

CRUD with Express

by sandeep chopra

thanks for the feedback

additional lecture activities & resources

Slides with learning objectives outlined

VS-Code Live Share collaboration is enabled

Zoom Co-Annotation is enabled

Breakout Sessions & Pair Programming

Optional quizzes

Office hours and 1:1 Support

Let's get started


learning outcomes

Recap & Understand 3 Tier Web App Architecture

Quick Recap of ExpressJS

Understand CRUD operations & Restful Convention

Create & Implement Restful Routes

 Build a Basic Web Info App for SpaceX Rockets using ExpressJS

Learn Basic ExpressJS Debugging

Optional Quiz

What is 3 tier web architecture?

A 3-tier application architecture is a modular client-server architecture that consists of a presentation tier, an application tier and a data tier.

1. Presentation Tier (Client)
 - a. is a graphical user interface (GUI) that communicates with the other two tiers for e.g. browser on your computer
2. Application Tier (Server)
 - a. Handles the logic for the application for e.g. Flight Tickets Reservation System
3. Data Tier (Database)
 - a. The data tier stores the information and data related to the entire system for e.g. a user database

Quick Recap: Web 3 tier architecture

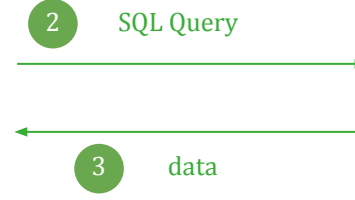
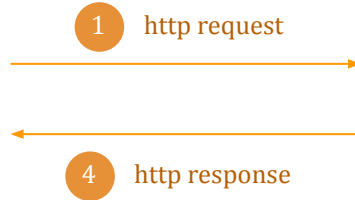
Client e.g. Browser



Server for e.g. Web Server



Database



Presentation e.g.
using HTML

Accepts HTTP requests,
handle logic & send back
response

Accepts HTTP requests,
handle logic & send back
response

Express 4.17.1

Fast, unopinionated, minimalist
web framework for [Node.js](#)

Express 4.17.1

Fast, unopinionated, minimalist
web framework for Node.js

Quick Recap of ExpressJS

It provides some useful **abstraction** and tools to help us with:

- Quickly setup a server
- Route Handling
- Templating Engine for e.g. EJS
- Middleware Functions



Let's create a spaceX web app

<https://bit.ly/3FTIKVq>

Restful API & http verbs

our **Resource** will be a single rocket 🚀

- Create a new Rocket [**C**: Create]
- List all Rockets [**R**: Read]
- Update a Rocket [**U**: Update]
- Delete a Rocket [**D**: Delete]

CRUD & REST conventions

REST stands for **R**epresentational **s**tate **t**ransfer

A map between HTTP verb/path combinations and CRUD actions users want to perform on a resource

- Create a new Rocket [**C**: Create]
- List all Rockets [**R**: Read]
- Update a Rocket [**U**: Update]
- Delete a Rocket [**D**: Delete]



- ★ POST */rockets*
- ★ GET */rockets*
- ★ PUT */rocket/:id/update*
- ★ DELETE */rocket/:id/delete*

Creating routes for our APP

our **Resource** will be a single rocket 🚀

HTTP Verb	Route	Action
GET	/rockets	List all the rockets .
GET	/rockets/:id	Get a specific rocket.
GET	/rockets/new	Display the new rocket form.
POST	/rockets	Create a new rocket.
GET	/rockets/:id/update	Get the form to Update the existing rocket.
PUT	/rockets/:id	Update the existing
DELETE	/rockets/:id/delete	Delete a specific rocket

* browsers only support GET and POST, so our routes may vary slightly



Questions?



hands on practice