# HTTP and REST Services

## HTTP, Request Headers, RESTful Web Services

**SoftUni Team**
**Technical Trainers**

Software University

SoftUni

Software University

https://softuni.bg

# Have a Question?

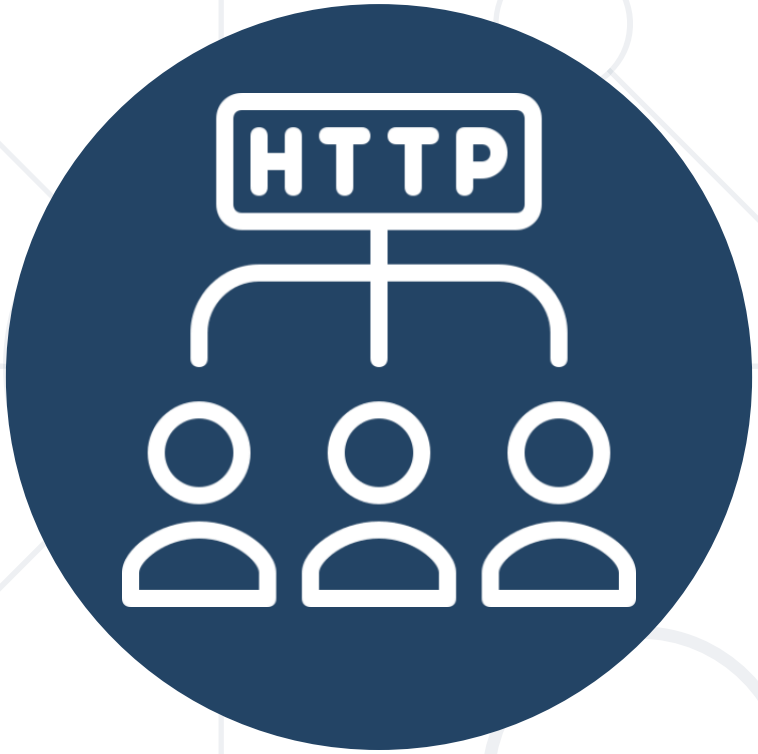
Software University

sli.do

# #js-front-end

# Table of Contents

1. HTTP Overview & Developer Tools
2. REST & RESTful Services
3. Accessing the GitHub API
4. NPM & package.json
5. CRUD Operations to a NodeJS REST API

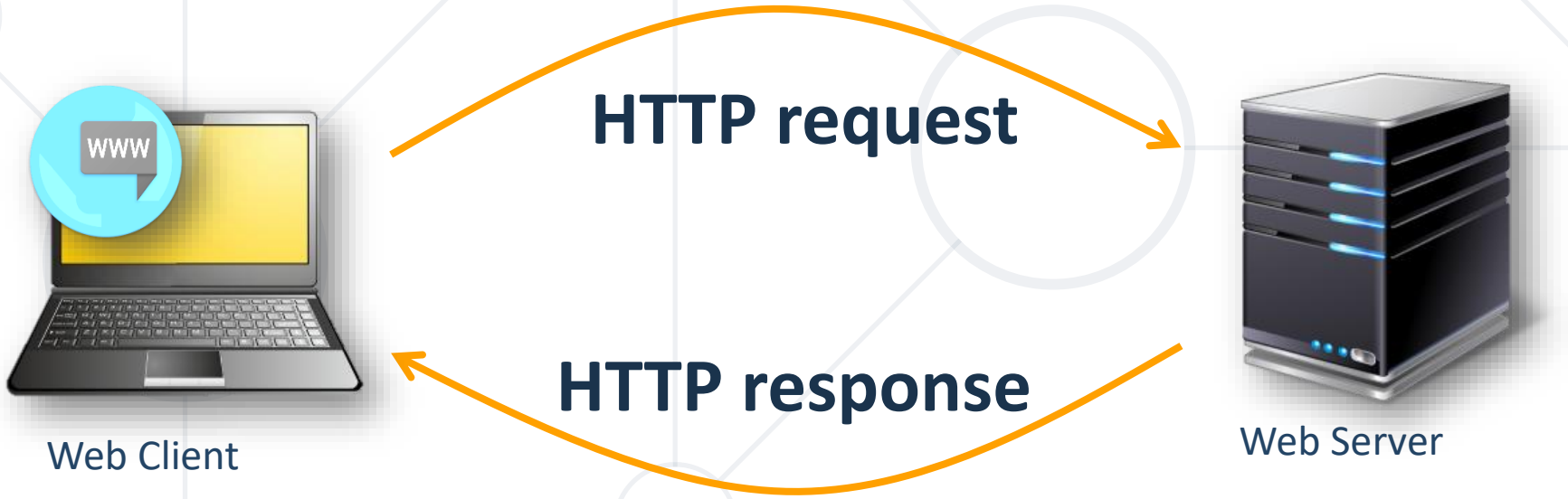



# HTTP Overview

## Hypertext Transfer Protocol

# HTTP Basics

- HTTP (**H**yper **T**ext **T**ransfer **P**rotocol)
  - Text-based client-server protocol for the Internet
  - For transferring Web resources (HTML files, images, styles, etc.)
  - Request-response based



# HTTP Request Methods

- **HTTP** defines **methods** to indicate the desired action to be performed on the identified resource

| Method | Description |
|---|---|
| GET | Retrieve / load a resource |
| POST | Create / store a resource |
| PUT | Update a resource |
| DELETE | Delete (remove) a resource |
| PATCH | Update resource partially |
| HEAD | Retrieve the resource's headers |
| OPTIONS | Returns the HTTP methods that the server supports for the specified URL |

# HTTP GET Request – Example

```
GET /users/testnakov/repos HTTP/1.1
Host: api.github.com
Accept: */*
Accept-Language: en
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/54.0.2840.71 Safari/537.36
Connection: Keep-Alive
Cache-Control: no-cache
<CRLF>
```

HTTP request line

HTTP headers

The request body is empty

# HTTP POST Request – Example

```
POST /repos/testnakov/test-nakov-repo/issues HTTP/1.1
Host: api.github.com
Accept: */*
Accept-Language: en
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible;MSIE 6.0; Windows NT 5.0)
Connection: Keep-Alive
Cache-Control: no-cache
<CRLF>
{"title":"Found a bug",
 "body":"I'm having a problem with this.",
 "labels":["bug","minor"]}
<CRLF>
```

HTTP request line

HTTP headers

The request body holds the submitted data

# HTTP Response – Example

```
HTTP/1.1 200 OK
Date: Fri, 11 Nov 2016 16:09:18 GMT+2
Server: Apache/2.2.14 (Linux)
Accept-Ranges: bytes
Content-Length: 84
Content-Type: text/html
<CRLF>
<html>
  <head><title>Test</title></head>
  <body>Test HTML page.</body>
</html>
```

HTTP response status line

HTTP response headers

HTTP response body

# HTTP Response Status Codes

Software University

| Status Code | Action | Description |
|---|---|---|
| 200 | OK | Successfully retrieved resource |
| 201 | Created | A new resource was created |
| 204 | No Content | Request has nothing to return |
| 301 / 302 | Moved | Moved to another location (redirect) |
| 400 | Bad Request | Invalid request / syntax error |
| 401 / 403 | Unauthorized | Authentication failed / Access denied |
| 404 | Not Found | Invalid resource |
| 409 | Conflict | Conflict was detected, e.g. duplicated email |
| 500 / 503 | Server Error | Internal server error / Service unavailable |

# Content-Type and Disposition

- The **Content-Type** / **Content-Disposition** headers specify how the HTTP request / response body should be processed

```
Content-Type: application/json
```

JSON-encoded data

```
Content-Type: text/html; charset=utf-8
```

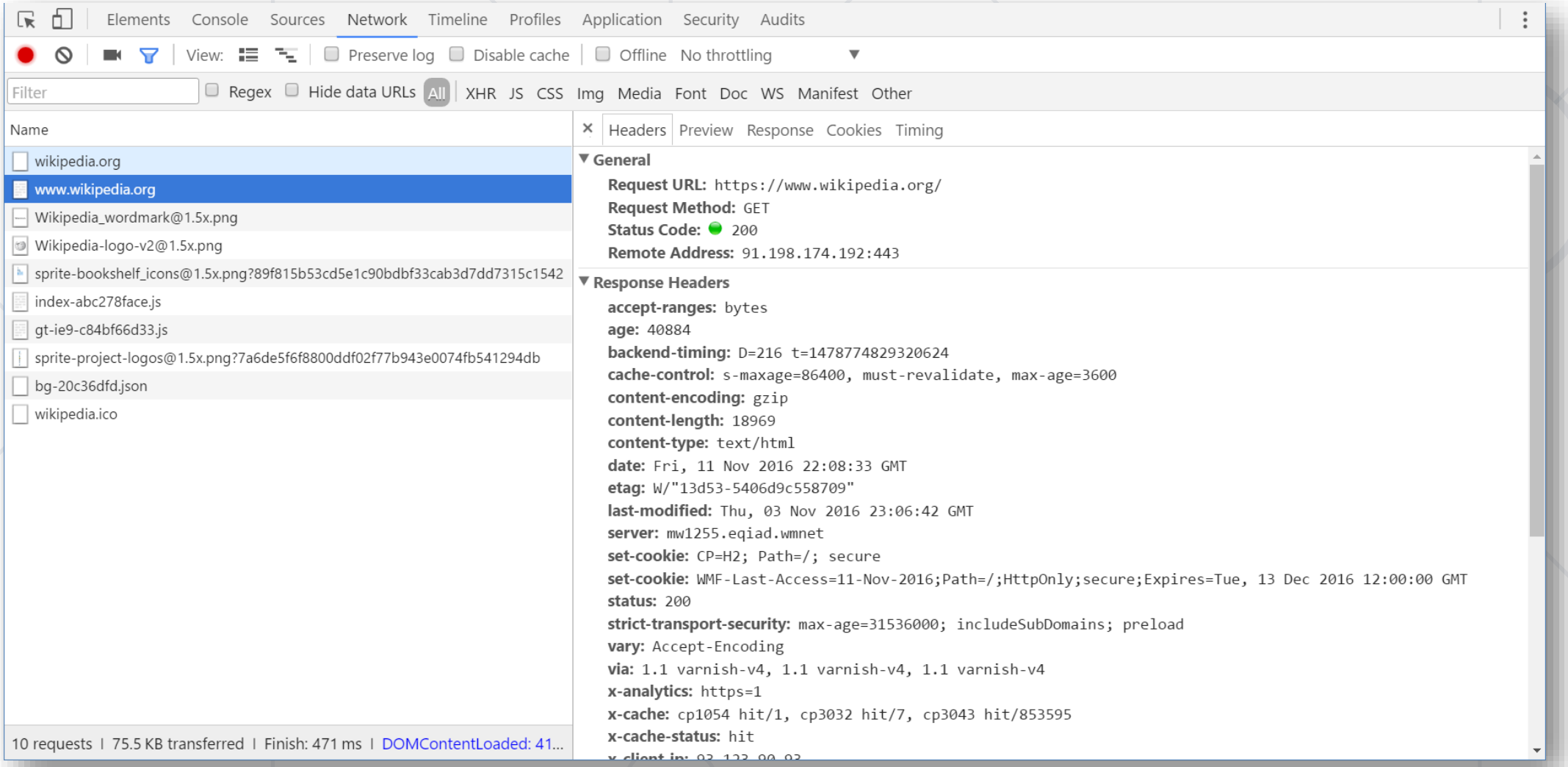
UTF-8 encoded HTML page. Will be shown in the browser

```
Content-Type: application/pdf
Content-Disposition: attachment;
filename="Financial-Report-April-2016.pdf"
```

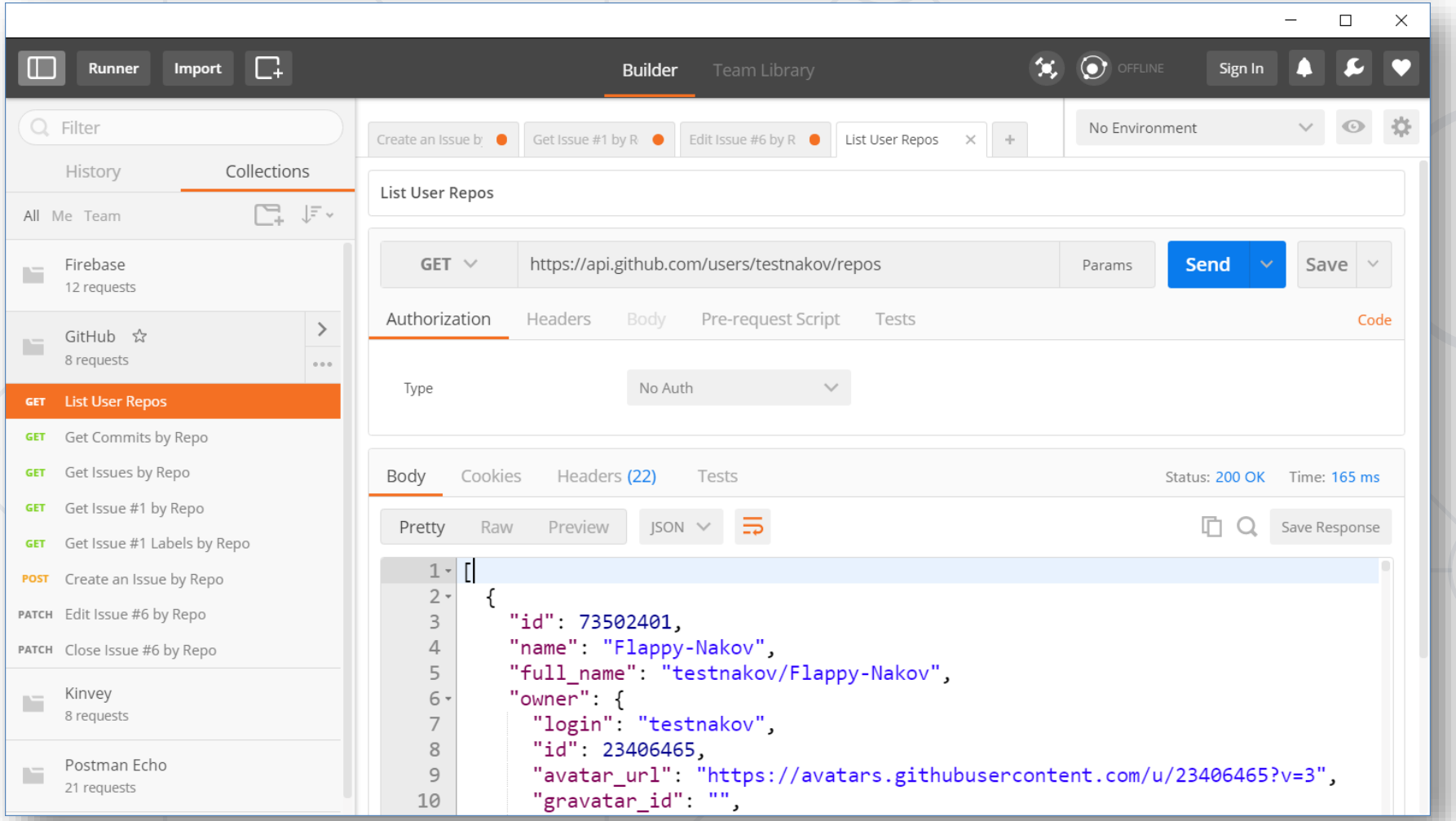
This will download a PDF file named Financial-Report-April-2016.pdf

# Browser Developer Tools

Software University



# Postman

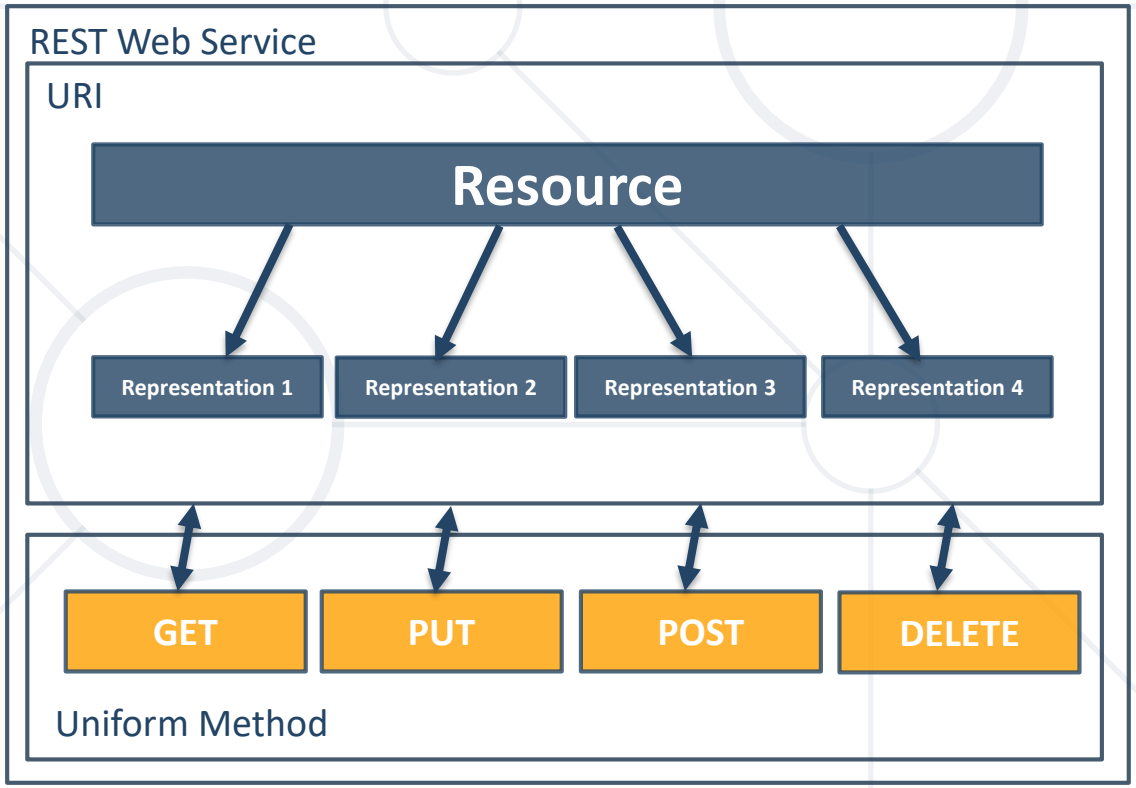



## Read more about Postman REST Client

# REST and RESTful Services

# REST and RESTful Services

- Representational State Transfer (REST)
  - Architecture for client-server communication over HTTP
  - Resources have URI (address)
  - Can be created/retrieved/modified/deleted/etc.
- RESTful API/RESTful Service
  - Provides access to server-side resources via HTTP and REST



# REST and RESTful Services – Example

Software University

- Create a new post

| POST | http://some-service.org/api/posts |
|---|---|

- Get all posts / specific post

| GET | http://some-service.org/api/posts |
|---|---|

| GET | http://some-service.org/api/posts/17 |
|---|---|

- Delete existing post

| DELETE | http://some-service.org/api/posts/17 |
|---|---|

- Replace / modify existing post

| PUT/PATCH | http://some-service.org/api/posts/17 |
|---|---|

# Accessing GitHub Through HTTP

## GitHub REST API

# GitHub API

- List user's all public repositories:

| GET | https://api.github.com/users/testnakov/repos |
|---|---|

- Get all commits from a public repository:

| GET | https://api.github.com/repos/testnakov/softuniada-2016/commits |
|---|---|

- Get all issues/issue #1 from a public repository

| GET | /repos/testnakov/test-nakov-repo/issues |
|---|---|

| GET | /repos/testnakov/test-nakov-repo/issues/1 |
|---|---|

# Github: Labels Issue

- Get the first issue from the "**test-nakov-repo**" repository
- Send a **GET** request to:
  - **https://api.github.com/repos/testnakov/test-nakov-repo/issues/:id**
  - Where **:id** is the current issue

# GitHub API

- Get all labels for certain issue from a public repository:

| GET | https://api.github.com/repos/testnakov/test-nakov-repo/issues/1/labels |
|---|---|

- Create a new issue to certain repository (with authentication)

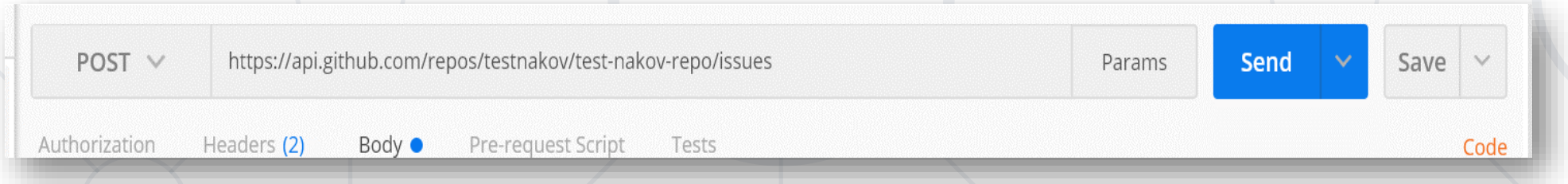
| POST | https://api.github.com/repos/testnakov/test-nakov-repo/issues |
|---|---|

| Headers | Authorization: Basic base64(user:pass) |
|---|---|

| Body | {"title":"Found a bug",<br>"body": "I'm having a problem with this."} |
|---|---|

# Github: Create Issue

- Create an issue when you send a "**POST**" request
- Use your Github account **credentials** to submit the issue



# Node Package Manager (NPM)

# NPM: Package Management for JavaScript Projects

- **Node Package Manager (NPM)** is a package manager for JavaScript and Node.js projects
- Simplifies installing, managing, and sharing libraries and tools in web development
- Bundled with Node.js, it offers a command-line interface for various tasks
- Facilitates integrating third-party packages to reuse existing code and speed up development
- Helps manage project dependencies, ensuring required packages are available
- Allows defining and running scripts through the project‘s package.json file

# package.json

- **package.json** is a metadata file used in Node.js projects to provide information about the project, its dependencies, and various configurations
- It includes details such as the project's name, version, description, author, license, and more, making it a central place to document essential project information
- integral to the Node.js and NPM ecosystems. When you share your project, including the **package.json** file lets others quickly understand and recreate your project's environment. It also makes it easy for others to install the correct dependencies with a single command (**npm install**)

# Package.json– Example

```
{
  "name": "test",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "scripts": {
    "start": "http-server -a localhost -p 3000 -P http://localhost:3000? -c-1"
  },
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "http-server": "0.12.3"
  }
}
```

# Install Packages From The package.json

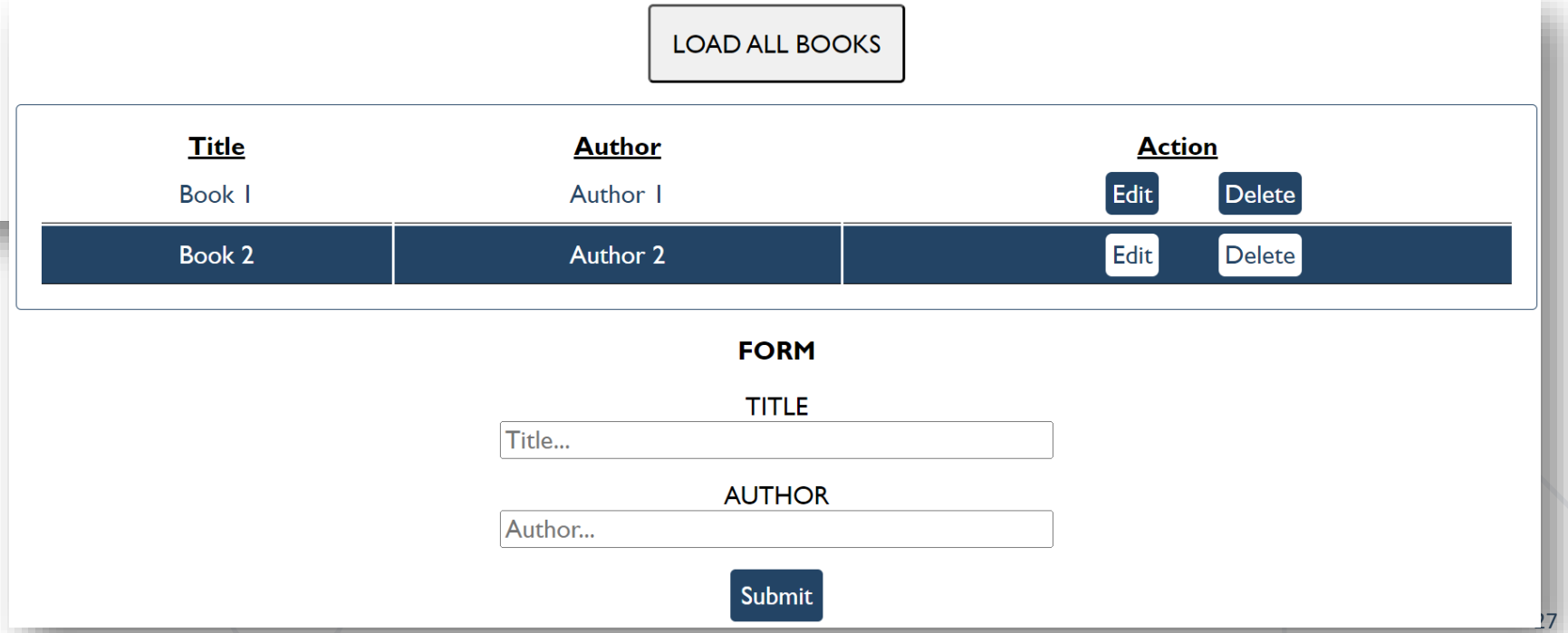
- **Navigate to Project**: Open your command-line interface and navigate to your project's directory
- Run the following command to install the project's dependencies listed in the **package.json**

```
npm install
```

- NPM will read the **package.json** file and install all the dependencies

# Problem: Book Library

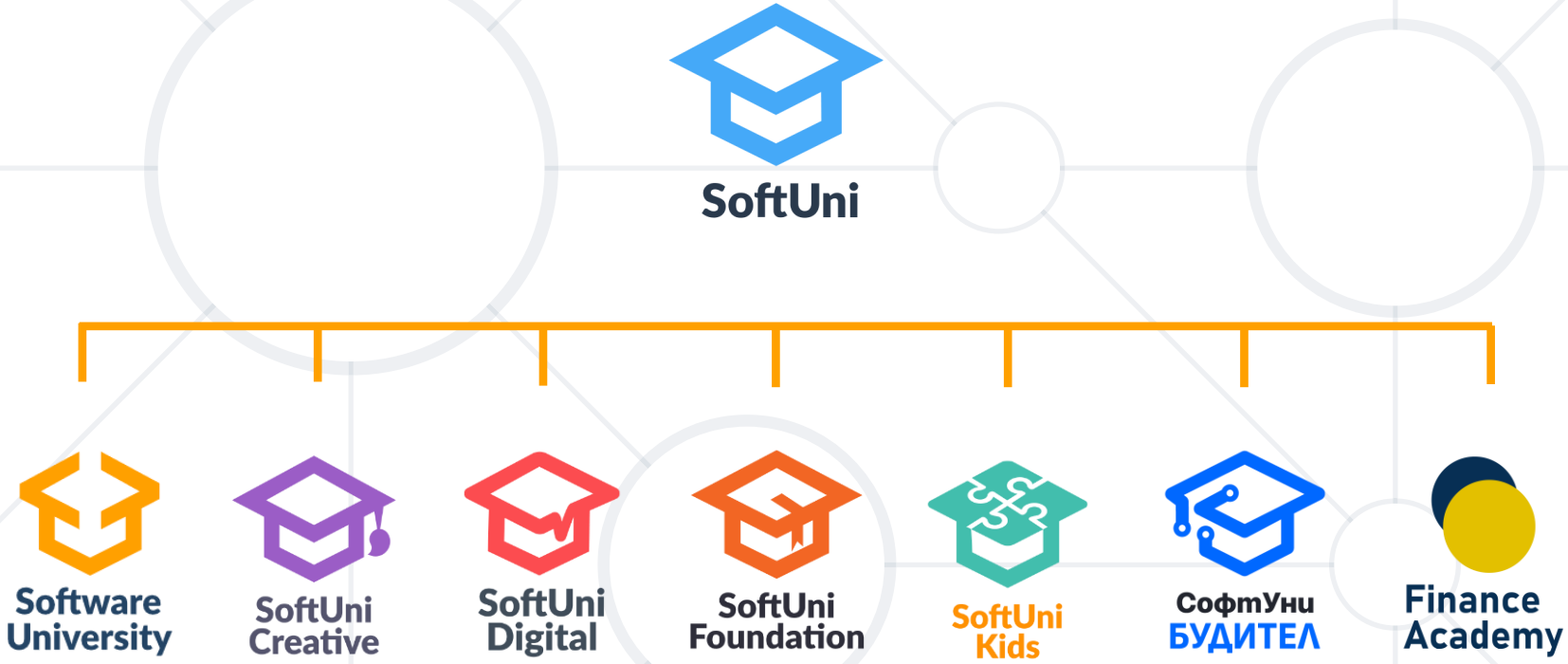
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Books</title>
    <link rel="stylesheet" href="./styles.css">
</head>
<body>
    <button id="loadBooks">LOAD ALL BOOKS</button>
    <table>
        <thead>
            <tr>
                <th>Title</th>
                <th>Author</th>
                <th>Action</th>
```



# Summary

- HTTP is text-based request-response protocol
- RESTful services address resources by URL
- Provide CRUD operations over HTTP
- RESTful services address resources by URL
- NPM is used for installing, managing, and sharing libraries and tools

# Questions?




© SoftUni – https://about.softuni.bg. Copyrighted document. Unauthorized copy, reproduction or use is not permitted.

# SoftUni Diamond Partners

Software University

SmartIT

Coca-Cola HBC Bulgaria

SUPER HOSTING .BG

createX

Postbank

Решения за твоето утре

DRAFT KINGS

VIVACOM

AMBITIONED

INDEAVR

Serving the high achievers

SOFTWARE GROUP

PHAR VISION

# Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, softuni.org
- 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://softuni.org
- © Software University – https://softuni.bg