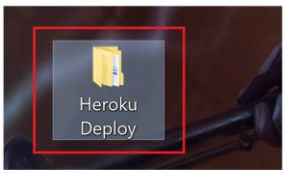
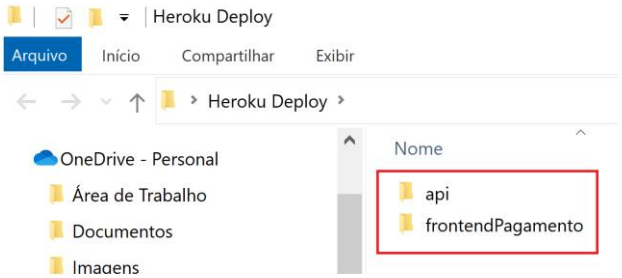
Processo de deploy

Primeiro de tudo eu crio uma pasta para onde eu vou COPIAR tanto o backend e o frontend da aplicação *(porque esses arquivos vão ser modificados)*:





Build do projeto Angular:

Os endpoints do front end da aplicação agora serão consumidos diretamente pela **própria url do heroku**, então, é necessário efetuar essa mudança:

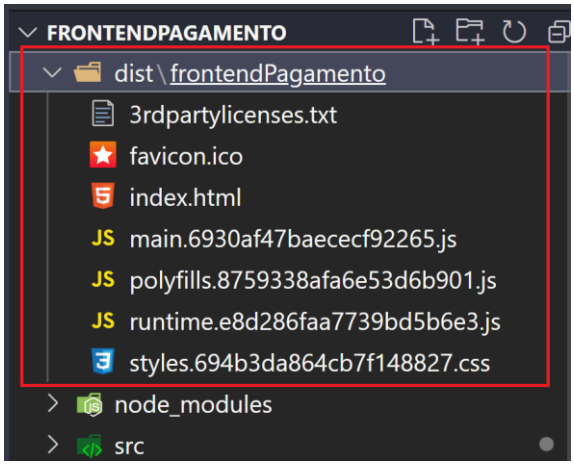
```
export class FuncionariosService {

  readonly API="https://nome-da-aplicacao.herokuapp.com/api/empresas";

  constructor(private http:HttpClient) { }

  getFuncionarios() : Observable<IFuncionario[]> {
    return this.http.get<IFuncionario[]>(this.API + "/funcionarios");
  }
```

```
export class HoleritesService {


  readonly API="https://nome-da-aplicacao.herokuapp.com/api/holerites";

  constructor(private http:HttpClient) { }

  getHoleriteById(idHolerite: number) : Observable<IHolerite> {
    return this.http.get<IHolerite>(`${this.API}/${idHolerite}`);
  }
```

O próximo passo é fazer build do projeto Angular com o comando **“ng build --prod"**

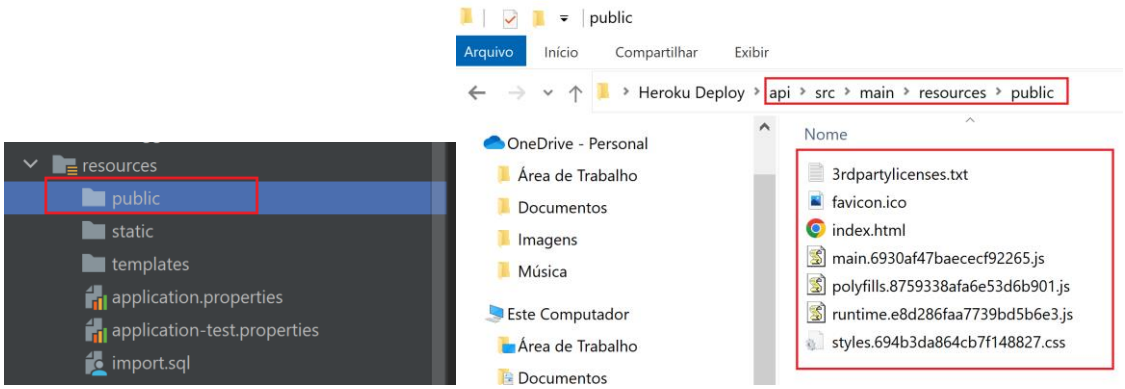
```
Microsoft Windows [versão 10.0.19044.1645]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\lucas\Desktop\Heroku Deploy\frontendPagamento>ng build --prod
```

A ação do build do projeto angular cria uma pasta chamada **“dist”,** essa pasta contém o **index.html** da aplicação e todos os **componentes já transformados em javascript.** Porém esse index não roda localmene, ele precisa de um node server:

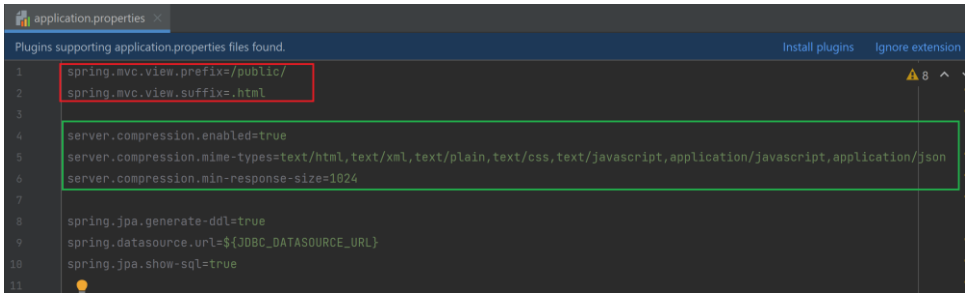


Linkando o Front End buildado com o Backend e tendo finalmente o view

Agora temos a camada de view do nosso MVC *(os arquivos gerados dentro de "dist" no build do projeto Angular),* precisamos **adiciona-lo ao projeto java**. Então na pasta de resources vamos criar uma pasta "public" e copiar todos os arquivos da "dist" :



Agora é preciso mudar as configurações no application properties para indicar **aonde está o view do projeto**, algumas configurações de **compressão para performance**:



Outra mudança necessária é a dependência do banco de dados, o Heroku só fornece gratuidade se o banco de dados for postgres:

Configurando o Spring

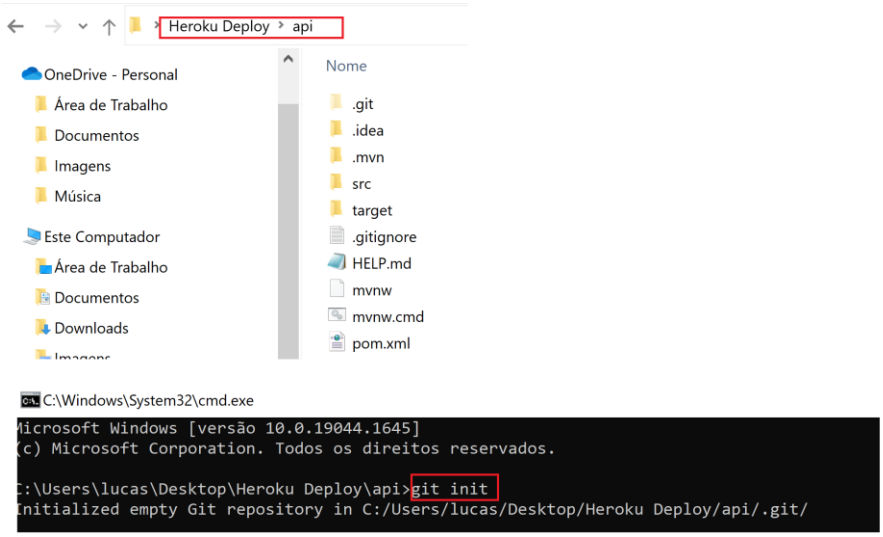
Colocamos o frontend na pasta public, agora é necessário adicionar uma configuração para que o Heroku assuma as rotas do Angular e para que o Spring fale pro Heroku qual o nosso front end:

```java
@Configuration
public class MyAppWebMvcConfigurer implements WebMvcConfigurer {

    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {
        registry.addResourceHandler( ...pathPatterns: "/**/*") ResourceHandlerRegistration
                .addResourceLocations("classpath:/public/")
                .resourceChain( cacheResources: true) ResourceChainRegistration
                .addResolver(new PathResourceResolver(){
                    @Override
                    protected Resource getResource(String resourcePath, Resource location) throws IOException{
                        Resource requestedResource = location.createRelative(resourcePath);
                        return requestedResource.exists() && requestedResource.isReadable() ?
                                requestedResource : new ClassPathResource("/public/index.html");
                    }
                });
    }

}
```

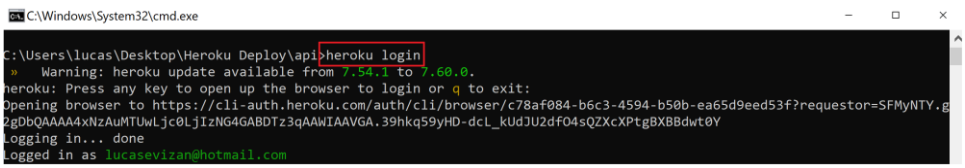
Então essa configuração basicamente indica aonde está o front end da aplicação.

Deploy no Heroku

Agora que o backend possuí o view da aplicação já dá pra subir no heroku. Essa pasta que tem o backend vai virar um repositório, só dar um “git init” nela:



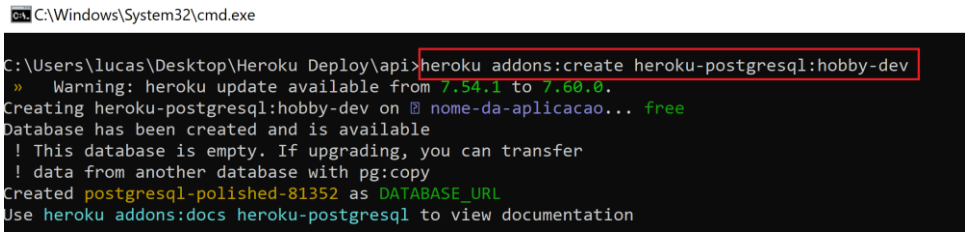
Depois disso a gente faz login no heroku com “heroku login” *(precisa ter instalado na máquina):*



Depois disso temos que criar o projeto, com o nome que a gente definiu nos endpoints do angular lá trás:

```
C:\Users\lucas\Desktop\Heroku Deploy\api>heroku create nome-da-aplicacao
 »   Warning: heroku update available from 7.54.1 to 7.60.0.
Creating ⬀ nome-da-aplicacao... done
https://nome-da-aplicacao.herokuapp.com/ | https://git.heroku.com/nome-da-aplicacao.git
```

Agora tem que adicionar o postgres na máquina virtual do heroku:



Agora só dar um git push pra branch do Heroku que se tiver tudo certo a aplicação vai subir e ficar disponível: