 Summary - Express Basics

API vs REST API

- API(Application Programming Interface) is a way for two programs communicate with each other.
- REST API stands for REpresentational State Transfer API and it is a specific way of building our APIs.
- REST API is the same as API but we have to follow some rules to build that API. If we follow some rules then simple API become REST API.
- Rule 01 - Define separate API url for each piece of data
- Rule 02 - Different type of actions should use specific HTTP methods.

HTTP methods

- GET - For getting data from the server
- POST - For create a new data in server.
- PUT - For updating whole data
- PATCH - For updating little data in the existing data
- DELETE - For deleting data from the server

Create express server

```
const express = require("express");
const app = express();

app.listen(3000, () => console.log("Server is listening on port 3000..."));
```

APIs Template in express

```
app.method("endpoint", (req, res) => {
  // Logic for handling this route
});

// method - get, post, put, patch, delete
// endpoint - i.g. "/posts", "/users/follow"
```

Create a Todo

```
app.post("/todos", (req, res) => {
  const todo = req.body;

  const newTodo = {
    id: todosArr[todosArr.length - 1].id + 1,
    task: todo.task,
    tags: todo.tags,
    status: todo.status,
  };

  todosArr.push(newTodo);
  res.status(201).json(newTodo);
});
```

Getting Todos

```
// getting all todos
app.get("/todos", (req, res) => {
  res.json(todosArr);
});

// getting specific todo by id (Route parameter)
app.get("/todos/:id", (req, res) => {
  const todoId = parseInt(req.params.id);
```

```
  const todo = todosArr.find((t) => t.id === todoId);
  res.json(todo);
});
```

- When we are working on node application we have to manually restart our application in terminal which is really annoying.
- So for automatically restarting our server we use nodemon.

```
// Installing nodemon as global package
npm i -g nodemon

// Starting node application using nodemon
nodemon index.js
```

Environment Variables

- Sometimes we have to define some secret data in our application. At that time we can define them into .env file.

```
// Accessing Environment Variable
const port = process.env.PORT
```

Route Parameter & Query Parameter

- Both are used to pass data in URL. But route parameters are used to pass data which are must required. Where query parameters are used to pass data which are additional.
- If frontend didn't pass route parameters then our API will give us error, but if we didn't pass query parameters then our API will not give us error. It is an additional details.

```
// Accessing Route Parameters
app.get("/todos/:id", (req,res) => {
```

```
    const id = req.params.id;
})

// Accessing Query Parameters
app.get("/todos/", (req,res) => {
    const sortBy = req.query.sortBy;
})
```

- We use `res.send()` for simple responses, whether text, HTML, or objects. and we will use `res.json()` when you want to ensure that the response is in JSON format and properly formatted for JSON clients.

Updating Todos

```
app.put("/todos/:id", (req, res) => {
  const id = parseInt(req.params.id);
  const { task, tags, status } = req.body;

  const todoIndex = todosArr.findIndex((t) => t.id === id);

  if (todoIndex === -1) {
    return res.status(404).json({ message: "Todo not found!" });
  }

  if (task) {
    todosArr[todoIndex].task = task;
  }
  if (tags) {
    todosArr[todoIndex].tags = tags;
  }
  if (status) {
    todosArr[todoIndex].status = status;
  }

  res.json(todosArr[todoIndex]);
});
```

Deleting Todo by id

```
app.delete("/todos/:id", (req, res) => {
  const id = parseInt(req.params.id);

  const todoIndex = todosArr.findIndex((t) => t.id === id);

  if (todoIndex === -1) {
    return res.status(404).json({ message: "Todo not found!" });
  }

  todosArr.splice(todoIndex, 1);
  res.json({ message: "Todo deleted successfully!" });
});
```

Some useful status code

- 200 - all ok
- 201 - created successfully
- 400 - bad request done by frontend
- 404 - not found
- 500 - Internal server error (something goes wrong on server)

The ultimate Node JS Course ~ Code Bless You❤️