HTTP Protocol

HTTP

SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

Questions?

Software University

sli.do

#java-web

Table of Contents

Web Application

What is a Web Application?

What is a Web Application?

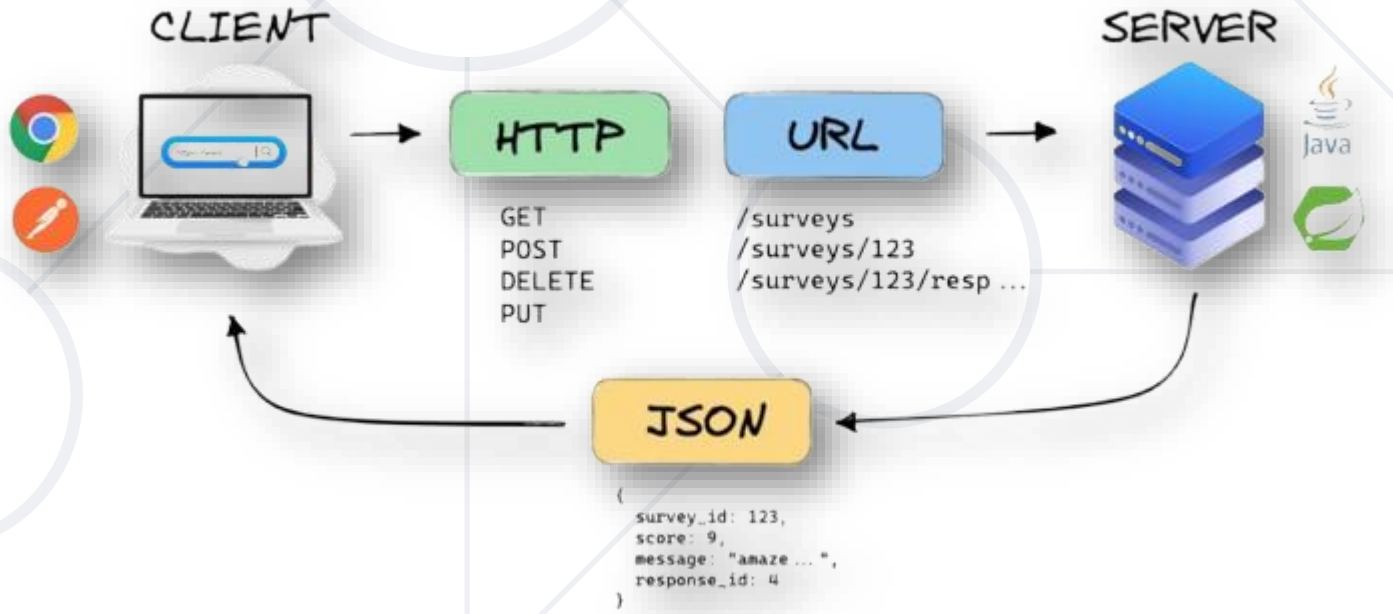
- **Web application** is a program accessible over the **web**
- System that receives **web requests** (e.g. HTTP)
- Sends **web responses** (e.g. HTTP)

How Does It Work?

- The **client** (web browser) sends an **HTTP request** to access a resource (e.g., a webpage)
- The **web application** receives the request
- **Processes** the request
- A **response** (e.g., HTML, JSON) **is sent back** to the client

Client vs. Server

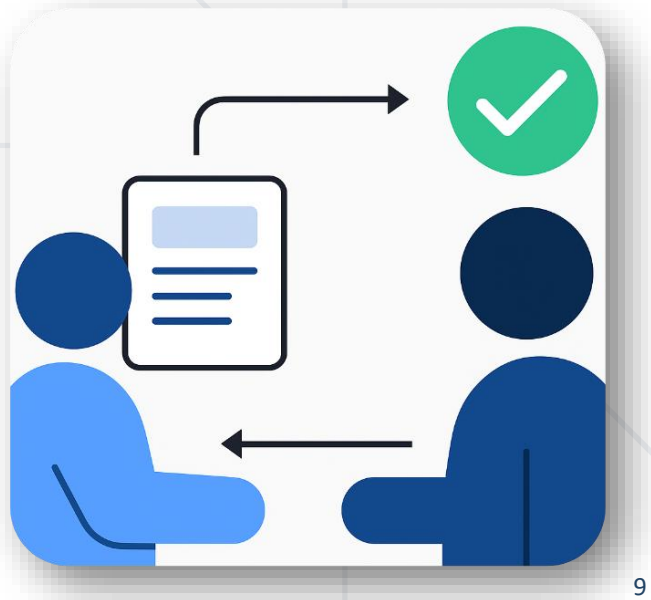
- **Client**: Initiates requests (e.g., a browser, Postman)
- **Server**: Responds to requests, providing data or resources



HyperText Transfer Protocol

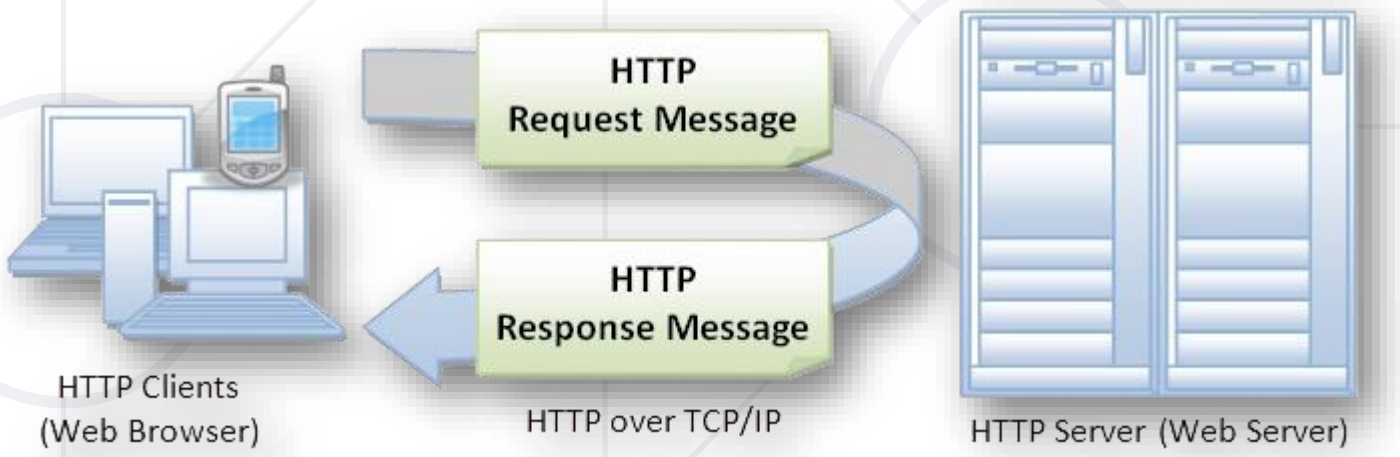
Structured Communication

- To communicate effectively, both sides must follow a shared **format and rules** - this prevents chaos and misunderstandings
- **Examples:**
  - University application (form, documents)
  - Job application (CV, cover letter)
  - Sending a delivery (address, order details)



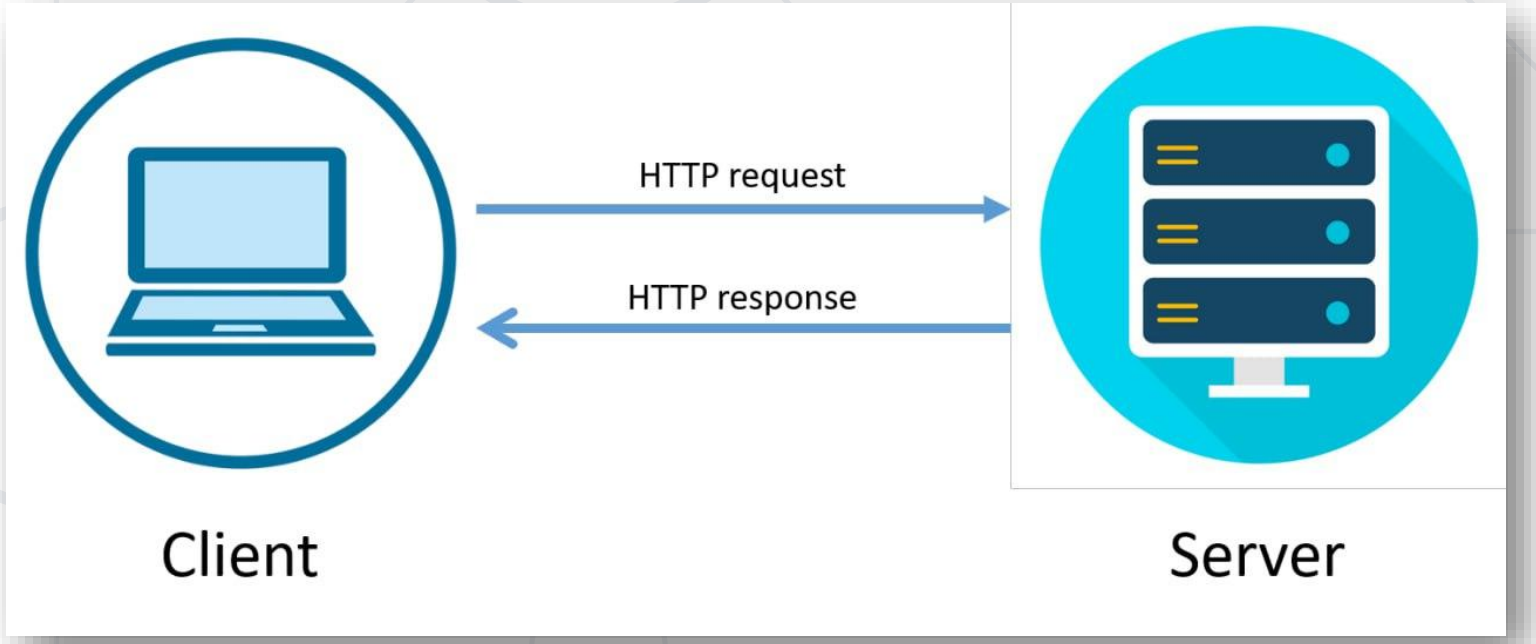
What is HTTP?

- **HyperText Transfer Protocol** The foundational protocol for web communication
- Easy way of sending **requests** and **responses** in a structured way between clients and servers



How Does It Work?

- **Client sends** an HTTP **request** to access a resource
- **Server** receives the request and **returns a response**
- Communication happens over **IP**



Is HTTP the Only Way?

- **No! Alternatives include:**
  - **WebSocket:** Real-time bidirectional communication
  - **gRPC:** For efficient communication between services
  - **FTP:** File transfer protocol
  - Just HTTP is a convenient way to build a web application communication

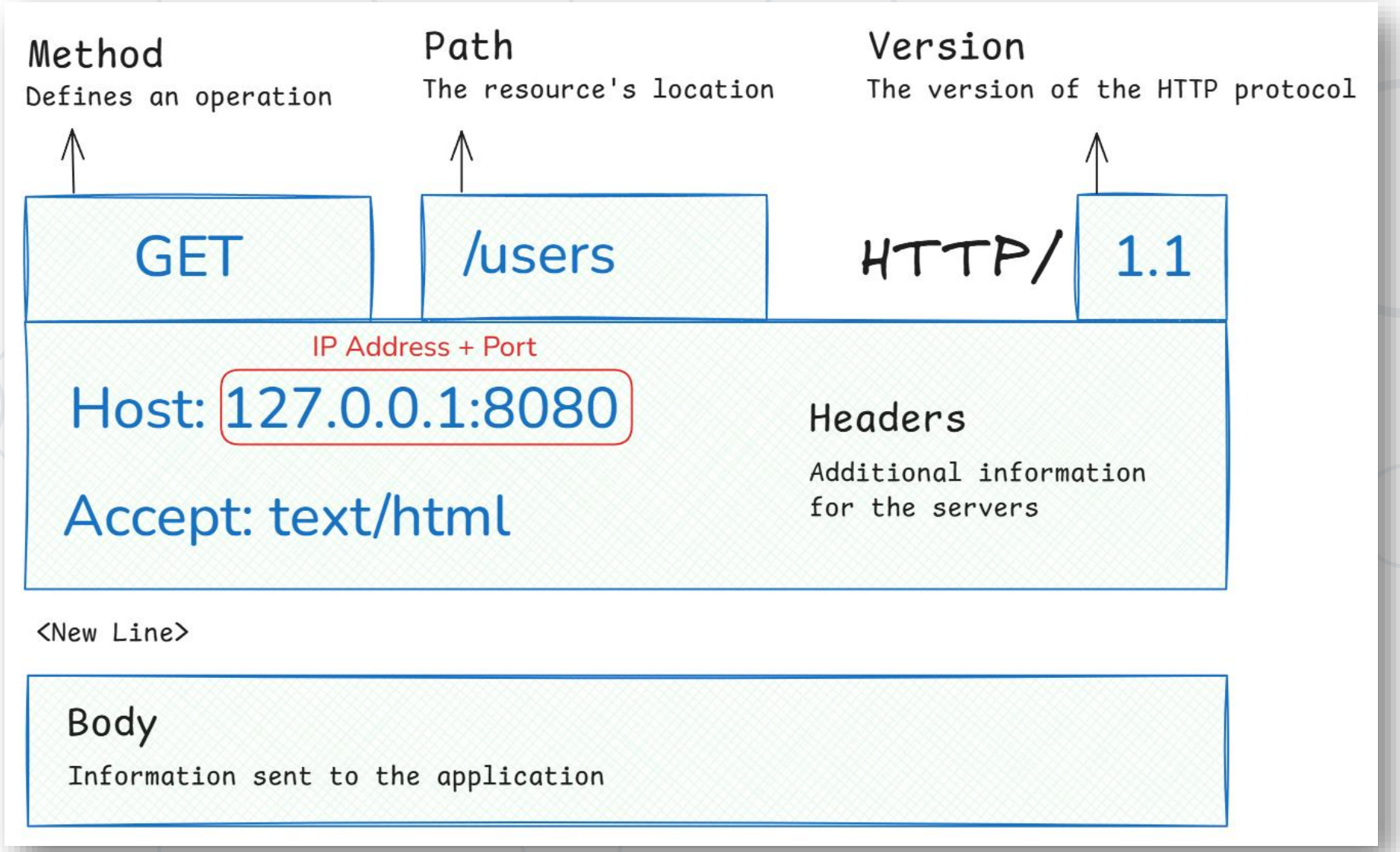
HTTP Request and Response

HTTP Request and Response

- **HTTP Request**: Sent by the client, asking for a resource

- **HTTP Response**: Sent by the server, providing the resource

HTTP Request Structure



HTTP GET Request – Example

Software University

- Example of HTTP **GET** request:

```
GET /products/42 HTTP/1.1
Host: 192.168.1.100:8080
```

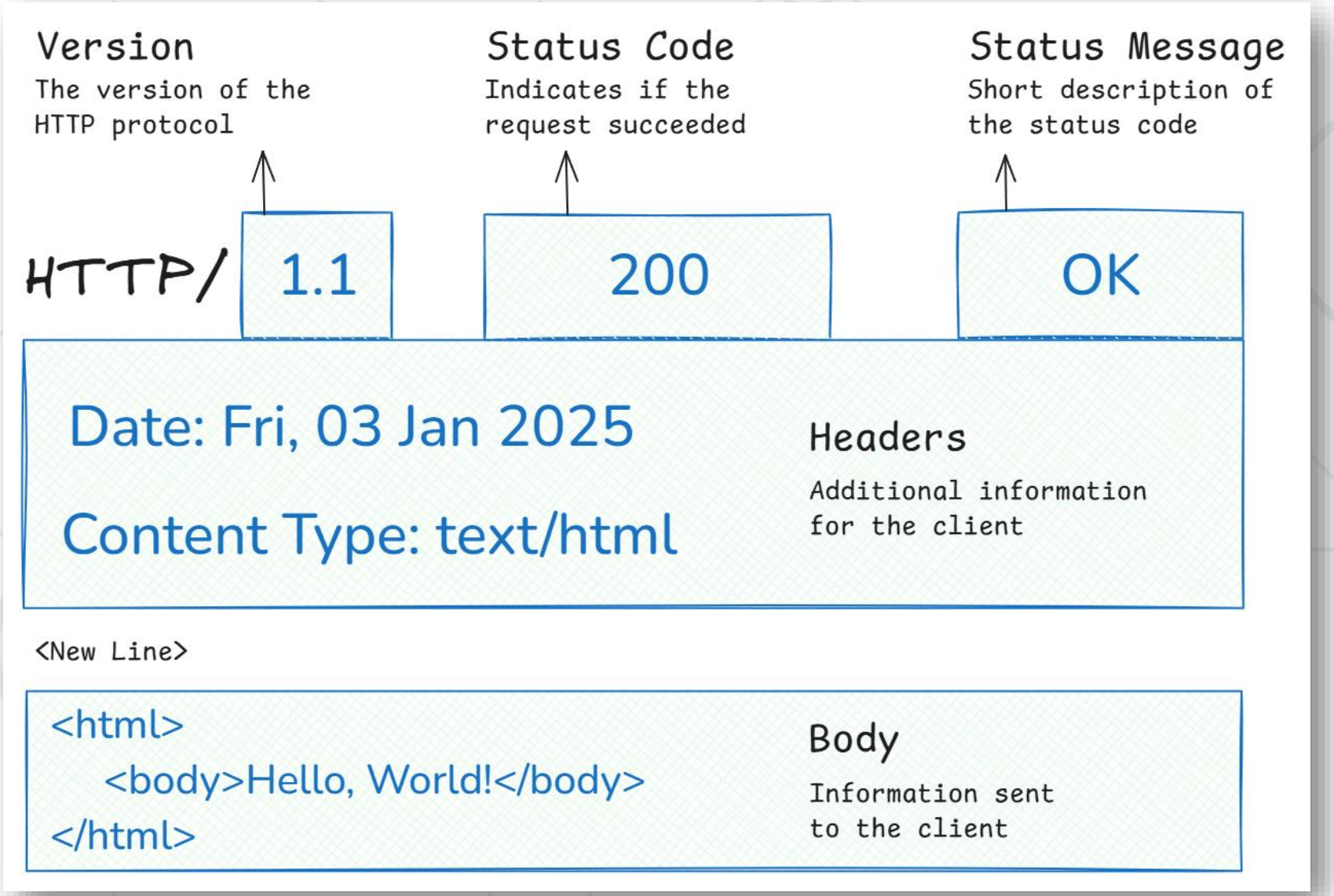
HTTP POST Request – Example

- Example of HTTP **POST** request:

```
POST /orders HTTP/1.1
Host: 192.168.1.100:8080
Content-Type: application/json

{
  "productId": 42,
  "quantity": 2,
  "deliveryMethod": "standard"
}
```

HTTP Response Structure



HTTP Response – Example

- Example of HTTP **response** from the Web server:

```
HTTP/1.1 200 OK
Date: Sat, 31 May 2025 13:20:00 GMT
Content-Type: application/json

{
  "id": 42,
  "name": "Whey Protein",
  "price": 29.99,
  "available": true
}
```

HTTP Error Response – Example

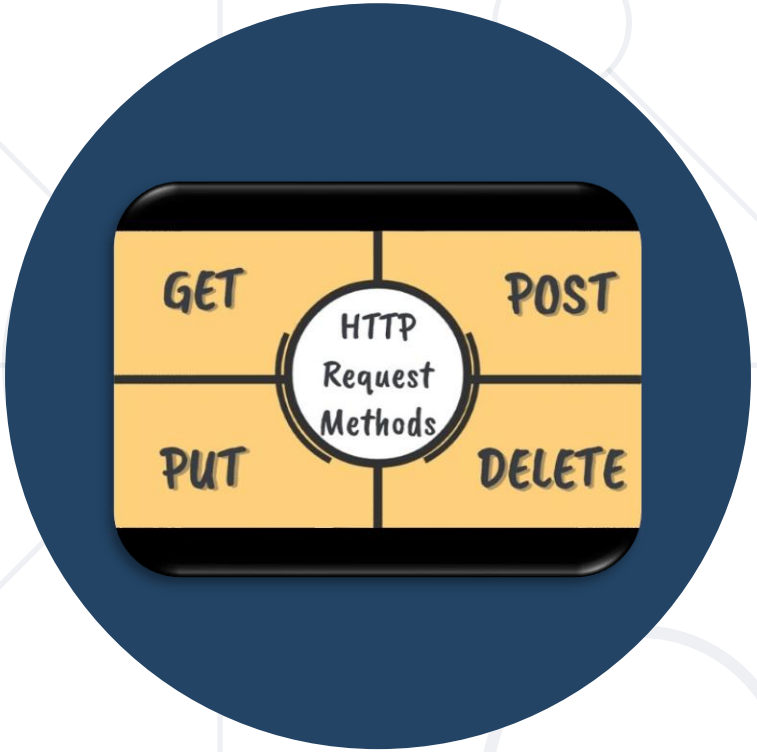
- Example of **HTTP response** with error result:

```
HTTP/1.1 404 Not Found
Date: Sat, 31 May 2025 13:21:00 GMT
Content-Type: application/json

{
  "errorMessage": "Product not found"
}
```

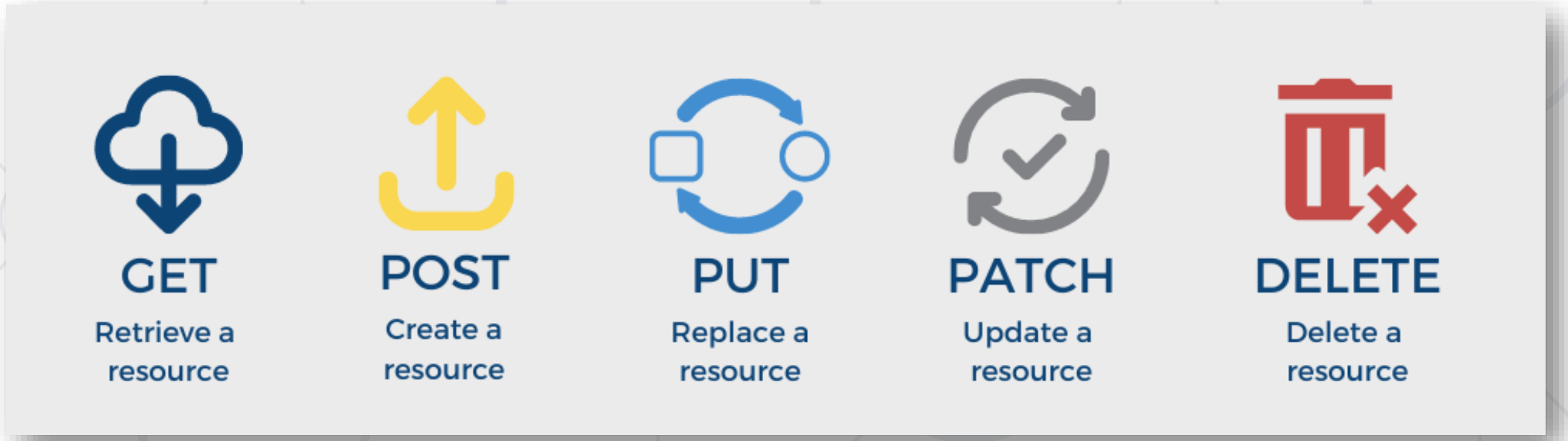
Uniform Resource Locator (URL)

- URL is a formatted string, consisting of:
  - Protocol for communicating (**http**, **ftp**, **https**...) – HTTP in most cases
  - Host or IP address (**www.softuni.bg**, **gmail.com**, **127.0.0.1**, **web**)
  - Port (the default port is **80**) – a number in range [0…65535]
  - Path (**/forum**, **/path/index.html**)
  - Query string (**?id=27&lang=en**)
  - Fragment (**#lectures**) – used on the client to navigate to some section



HTTP Methods

Common HTTP Methods



When to Use Each Method

- **POST**: Add a new product to the catalog
  - **Example**: POST /products with product details in the request body
- **GET**: Retrieve all products
  - **Example**: GET /products
- **DELETE**: Remove a product from the catalog
  - **Example**: DELETE /products/123

HTTP Status Codes

- HTTP response code classes
  - **1xx**: informational ("**100** Continue")
  - **2xx**: successful ("**200** OK", "**201** Created")
  - **3xx**: redirection ("**304** Not Modified", "**301** Moved Permanently", "**302** Found")
  - **4xx**: client error ("**400** Bad Request", "**404** Not Found", "**401** Unauthorized", "**409** Conflict")
  - **5xx**: server error ("**500** Internal Server Error", "**503** Service Unavailable")

Idempotency

What is Idempotency?

- **Idempotent Methods**:
  - **GET**: Retrieving a resource doesn't change its state
  - **PUT**: Updating a resource results in the same state, regardless of how often it's called
  - **DELETE**: Removing a resource remains the same after multiple attempts
- **Non-Idempotent Methods**:
  - **POST**: Repeated submissions can create duplicate entries or actions
- **Importance**
  - Ensures predictable behavior in network communication

MIME Types

Multi-Purpose Internet Mail Extensions

What are MIME types?

- **MIME** == **M**ulti-Purpose **I**nternet **M**ail **E**xtensions
  - Defines the format of data transmitted over the internet
  - Used in HTTP to specify the nature of data in requests and responses
- Used in HTTP through the **Content-Type** header

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": 101,
  "username": "User123",
  "email": "user@gmail.com"
}
```

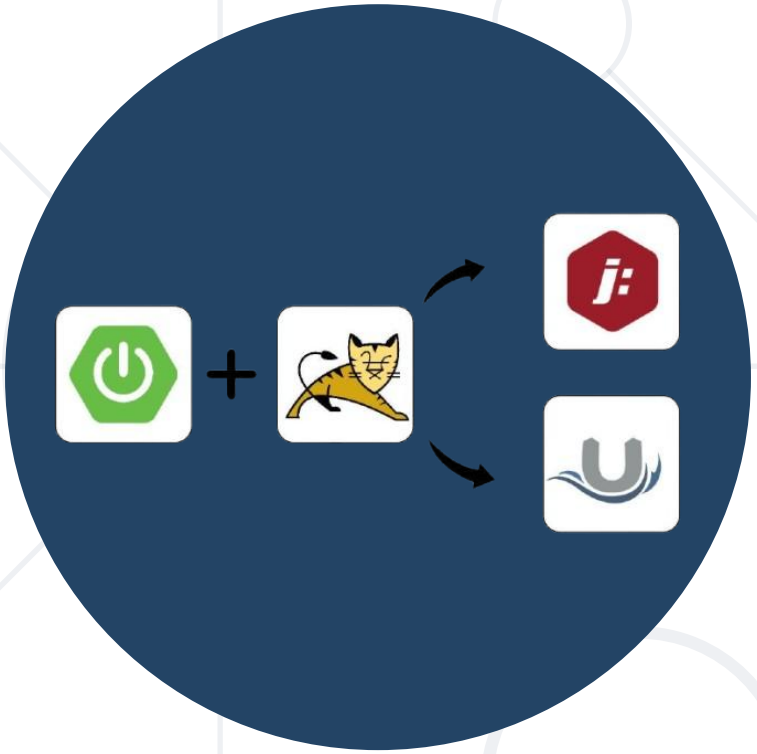
Common MIME Types

Software University

MIME Type	Description
application/json	JSON data
image/png	PNG images
image/gif	GIF images
text/html	HTML documents
text/plain	Plain text
text/xml	XML data
video/mp4	MP4 videos
application/pdf	PDF documents

Why MIME Types Matter?

- Helps the browser or client application understand how to **process the content**, it essentially gives the client semantic information about the content
- Ensures **proper rendering** or **handling of data** (e.g., display an image, download a file)



Web Servers

What is a Web Server?

- They are **software components** that handle HTTP requests and responses
- **Java** itself **doesn't** know how to process HTTP requests
- **Java** applications **require** a **web server** like Tomcat, Jetty, or Undertow to handle HTTP communication

Key Points

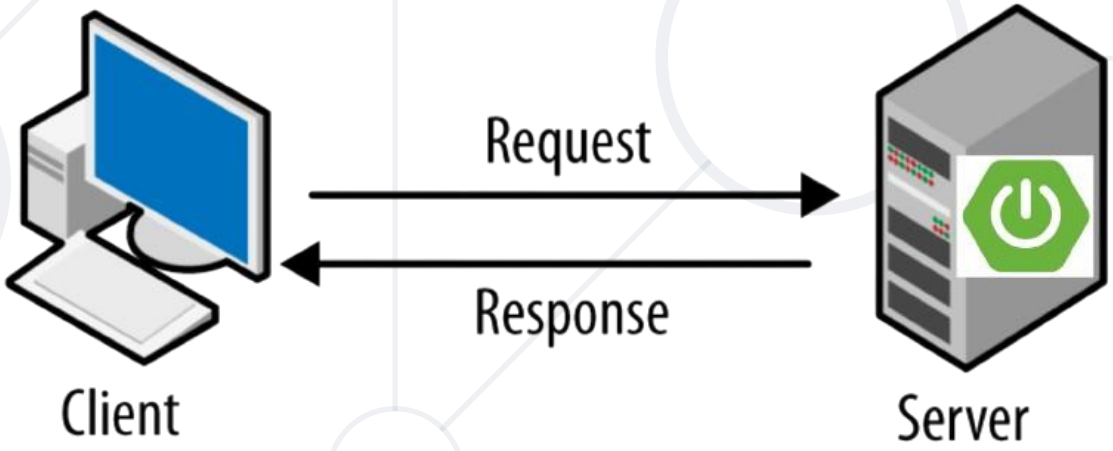
- Web servers are **not part of** the standard Java distribution (**Java SE**)
- They are **installed manually** on our machines
- Manual **configuration** and **deployment** is required (e.g. **.war** files)

Examples of Web Servers

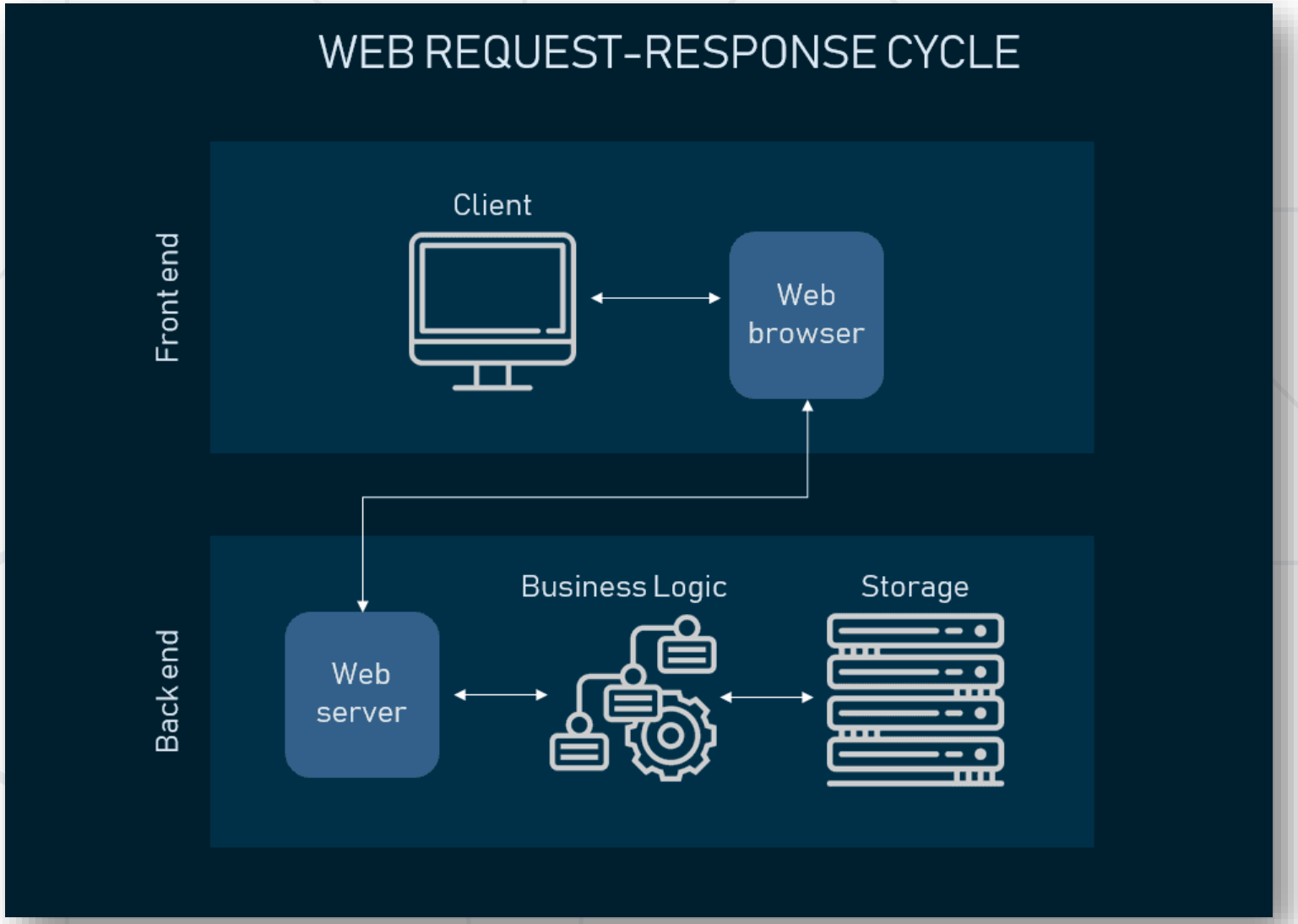
Web Server	Description
Apache Tomcat	Most common Java web server, used by Spring
Jetty	Lightweight and highly embeddable
Undertow	High-performance web server by JBoss

How Spring Simplifies Web Servers?

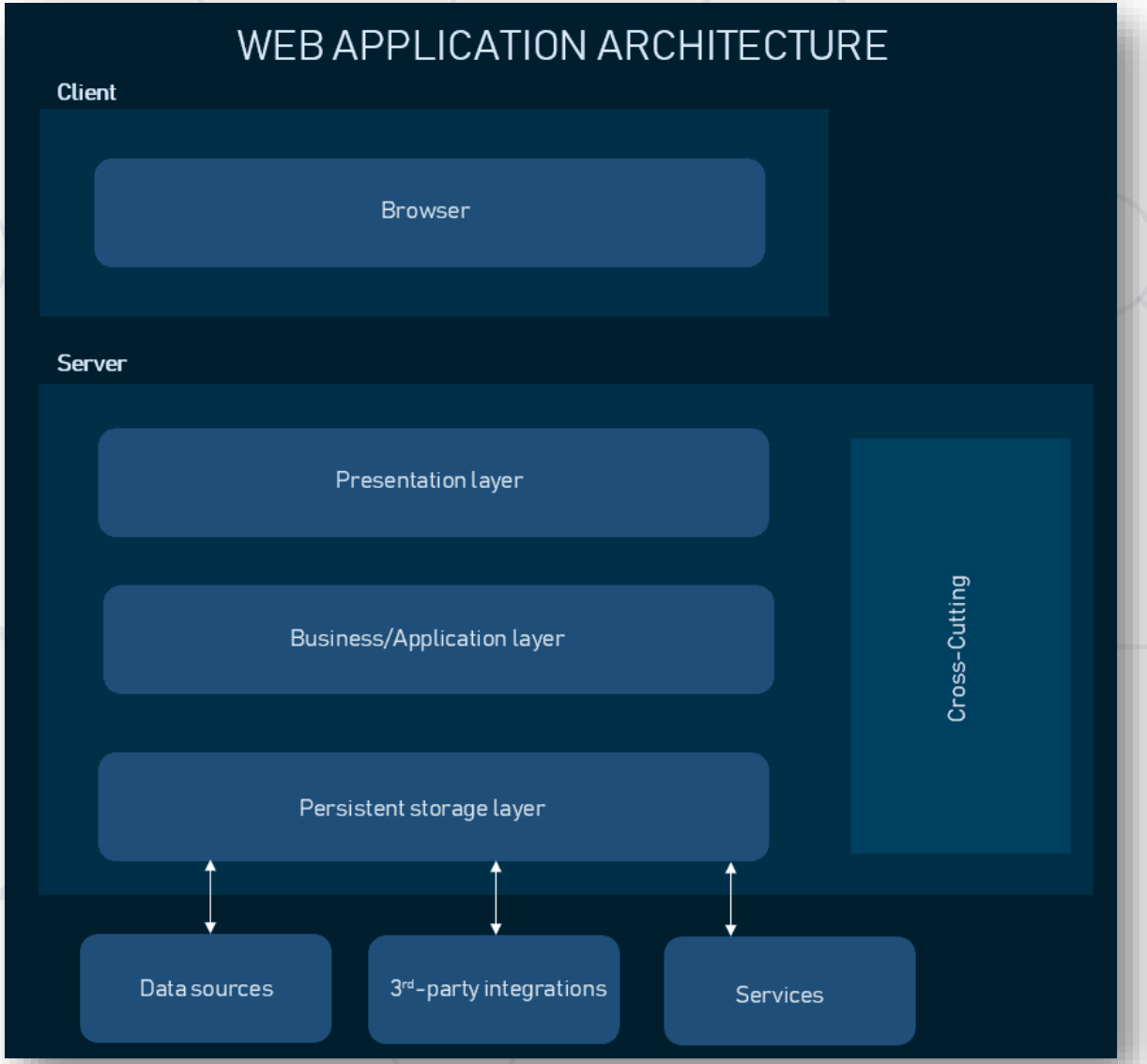
- **Spring Boot** includes an embedded Tomcat server by default
- No need to install or configure **Tomcat** manually—just add your code and run it
- Spring **simplifies** integration, making it ideal for building modern Java web applications



How Web Applications Work?



How Web Applications Work?



How Web Applications Work?

- **The Basics**
  - **Client**: The user's device, web browser, or anything capable of sending **requests** (communicating to web applications)
  - **Server (Web Application)**: Hosts the application and **responds** to client requests
- **The Communication Process**
  - Browser sends **HTTP** requests to the server
  - Server **processes** the request and interacts with the **database** if necessary
  - **Response** (HTML, JSON, etc.) is **returned** to the browser
  - Browser **renders** the content for the user

Summary

- Web Application
- HTTP
- HTTP Request and Response
- HTTP Methods
- Idempotency
- MIME and Media types
- Embedded Web Servers

Questions?

SoftUni

Software University | SoftUni AI | SoftUni Digital | SoftUni Creative | SoftUni Foundation | СофтУни БУДИТЕЛ | Finance Academy

SoftUni Diamond Partners

Software University

Coca-Cola HBC Bulgaria

Postbank

top EMPLOYER България Bulgaria 2025

FOR A BETTER WORLD OF WORK

SUPER HOSTING .BG

encorp.ai

createX

DRAFT KINGS

THE CROWN IS YOURS

INDEAVR

Serving the high achievers

VIVACOM

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg