DBC
DIGITAL BUSINESS COMPANY®

VEM SER
DBC

# Spring Web

## Aula 1 – Fundamentos

# Sumário do Módulo

- Spring Web (2 semanas)
- Spring Data (1 semana)
- Spring Security (1 semana)

# Sumário Spring Web

- Fundamentos da Web
- Desenvolvimento Web com Spring Boot
- Injeção de Dependencia
- Bean Validation
- Lombok e ObjectMapper
- FeignClient
- E-mail
- OpenAPI (Swagger)

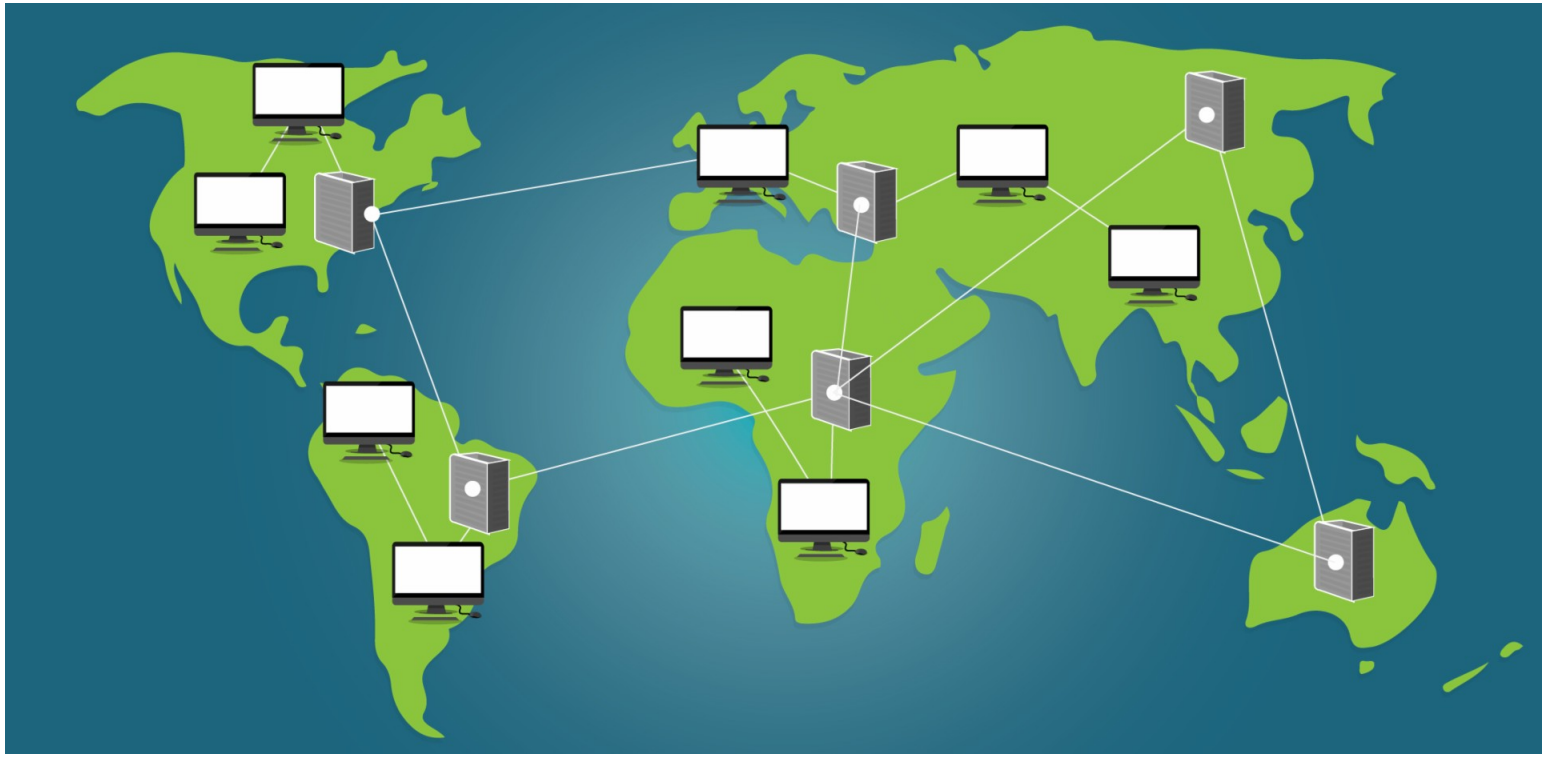
# Sumário

- Conceito de WEB
- Protocolos
- HTTP
- Formato de Requisição e Resposta
- URL e URI
- Verbos
- Corpo da Requisição
- JSON
- Kahoot
- Consumo de APIs com POSTMAN
- Exercício
- Homework

# O que é WEB?

# O que é WEB?

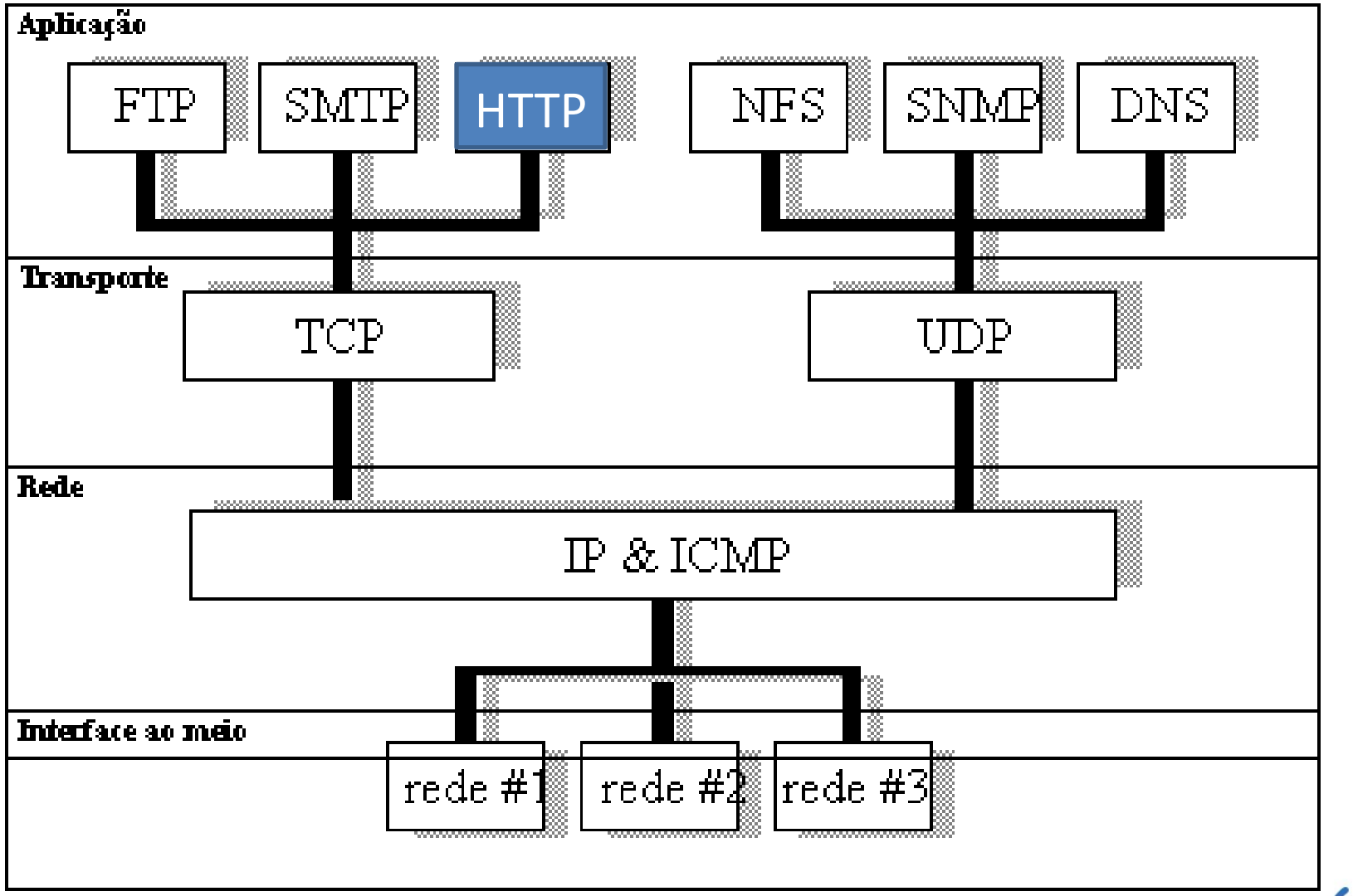
- É a nossa rede mundial de computadores (vulgo internet)



# Protocolos da WEB

- Protocolos da web são conjuntos de regras de como as coisas devem acontecer na Internet.
- Antes de mais nada são protocolos de Rede.
- A Internet e seu funcionamento, nada mais é do que muitas redes conectadas entre si.
- Os protocolos de Internet existem e foram criados para possibilitar a comunicação entre dois pontos da rede.

# Protocolos da WEB



# Protocolos da WEB

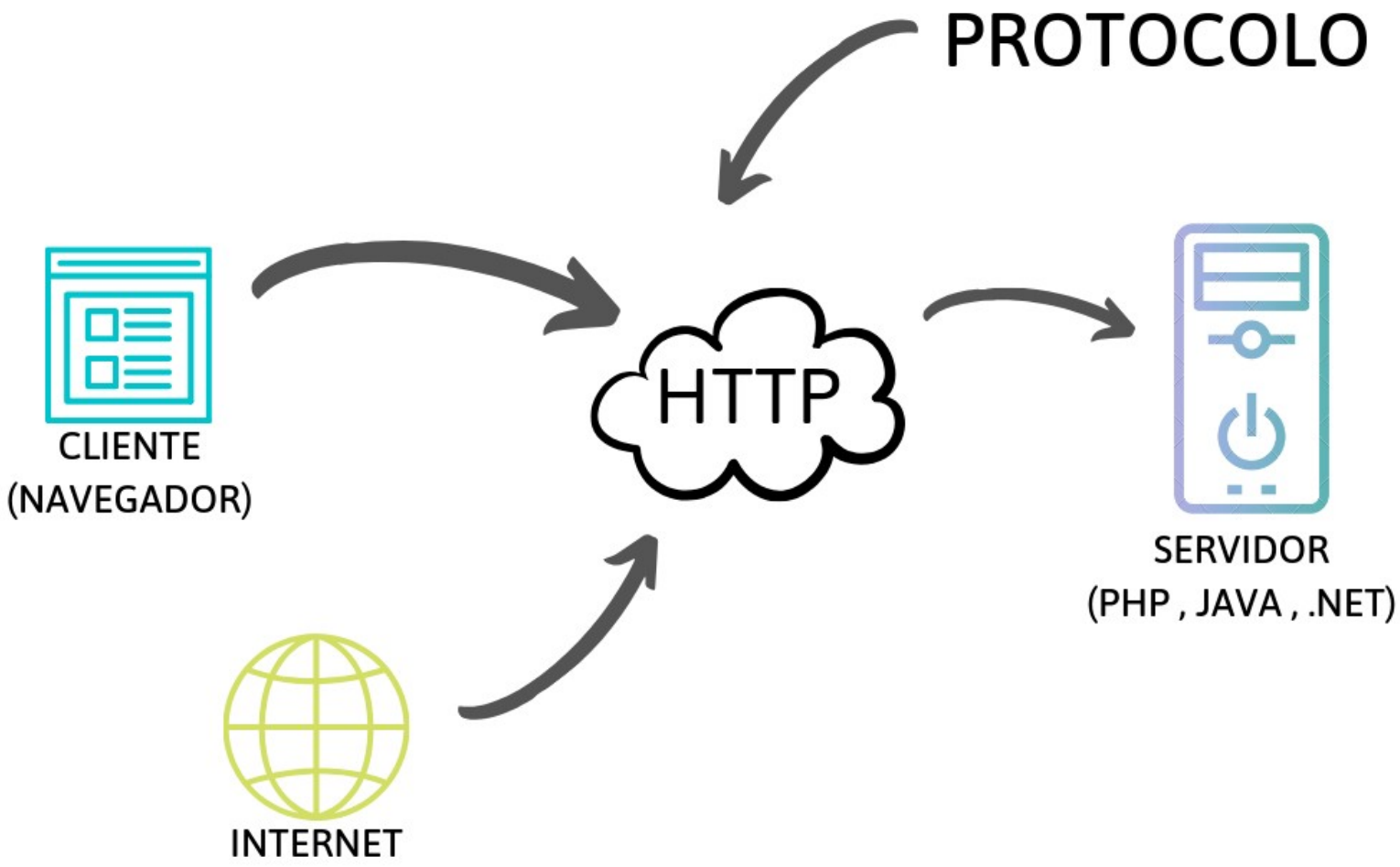
- TCP/IP
- **HTTP**
- POP, IMAP e SMTP
- FTP, SFTP e FTPS
- SSH

# Hypertext Transfer Protocol (HTTP)

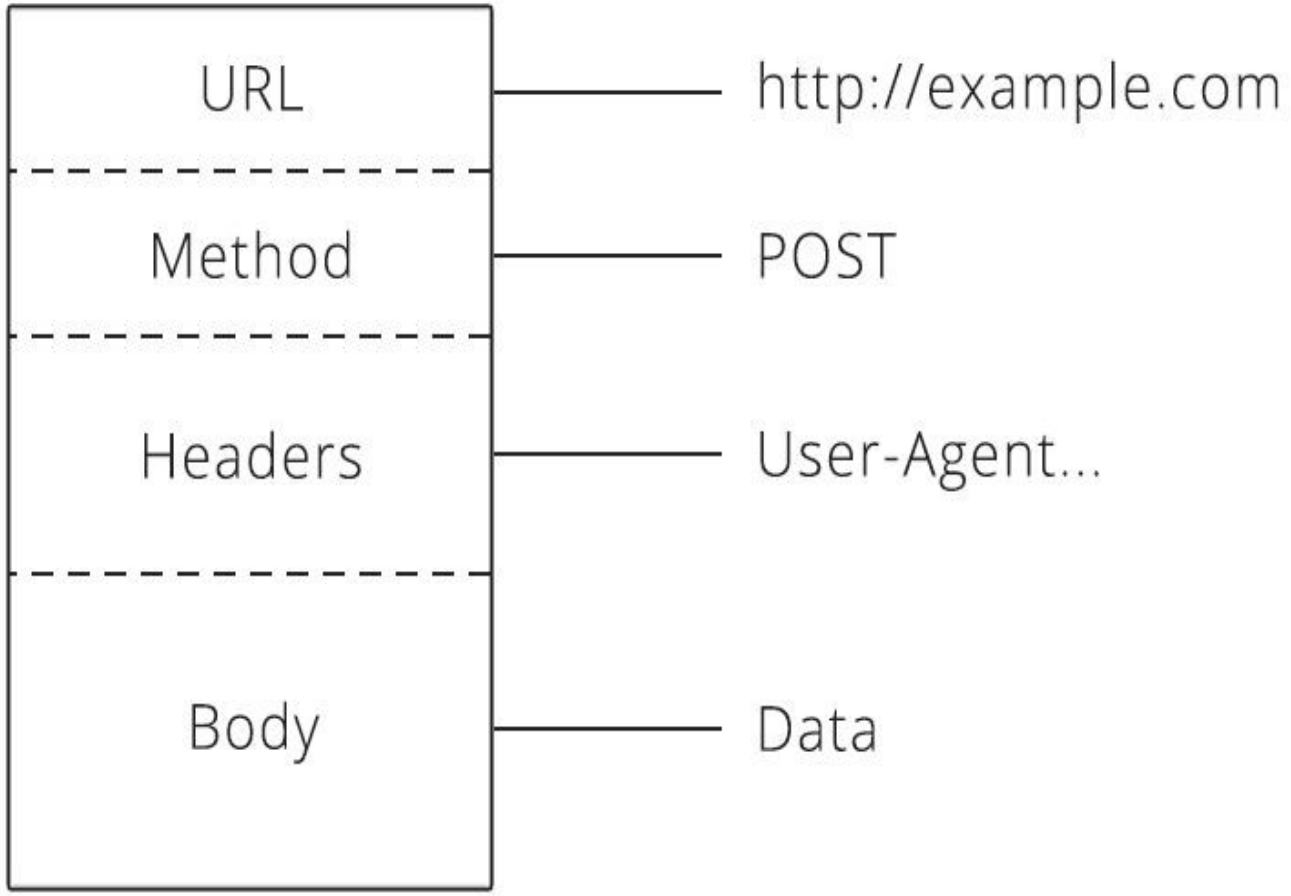
- HTTP possibilita que as pessoas que inserem a URL do seu site na Web possam ver os conteúdos e dados que nele existem.
- Berners-Lee propôs pela primeira vez o projeto “WorldWideWeb” em **1989** – hoje conhecido como World Wide Web (WWW) . A primeira versão do protocolo tinha apenas um método, chamado **GET.**
- O padrão HTTP / 1.1, como definido no RFC 2068, foi oficialmente lançado em janeiro de **1997**, enquanto melhorias e atualizações do padrão HTTP / 1.1 vieram em junho de **1999.**
- Em **2007**, o HTTPbis Working Group foi formado, em parte, para revisar e esclarecer a especificação HTTP / 1.1. Em junho de **2014**, divulgou uma especificação atualizada.

- https://rockcontent.com/br/blog/http/

# Hypertext Transfer Protocol (HTTP)



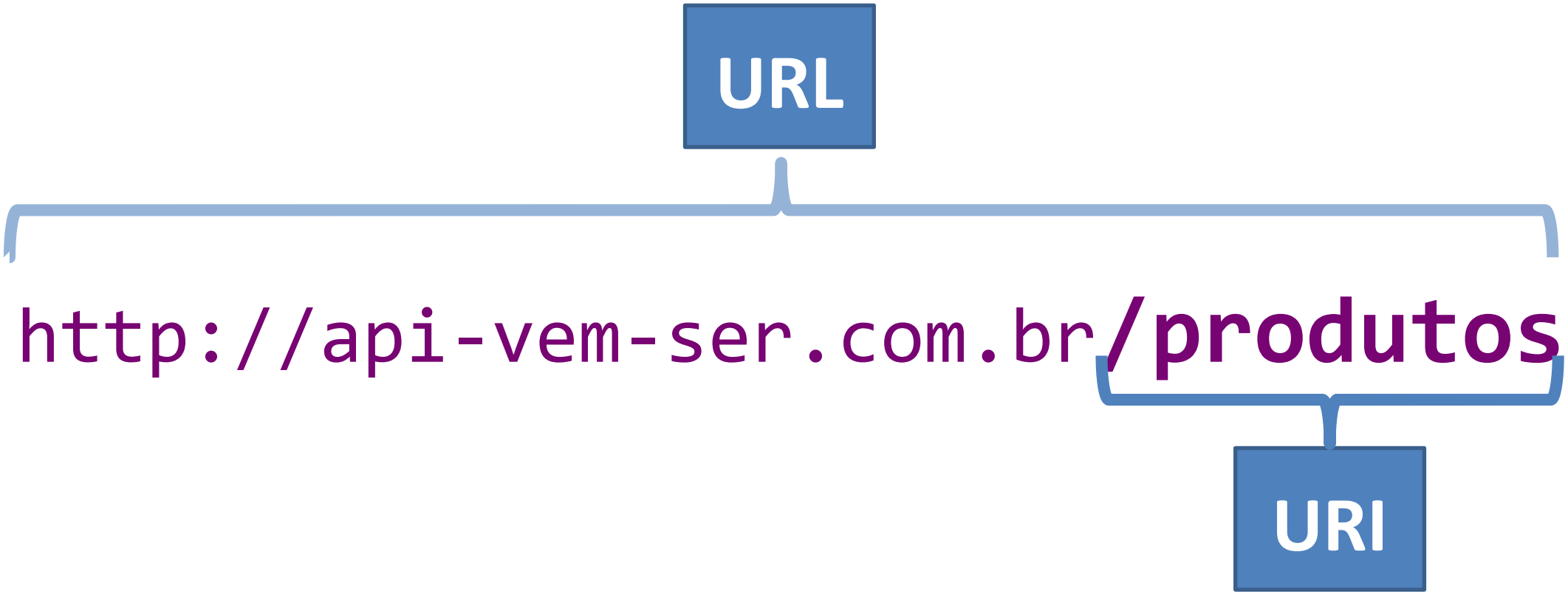
# Hypertext Transfer Protocol (HTTP)




*Request*

# URL e URI

- **URL** – Uniform Resource Locator
- **URI** – Uniform Resource Identifier



# Verbos HTTP

- São as **ações** aplicadas aos nossos recursos HTTP **(endpoints)**

- Os verbos mais utilizados são:
  - **GET:** Solicita informações do recurso;
  - **POST:** Criar um novo recurso (utiliza body);
  - **PUT:** Atualizar um recurso (utiliza body);
  - **DELETE:** Remove recurso;

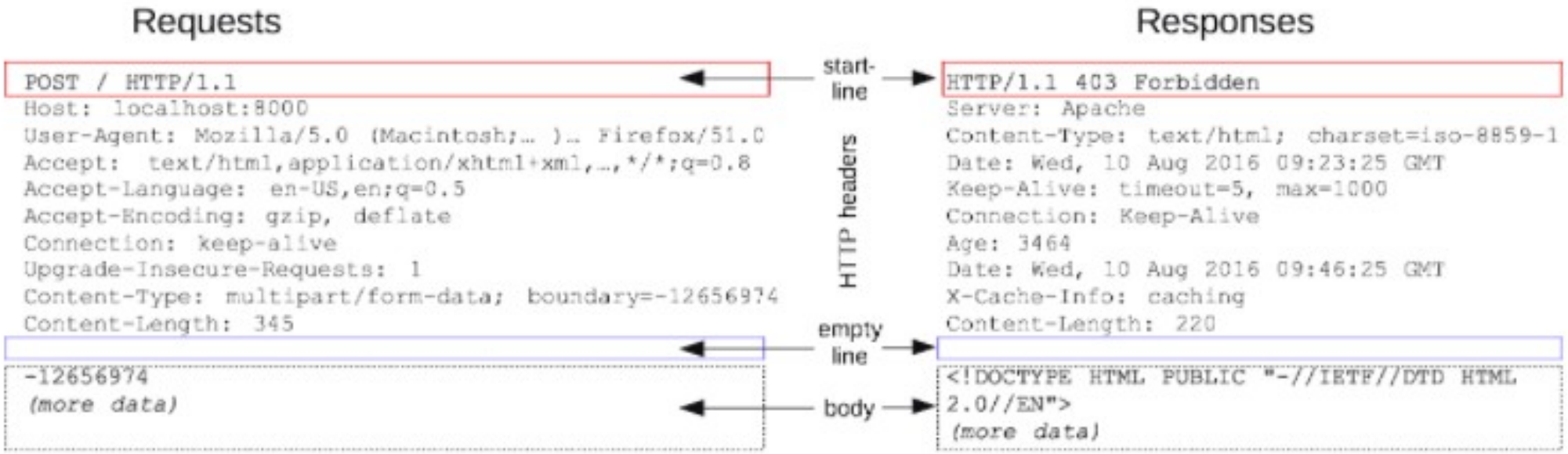
- Outros:  HEAD, OPTION, CONNECT, TRACE, PATCH

# Verbos HTTP

| Task | Method | Path |
| --- | --- | --- |
| Create a new task | POST | /tasks |
| Delete an existing task | DELETE | /tasks/{id} |
| Get a specific task | GET | /tasks/{id} |
| Search for tasks | GET | /tasks |
| Update an existing task | PUT | /tasks/{id} |

# Body (corpo)

- Corpo da nossa requisição ou resposta.
- Dados que enviamos para o recurso por meio dos métodos **POST e PUT.**
- Recebemos um corpo também na resposta de um **GET** ou demais.



# JavaScript Object Notation (JSON)

- **JSON** é um formato de dados baseado em texto seguindo a sintaxe do objeto **JavaScript.**

- Um objeto JSON pode ser armazenado em seu próprio arquivo, que é basicamente apenas um arquivo de texto com uma extensão de .json, e um MIME type **application/json.**

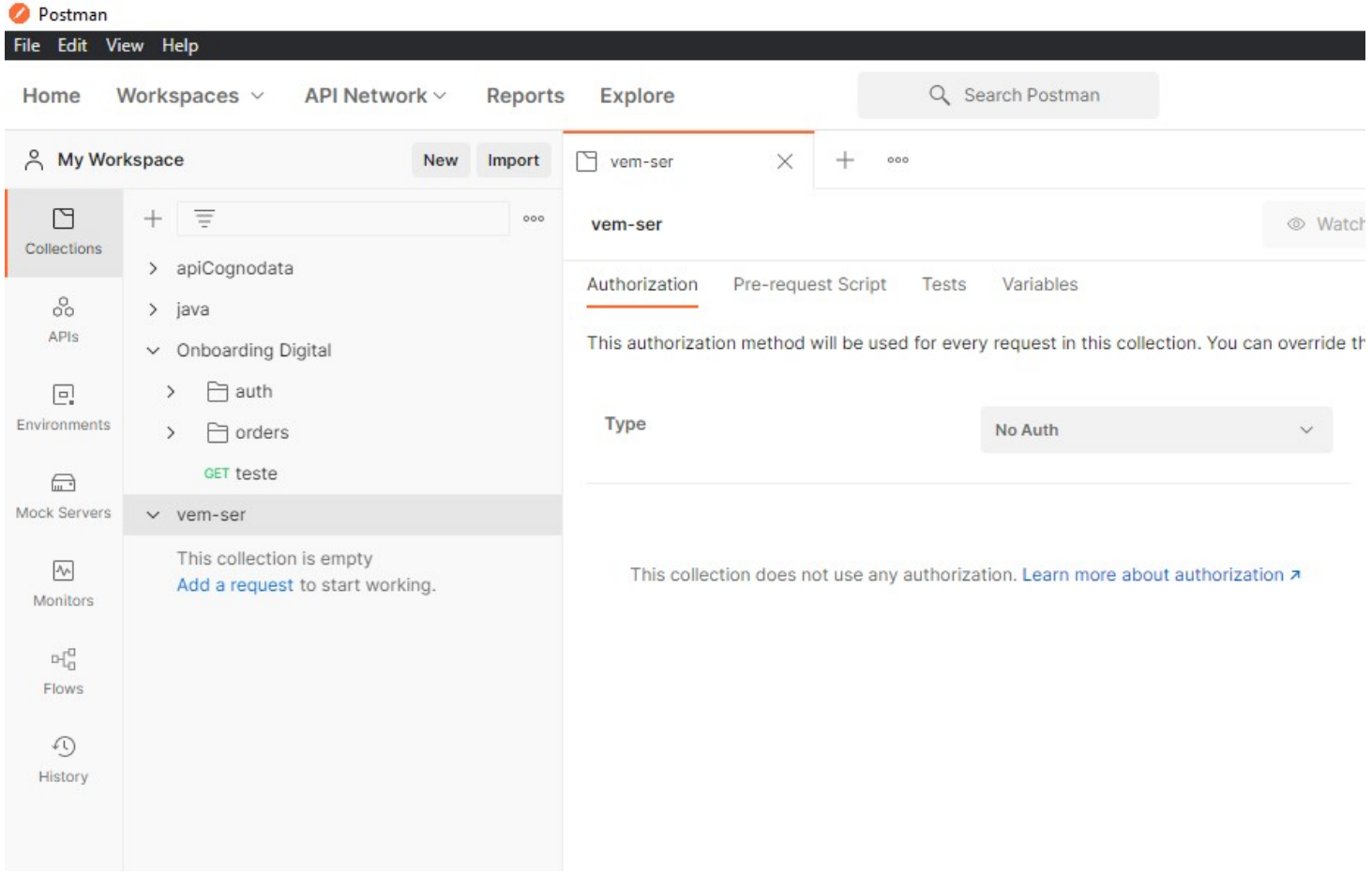
# JSON

```
{
  "name": "Molecule Man",
  "age": 29,
  "secretIdentity": "Dan Jukes",
  "powers": [
    "Radiation resistance",
    "Turning tiny",
    "Radiation blast"
  ]
}
```

- https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Objects/JSON

# Postman

- Software para fazer chamadas HTTP




- https://www.postman.com/downloads

# Vamos praticar!

# Exercício #1

- Utilizando o Postman e a API https://jsonplaceholder.typicode.com

- Utilizar GET, POST, PUT e DELETE para posts.

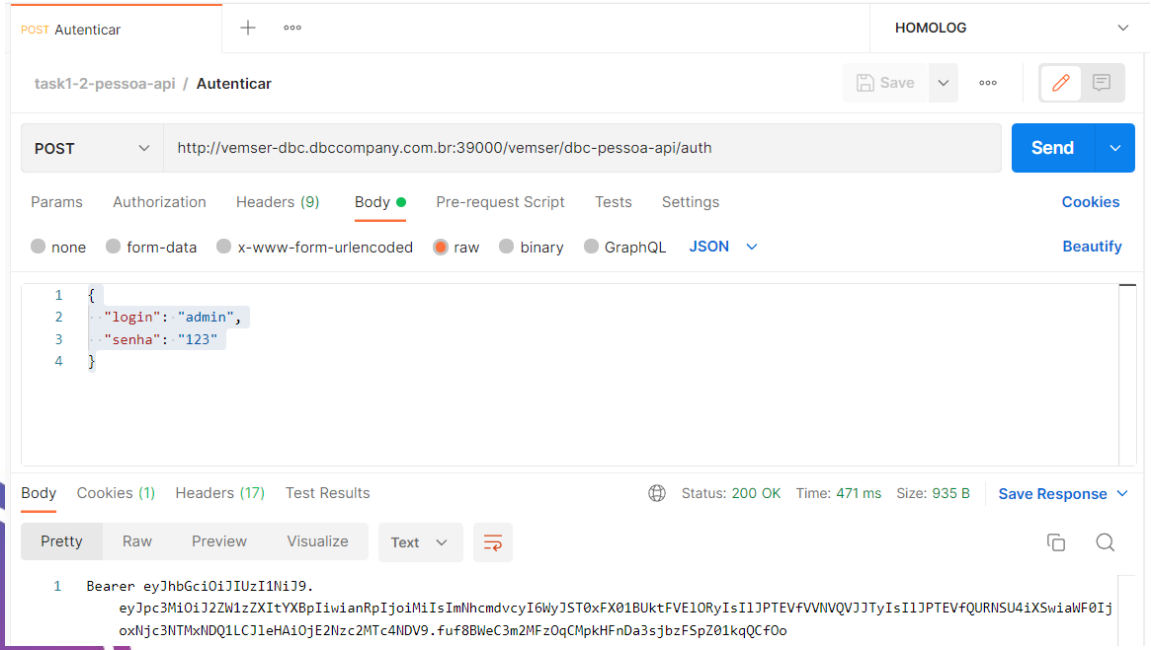
# Task #1

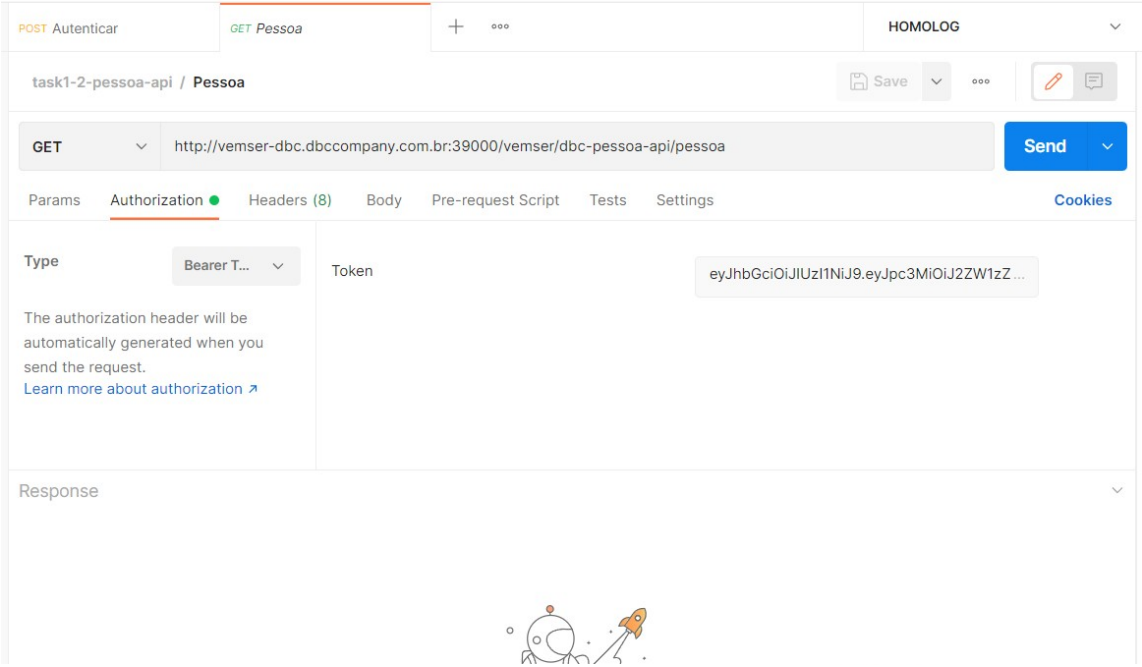
- Criar uma pasta “modulo-3-1” no git, criar uma collection no postman com o nome “task-1-1” e implementar os seguintes métodos:
- Utilizando o Postman e a API https://jsonplaceholder.typicode.com
- Utilizar GET, POST, PUT e DELETE para:
  - /comments
  - /albums
  - /photos
  - /todos
  - /users

# Task #2

- Criar uma pasta “modulo-3-1” no git, criar uma collection no postman com o nome “task-1-2” e implementar os seguintes métodos:
- Utilizando o Postman e a API:

  http://vemser-dbc.dbccompany.com.br:39000/vemser/dbc-pessoa-api
- Utilizar Autenticar com POST /auth para Autenticar e pegar o TOKEN e após usar ele nas chamadas, na aba Authorization com tipo Bearer Token ({"login": "admin",  "senha": "123"}):



# Task #2

- Utilizando o Postman e a API:
  http://vemser-dbc.dbccompany.com.br:39000/vemser/dbc-pessoa-api
- Utilizar GET, POST, PUT e DELETE para:
  - GET /pessoa
  - PUT /pessoa/{id}
  - DELETE /pessoa/{id}
  - POST /pessoa
  - GET /contato
  - GET /contato/{idPessoa}
  - POST /contato/{idPessoa}
  - PUT /contato/{idContato}
  - DELETE /contato/{idContato}

# Obrigado!

DBC

DIGITAL BUSINESS COMPANY®

/dbc.company /dbccompany /dbccompany.com.br /company/dbc-company