

# EMBARCADERO CONFERENCE



 embarcadero®

# Saia do básico para turbinar sua API

RAFAEL ARAUJO



CONFERENCE

EMBARCADERO CONFERENCE 2022



# Saia do básico para turbinar sua API



- Ajudo empresas a atingirem seus objetivos de negócio com o uso da tecnologia
- Coordenador de Sustentação na Softplan
- Ajudo desenvolvedores a evoluir tecnicamente e potencializar os resultados de seu esforço
- Fundador da TaskIP Treinamentos e Consultoria
- Palestrante em diversos eventos da Comunidade Delphi
- Aprender, Impactar e Divertir

# Saia do básico para turbinar sua API



O que seria o básico em  
construção de APIs?

# Saia do básico para turbinar sua API

HTTP Method	Resource	Action	Status Code
GET	/findCustomer/1	READ	200
POST	/saveCustomer	CREATE	200
POST	/alterCustomer/1	UPDATE	200
GET/POST	/deleteCustomer/1	DELETE	200



# Saia do básico para turbinar sua API



# Saia do básico para turbinar sua API



# Saia do básico para turbinar sua API





# Saia do básico para turbinar sua API



# Saia do básico para turbinar sua API

- **Verbos / Métodos HTTP**

- GET - Read
- POST - Create
- PUT - Replace
- PATCH - Partial Modifications
- DELETE - Delete



HTTP Method	Resource	Action
GET	/customers/1	READ
POST	/customers	CREATE
PUT	/customers/1	UPDATE
DELETE	/customers/1	DELETE

# Saia do básico para turbinar sua API

- **Response Status Codes**

- Informational (100 – 199)
- Successful (200 – 299)
- Redirection Messages (300 – 399)
- Client Error (400 – 499)
- Server Error (500 – 599)



HTTP Method	Resource	Action	Status Code
GET	/customers/1	READ	200
POST	/customers	CREATE	201
PUT	/customers/1	UPDATE	204
DELETE	/customers/1	DELETE	204

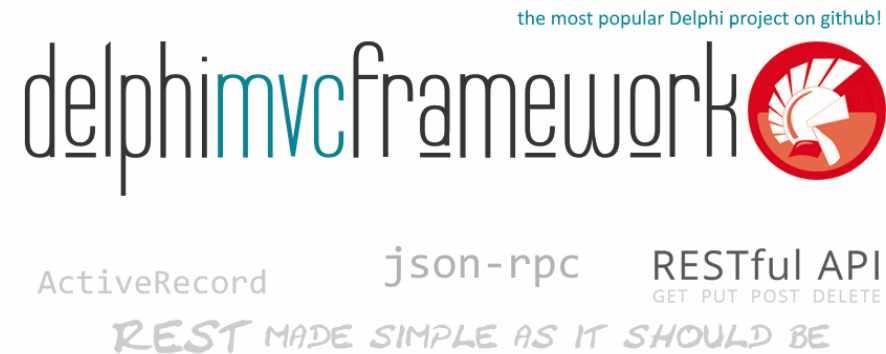


# Saia do básico para turbinar sua API



# Saia do básico para turbinar sua API

- WebBroker
- Frameworks
  - DataSnap
  - RAD Server
  - Horse
  - Mars
  - **DelphiMVCFramework**

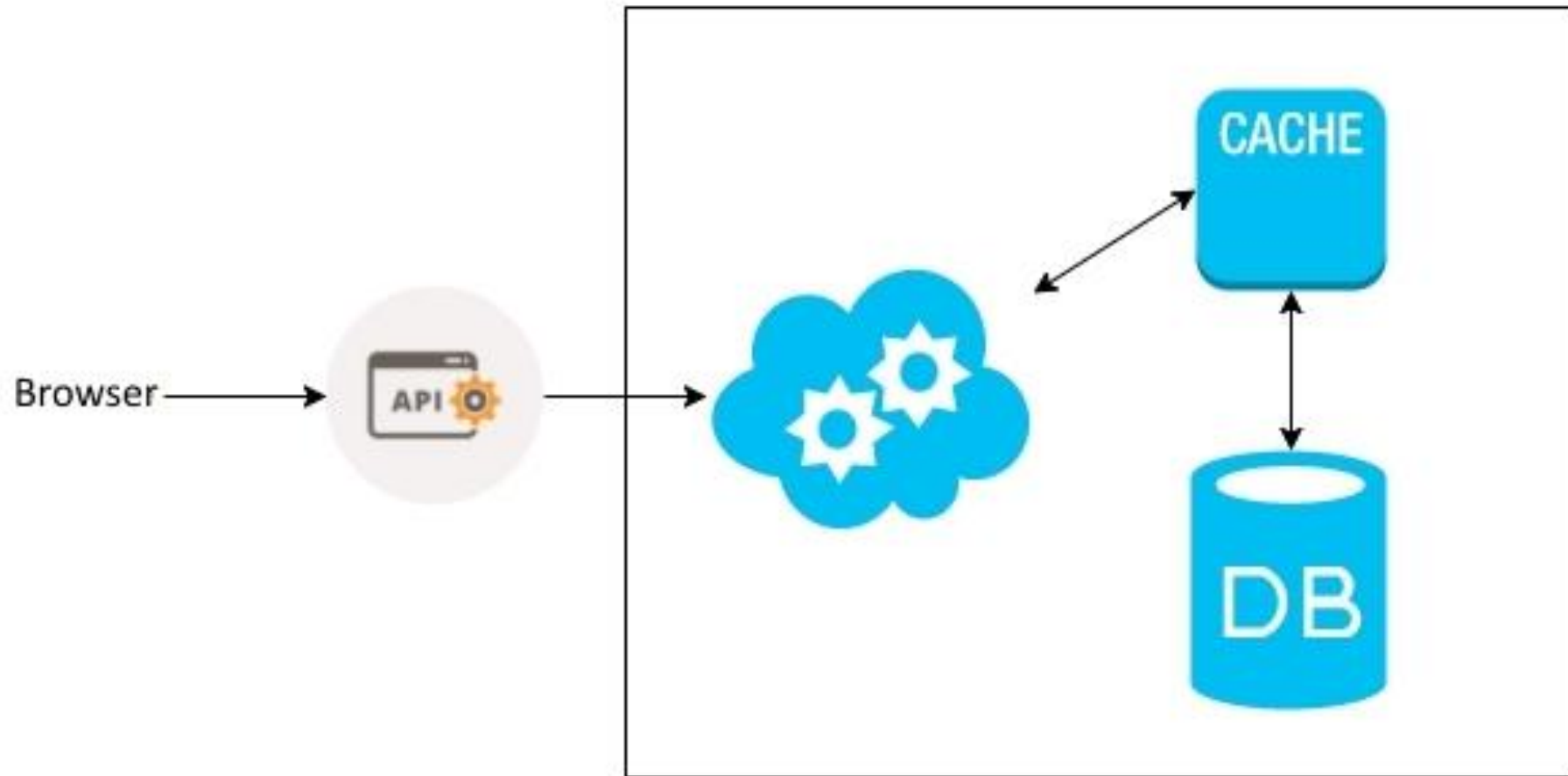


# Filtros e Paginação





# Cache de Dados



# Cache de Dados

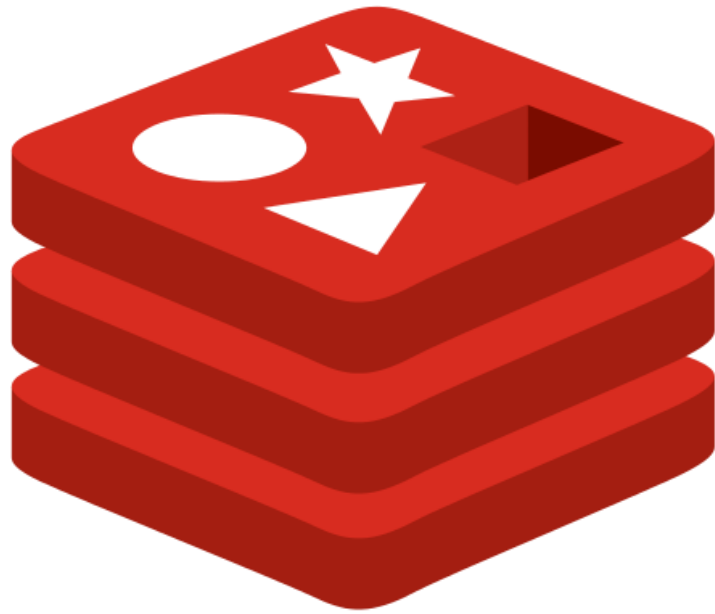
“Os objetivos do cache de dados HTTP são **eliminar o envio de requisições** o máximo possível e, se ela se fizer necessária, **reduzir os dados de resposta.**”

# Cache de Dados

- Reduzir o custo para o funcionamento aplicação
- De difícil acesso, acessado constantemente e/ou computacionalmente custoso de se obter
- Pode salvar uma enorme quantidade de tempo
- Diminuir tempo da requisição = menor necessidade de poder de processamento
- Escalar mais facilmente
- Dados real-time não deve ser cacheado



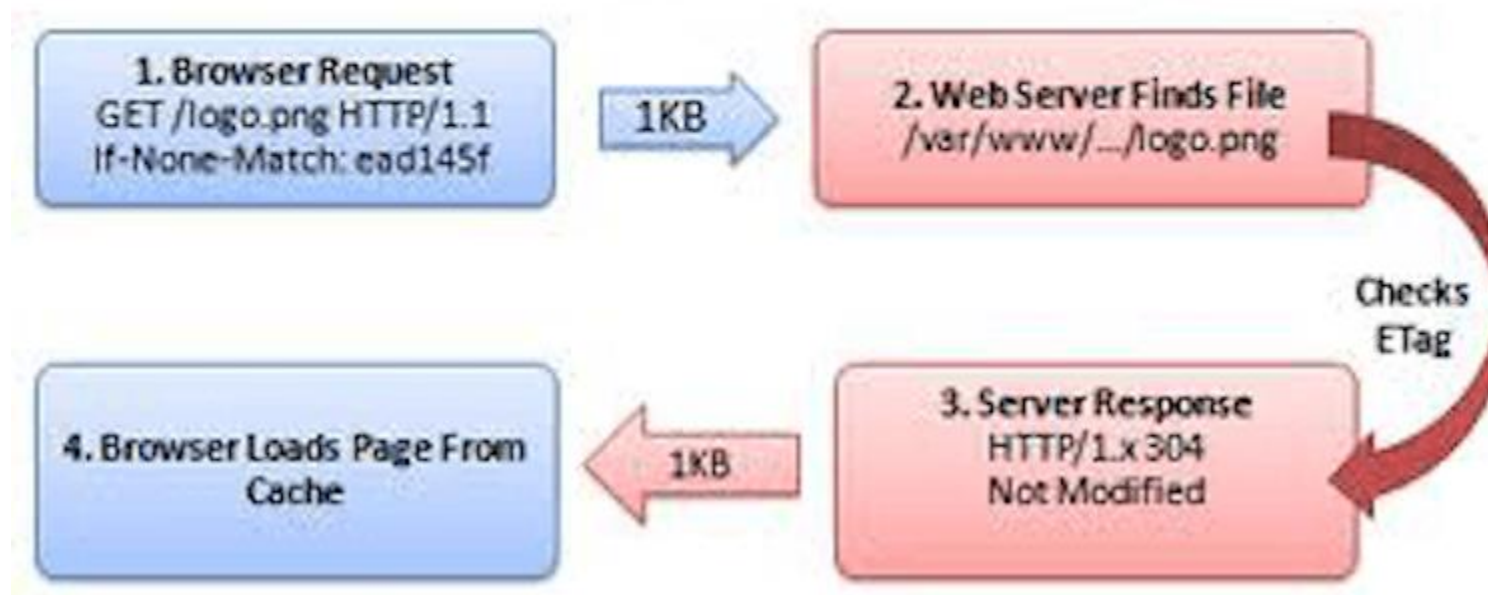
# Cache de Dados



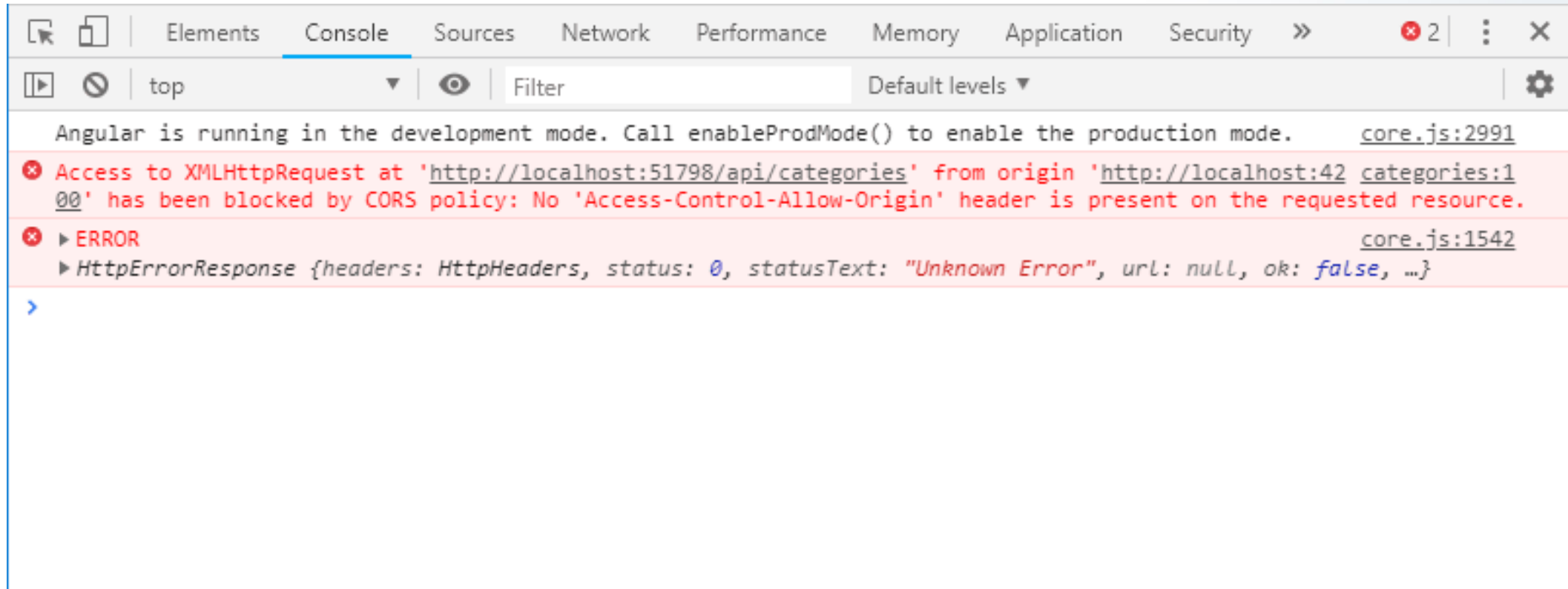
redis

# Cache de Dados

## HTTP Cache: If-None-Match



# CORS – Cross-Origin Resource Sharing

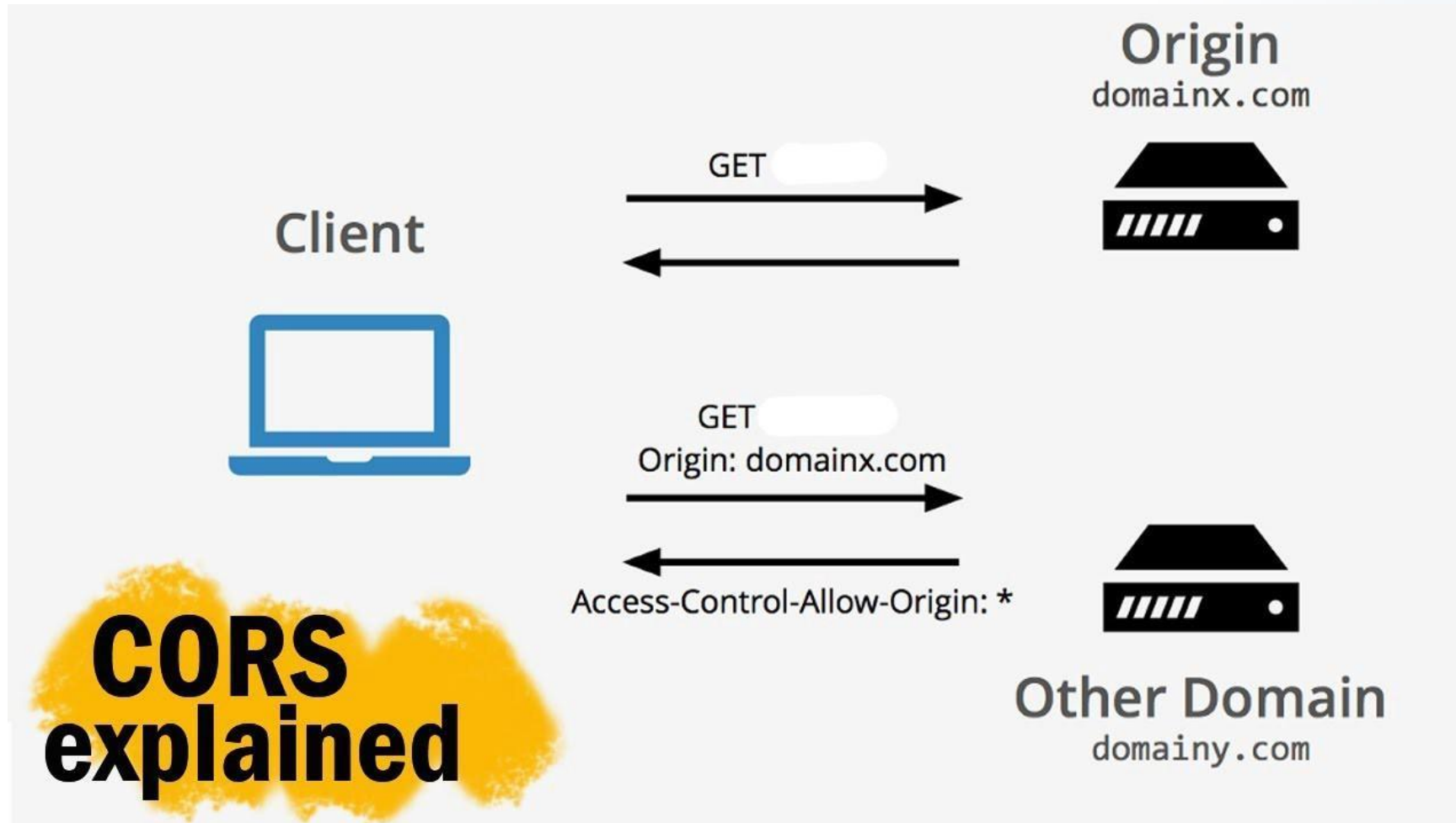


The screenshot shows a web browser's developer console with the 'Console' tab selected. The console displays a message from Angular about development mode and two error messages. The first error is a CORS policy violation: 'Access to XMLHttpRequest at 'http://localhost:51798/api/categories' from origin 'http://localhost:4200' has been blocked by CORS policy: No 'Access-Control-Allow-Origin' header is present on the requested resource.' The second error is an 'ERROR' message from 'core.js:1542' indicating an 'Unknown Error' in an 'HttpErrorResponse' object.

```
Angular is running in the development mode. Call enableProdMode() to enable the production mode.    core.js:2991
✖ Access to XMLHttpRequest at 'http://localhost:51798/api/categories' from origin 'http://localhost:4200' has been blocked by CORS policy: No 'Access-Control-Allow-Origin' header is present on the requested resource.
✖ ▶ ERROR    core.js:1542
  ▶ HttpErrorResponse {headers: HttpHeaders, status: 0, statusText: "Unknown Error", url: null, ok: false, ...}
```



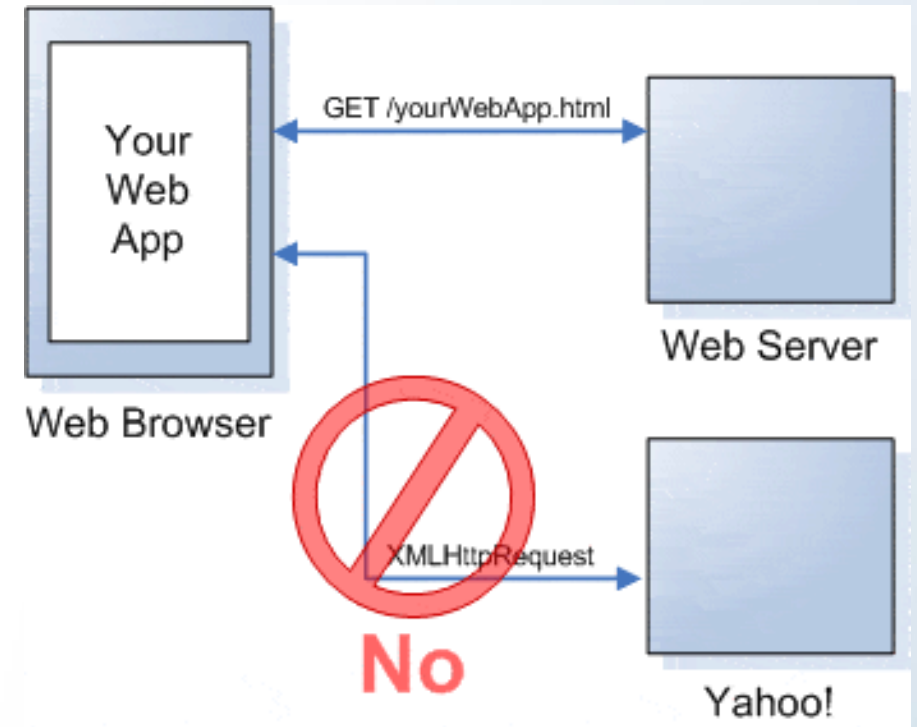
# CORS – Cross-Origin Resource Sharing



# CORS – Cross-Origin Resource Sharing

- **Same-Origin Policy**

- Política da mesma origem
- Padrão dos navegadores
- Só é possível acessar recursos localizados na mesma origem que da solicitação
- **Protocolo** / Host / **Porta**
- **http://**www.minhaaplicacao.com.br:**80**



# CORS – Cross-Origin Resource Sharing

- Compartilhamento de Recursos de Origem Cruzada
- Permite que você restrinja quais domínios podem usar seus recursos de backend
- Não é um mecanismo de segurança forte
  - Basic Authentication
  - OAuth2
  - JWT



# CORS – Cross-Origin Resource Sharing

*“Garantir que apenas domínios autorizados tenham acesso às informações fornecidas por sua API.”*

# CORS – Cross-Origin Resource Sharing

- Compartilhando recursos de origem diferente
- Ao habilitar o CORS padrão no Projeto, é liberado o acesso de qualquer domínio a nossa API
  - Access-Control-Allow-Origin(origem que pode acessar)
  - Access-Control-Allow-Methods(verbos que podem ser utilizados)
  - Access-Control-Allow-Headers(quais itens são aceitos no cabeçalho)

Swagger



Swagger™

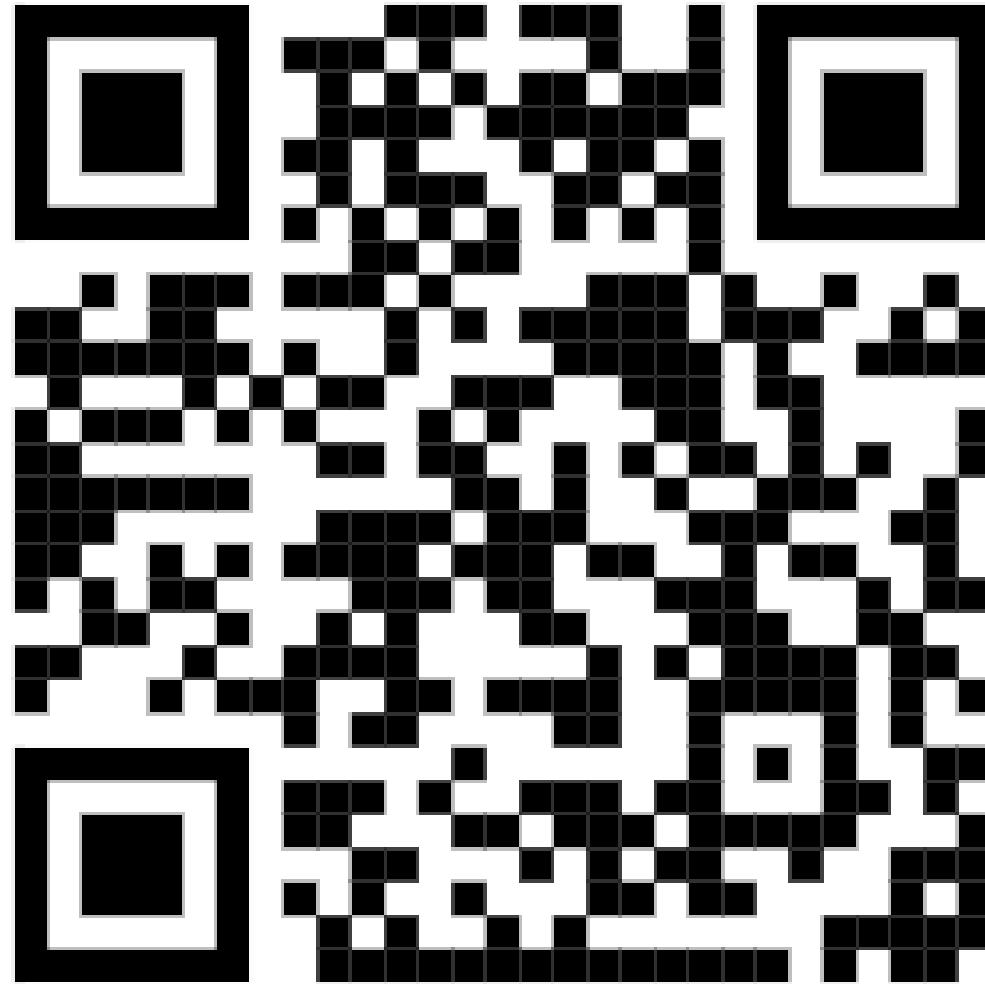




# Runners e Actions



# Runners e Actions



# From Zero To Hero

- Zero
- Preparação e Nivelamento
- Autenticação e Autorização
- Segurança
- Database
- Documentação
- Deploy
- Avançado
- Hero







[rafael@taskip.com.br](mailto:rafael@taskip.com.br)



[facebook.com/araujor82](https://facebook.com/araujor82)



[instagram.com/araujor82](https://instagram.com/araujor82)



[linkedin/in/araujor82](https://linkedin/in/araujor82)

# OBRIGADO