Desenvolvimento Web 1

Atividade prática

Introdução ao Angular + Azure

Beatriz Nataly Silva (SP3070263)
Carlos H. Biagolini Junior (SP3076113)
Gabriel S. Parreira (SP3072819)
Davi Da Silva Pinheiro Neto (Sp3070786)
Fabíola Velocini Ramirez (Sp3069672)
Iurinicolas Henrique Garcia (Sp3074048)

São Paulo / 2022

Sumário

Visão geral

Proposta

Desenvolvimento de uma página web, que apresente um curriculum vitae (CV). A página deverá apresentar a opção de leitura do CV em dois idiomas, sendo eles Português e Inglês. A página web será desenvolvida em Angular. O compartilhamento da página será feito pela plataforma Azure, utilizando a extensão "Azure Static Web Apps" na IDE Visual Studio Code.

Git

https://github.com/biagolini/ifspDw1Projeto

Referências externas

Loiane Groner (2021) CRUD Angular + Spring. Disponível em: https://www.youtube.com/playlist?list=PLGxZ4Rq3BOBpwaVgAPxTxhdX_TfSVlTcY. Data de acesso: 24 de março de 2022.

Microsoft (2022) Quickstart: Building your first static site with Azure Static Web Apps. Disponível em: https://docs.microsoft.com/en-us/azure/static-web-apps/getting-started?tabs=vanilla-javascript. Data de acesso: 27 de março de 2022.

SO de desenvolvimento

Edition	Windows 11 Pro
Version	21H2
OS build	22000.556
Experience	Windows Feature Experience Pack 1000.22000.556.0

Linguagens

Verificado pelo seguinte comando no powershell

```
ng --version
```

Node.js: 16.14.2

Typescript : 4.5.5

Angular CLI: 13.1.2

IDE

Verificado pelo seguinte comando no powershell

```
code --version
```

Visual Code Studio: 1.65.2

Plugin

Azure Static Web Apps: 0.10.0

Base do projeto Angular

Passo 1: Selecione um diretório (de sua preferência) onde serão salvos os arquivos relacionados ao desenvolvimento do seu projeto. Abra o terminal nessa pasta. Em seguida crie o projeto base, com o comando:

```
ng new cv --routing
```

 # Selecione a opção:

? Which stylesheet format would you like to use? SCSS

Passo 2: Entre na pasta do projeto criado:

```
cd cv
```

Passo 3: Adicione o Angular Material no seu projeto

```
ng add @angular/material
```

 # Selecione as opções

Would you like to proceed? Yes

? Choose a prebuilt theme name, or "custom" for a custom theme: Custom

? Set up global Angular Material typography styles? Yes

? Set up browser animations for Angular Material? Yes

Passo 4: Instale o FlexLayout.

```
npm install @angular/flex-layout
```

Passo 5: Instalar o pacote ngx-translate. Para mais informações veja: https://www.npmjs.com/package/@ngx-translate/core.

```
npm install @ngx-translate/core --save
npm install @ngx-translate/http-loader --save @biesbjerg/ngx-translate-extract --saveDev
```

Texto a ser apresentado

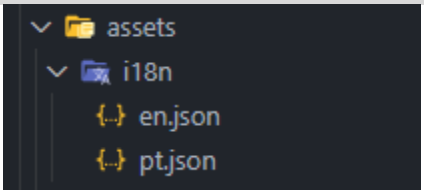
Passo 1. Importe os pacotes necessários no app.module (TranslateModule, TranslateLoader, TranslateHttpLoader, HttpClient, HttpClientModule, HTTP_INTERCEPTORS). Crie uma função que fará a tradução. O ngx-translate usa uma lógica de vocês escrever suas strings de textos a serem apresentado em arquivos json (você terá um arquivo para cada idioma), e uma função aqui descrita (HttpLoaderFactory) faz a ponte entre os jsons e os templates. Essa função criada vai ficar aqui dentro mesmo, e como ela está descrita no app.module, ficará disponível para todas as páginas da sua aplicação.

src\app\app.module.ts

```
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { BrowserModule } from '@angular/platform-browser';
import { FlexLayoutModule } from '@angular/flex-layout';
import { HttpClient, HttpClientModule } from '@angular/common/http';
import { NgModule } from '@angular/core';
import { TranslateHttpLoader } from '@ngx-translate/http-loader';
import { TranslateLoader, TranslateModule } from '@ngx-translate/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';


export function HttpLoaderFactory(http: HttpClient): TranslateLoader{
  return new TranslateHttpLoader(http, './assets/i18n/', '.json');
}

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    BrowserAnimationsModule,
    FlexLayoutModule,
    HttpClientModule,
    TranslateModule.forRoot({
      loader: {
        provide: TranslateLoader,
        useFactory: HttpLoaderFactory,
        deps: [HttpClient],
      },
      defaultLanguage: 'pt',
    }),
  ],

  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Passo 2. Dentro da pasta assets, crie uma pasta chamada i18n, e dentro dela crie 1 json para cada linguagem. Será dentro desse JSON que vamos inserir as strings a serem apresentada no nosso site. Os diferentes arquivos json devem ter a mesma estrutura (nome dos campos deve ser o mesmo), a única coisa que muda é o conteúdo (string) dos textos a serem apresentado em cada um dos idiomas.

```
mkdir src\assets\i18n
```



Passo 3. Desenvolva o conteúdo a ser apresentado na nossa página.

src\assets\i18n\en.json

```
{
  "header": {
    "nameHeader": "Lorem ipsum",
    "profile": "Candidate profile",
    "background": "Professional background ",
    "experience": "Professional experience",
    "ademic": "Academic Background",
    "courses": "Extracurricular courses ",
    "languanges": "Languanges"
  },
  "context": {
    "profile": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse suscipit, libero et bibendum aliquam, quam felis laoreet dolor, a malesuada sapien nisi nec lectus. Ut id erat tristique, porttitor lacus eu, gravida lacus. Vivamus tempor turpis eu leo luctus, vel pellentesque turpis ultrices. Curabitur ut dignissim arcu, eget sagittis ante. Vivamus mi dui, pulvinar dictum volutpat bibendum, tempor sed turpis. Fusce et nulla massa. Phasellus tincidunt malesuada risus id imperdiet. Aliquam pellentesque risus sit amet quam accumsan vehicula. Sed ullamcorper enim urna, non luctus mi sodales vitae. Vestibulum id vestibulum felis, id luctus est. Nulla congue eu purus at bibendum.",
    "background": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. In et libero sed mi ultrices aliquam viverra non tellus. Cras ultricies tellus non tristique luctus. Phasellus purus ante, ultrices ut augue nec, pharetra suscipit sapien. Donec maximus diam lacus, fringilla pulvinar dui finibus ac. Donec molestie, mauris eget tristique aliquet, metus nibh sagittis ante, eu dapibus tellus sapien vel lectus. Vivamus a mi magna. Mauris non nisl magna. Pellentesque interdum enim at tellus porttitor, id feugiat diam aliquam. Donec est lectus, faucibus ut porttitor et, dictum id tortor. Etiam aliquam massa in purus tempor ultrices. Praesent porttitor ante vitae placerat blandit.",
    "experience1": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
    "experience2": "Praesent mollis arcu ac velit tempor pretium.",
    "experience3": "Aliquam in urna eget nisl tincidunt iaculis.",
    "ademic1": "Sed eget ligula sed augue fermentum cursus.",
    "ademic2": "Nullam vulputate leo ut lorem suscipit eleifend.",
    "courses1": "Proin volutpat nunc tempus libero commodo, sed euismod nulla dignissim.",
    "courses2": "Vestibulum aliquam neque mollis lectus viverra aliquam.",
    "languanges1": "Nullam mattis felis dictum ligula pretium iaculis.",
    "languanges2": "Fusce tincidunt urna et dui tempor accumsan ac eu nisi."
  }
}
```

src\assets\i18n\pt.json

```
{
  "header": {
    "nameHeader": "Lorem ipsum",
    "profile": "Perfil do candidato",
    "background": "Histórico profissional ",
    "experience": "Experiência profissional",
    "ademic": "Formação Acadêmica",
    "courses": "Cursos extracurriculares",
    "languanges": "Linguagens"
  },
  "context": {
    "profile": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum rhoncus massa id mauris rutrum, eu semper ligula auctor. Quisque posuere convallis justo ac fermentum. Aliquam lacus urna, mattis ac eros nec, ultrices semper est. Fusce eget urna rhoncus, vehicula lectus eu, cursus arcu. Cras bibendum tincidunt pharetra. Donec malesuada sagittis massa et lobortis. In ac lectus a purus elementum scelerisque. Aliquam tincidunt imperdiet mauris, ut rutrum neque dictum at. Pellentesque tristique sed leo a elementum. Mauris ac leo tellus. Etiam eu purus quis libero aliquam tempor. Praesent ante massa, faucibus eu imperdiet id, pulvinar ut ipsum. Phasellus venenatis, risus nec.",
    "background": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer at felis sit amet tortor fringilla cursus a vel risus. Duis quis sagittis neque. In at lorem semper nisi vulputate efficitur. Fusce venenatis rutrum lacus, a convallis mi. Vivamus velit sem, sollicitudin et sem vitae, vulputate luctus magna. Maecenas maximus enim sed rutrum aliquet. In iaculis risus sem, vitae placerat nulla egestas sit amet. Aliquam eleifend, nulla id dignissim fermentum, ex neque lacinia ipsum, eget mattis lacus felis eget massa. Duis semper, nisl non malesuada semper, tellus tortor cursus orci, sed condimentum ex est facilisis sem. Praesent luctus placerat justo eget.",
    "experience1": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
    "experience2": "Praesent mollis arcu ac velit tempor pretium.",
    "experience3": "Aliquam in urna eget nisl tincidunt iaculis.",
    "ademic1": "Sed eget ligula sed augue fermentum cursus.",
    "ademic2": "Nullam vulputate leo ut lorem suscipit eleifend.",
    "courses1": "Proin volutpat nunc tempus libero commodo, sed euismod nulla dignissim.",
    "courses2": "Vestibulum aliquam neque mollis lectus viverra aliquam.",
    "languanges1": "Nullam mattis felis dictum ligula pretium iaculis.",
    "languanges2": "Fusce tincidunt urna et dui tempor accumsan ac eu nisi."
  }
}
```

Pagina web

Neste tópico vamos desenvolver o template da nossa página web.

Passo 1. Dentro da pasta assets, crie uma pasta chamada images, e deposite nessa pasta as imagens que serão carregadas no template da usa página. No nosso projeto, vamos depositar um desenho de uma bandeira do Brasil e outro desenho de uma bandeira do Reino Unido. Esse desenhos serão usados dentro de botões para a troca de idioma da página.

```
mkdir src\assets\images
```

Passo 2. Desenvolva o app.compnent.ts:

src\app\app.component.ts

```ts
import { Component } from '@angular/core';
import { TranslateService } from '@ngx-translate/core';


@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'cv';

  constructor(
    private translateService: TranslateService,
  ) {}

  changeLanguage(languange: string): void {
    this.translateService.use(languange);
  }

}
```

Passo 3. Desenvolva o app.compnent.html:

src\app\app.component.html

```html
<div fxLayout="column" fxLayoutAlign="start stretch">
  <div fxLayout="row" fxLayoutAlign="end start" class="bg-flags">
    <mat-icon (click)="changeLanguage('pt')" class="flags">
      <img src="/assets/images/br-flag.png" alt="Brazil flag" class="flags"
    /></mat-icon>

    <mat-icon (click)="changeLanguage('en')" class="flags">
      <img
        src="/assets/images/uk-flag.png"
        alt="United Kingdom flag"
        class="flags"
    /></mat-icon>
  </div>

  <div class="page">
    <div class="cv-page">
      <div class="cv-body" fxLayout="column" fxLayoutAlign=" stretch">
        <div
```

```
          fxLayoutGap="20px"
          class="cv-head full-width"
          fxLayout="column"
          fxLayoutAlign="center center"
        >
          <h1>{{ "header.nameHeader" | translate }}</h1>
        </div>

        <div
          class="cv-description"
          fxLayout="column"
          fxLayoutAlign="start stretch"
          fxLayoutGap="20px"
        >
          <h2>{{ "header.profile" | translate }}</h2>
          <p>{{ "context.profile" | translate }}</p>

          <h2>{{ "header.background" | translate }}</h2>
          <p>{{ "context.background" | translate }}</p>

          <h2>{{ "header.experience" | translate }}</h2>
          <ul>
            <li>{{ "context.experience1" | translate }}</li>
            <li>{{ "context.experience2" | translate }}</li>
            <li>{{ "context.experience3" | translate }}</li>
          </ul>

          <h2>{{ "header.ademic" | translate }}</h2>
          <ul>
            <li>{{ "context.ademic1" | translate }}</li>
            <li>{{ "context.ademic2" | translate }}</li>
          </ul>

          <h2>{{ "header.courses" | translate }}</h2>
          <ul>
            <li>{{ "context.courses1" | translate }}</li>
            <li>{{ "context.courses2" | translate }}</li>
          </ul>

          <h2>{{ "header.ademic" | translate }}</h2>
          <ul>
            <li>{{ "context.ademic1" | translate }}</li>
            <li>{{ "context.ademic2" | translate }}</li>
          </ul>

          <h2>{{ "header.languanges" | translate }}</h2>
          <ul>
            <li>{{ "context.languanges1" | translate }}</li>
            <li>{{ "context.languanges2" | translate }}</li>
          </ul>
        </div>
      </div>
    </div>
  </div>
</div>
```

Passo 4. Desenvolva o app.compnent.html:

src\app\app.component.css

```css
h1 {
  font-family: Arial;
  font-weight: bold;
  font-size: 3em;
  color: #425264;
  letter-spacing: 2px;
  margin: 1em 0;
}

h2 {
  text-align: center;
}

p {
  text-align: justify;
  text-indent: 3rem;
}

.bg-flags {
  background-color: rgb(50, 50, 50);
}

.flags {
  width: 25px;
  height: auto;
}

.page {
  width: 100%;
  background-color: #425264;
  display: flex;
  justify-content: center;
}

.cv-page {
  max-width: 750px;
}

.cv-head {
  background: #788fa8;
}

.cv-body {
  background: #ffffff;
}

.cv-description {
  padding: 0 20px;
}
```

Controle de versionamento

Afim de criar um backup do nosso projeto, e facilitar o desenvolvimento de atualizações futuras, recomendamos o controle de versionamento por meio do Git.

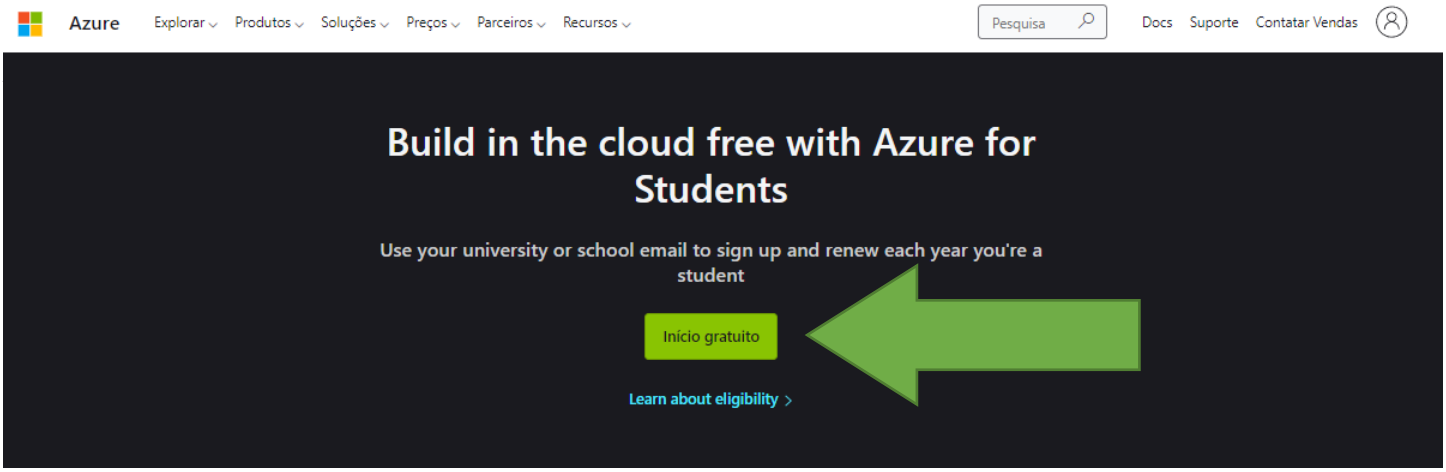
Passo 1: Acesse o link https://github.com/new e crie o projeto: ifspDw1Projeto

Passo 2: Envie para o GitHub uma cópia do seu projeto.

```
git init
git config --local user.name "SEU NOME"
git config --local user.email "SEU_EMAIL@email.com"
git remote add origin https://github.com/USUARIO_GITHUB/ifspDw1Projeto.git
git branch -M main
git add .
git commit -m "Dummy CV"
git push -u origin main
```

Publicação da página web

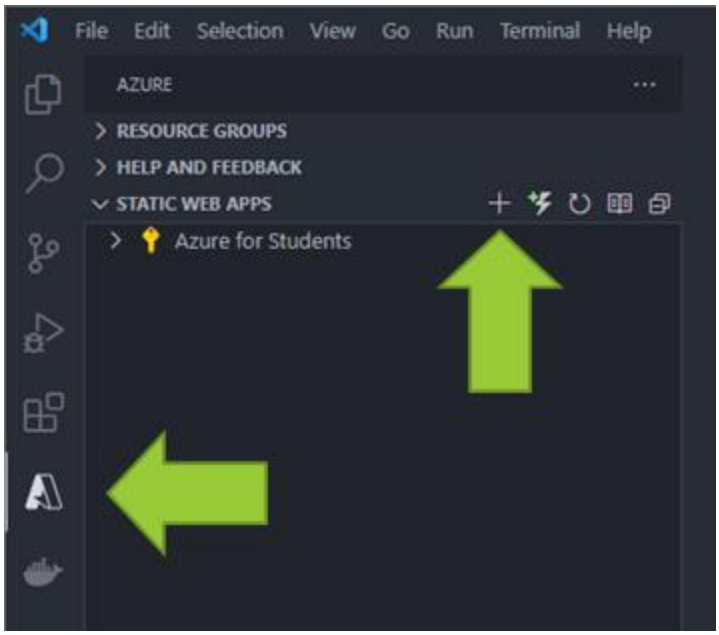
Passo 1: Acesse o link https://azure.microsoft.com/pt-br/free/students/, crie seu cadastro e inicie o teste gratuito.



Passo 2: Compile o projeto a ser publicado. Para mais informações sobre compilação, veja: https://angular.io/cli/build.

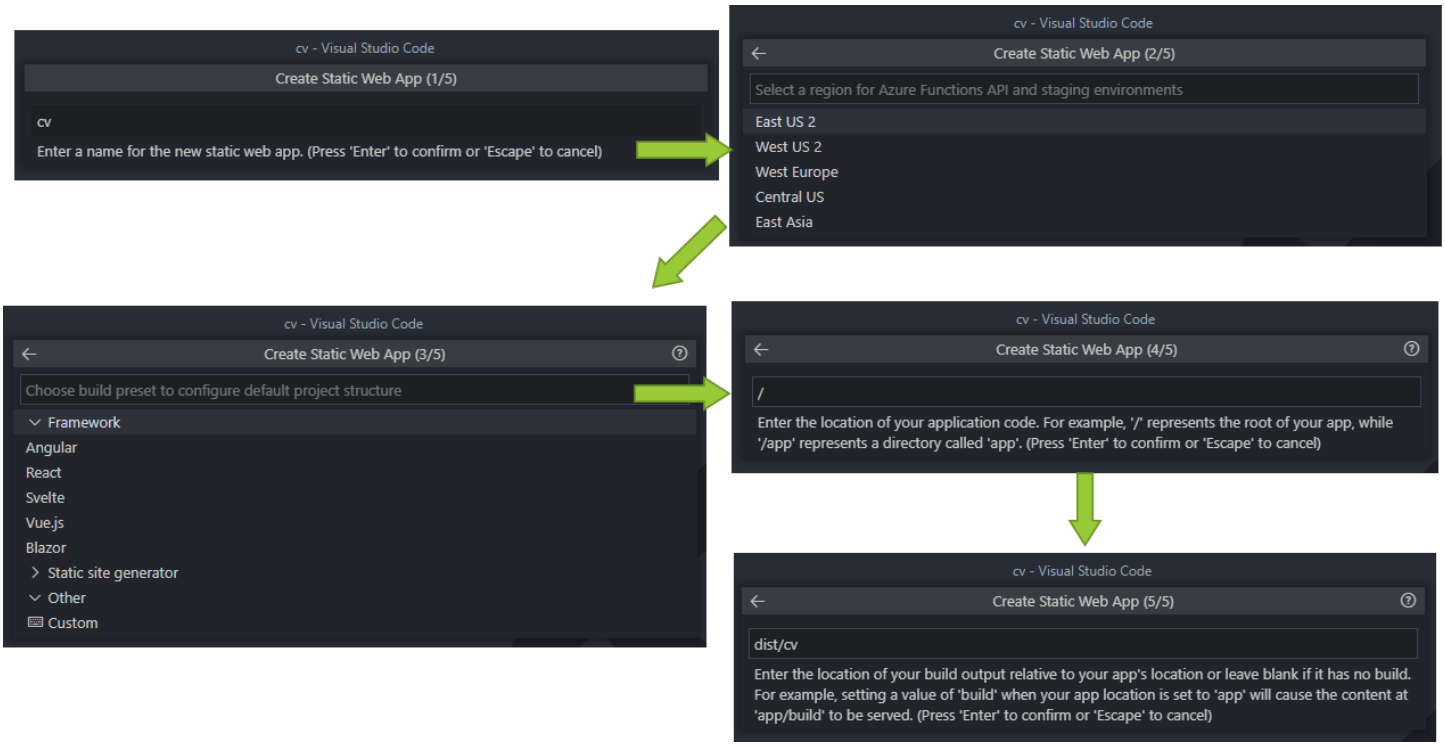
```
ng build
```

Passo 3: Localize e acesse o plugin "Azure Visual Studio Code" (disponível em https://marketplace.visualstudio.com/items?itemName=ms-azuretools.vscode-azurestaticwebapps). Em seguida clique no ícone de +.

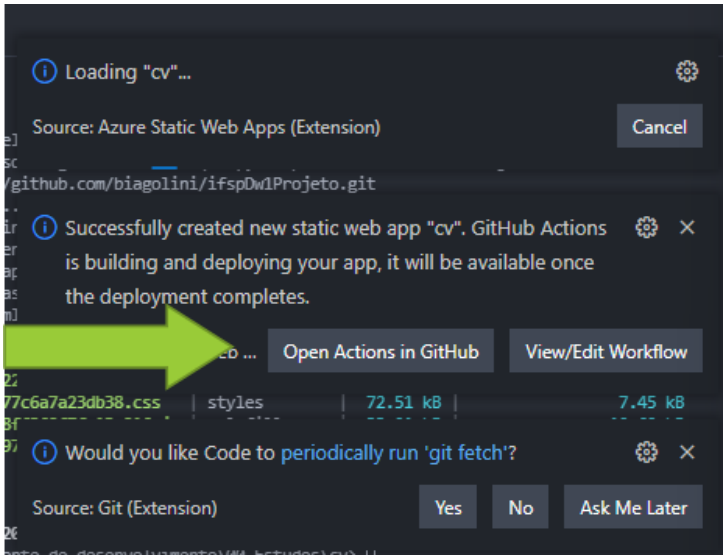


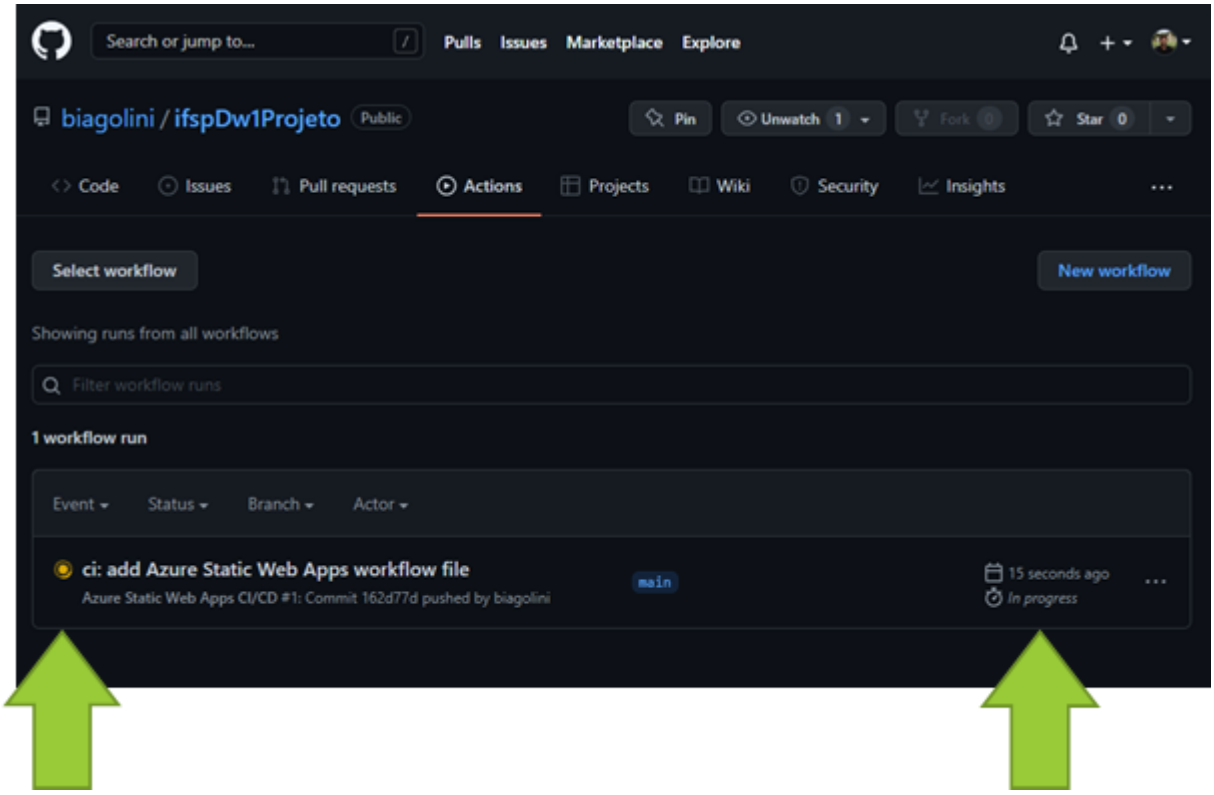
Passo 4: Siga os passos para configuração:

1. Nome da página: cv
2. Selecione a região para postar sua página: Selecione geograficamente mais próximo ao seu público alvo. Por exemplo, se você estivesse postando um CV para empresas na Europa, escolheria o servidor na Europa.
3. Informe o framework usado para construção da sua página web: Angular
4. Informe a localização da sua aplicação: Se você abriu o visual code diretamente na raiz do seu projeto, basta colocar "/".
5. Informe a direção para a pasta onde estão os arquivos a serem postados (nesse tutorial, geramos esse arqiovos com o comando ng build no passo 2): por default, o ng build salvará os documento em "/dist/{nome do projeto}", portanto, usaremos aqui "dist/cv".

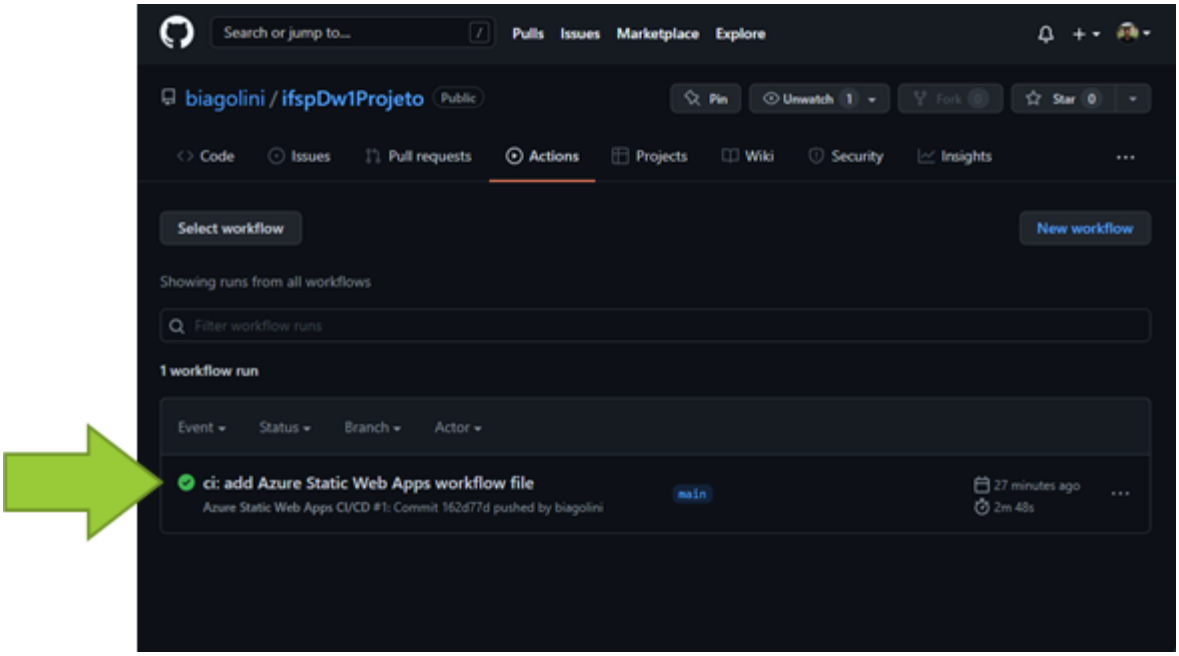


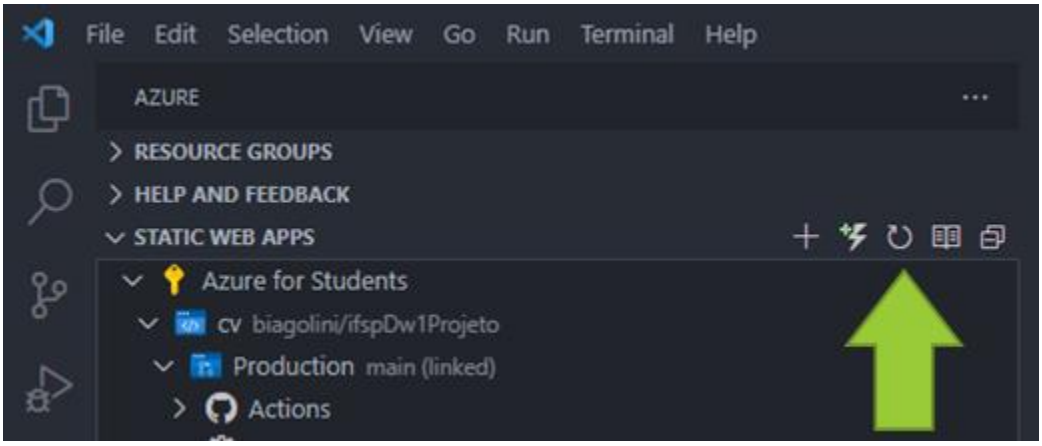
Passo 5: Agora é só aguardar. Se quiser ver o status da construção do seu site, localize a mensagem de confirmação de envio do projeto para deploy, e clique em “Open Actions in GitHub”.





Passo 6: Quando o status do projeto indicar que o mesmo foi finalizado (ícone muda de um check box verde, e o texto “in progress” muda para o tempo desde que a aplicação está de pé), atualize o status da aplicação no plugin “Azure Visual Studio Code”, clicando no botão de reload.





Passo 7: Para acessar sua página, clique com o botão direito sobre “Production” em seguida clique em “Browse site”, seu navegador será aberto e levará você para a página da sua aplicação.