

Escola Superior de Tecnologia e Gestão

# **MULTI SNAKE**

**v1.0**

Luís Marques  
8170485

Licenciatura em Engenharia Informática

Laboratório de Desenvolvimento de Software  
3º Ano, 1º Semestre

3 de dezembro de 2020

[Página propositadamente deixada em branco.]

# **Multi Snake**

## **v1.0**

Luís Marques

Licenciatura em Engenharia Informática

[Página propositadamente deixada em branco.]

# Histórico de Versões

Name	Date	Reason for change(s)	Version
Luís Marques	25-11-2020	Início do relatório	0.4
Luís Marques	02-12-2020	Adição da stack tecnológica e estado atual do projeto	1.0

[Página propositadamente deixada em branco.]

# Conteúdo

<b>Lista de Figuras</b>	<b>v</b>
<b>Glossário</b>	<b>vii</b>
<b>Abreviaturas</b>	<b>ix</b>
<b>1 Introdução</b>	<b>1</b>
1.1 Contextualização . . . . .	1
1.2 Âmbito . . . . .	1
1.3 Objetivos . . . . .	1
<b>2 Visão Geral</b>	<b>3</b>
2.1 Tecnologias Utilizadas . . . . .	3
2.2 Aplicação Web . . . . .	4
2.3 API REST . . . . .	5
<b>3 Requisitos</b>	<b>7</b>
3.1 Funcionais . . . . .	8
3.2 Não Funcionais . . . . .	10
<b>4 Estado Atual da Aplicação</b>	<b>11</b>
<b>References</b>	<b>13</b>

[Página propositadamente deixada em branco.]



# Lista de Figuras

2.1	Stack tecnológica do projeto . . . . .	3
2.2	Diagrama de Casos de Uso da Web App . . . . .	4
2.3	Arquitetura da API REST . . . . .	5
4.1	Componente de login de utilizadores . . . . .	11
4.2	Componente de registo de utilizadores . . . . .	12
4.3	Página privada de um utilizador com a versão singleplayer do jogo da cobra . . .	12

[Página propositadamente deixada em branco.]

# Glossário

**ASP.NET** é a plataforma da Microsoft para o desenvolvimento de aplicações web. 2

**Git** é um sistema de controlo de versões utilizado principalmente no desenvolvimento de software. vii, 1

**GitLab** é um gestor de repositórios de software baseado em Git. 1

**Hypertext Transfer Protocol** é um protocolo de comunicação utilizado para sistemas de informação. ix

**Representational State Transfer** (Transferência Representacional de Dados) é um estilo de arquitetura de software que define um conjunto de restrições a serem utilizadas por serviços web. ix, 5

**Scrum** é uma *framework* de gestão de projetos de desenvolvimento ágil. 1

**Swagger** é uma linguagem de descrição de interface para descrever API's RESTful expressas usando JSON. 10

[Página propositadamente deixada em branco.]

# Abreviaturas

**API** *Application Programming Interface*. 2, 5

**BD** Base de Dados. 5, 10

**DTO** *Data Transfer Object*. 5

**ESTG** Escola Superior de Tecnologia e Gestão. 1

**HTTP** *Hypertext Transfer Protocol*. 5

**JSON** *JavaScript Object Notation*. vii, ix, 2, 5

**JWT** *JSON Web Token*. 1, 2, 8

**LDS** Laboratório de Desenvolvimento de Software. 1

**LEI** Licenciatura em Engenharia Informática. 1

**REST** *Representational State Transfer*. 5

[Página propositadamente deixada em branco.]

# Capítulo 1

## Introdução

---

1.1	Contextualização . . . . .	1
1.2	Âmbito . . . . .	1
1.3	Objetivos . . . . .	1

---

### 1.1 Contextualização

O presente documento visa descrever todo o trabalho realizado no desenvolvimento deste projeto, que foi realizado para a prova oral da unidade curricular de Laboratório de Desenvolvimento de Software da Licenciatura em Engenharia Informática na Escola Superior de Tecnologia e Gestão do Instituto Politécnico Porto no ano de 2020.

### 1.2 Âmbito

Este projeto consiste no desenvolvimento de uma aplicação web para um jogo da cobra com a vertente multijogador. Existe também um serviço de gestão de utilizadores com autenticação JWT. O utilizador terá a possibilidade de criar e de se juntar a uma sessão de jogo.

### 1.3 Objetivos

Para a realização deste projeto, foram definidos alguns objetivos principais com a finalidade de manter uma boa coesão e garantia de qualidade do produto e dos métodos de desenvolvimento do mesmo. Alguns dos objetivos são obrigatórios da unidade curricular em que se insere o trabalho e outros foram definidos pelo dono do produto e pelo gestor do projeto.

São eles:

- Utilização da metodologia Scrum e uso da plataforma ;
- Uso do GitLab e Git para gestão e versionamento do projeto, gestão de tarefas e *wiki* do projeto;

- Desenvolvimento de serviços em ASP.NET Core sob a forma de uma *Application Programming Interface* (API) com autenticação de utilizadores via *JSON Web Token* (JWT);
- Documentação do projeto com ferramentas adequadas;
- Aplicação para o cliente capaz de consumir os serviços criados.



## Capítulo 2

# Visão Geral

---

2.1	Tecnologias Utilizadas . . . . .	3
2.2	Aplicação Web . . . . .	4
2.3	API REST . . . . .	5

---

### 2.1 Tecnologias Utilizadas

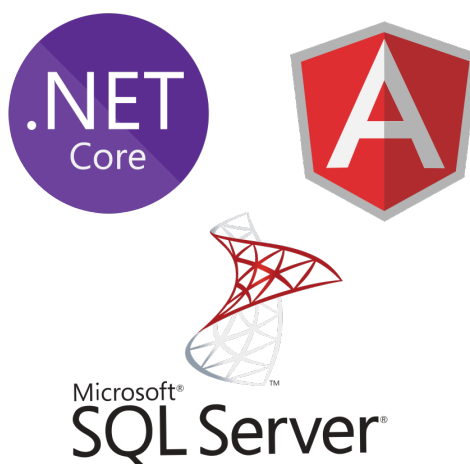


Figura 2.1: Stack tecnológica do projeto

Na figura 2.1 está representado o grupo de tecnologias utilizadas no desenvolvimento deste projeto. Para o *backend* foi utilizada a versão 3.1 do *ASP.NET*, para o *frontend* foi utilizada framework *AngularJS* na sua versão 11.0.2. Para a gestão de dados da aplicação foi utilizada um sistema de gestão de bases de dados relacionais chamado *Microsoft SQL Server* alojado num serviço *Microsoft Azure*.

## 2.2 Aplicação Web

Na figura 2.2 é possível analisar um diagrama de Casos de Uso relativo à aplicação web de um modo geral.



Figura 2.2: Diagrama de Casos de Uso da Web App

A aplicação web deve consistir numa plataforma simples e de fácil utilização, por isso a sua responsividade e usabilidade devem ser muito elevadas. O utilizador deve ser capaz de se registar e de se autenticar. Com o login realizado, deverá ser capaz de criar ou de se juntar a uma sessão de jogo para disfrutar do magnífico e único jogo da cobra, mas numa versão mais competitiva e aliciante. Para além disto, o utilizador poderá ser capaz de visitar a sua página pessoal com as suas estatísticas de partidas realizadas, bem como ter a possibilidade de atualizar qualquer dado pessoal introduzido no registo.

## 2.3 API REST

Na figura 2.2 está representado um desenho concetual da arquitetura da API.

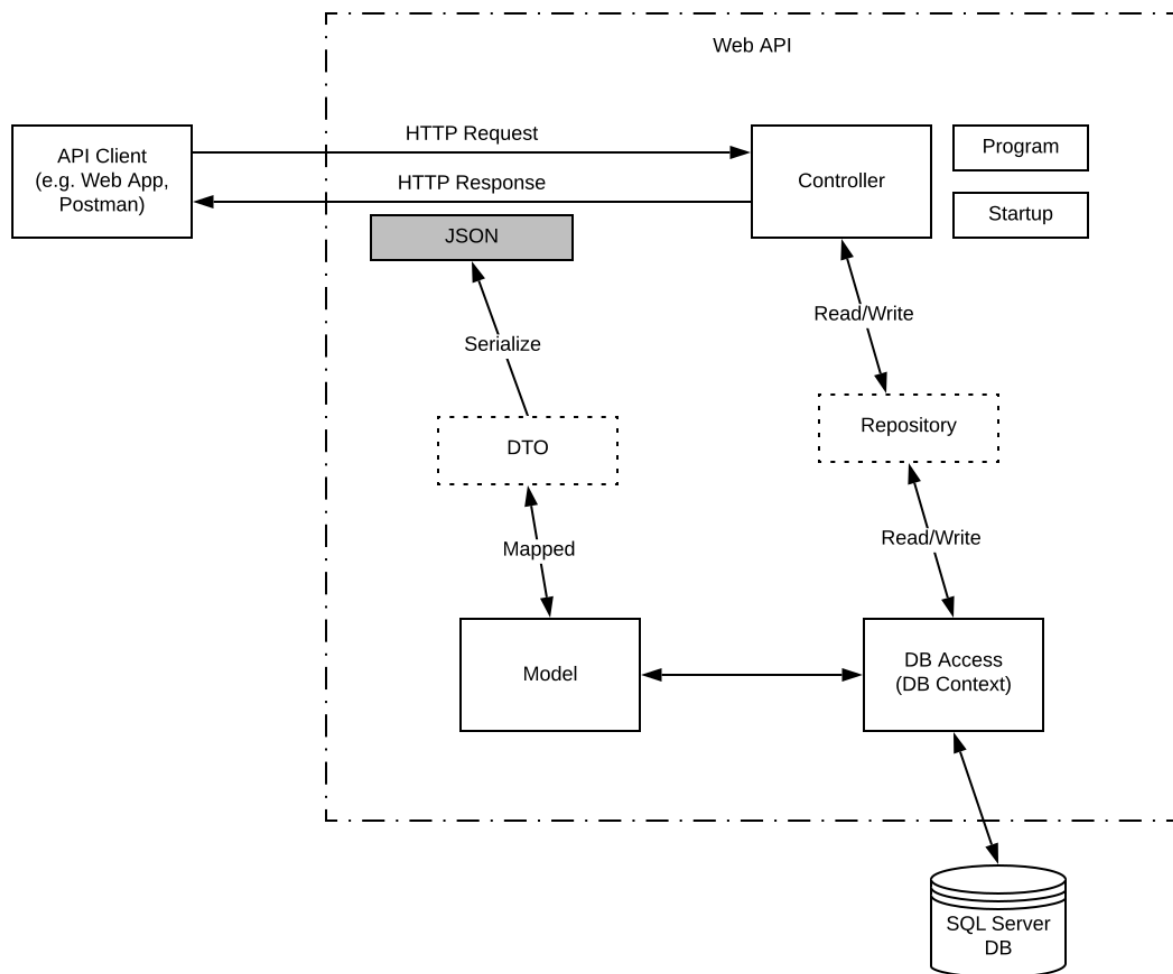


Figura 2.3: Arquitetura da API REST

A API será desenvolvida com a utilização do padrão *Representational State Transfer* (REST). Para iniciar, quando é realizado um pedido HTTP através de um cliente da API (e.g., a nossa aplicação web), este é recebido e tratado por um *Controller* que irá comunicar com a Base de Dados (BD) do sistema através do componente *DBContext*. Para que o controlador consiga comunicar com este componente de forma mais genérica e modularizada foi desenvolvido um componente *Repository*.

A informação retornada pela BD é estruturada pelo *Model* e existirá também um componente *Data Transfer Object* (DTO) para gerir a informação que deve ser enviada na resposta, de modo a evitar fugas de informação. Para finalizar, a resposta é enviada num formato JSON.

[Página propositadamente deixada em branco.]

## Capítulo 3

# Requisitos

---

3.1	Funcionais . . . . .	8
3.2	Não Funcionais . . . . .	10

---

Para a definição dos requisitos do projeto foram definidas três prioridades possíveis que podem ser atribuídas a cada requisito e os respectivos pontos de realização do mesmo:

- Alta: 8 pontos;
- Média: 5 pontos;
- Baixa: 2 pontos.

Estes pontos foram atribuídos com base na sequência de *Fibonacci*. Analisando os pontos associados a cada prioridade podemos ver que a realização de um requisito com prioridade alta tem mais importância que a junção de um requisito de prioridade média com um de prioridade baixa.

## 3.1 Funcionais

### RF\_1.1

**Nome:** O sistema tem de permitir o registo de um novo utilizador

**Categoria:** Gestão de utilizadores

**Descrição:** O registo de um novo utilizador tem de ser realizado com o preenchimento de alguns campos obrigatórios (email, nome de utilizador e palavra-passe).

**Prioridade:** Alta

### RF\_1.2

**Nome:** O sistema tem de permitir a atualização de um utilizador

**Categoria:** Gestão de utilizadores

**Descrição:** A atualização de um utilizador só é permitida para utilizadores já registados no sistema e com o login realizado.

**Prioridade:** Média

### RF\_1.3

**Nome:** O sistema tem de permitir a atualização da palavra-passe de um utilizador

**Categoria:** Gestão de utilizadores

**Descrição:** Para atualizar a palavra-passe de um utilizador tem de ser enviado uma hiperligação para o email do utilizador em questão com uma validade de 5 minutos.

**Prioridade:** Média

### RF\_1.4

**Nome:** O sistema pode permitir a consulta da página de perfil de um utilizador

**Categoria:** Gestão de utilizadores

**Descrição:** Qualquer utilizador sem estar autenticado pode consultar a página de um utilizador em específico e ver as estatísticas dos jogos desse utilizador.

**Prioridade:** Baixa

### RF\_2.1

**Nome:** O sistema tem permitir a autenticação de um utilizador

**Categoria:** Gestão de sessões

**Descrição:** O login de um utilizador registado tem de ser suportado por uma autenticação com JWT.

**Prioridade:** Alta

### RF\_2.2

**Nome:** O sistema tem permitir o logout de um utilizador

**Categoria:** Gestão de sessões

**Descrição:** Um utilizador autenticado tem de ter a possibilidade de sair da aplicação.

**Prioridade:** Alta

### RF\_3.1

**Nome:** O sistema deve permitir o modo de jogo de um único jogador

**Categoria:** Gestão o jogo

**Descrição:** O modo de jogo *single player* deve estar disponível para qualquer utilizador autenticado.

**Prioridade:** Média

### RF\_3.2

**Nome:** O sistema tem permitir o modo de jogo multijogador

**Categoria:** Gestão o jogo

**Descrição:** O modo de jogo *multiplayer* deve estar disponível para qualquer utilizador autenticado.

**Prioridade:** Alta

### RF\_3.3

**Nome:** O sistema tem permitir a criação de uma sala de jogo multijogador

**Categoria:** Gestão o jogo

**Descrição:** Qualquer utilizador autenticado tem a possibilidade de criar uma sala de jogo *multiplayer*. Apenas é permitido criar uma sala de jogo de cada vez por cada utilizador.

**Prioridade:** Alta

### RF\_3.4

**Nome:** O sistema tem permitir a junção a uma sala de jogo multijogador

**Categoria:** Gestão o jogo

**Descrição:** Qualquer utilizador autenticado tem a possibilidade de se juntar a uma sala de jogo *multiplayer*. Apenas é permitido juntar-se a uma sala de jogo de cada vez por cada utilizador.

**Prioridade:** Alta

## 3.2 Não Funcionais

### RNF\_1.1

**Nome:** A aplicação web tem de ser responsiva

**Categoria:** Aplicação web

**Descrição:** A aplicação web tem de ser o mais responsiva possível para que possa ser utilizada a partir de qualquer dispositivo, visto que não irá existir aplicação nativa.

**Prioridade:** Alta

### RNF\_1.2

**Nome:** A aplicação web tem de ter boa usabilidade

**Categoria:** Aplicação web

**Descrição:** A aplicação web tem de ser muito intuitiva e amiga do utilizador, para que não seja difícil a adaptação à mesma.

**Prioridade:** Alta

### RNF\_2.1

**Nome:** A API tem de estar fortemente documentada

**Categoria:** API REST

**Descrição:** A API tem de estar fortemente documentada com a utilização do Swagger.

**Prioridade:** Alta

### RNF\_3.1

**Nome:** A Base de Dados relacional tem de ser alojada num serviço cloud

**Categoria:** Base de Dados

**Descrição:** A Base de Dados relacional tem de ser alojada num serviço cloud da *Microsoft Azure*.

**Prioridade:** Alta



## Capítulo 4

# Estado Atual da Aplicação

Atualmente, o componente de gestão de utilizadores da aplicação está totalmente funcional, com o registo e o login de utilizadores. Estão a ser corrigidos alguns precalços ocorridos no desenvolvimento da gestão de sessões de utilizadores, para que apenas utilizadores registados e logados no sistema possam jogar.

Na figura 4.1 e 4.2 estão representados o componente de login e registo de um utilizador, respetivamente.

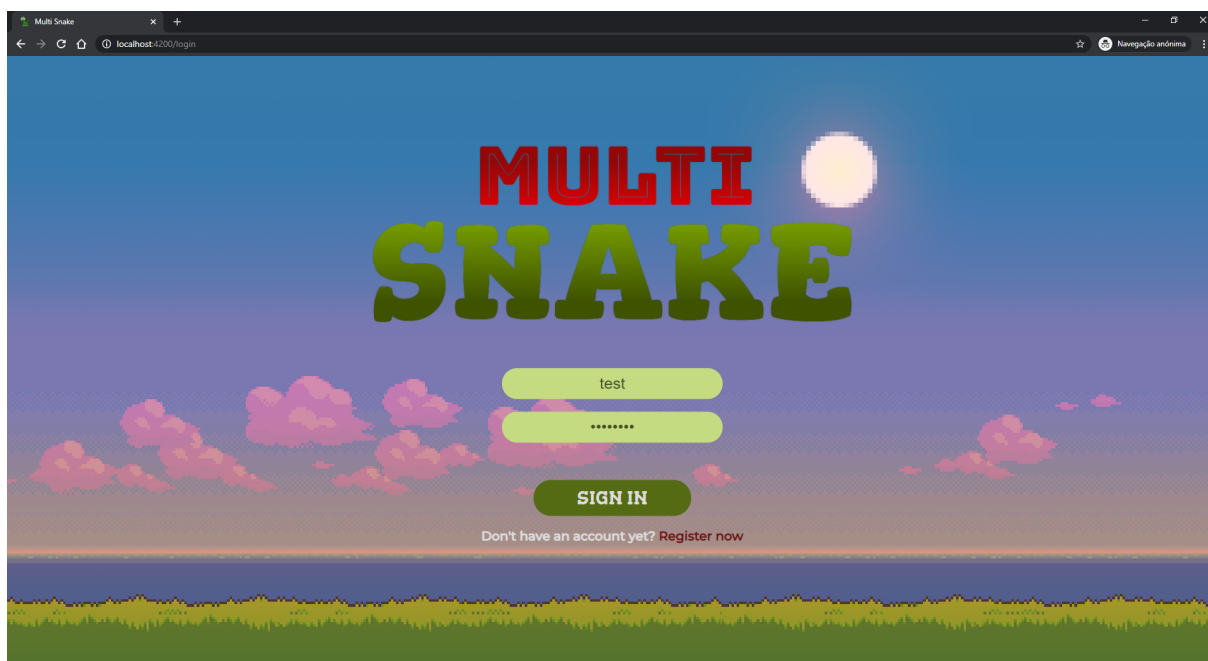


Figura 4.1: Componente de login de utilizadores

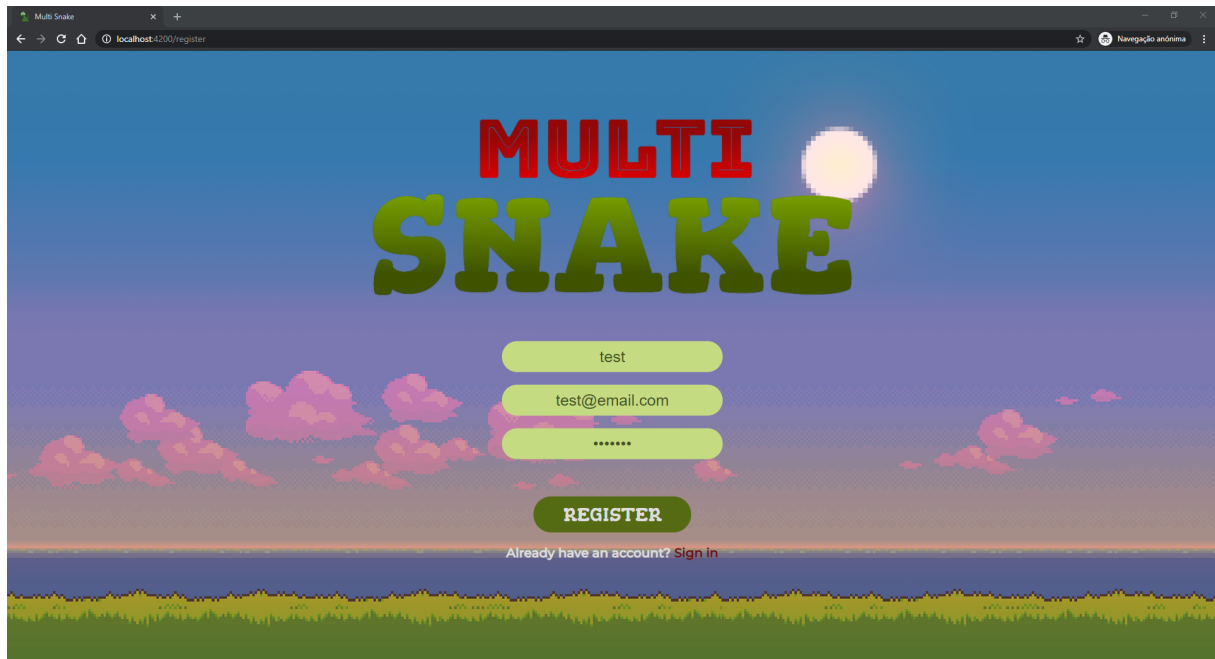


Figura 4.2: Componente de registo de utilizadores

Em desenvolvimento está também a página pessoal de cada utilizador, dentro da qual foi realizada uma versão singleplayer do jogo da cobra, para servir de apoio e modelo para o desenvolvimento do modo multiplayer. A figura 4.3 demonstra uma fase bastante primordial da página privada do utilizador com a versão do jogo para um jogador.

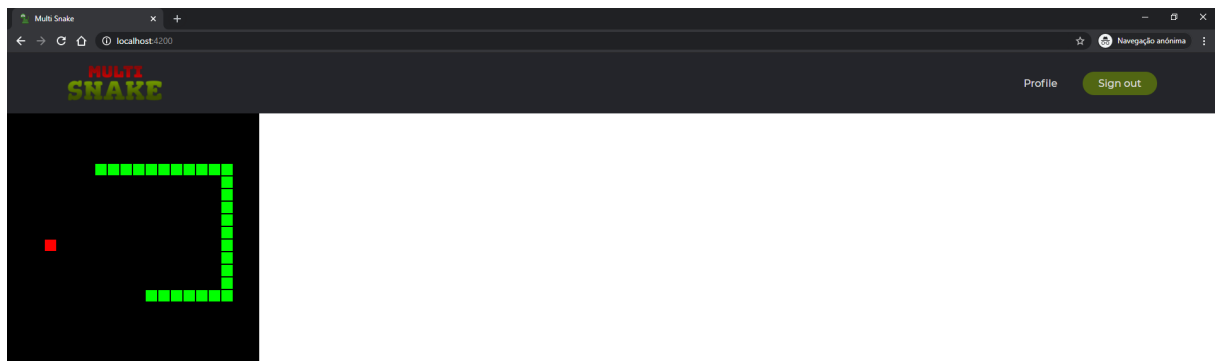


Figura 4.3: Página privada de um utilizador com a versão singleplayer do jogo da cobra

# Referências

- [1] [en.wikipedia.org](https://en.wikipedia.org)
- [2] [www.lucidchart.com](https://www.lucidchart.com)
- [3] [docs.gitlab.com/ee/user/project/description\\_templates.html](https://docs.gitlab.com/ee/user/project/description_templates.html)
- [4] [www.newthinktank.com/2019/01/latex-tutorial](https://www.newthinktank.com/2019/01/latex-tutorial)
- [5] [pt.overleaf.com](https://pt.overleaf.com)
- [6] [www.dickimaw-books.com/gallery/glossaries-styles](https://www.dickimaw-books.com/gallery/glossaries-styles)
- [7] [www.microsoft.com/pt-pt/sql-server/sql-server-2019](https://www.microsoft.com/pt-pt/sql-server/sql-server-2019)
- [8] [dotnet.microsoft.com/apps/aspnet/apis](https://dotnet.microsoft.com/apps/aspnet/apis)
- [9] [www.nuget.org](https://www.nuget.org)
- [10] [swagger.io](https://swagger.io)
- [11] [jasonwatmore.com](https://jasonwatmore.com)
- [12] [www.youtube.com/user/binarythistle](https://www.youtube.com/user/binarythistle)
- [13] [angular.io](https://angular.io)
- [14] [getbootstrap.com](https://getbootstrap.com)
- [15] [www.npmjs.com](https://www.npmjs.com)
- [16] [www.youtube.com/channel/UC80PWRj\\_ZU8Zu0HSMNVwKWw](https://www.youtube.com/channel/UC80PWRj_ZU8Zu0HSMNVwKWw)
- [17] [www.youtube.com/channel/UCSJbGtTlrDami-tDGPUV9-w](https://www.youtube.com/channel/UCSJbGtTlrDami-tDGPUV9-w)