BLACKBUCK ENGINEERS

The MERN Development Lab

Session 19 - ExpressJS REST API

BY AQUIB AJANI

BLACKBUCK ENGINEERS

GitHub Repository

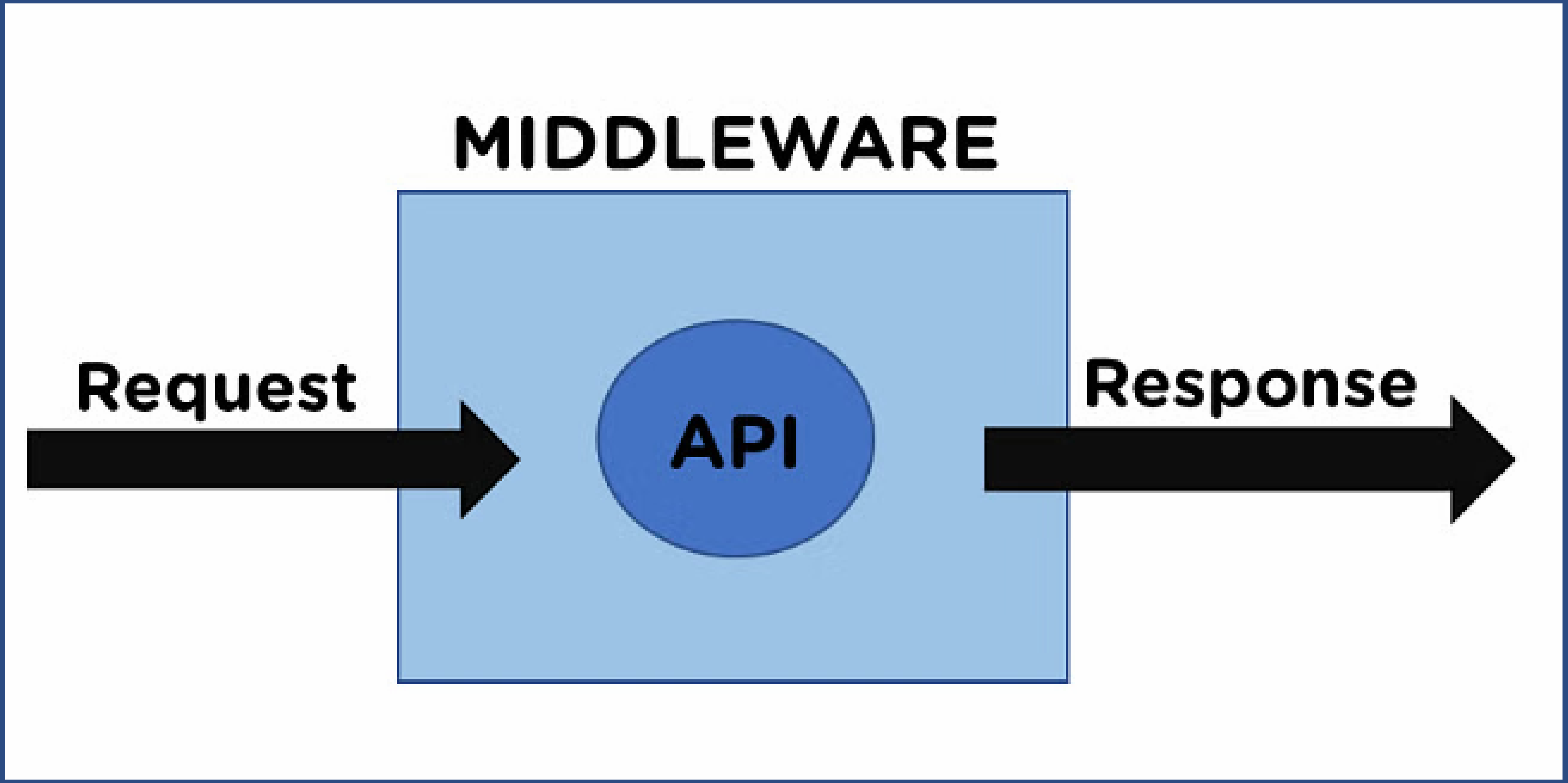
tabdeelstudios/blackbucks-iidt-internship-full-stack-feb-2024

https://github.com/tabdeelstudios/blackbucks-iidt-internship-full-stack-feb-2024

What is ExpressJS?

Express JS

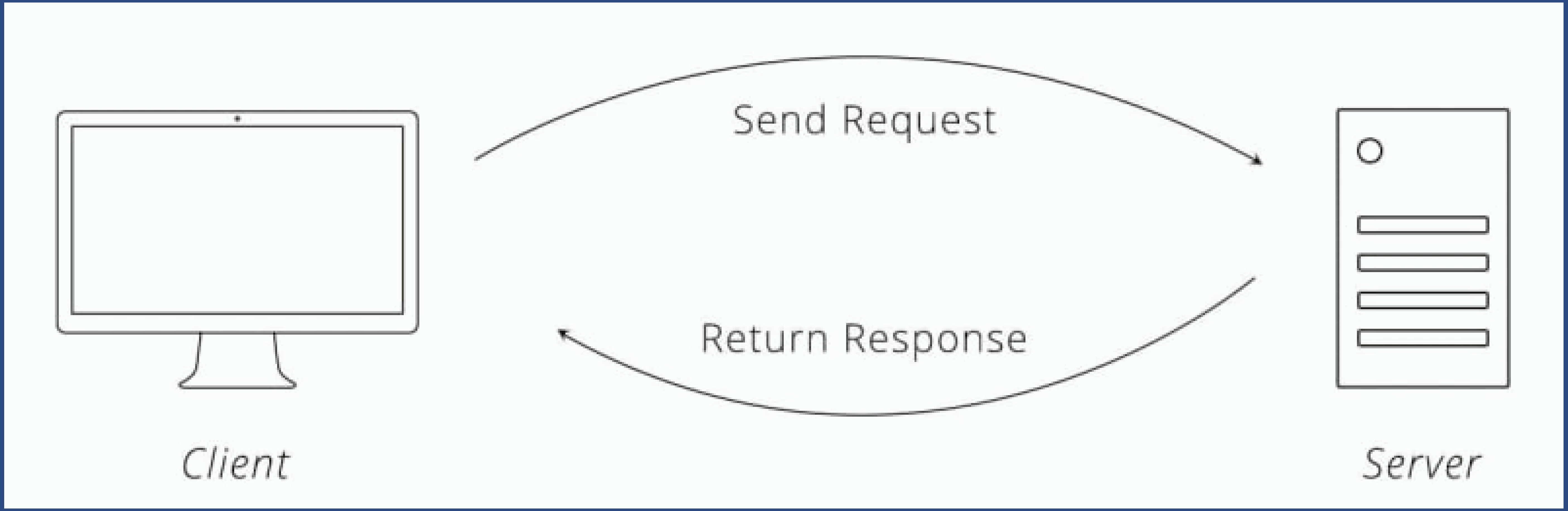
The role of ExpressJS



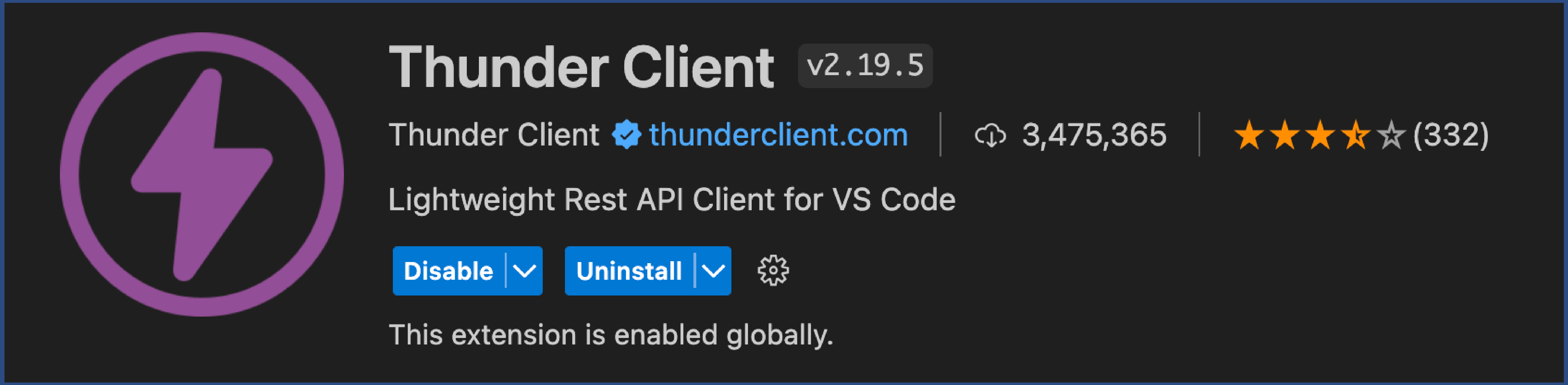
Creating a new Express Server

```
>_  npm init -y
```

Returning a “response”

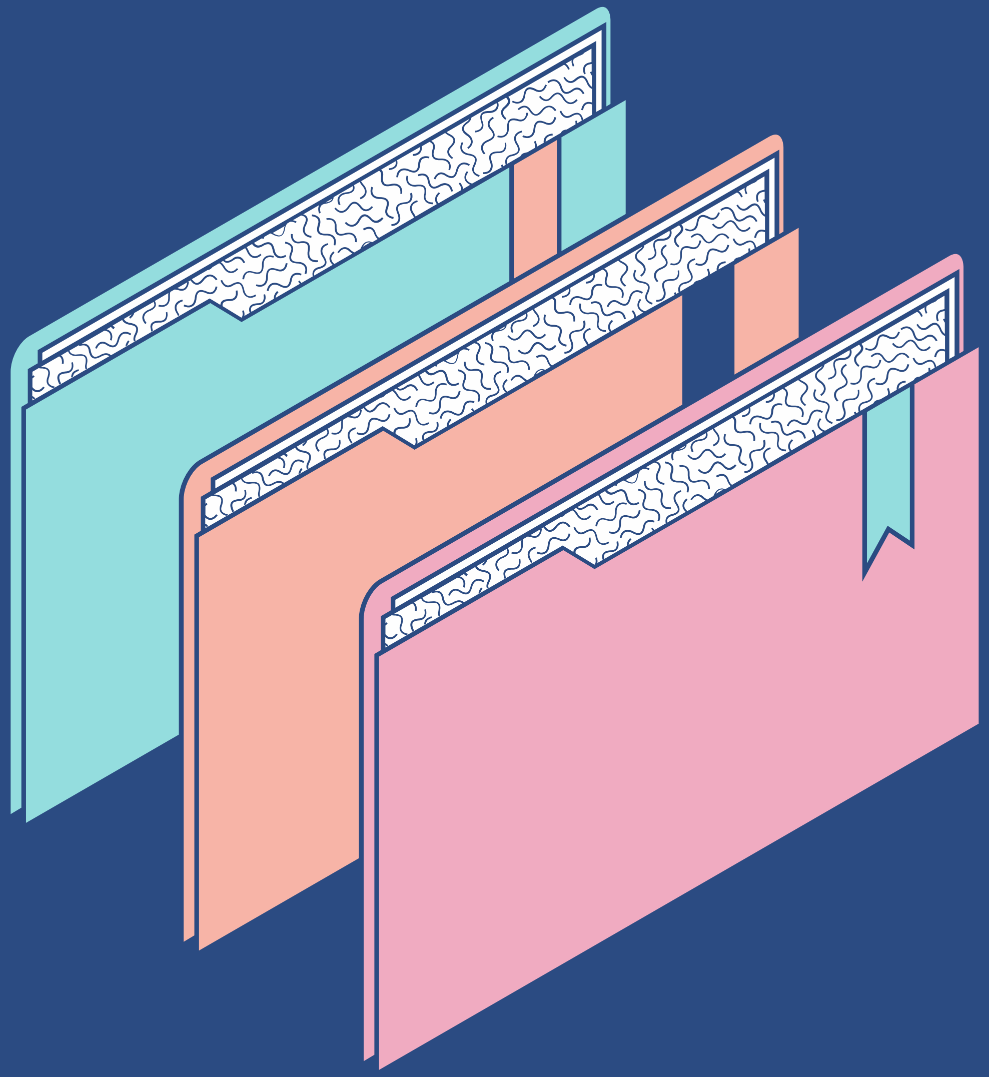


Testing Express Server with Thunder Client



Lightweight Rest API Client for VS Code

This extension is enabled globally.



Session Agenda

EXPRESSJS REST API

- What is an API?
- Properties of RESTful API
- HTTP Methods
- The GET Method
- The POST Method
- The PUT Method
- The PATCH Method
- The DELETE Method

Application Programming Interface

Properties of RESTful API

Properties of RESTful API

- **Stateless**: Requests contain all necessary information; no server state is stored.

Properties of RESTful API

- **Stateless**: Requests contain all necessary information; no server state is stored.
- **Client-Server Architecture**: Separation of concerns between client and server.

Properties of RESTful API

- **Stateless**: Requests contain all necessary information; no server state is stored.
- **Client-Server Architecture**: Separation of concerns between client and server.
- **Uniform Interface**: Standardised operations for all resources using HTTP methods.

Properties of RESTful API

- **Stateless**: Requests contain all necessary information; no server state is stored.
- **Client-Server Architecture**: Separation of concerns between client and server.
- **Uniform Interface**: Standardised operations for all resources using HTTP methods.
- **Resource-Based**: Resources identified by URIs; manipulated via HTTP methods.

Properties of RESTful API

- **Stateless**: Requests contain all necessary information; no server state is stored.
- **Client-Server Architecture**: Separation of concerns between client and server.
- **Uniform Interface**: Standardised operations for all resources using HTTP methods.
- **Resource-Based**: Resources identified by URIs; manipulated via HTTP methods.
- **Representation**: Resources represented using standard formats like JSON or XML.

Properties of RESTful API

- **Stateless**: Requests contain all necessary information; no server state is stored.
- **Client-Server Architecture**: Separation of concerns between client and server.
- **Uniform Interface**: Standardised operations for all resources using HTTP methods.
- **Resource-Based**: Resources identified by URIs; manipulated via HTTP methods.
- **Representation**: Resources represented using standard formats like JSON or XML.
- **State Transfer**: Transfer of resource state between client and server.

Properties of RESTful API

- **Stateless**: Requests contain all necessary information; no server state is stored.
- **Client-Server Architecture**: Separation of concerns between client and server.
- **Uniform Interface**: Standardised operations for all resources using HTTP methods.
- **Resource-Based**: Resources identified by URIs; manipulated via HTTP methods.
- **Representation**: Resources represented using standard formats like JSON or XML.
- **State Transfer**: Transfer of resource state between client and server.
- **Cacheability**: Support for caching responses to improve performance.

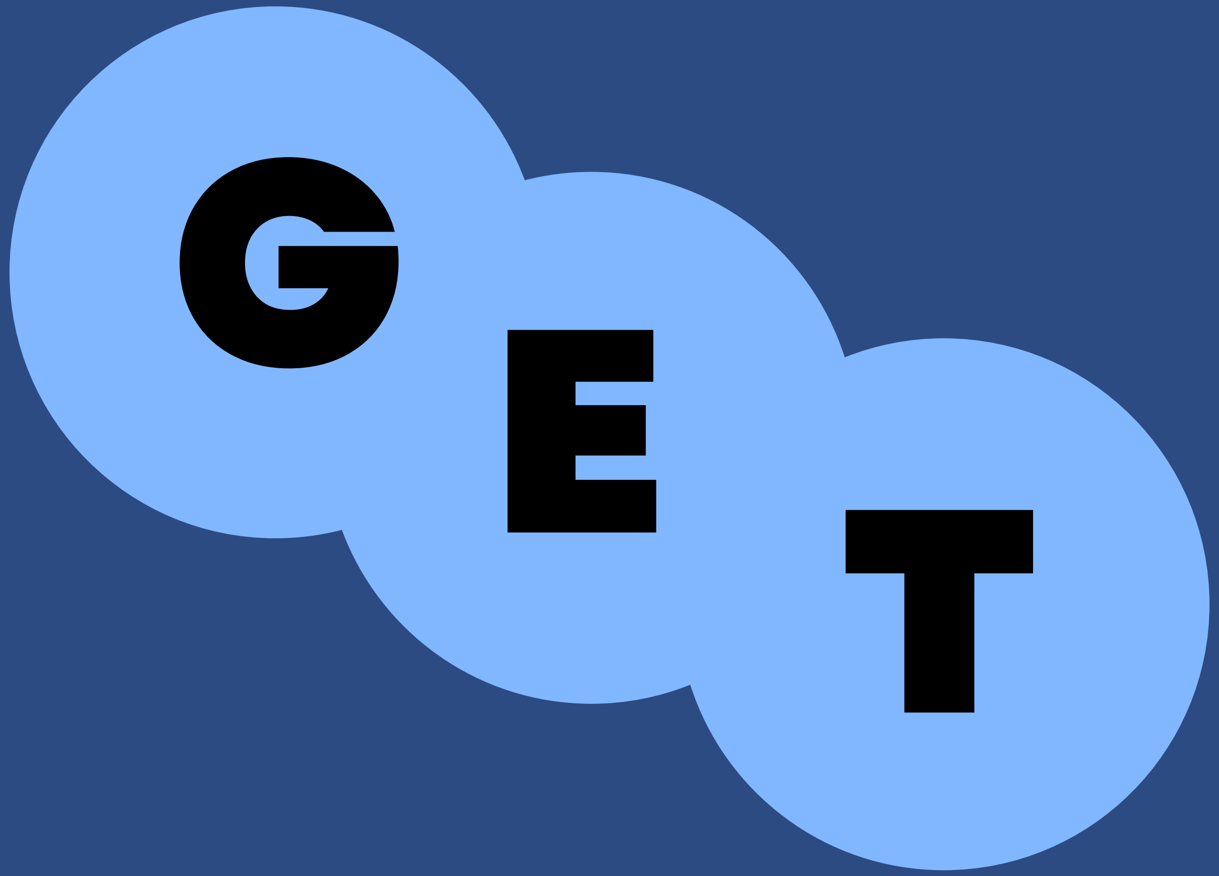
Properties of RESTful API

- **Stateless**: Requests contain all necessary information; no server state is stored.
- **Client-Server Architecture**: Separation of concerns between client and server.
- **Uniform Interface**: Standardised operations for all resources using HTTP methods.
- **Resource-Based**: Resources identified by URIs; manipulated via HTTP methods.
- **Representation**: Resources represented using standard formats like JSON or XML.
- **State Transfer**: Transfer of resource state between client and server.
- **Cacheability**: Support for caching responses to improve performance.
- **Layered System**: Allows for intermediaries like proxies for scalability and security.

HTTP Methods

The GET Method

Retrieves data from a server



The POST Method

Submits data to be processed by a server

The PUT Method

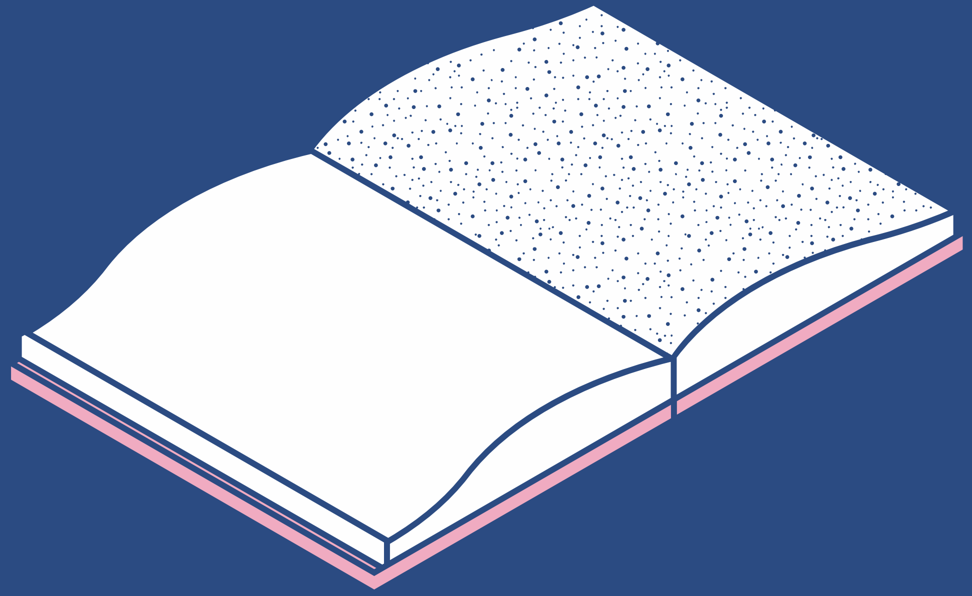
Updates or replaces data on a server

The PATCH Method

Partially updates data on a server

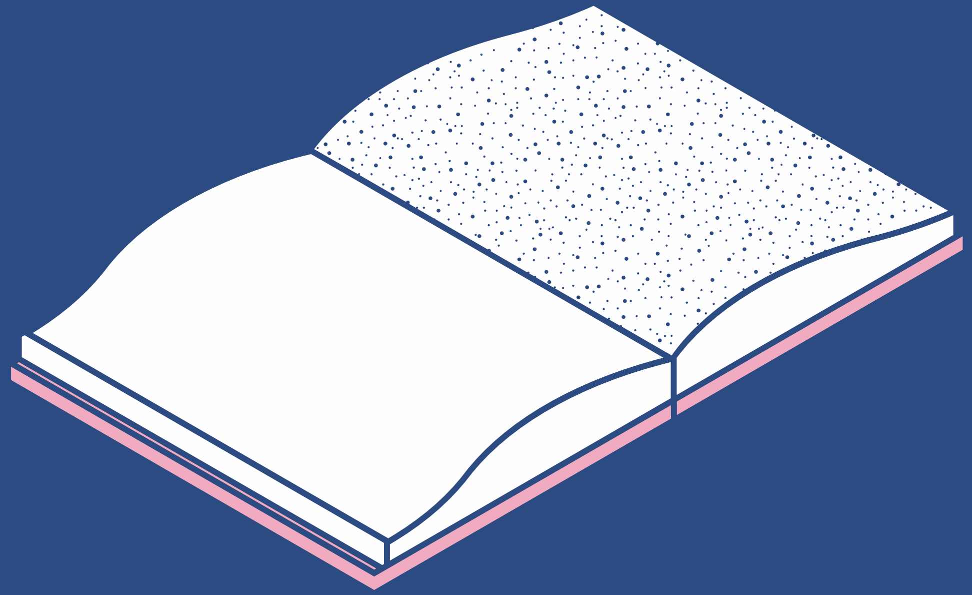
The DELETE Method

Removes data from a server



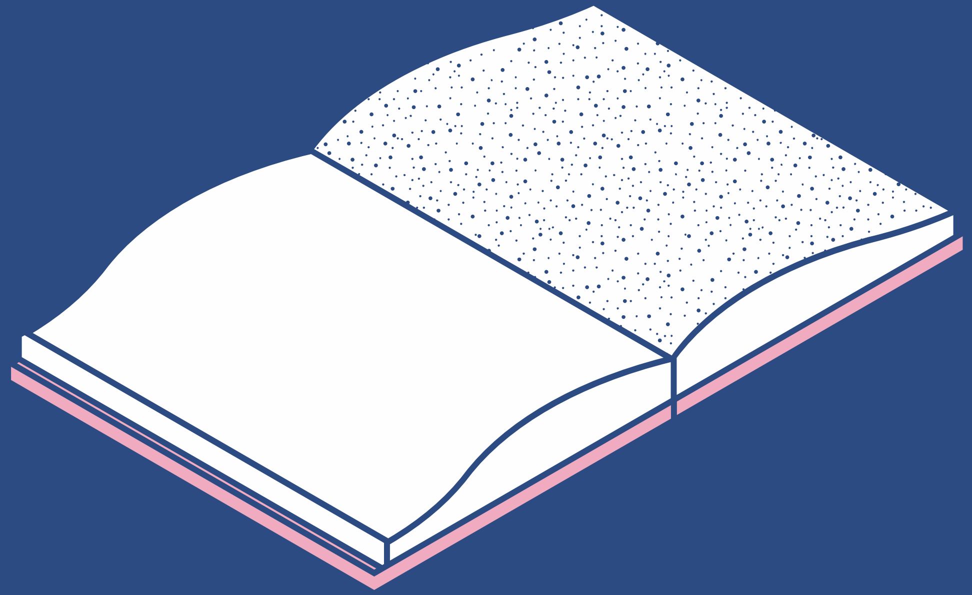
Which HTTP method is commonly used for creating new resources on a server in a RESTful API?

1. GET
2. POST
3. PUT
4. DELETE



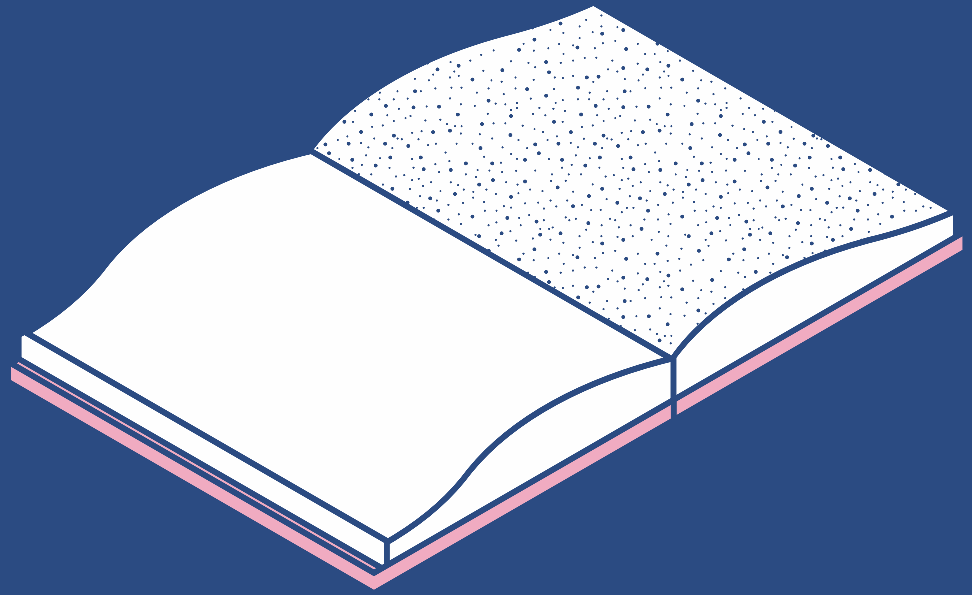
Which HTTP method is commonly used for creating new resources on a server in a RESTful API?

1. GET
2. POST
3. PUT
4. DELETE



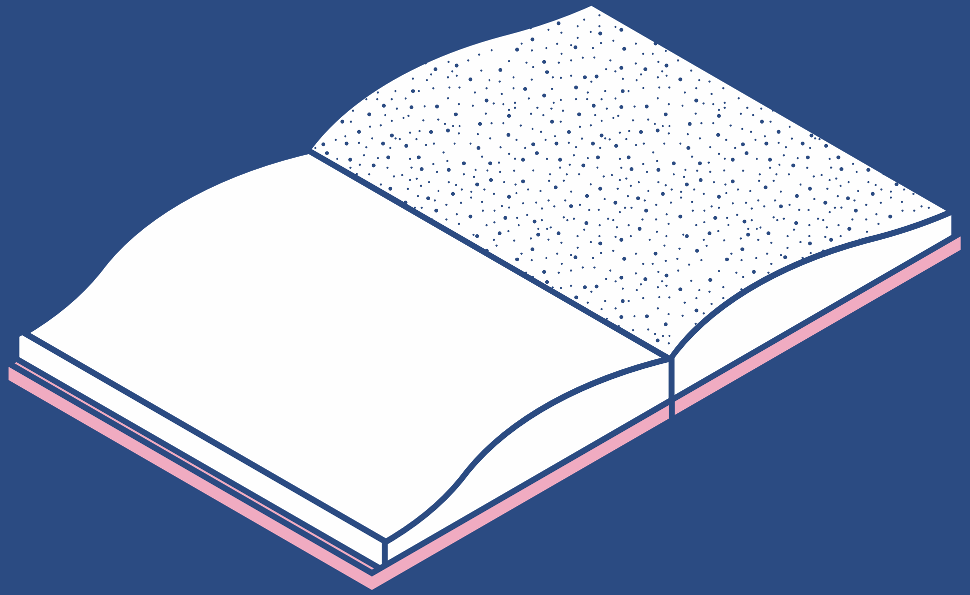
What does the DELETE HTTP method do in a RESTful API?

1. Retrieves data from a server
2. Submits data to a server
3. Removes data from a server
4. Partially updates data on a server



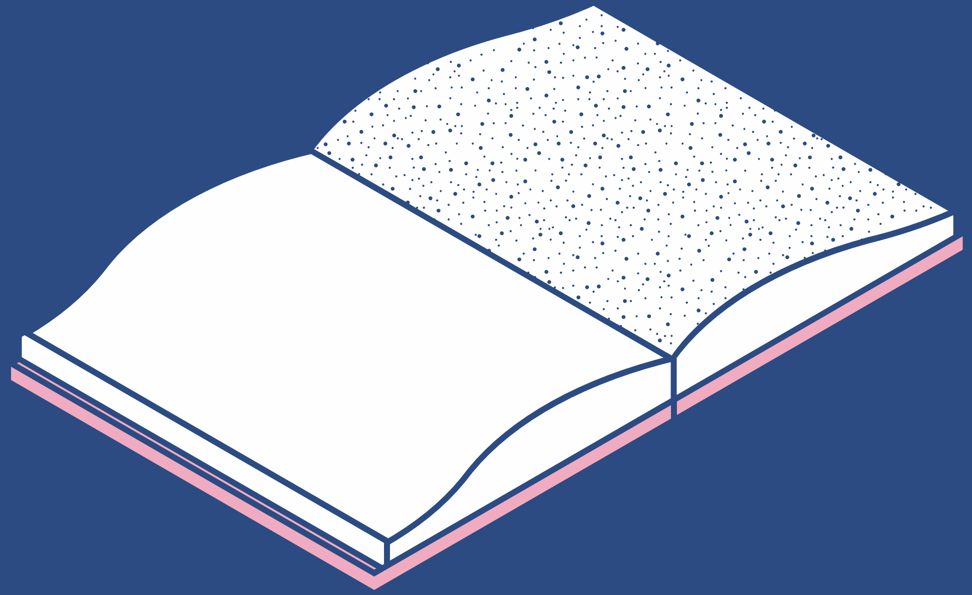
What does the DELETE HTTP method do in a RESTful API?

1. Retrieves data from a server
2. Submits data to a server
3. Removes data from a server
4. Partially updates data on a server



Which HTTP method should be used to update or replace an existing resource on a server in a RESTful API?

1. PUT
2. PATCH
3. POST
4. GET



Which HTTP method should be used to update or replace an existing resource on a server in a RESTful API?

1. PUT
2. PATCH
3. POST
4. GET