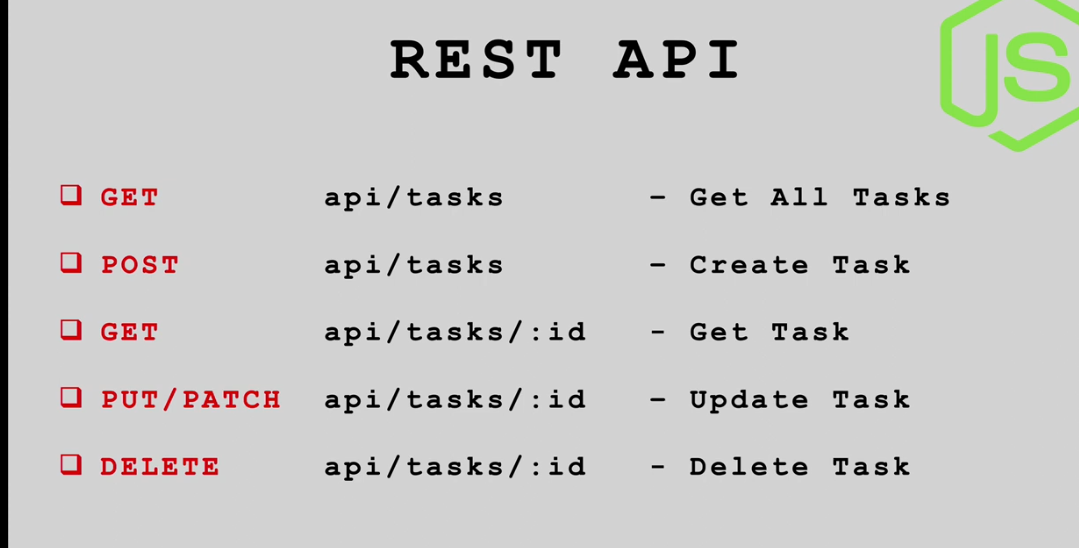
EVENT LOOP

JS has only one call stack.

Global execution context is created and push into the stack







 A REST Server simply provides access to resources and REST client accesses and modifies the resources using HTTP protocol.

**HTTP methods:**

* GET − Provides read-only access to a resource.
* PUT − Updates an existing resource or creates a new resource.
* DELETE − Removes a resource.
* POST − Creates a new resource.
* PATCH− Update/modify a resource



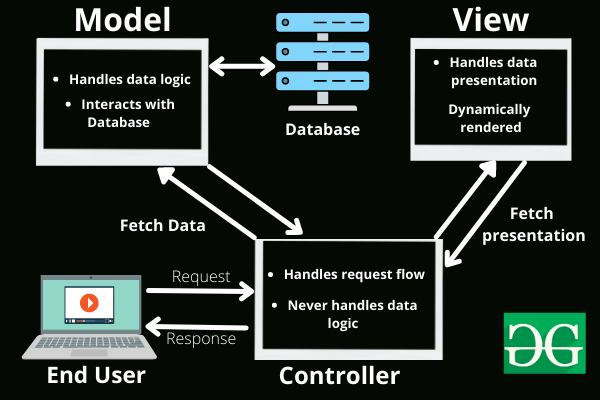
**Model** – It is a abstraction of the data in your mongodb database, which is represented

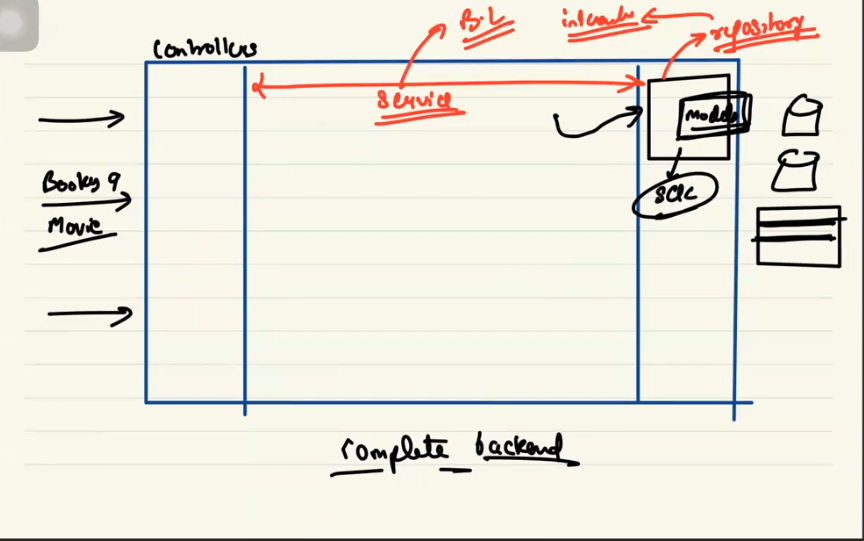
As a document . Because of this abstraction , you may use the ‘mongoose’ schemas

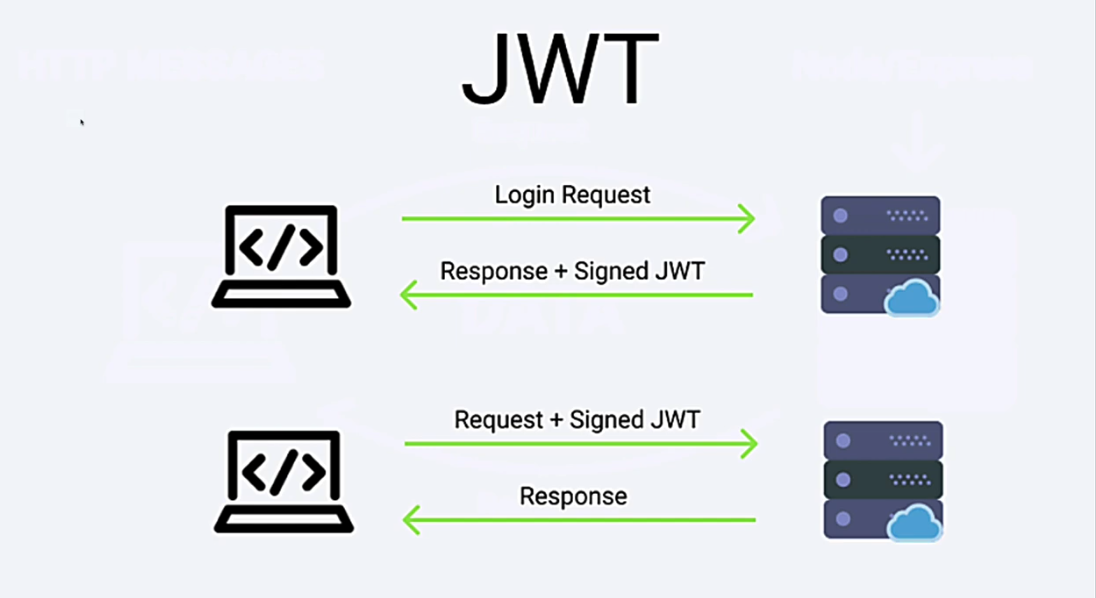
To construct a blueprint of how you want the added data to look and behave.

**Controllers** – controllers are responsible for handling incoming requests and returning

Responses to the client







\* The app.use() method in express is a \*middleware handler. An Express application is essentially a series of middleware function calls.

\* An Express application can use 5 different types of middleware, of which these two are majorly used:

1. Application-level middleware
2. Router-level middleware

App.use() is used to bind \*application-level middleware to an instance of the app object which is instantiated on the creation of the Express server (router.use() for router-level middleware).

Syntax : app.use(path, middleware function/s)

Here, the path is optional. When no path is specified the function gets executed every time the app receives a request, irrespective of which URL has been hit.

**FILE UPLOAD- ------------------------------------------------------**

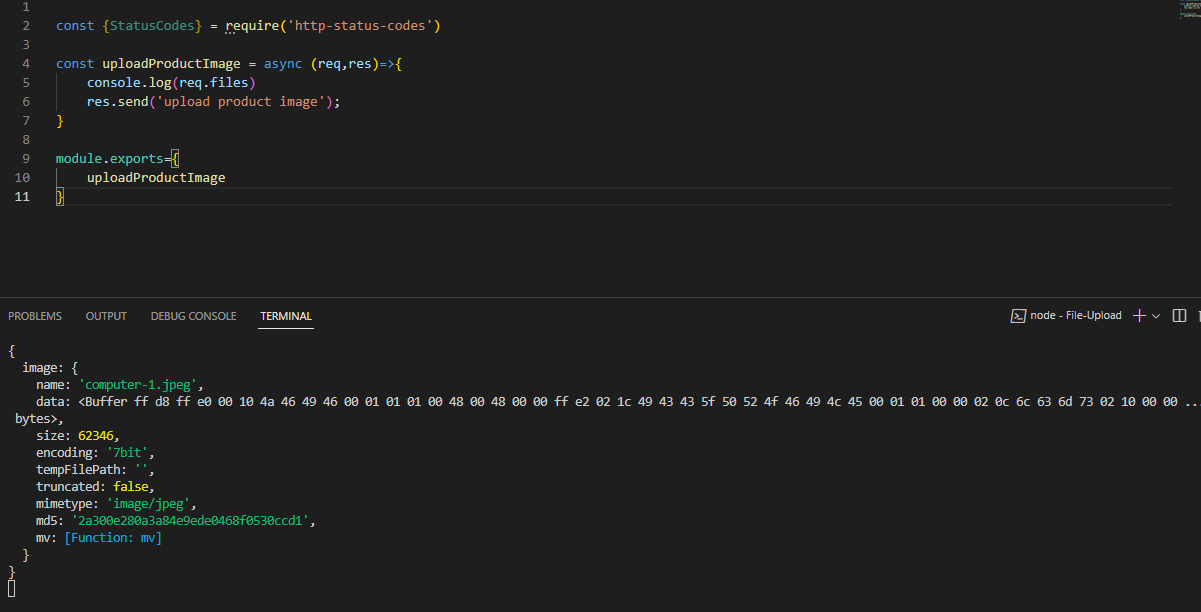
Const fileUpload = require(‘express-fileupload’)

App.use(fileUpload);

When you upload a file, the file will be accessible from req.files.

When a user upload what two situation we have?

First we want to store into our server and second we also want to publicly available.\_



When we test our application from postman or frontend , we upload image using postman it is in req.files and whatever the name we have given it in postman it is shown in the console.