Part 1 : Setting up applications :

Docker & MySQL Server

① Download & Install docker desktop

https://docs.docker.com/desktop/install/windows-install

Open Docker Desktop Software

② Installing mysql server https://hub.docker.com/r/mysql/mysql-server

In Command prompt :

i) Download image :

```
docker pull mysql/mysql-server
```

ii) Start a server instance

```
docker run -p 3306:3306 -p 33060:33060 --name=mysqlserver
-d mysql/mysql-server
```

. Check the generated root password from the logs.

. To display log in the CMD : [or check in Docker Desktop]

```
docker logs mysqlserver
```

iii) Change password :-

a) Get into docker shell

```
docker exec -it mysqlserver /bin/bash
```

b) Login as root user

```
mysql -u root -p
```

c) Change the password

```
alter user 'root'@'localhost' identified by 'password';
```

Single quotes are required

d) Allow root user to log in from any IP

```
update mysql.user set host ='%' where user ='root';
```

e) Fix issue where node.js can not connect to sql server

∵ ER_NOT_SUPPORTED_AUTH_MODE

```
alter user 'root'@'%' identified with
mysql_native_password by 'password';
```

f)
```
flush privileges;
exit;
```

g) Restart mysql server

```
docker restart mysqlserver
```

Node.js

① Download & Install Node.js

https://nodejs.org/en/download

② Verify that it is installed properly

```
// in CMD
node --version
// You should see the version of node js installed.
```

Part 2

① Open a project folder in VS Code

② Open VS Code terminal [Ctrl + Shift + `] ← The key above tab key

③ Initialize

```
npm init -y
```

-y → optional to skip the prompts.

④ Download modules :

```
npm i express mysql nodemon dotenv
```

// Create a .env file in project root

⑤ edit scripts in package.json to:

```
Scripts: {
    "dev" : "nodemon -r dotenv/config src/app.js -e js,html,env",
    "start" : "node -r dotenv/config src/app.js"
}
```

Conventions for folder structures [not a must]

Project Root Folder

- public
  - css
  - images
  - js
  - .html

  } files available to users

- src
  - app.js
  - db
  - middleware
  - routers

  } logic

- .env → Contains all the credentials
- package.json → metadata for the project

Sample file contents:

src/db/sql.js

```
const mysql = require('mysql')

var connection = mysql.createConnection({
        host: 'localhost'
        user: 'root'
        password: process.env.dbPassword
        database: 'miniproject'
        multipleStatements: true
})

connection.connect((err) => {
    if (err) {
        console.log('Unable to Connect', err)
    } else {
        console.log('Connection established')
})

module.exports = {connection, mysql}
```

Src/router/*.js

```
const express = require('express')
const router = new express.Router()

router.method (...)
        ⋮
module.exports = router
```

Src/app.js

```
const express = require('express')

const path = require('path')
const app = express()
const port = 3000

const myRouter = require(pathToMyRouter.js)
const pubDir = path.join(__dirname, '../public')
app.use(express.json())
app.use(express.static(pubDir))
app.use(myRouter)
app.listen(port, () = {
        console.log('Server is running')
})
```

req.body
req.query.variable (var=value)
req.params.key ('/:key')

public/js ...

```
fetch(path, {
      method: 'POST',
      headers: {
          'Content-Type': 'application/json',
      },
      body: JSON.stringify(...)
}).then((response) => {...}.catch((err) => {...})
```

for calling the APIs from javascript

Change according to the path

Note

* Do not upload node_modules & .env files onto github
* Explore modules like
    - hbs (handlebars) [to serve dynamic contents]
    - bcrypt [to hash password before saving it in database]
    - sendgrid [sending emails]
    - and much more

      [npmjs.org]