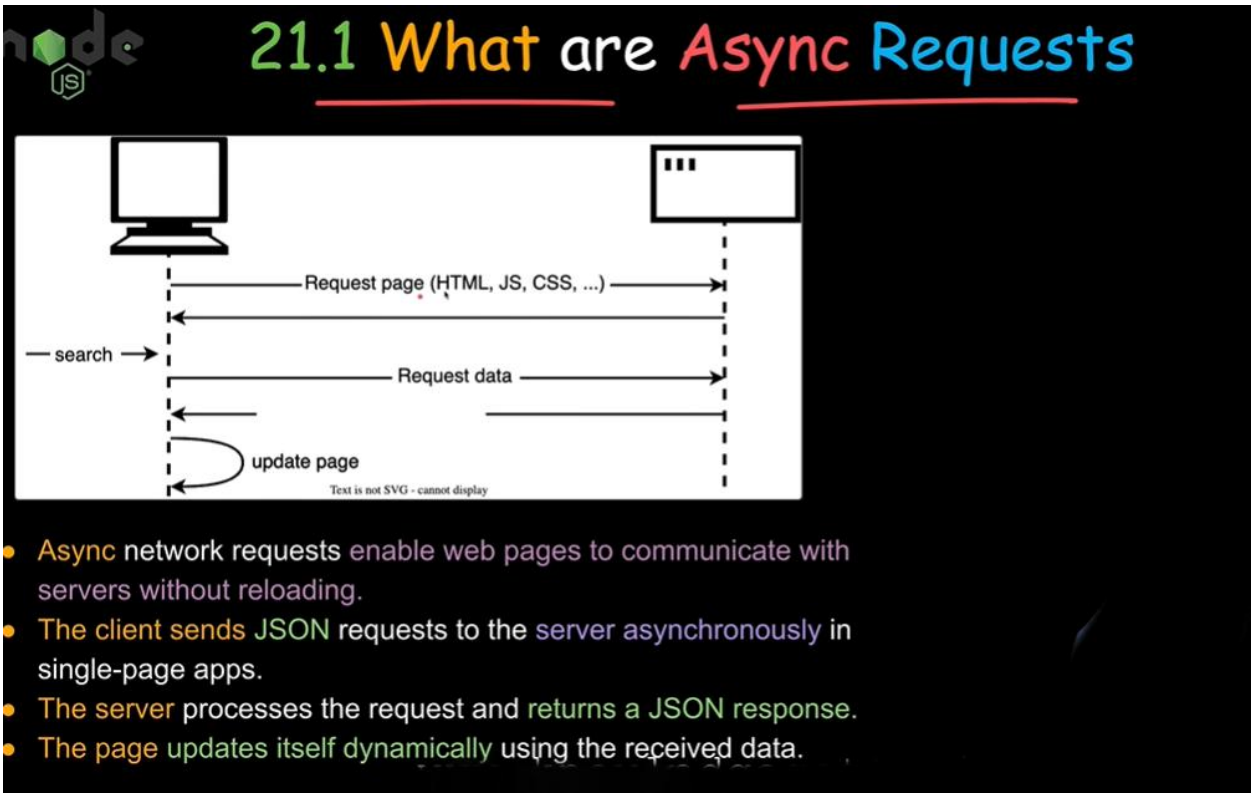
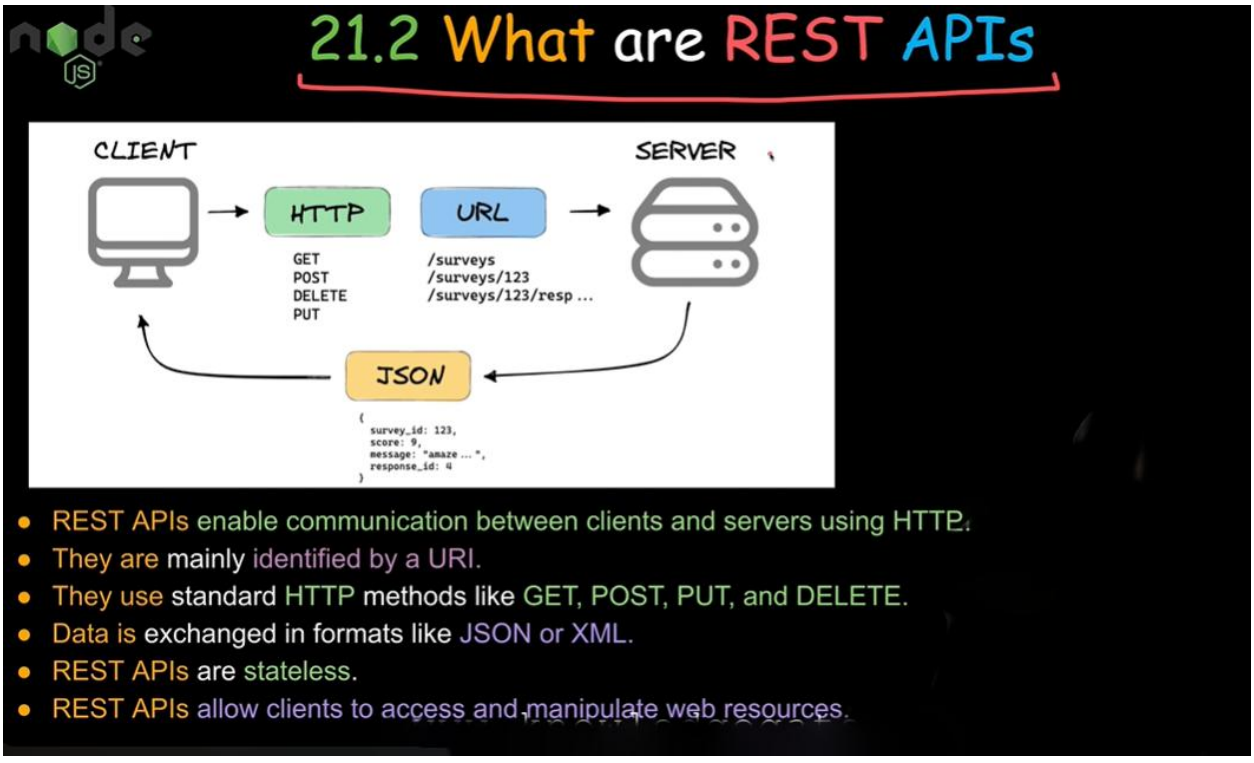
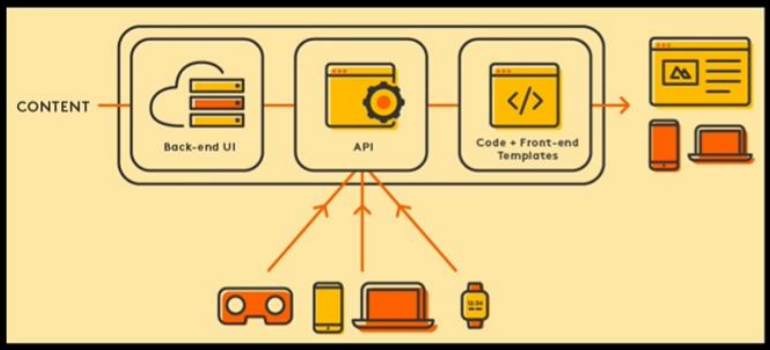
Json Request and Rest API





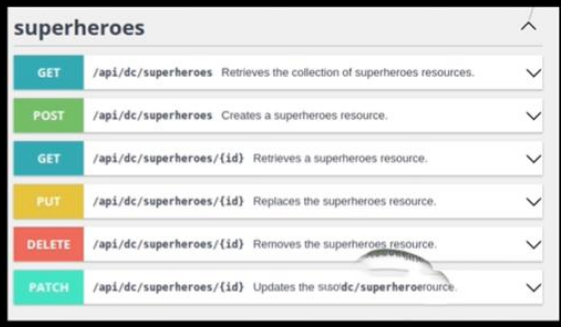
21.3 Decoupling Frontend & Backend



- Separating front-end and back-end allows independent development and scaling.
- REST APIs serve as a communication layer between them.
- Front-end interacts with back-end through standardized RESTful calls.
- Decoupling enhances flexibility and simplifies maintenance.
- REST APIs enable front-end updates without altering back-end code.

21.4 Routes & HTTP Methods

- REST API routes define the endpoints (URLs) where resources can be accessed by clients.
- GET: Retrieves data from the server at the specified route.
- POST: Sends new data to the server to create a resource.
- PUT: Updates or replaces an existing resource at a given route.
- DELETE: Removes a resource from the server at the specified route.
- PATCH: Partially updates an existing resource with new data.



21.5 REST Core Concepts

- Statelessness: Each request contains all necessary information; the server maintains no client session.
- Uniform Interface: Standardized communication using HTTP methods like GET, POST, PUT, DELETE.
- Client-Server Separation: Independent development of front-end and back-end components
- Cacheability: Responses indicate if they can be cached to improve performance.
- Layered System: Architecture allows for multiple layers between client and server.
- Code on Demand (Optional): Servers can extend client functionality by sending ex code.

21.6 First API Todo App

8,9.

```
const bodyParser = require("body-parser");
const cors = require("cors");

app.use(bodyParser.urlencoded({ extended: true }));
app.use(cors());
app.use(express.json());
app.use(itemsRouter);
app.use(errorController.get404);
```

itemsService.js

20-RestApi_JsonRequest > todoApp-Frontend > services > itemsService.js > markTodoItems > res

```
const mapServerItemToLocalItem = (serverItem) => ({
  Id: serverItem._id,
  Name: serverItem.task,
  Date: serverItem.date,
  Completed: serverItem.completed,
});

export const addItemToServer = async (task, date) => {
  const response = await fetch("http://localhost:3001/api/todos/items", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify({ task, date }),
  });

  if (!response.ok) {
    throw new Error(`Server responded with status ${response.status}`);
  }
  const data = await response.json();
  return mapServerItemToLocalItem(data);
};

export const getTodoItems = async () => {
  const res = await fetch("http://localhost:3001/api/todos/getall");
  const itemsList = await res.json();
  // console.log('itemsList:',itemsList);
  return itemsList.map(mapServerItemToLocalItem);
};

export const markTodoItems = async (id) => {
  const res = await fetch(`http://localhost:3001/api/todos/completed/${id}`, {
    method: "PUT",
  });
  const item = await res.json();
  return mapServerItemToLocalItem(item);
};

export const deleteTodoItems = async (id) => {
  const res = await fetch(`http://localhost:3001/api/todos/delete/${id}`, {
    method: "DELETE",
  });
  const item = await res.json();
  return item._id;
};
```

todoItemsRouter.js

20-RestApi_JsonRequest > todoApp-Backend > routes > todoItemsRouter.js > ...

```
const express=require('express');
const { createTodoItems, getTodoItems, deleteTodoItems, markTodoItems } = require('../controllers/todoItemsController');

const todoItemsRouter=express.Router();

todoItemsRouter.post('/items',createTodoItems);
todoItemsRouter.get('/getall',getTodoItems);
todoItemsRouter.delete('/delete/:Id',deleteTodoItems);
todoItemsRouter.put('/completed/:Id',markTodoItems);

module.exports=todoItemsRouter;
```

todoItemsController.js

20-RestApi_JsonRequest > todoApp-Backend > controllers > todoItemsController.js > ...

```
const TodoItem = require("../models/todoItem");

exports.createTodoItems = async (req, res, next) => {
  // console.log("todoItems:", req.body);
  const { task, date } = req.body;
  const todoItem = new TodoItem({ task, date });
  await todoItem.save();
  res.status(201).json(todoItem);
};

exports.getTodoItems = async (req, res, next) => {
  const todoItems = await TodoItem.find();
  // console.log('all items are:',todoItems);
  res.json(todoItems);
};

exports.deleteTodoItems = async (req, res, next) => {
  const Id=req.params.Id;
  const deleteItem = await TodoItem.findByIdAndDelete(Id);
  res.status(201).json({_id:Id});
};

// exports.markTodoItems = async (req, res, next) => {
//   const Id=req.params.Id;
//   const todoItem = await TodoItem.findById(Id);
//   todoItem.completed=true;
//   await todoItem.save();
//   res.json(todoItem);
// };

exports.markTodoItems = async (req, res, next) => {
  const Id = req.params.Id;
  try {
    const todoItem = await TodoItem.findById(Id);
    if (!todoItem) {
      return res.status(404).json({ error: "Todo item not found" });
    }

    // Toggle completed value
    todoItem.completed = !todoItem.completed;

    await todoItem.save();
    res.json(todoItem);
  } catch (error) {
    res.status(500).json({ error: "Server error" });
  }
};

```

Sagar Gautam (2 hours ago) Ln 27, Col 28 Spaces: 2 UTF-8 CRLF