Node JS course:

1.REPL: Read Eval Print Loop

-----------------------------

2.JS refresher:

Weakly typed : No explicit assignement.

Data types can be switched dynamically

OOL :Object oriented language

#To understand data types in JS:

https://academind.com/tutorials/reference-vs-primitive-values

#To undertsand more about this

https://academind.com/tutorials/this-keyword-function-references

#Spread operator: [...hobbies] = all the elements of existing array

#Rest operator:(...args) = To apply n unknown number of elements

-------------------------------

3.Node basics

#core modules :

http:launch server/send request

https:SSL server

fs(filesystem): read and write file

os, path, url

#Server creation:

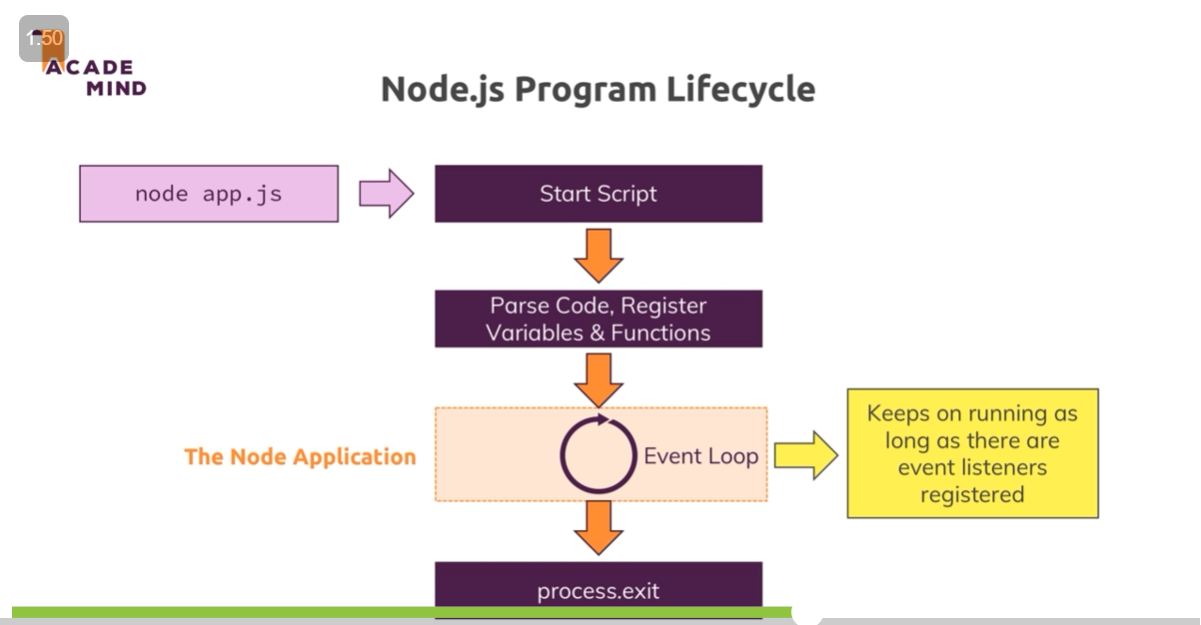
1.Import http- const http = require("http");

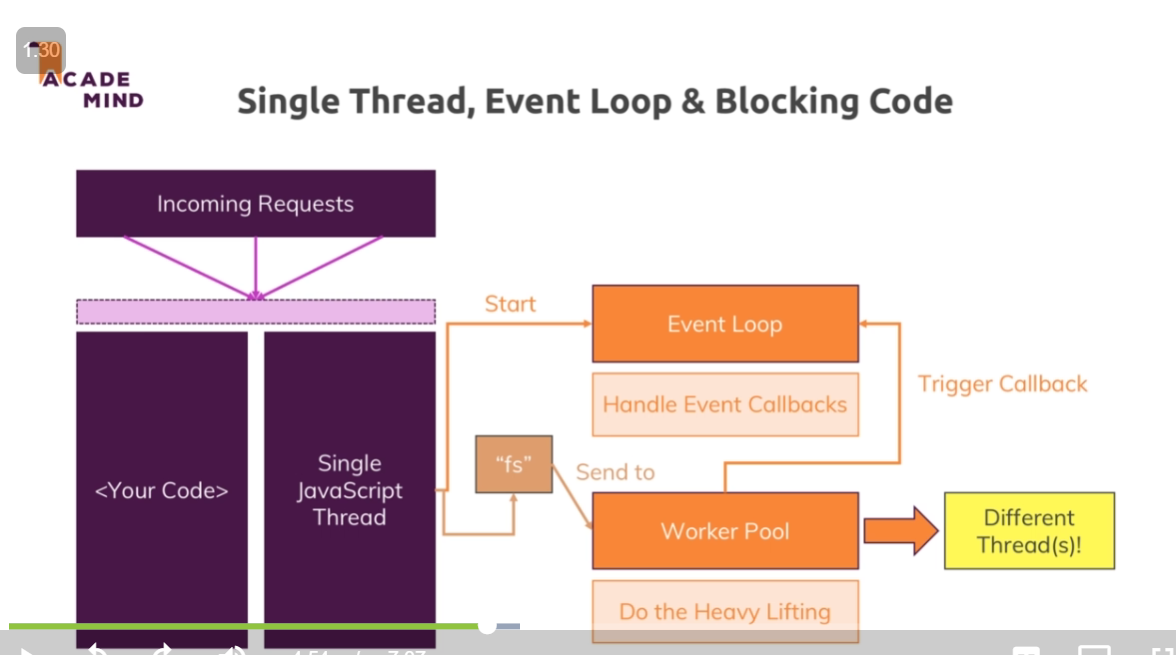
2.Create server by providing request listener to create server method

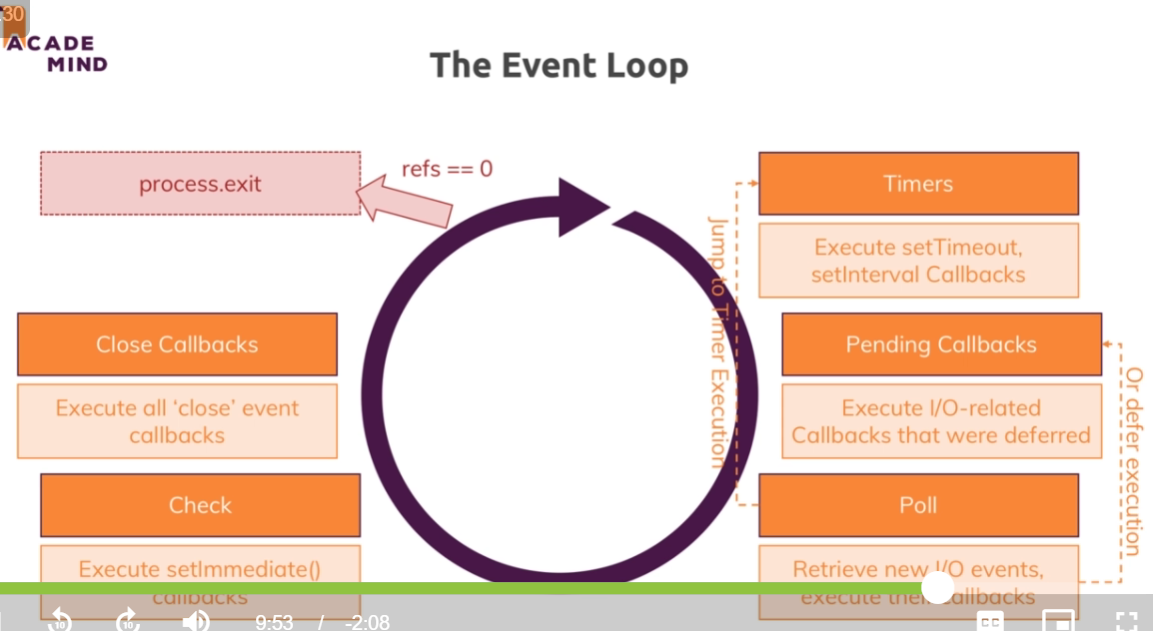
-const server = http.createServer((req, res) => {console.log(req)});

3.Listen for request on server- server.listen()

server.listen(port?: number, hostname?: string, backlog?: number, listeningListener?: () => void)







#Node uses event-driven approach

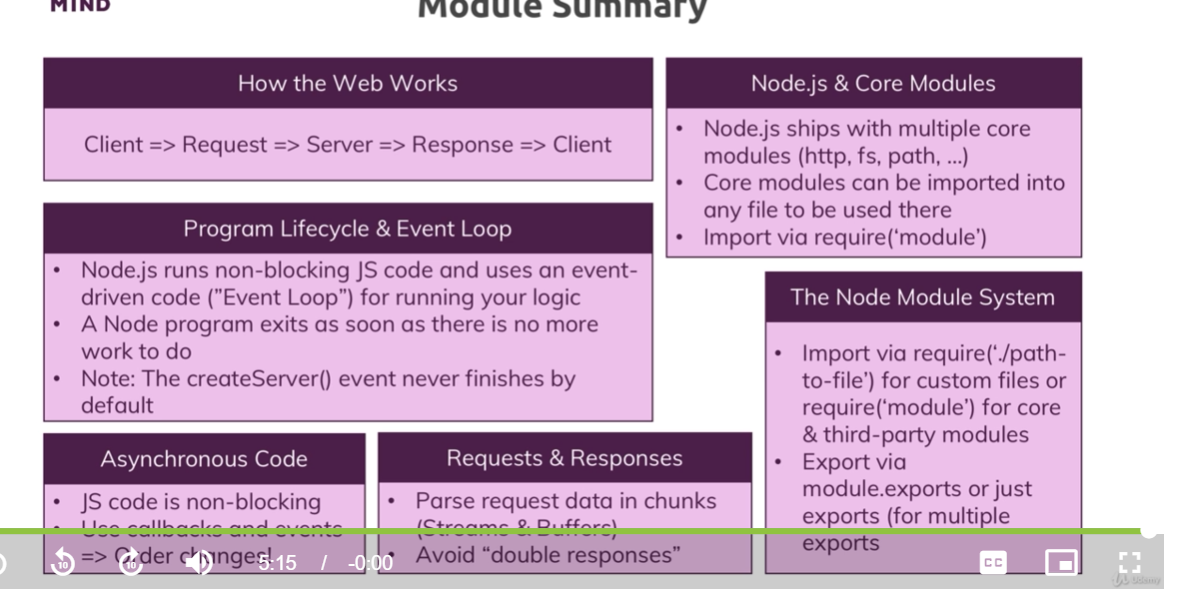
single thereaded js

# To disconnect server = process.exit()

# To set response header = response.setHeader('content-type','text/html')

#writeFile (fileName,content, error callback funtion)

#module.exports=function\_name;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

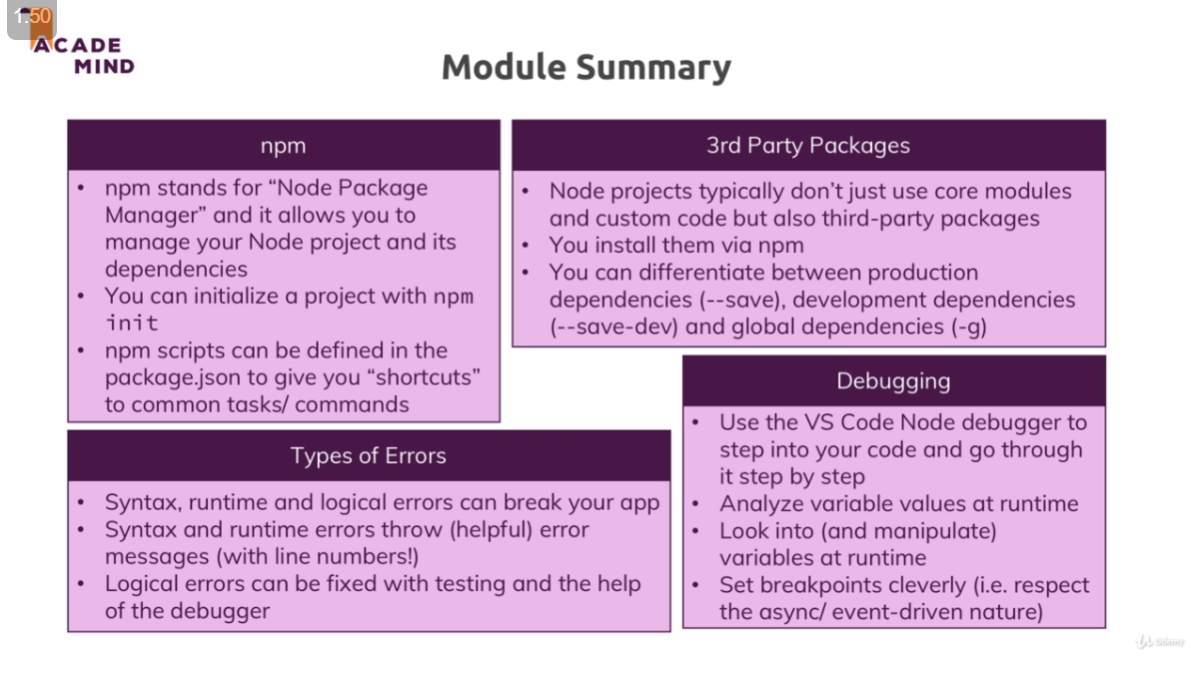
4.Debugging

Npm init : Will create package.json file

In package json file-> script: “start”:”node app.js”

npm start = will run script and thus project

for other scripts -> npm run script\_name



const express = require("express");

const app = express();

//Middleware function

app.use((req, res, next) => {

  console.log("In middleware");

  next(); //allows to proceed to next middleware function

});

app.use((req, res, next) => {

  console.log("In other middleware");

  res.send("<h1>express js course</h1>");

});

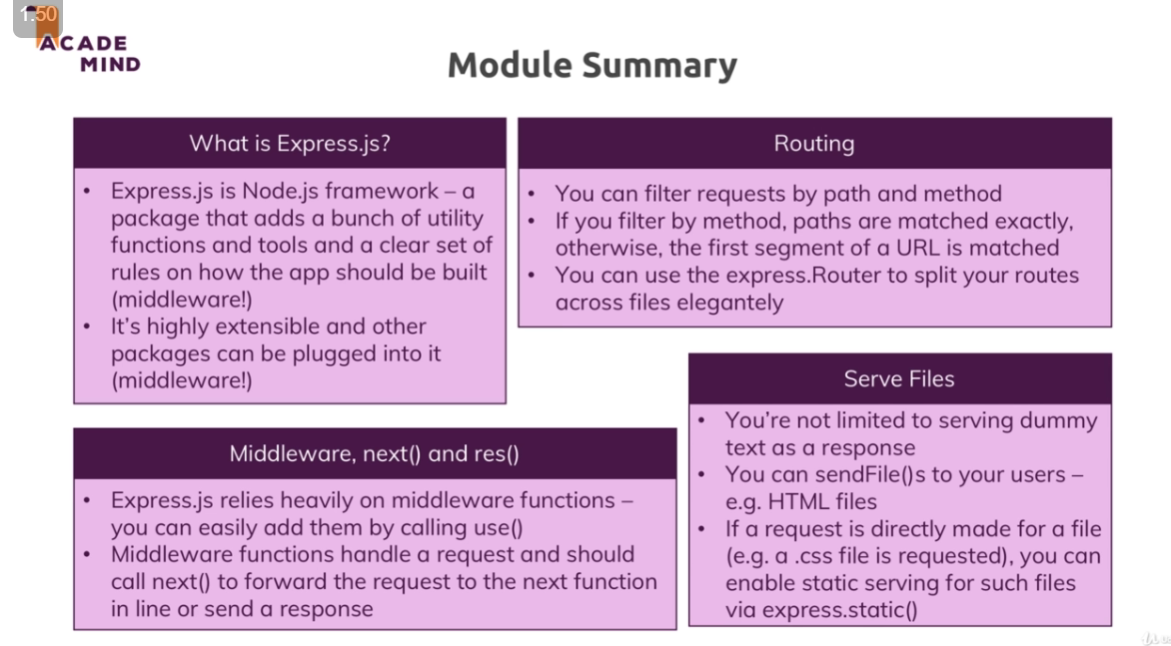
//In order to server running/ takes optional arguments:

app.listen(3000);

res.redirect=~res.location

app.get => for get request

app.post=> for post request



Express path fn:

module.exports = path.dirname(process.mainModule.filename);

const rootDir = require("../util/path");

res.sendFile(path.join(rootDir, "views", "shop.html"));

Static files:

In app.js

app.use(express.static(path.join(\_\_dirname, "public")));

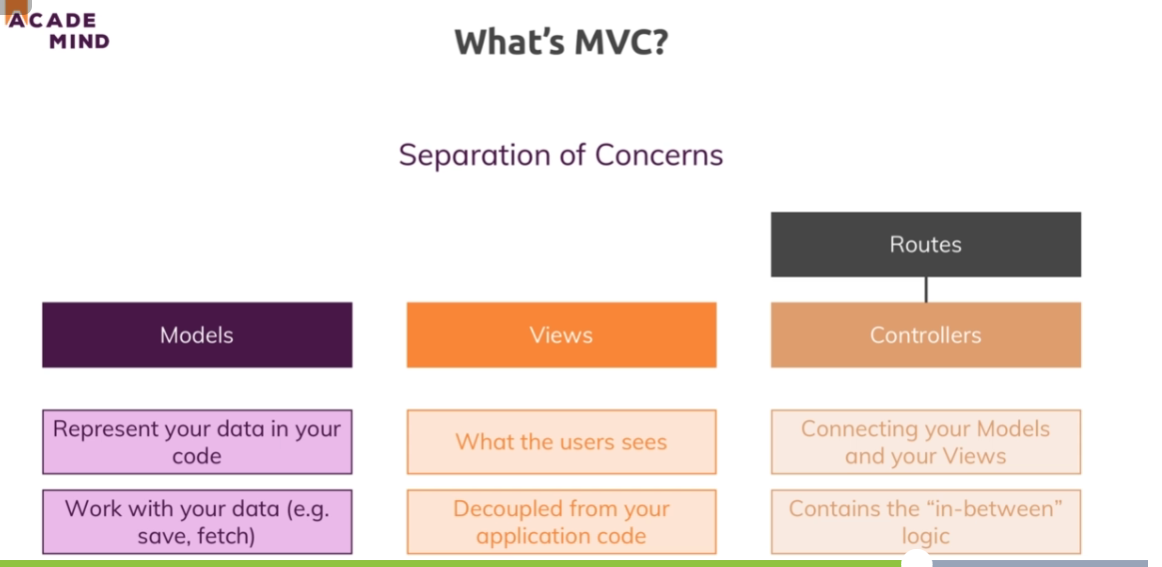
In html page :

    <link rel="stylesheet" href="/css/main.css" />

----- 5hr 27min course completed-----

Templating engines

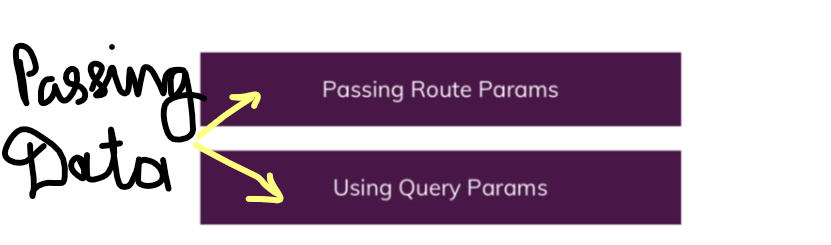
--- 7hr 30min----------



---- 8.15 min -------

Enhancing app

---- 8.50 min -------



To pass product id in path :=

<a href='/products/<%= product.id %>' class='btn'>Details</a>

To pass the param in route := ‘:’ symbol is used

router.get("/products/:productId");

Fb: Heroku@2312

To extract the param in controller:= req.params.productId

exports.getProduct = (req, res, next) => {

  const prodId = req.params.productId;

  Product.findProductById(prodId, (product) => console.log(product));

  res.redirect("/");

};

In model create method to get individual product which matches the id :

  static findProductById(id, cb) {

    readFromFile((products) => {

      const product = products.find((p) => p.id === id);

      cb(product);

    });

  }

A) Passing route params :

1. Req.params.prodId ->

2. Req.body.prodId -> In post method, u need to use data in hidden input

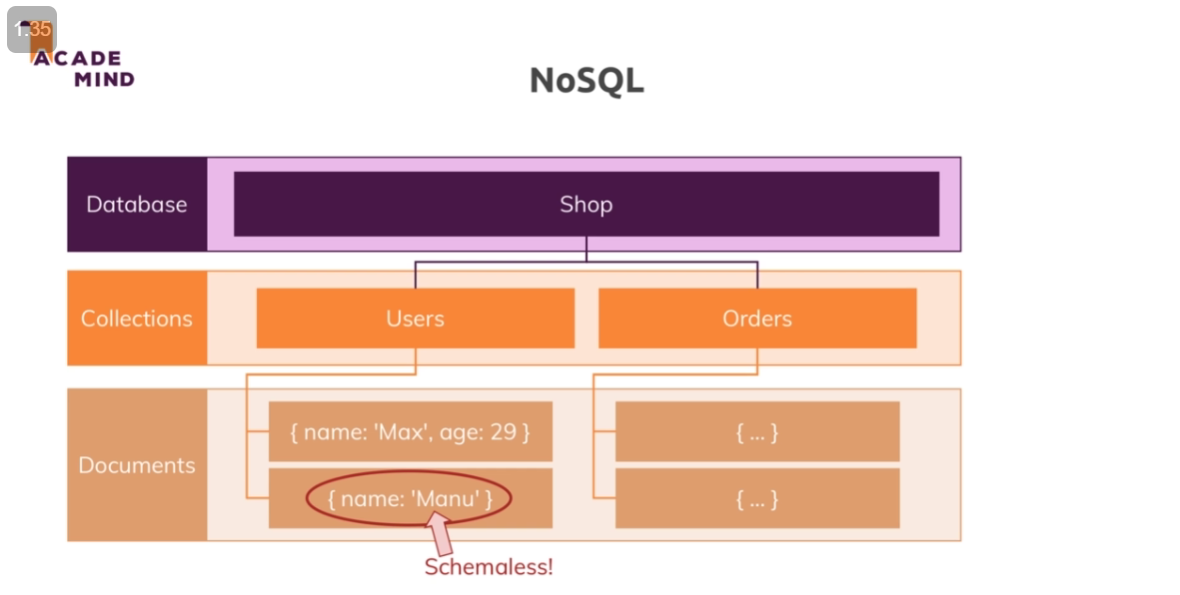
<input type="hidden" value="<%= product.id %>" name="productId" >

B)

--------10:30 hrs----------------

**SQL- 1) Strong schema 2) Relationships between table**

**NO SQL**



SQL password : root123

1. Create Pool of connection because every query requires connection :

const mysql = require("mysql2");

const pool = mysql.createPool({

  host: "localhost",

  user: "root",

  database: "node\_course",

  password: "root123",

});

module.exports = pool.promise();

export in form of promise because connections need to be processed asynchronously.

db.execute("SELECT \* from products")

  .then((res) => {

    console.log(res);

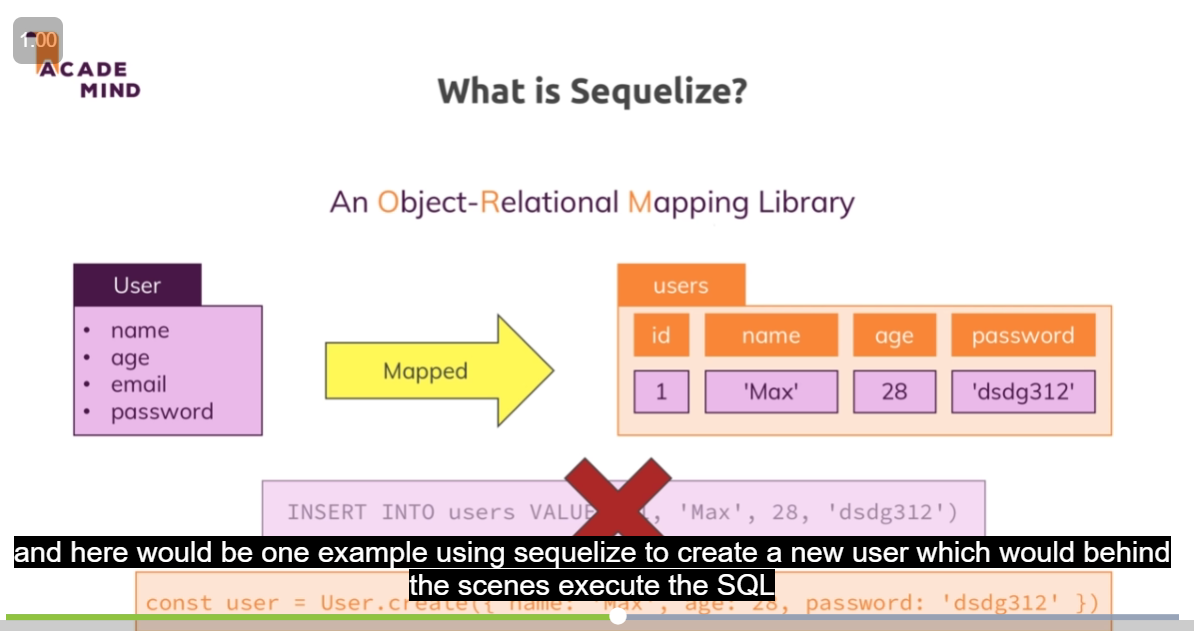
  })

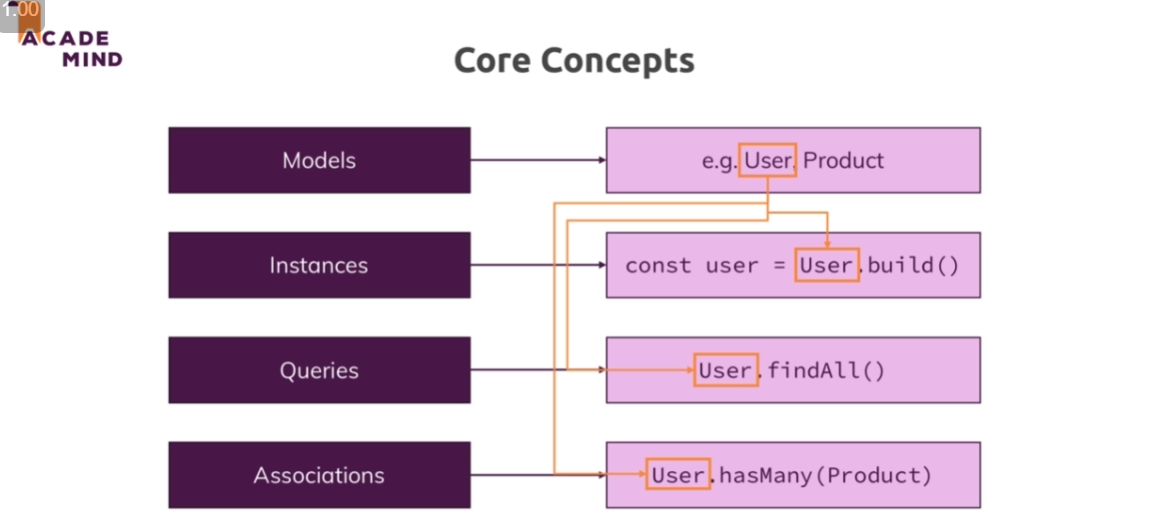
  .catch((err) => {

    console.log(err);

  });

------11:20hrs---------

**SEQUELIZE**

****