

Deploy do Backend no Heroku

O que é Deploy?

O verbo **deploy**, em inglês, significa **implantar**.

Em programação, seu sentido está intimamente relacionado à sua tradução literal: fazer um deploy, em termos práticos, significa colocar no ar alguma aplicação que teve seu desenvolvimento concluído.

Quando um site é finalizado por um desenvolvedor e, após seus testes, é finalmente hospedado na nuvem e colocado no ar, ele passa pelo processo de deploy.

De mesmo modo, quando um sistema sofre alguma melhoria ou alteração em seu código-fonte, implementar essa alteração ao sistema que está no ar também é um tipo de deploy.

O que veremos por aqui?

Esse documento é um passo a passo para você enviar a sua aplicação SPRING, gratuitamente para a nuvem (Deploy). Este processo irá gerar um link de acesso a sua aplicação, que poderá ser acessado de qualquer lugar, a partir de qualquer dispositivo com acesso a Internet.

Para efetuar o Deploy vamos precisar fazer algumas modificações em nosso projeto, que serão detalhadas nas próximas páginas.

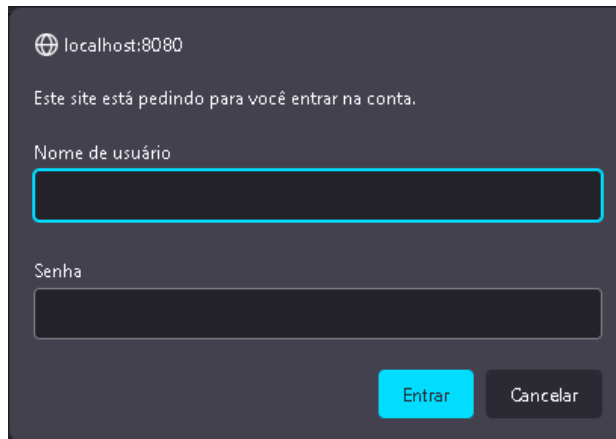


Passo 01 - Criar a Documentação da API

Para criar a Documentação da API no Swagger, utilize o **Guia de Configuração do SPringdoc**.

Passo 02 - Testar a API no seu computador

1. Execute a sua aplicação localmente pelo STS
2. Abra o endereço: <http://localhost:8080/> no seu navegador
3. A sua aplicação deverá exibir a tela de **Login (Usuário e Senha)**. Utilize o teste o **usuário: root** e a **Senha: root**, que foram criados em memória na **Classe BasicSecurityConfig**, na Camada Security.



4. Caso a aplicação **não** solicite o **Usuário** e a **Senha**, feche todas as janelas abertas do seu Navegador da Internet, abra novamente e acesse o endereço acima. Se o problema persistir, verifique a sua configuração do Swagger.
5. Verifique se após o login, o **Swagger** está inicializando automaticamente.
6. Caso você não tenha testado no **Insomnia**, execute os testes e verifique se tudo está funcionando.
7. Em especial, verifique se o Método **logar** está devolvendo o **Token**.
8. Antes de continuar a configuração do projeto para efetuar o Deploy, não esqueça de **parar a execução do Projeto no STS**.



IMPORTANTE: Não altere a senha do usuário root. Os instrutores da sua turma utilizarão este usuário para abrir, testar e corrigir a sua aplicação.

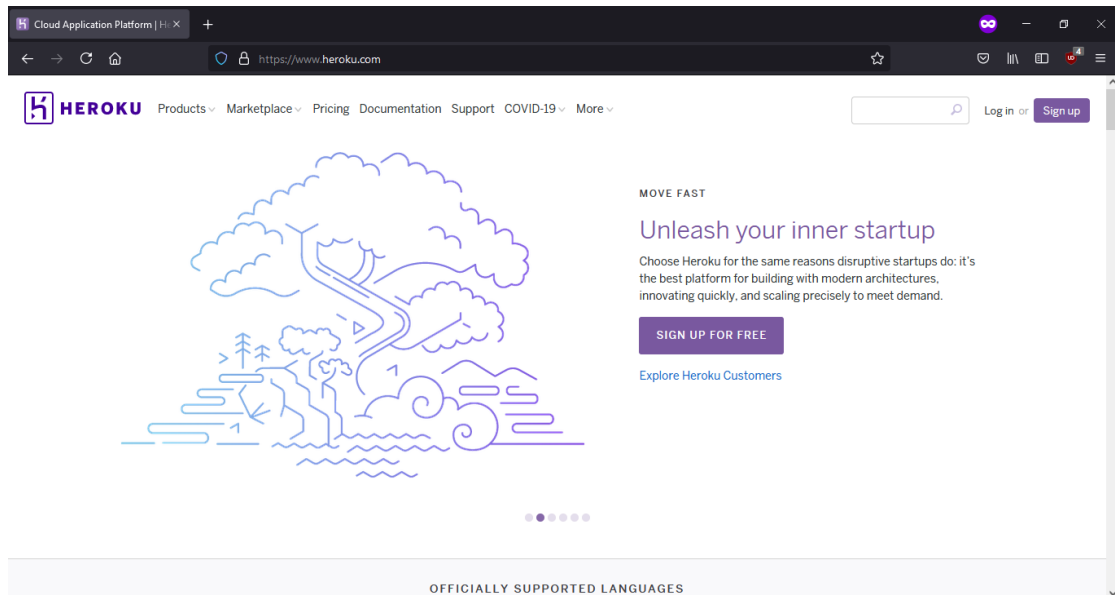


ATENÇÃO: Lembre-se que antes de fazer o Deploy é fundamental que a API esteja rodando e sem erros. Não faça os testes via Swagger porquê o usuário root (em memória, não utiliza todos os recursos da Spring Security, em especial o Token).




Passo 03 - Criar uma conta grátis no Heroku

1. Acesse o endereço: <https://www.heroku.com>




2. Crie a sua conta grátis no Heroku clicando no botão **SIGN UP FOR FREE**.

3. Preencha os dados do formulário e clique no botão **CREATE FREE ACCOUNT**.




Free account

Create apps, connect databases and add-on services, and collaborate on your apps, for free.



Your app platform

A platform for apps, with app management & instant scaling, for development and production.



Deploy now

Go from code to running app in minutes. Deploy, scale, and deliver your app to the world.

First name *

Last name *

Email address *

Company name

Role *

Role


Country *

Brazil

Primary development language *

Select a language

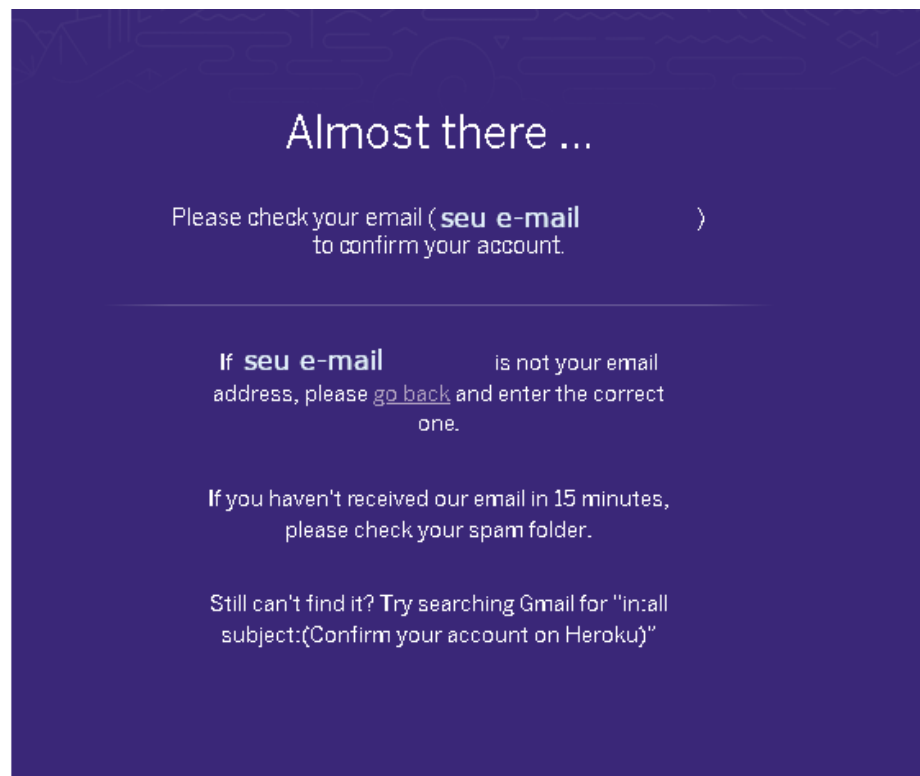
☐ I'm not a robot


reCAPTCHA
Privacy - Terms

CREATE FREE ACCOUNT

Signing up signifies that you have read and agree to the [Terms of Service](#) and our [Privacy Policy](#).
[Cookie Preferences](#).

4. Será exibida a mensagem abaixo informando que você receberá uma mensagem no seu e-mail para ativar a sua conta no Heroku. Acesse o seu e-mail e ative a sua conta.



5. O e-mail que você receberá será semelhante a imagem abaixo. Clique no link indicado em vermelho para ativar a sua nova conta



Thanks for signing up with Heroku! You must follow this link within 30 days of registration to activate your account:

<https://id.heroku.com/account/accept/10514805/556bc551c22058e09cc4986c24caaaf6>

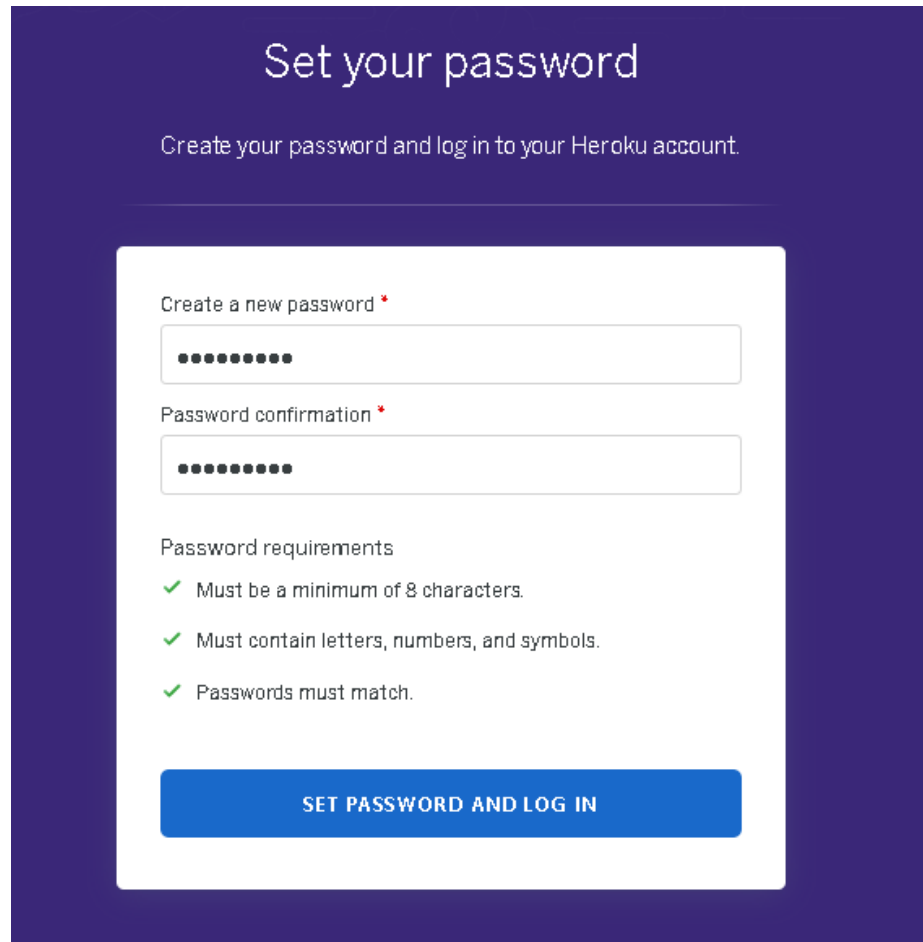
Have fun, and don't hesitate to contact us with your feedback.

The Heroku Team
<https://heroku.com>

Heroku is the cloud platform for rapid deployment and scaling of web applications. Get up and running in minutes, then deploy instantly via Git.

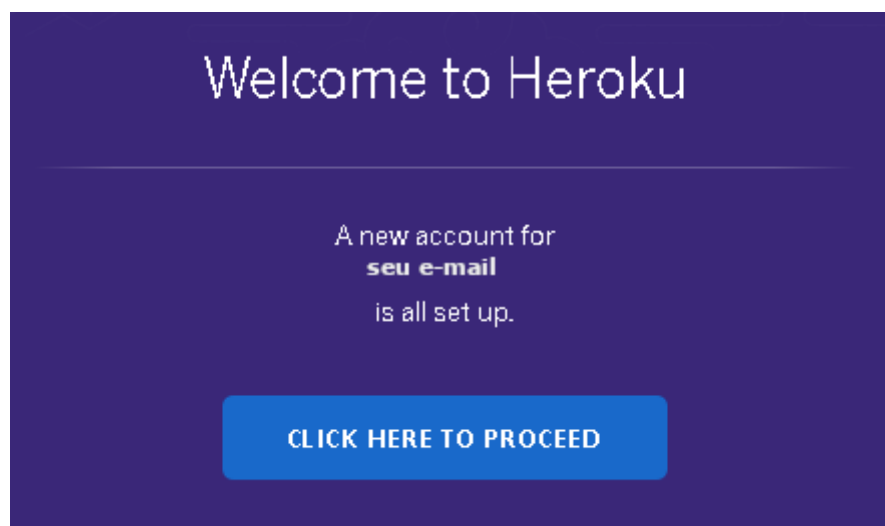
To learn more about Heroku and all its features, check out the Dev Center:
<https://devcenter.heroku.com/articles/quickstart>

6. Será aberta a janela abaixo para criar a senha da sua conta. Crie uma senha e clique no botão **SET PASSWORD AND LOGIN**.



ATENÇÃO: A senha deve ter no mínimo 8 caracteres e pelo menos 1 letra maiúscula, 1 caracter especial e 1 numero.

7. Será exibida a tela de Boas Vindas. Clique no botão **CLICK HERE TO PROCEED**.



8. Na próxima tela, concorde com os termos de uso da plataforma clicando no botão **Accept**.

Terms of Service

All Customers

Effective October 1, 2020, you agree that your use of the Heroku Services is governed by the [Salesforce Master Subscription Agreement](#), unless (except for free customers of the Heroku Services) you have a written master subscription agreement executed by salesforce.com for such Heroku Services as referenced in the Documentation, in which case such written salesforce.com master subscription agreement will govern. You further agree that your use or purchase of Heroku Add-ons, Buildpacks or Buttons that are not Heroku Services ("Heroku Elements") are governed by the [Heroku Elements Terms of Use \(Default\)](#), unless the Heroku Elements Marketplace provider has furnished to Heroku a separate terms of use, in which case such provider's terms of use govern the use or purchase of the applicable Heroku Elements. [Additional Terms](#) apply to credit card customers of the Heroku Services.

Heroku Marketplace Providers

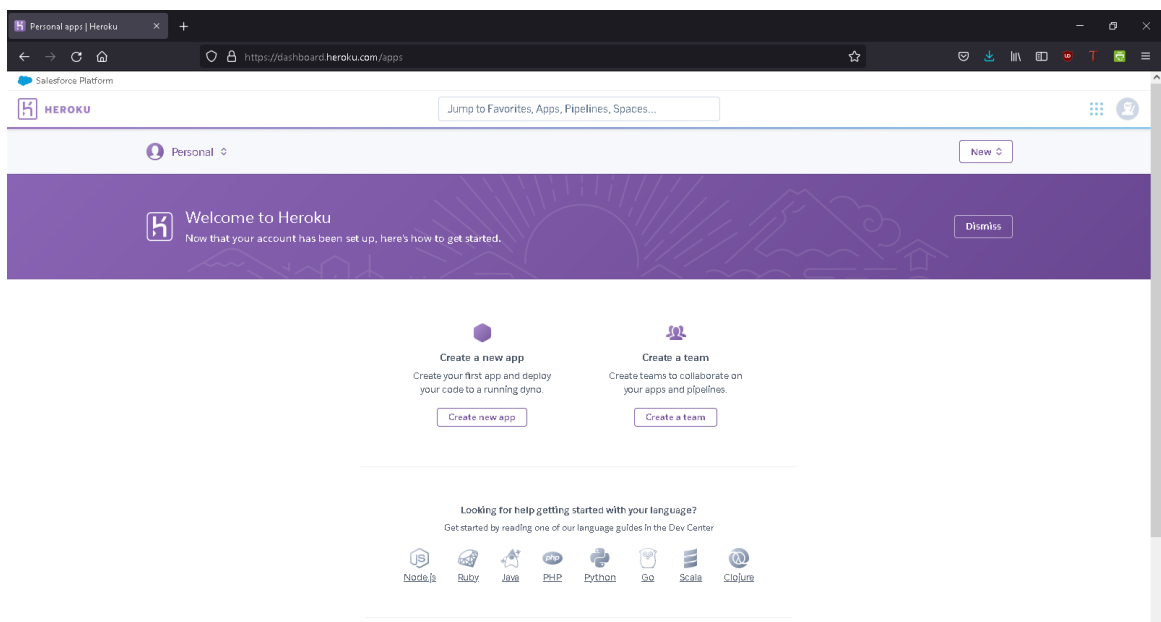
If you are or become a Heroku Elements Marketplace Provider, you further agree that your participation in the Heroku Elements Marketplace is governed by the [Salesforce License and Distribution Agreement for the Heroku Elements Marketplace](#).

Italian Customers

Are you domiciled in Italy?
☐ No.

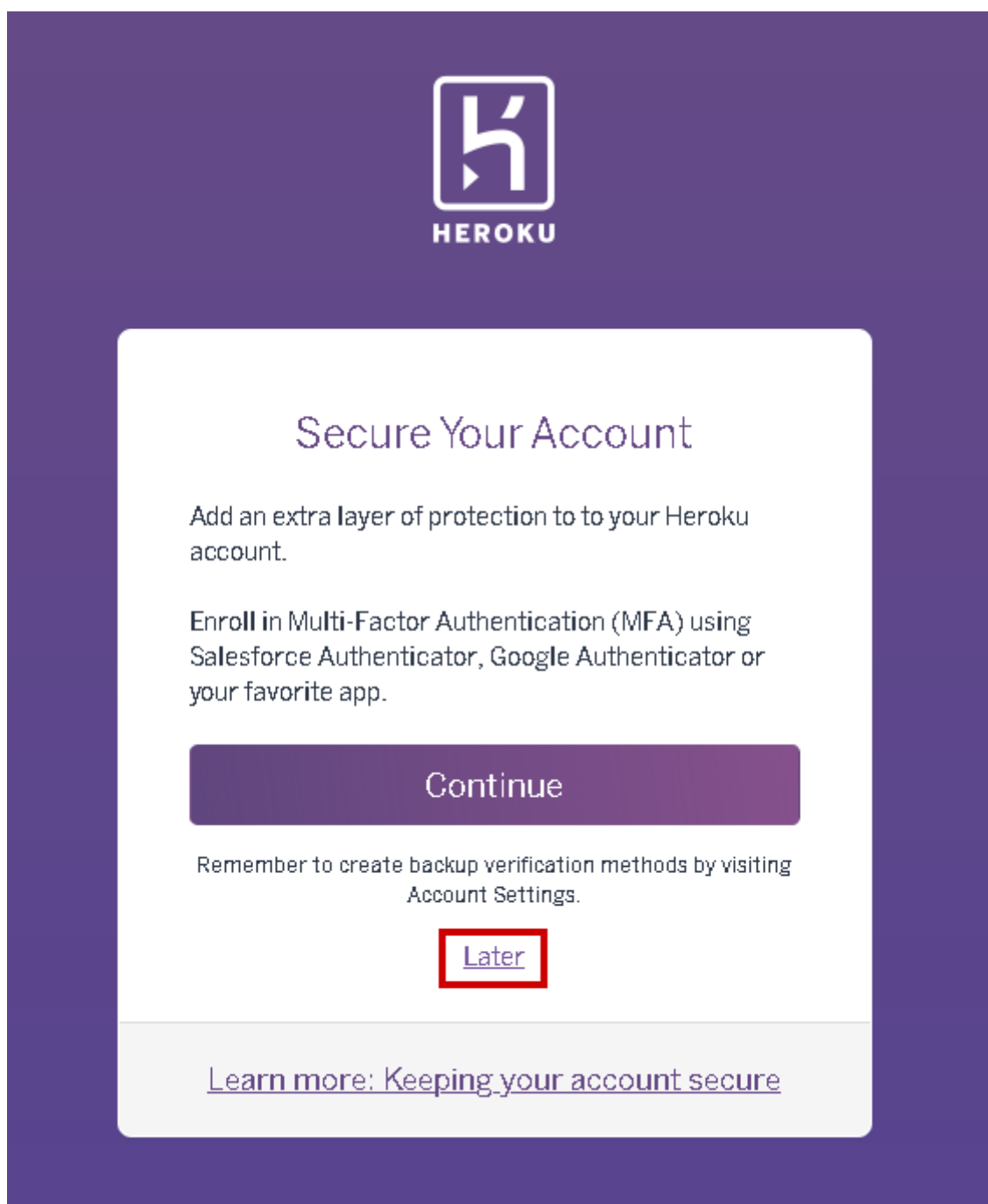
Accept

9. Você será redirecionado para o **Dashboard do Heroku**. Agora você está pronto para criar as suas aplicações na Nuvem do Heroku.



ATENÇÃO: Conclua todas etapas do processo de criação da conta no Heroku antes de avançar para o próximo passo do Deploy.

10. Caso o Heroku exiba a mensagem abaixo, solicitando a ativação do **MFA (Multi-Factor Authentication)**, não habilite esta opção. Clique no link **Later**, como mostra a figura abaixo, no item marcado em vermelho.

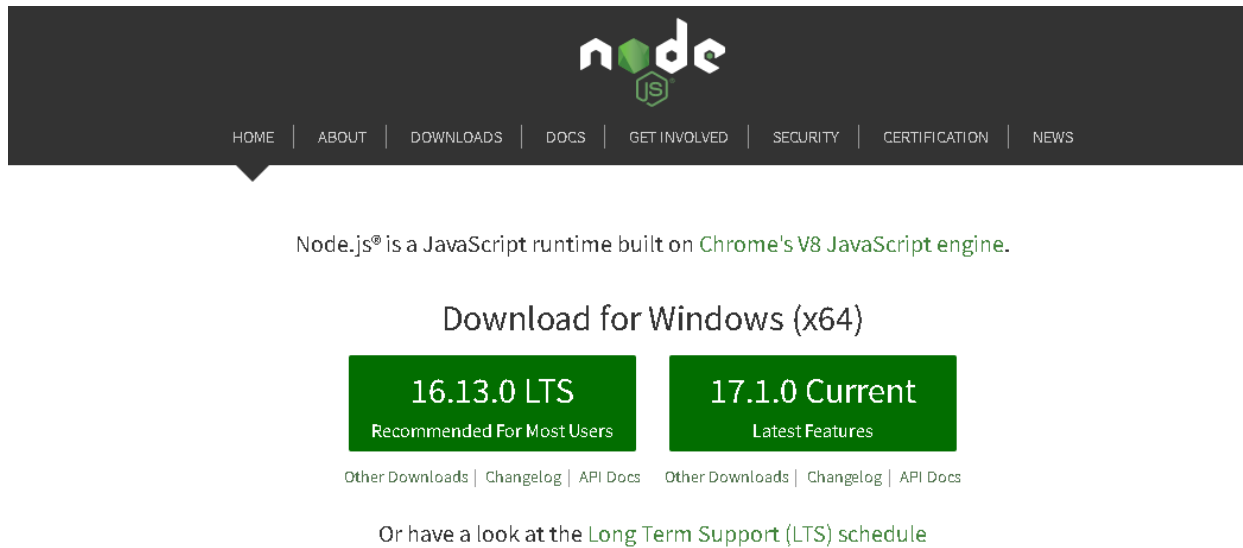


ATENÇÃO: Não habilite em sua conta no Heroku a opção MFA (Multi-Factor Authentication), ou seja, o login em 2 etapas. Em alguns servidores não é possível efetuar login via Heroku Client com o MFA habilitado.



Passo 04 - Instalação do Node.js

1. Acesse o endereço: <https://nodejs.org/en/>



2. Faça o download da última versão LTS disponível do Node.js e instale no seu computador.



ATENÇÃO: No momento em que este e-book foi escrito, a versão LTS mais atual do Node.js era a versão 16.13.0. Hoje pode ser que a versão mais atual seja outra*

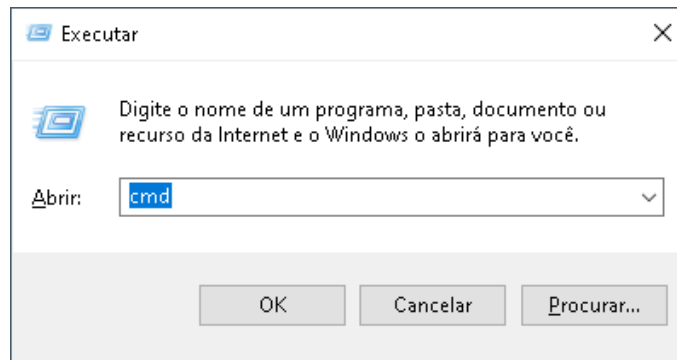


DICA: *Caso você tenha alguma dúvida quanto a instalação do Node.js, consulte o Guia de Instalação do Node.

Passo 05 - Instalação do Heroku Client

Para instalar e executar os comandos do Heroku Client utilizaremos o **Prompt de comando do Windows (cmd)**.

1. Para instalar, execute o atalho  +  para abrir a janela **Executar**.



2. Digite o comando **cmd** para abrir o **Prompt de comando do Windows**
3. Antes de instalar o **Heroku Client**, verifique se o Node já está instalado através do comando:

```
npm -version
```

```
C:\Users\rafae\Documents\workspace-spring-tool-suite-4-4.11.0.RELEASE>npm -version
7.19.1
```

**** A versão pode ser diferente da imagem***

4. Caso o Node não esteja instalado, volte ao passo **Instalação do Node.js**.
5. Para instalar o **Heroku Client** digite o comando:

```
npm i -g heroku
```

```
C:\Users\rafae\Documents\workspace-spring-tool-suite-4-4.11.0.RELEASE\blogpessoal>npm i -g heroku
npm WARN deprecated strip-eof@2.0.0: Renamed to `strip-final-newline` to better represent its functionality.
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js
environment. It is recommended you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated uuid@3.2.1: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances,
which is known to be problematic. See https://v8.dev/blog/math-random for details.
npm WARN deprecated uuid@3.3.2: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances,
which is known to be problematic. See https://v8.dev/blog/math-random for details.

added 13 packages, changed 620 packages, and audited 691 packages in 2m

22 packages are looking for funding
  run 'npm fund' for details

7 vulnerabilities (2 moderate, 5 high)

To address all issues, run:
  npm audit fix

Run 'npm audit' for details.
npm notice
npm notice New minor version of npm available! 7.17.0 -> 7.19.1
npm notice Changelog: https://github.com/npm/cli/releases/tag/v7.19.1
npm notice Run npm install -g npm@7.19.1 to update!
npm notice
```

6. Confirme a instalação do Heroku Client através do comando:

```
heroku version
```

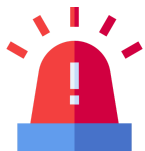
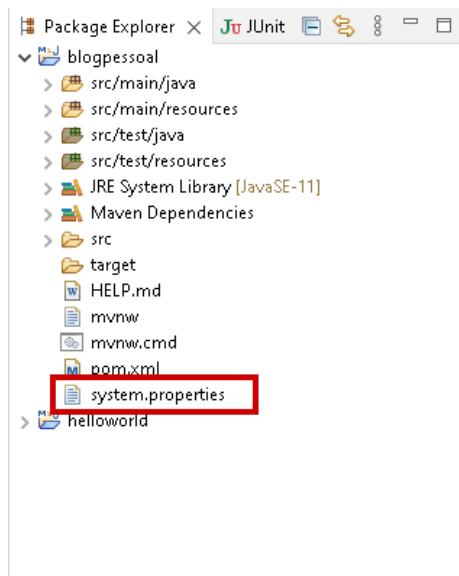
```
heroku/7.56.0 win32-x64 node-v14.17.3
```

**A versão pode ser diferente da imagem*

Passo 06 - Criar o arquivo system.properties

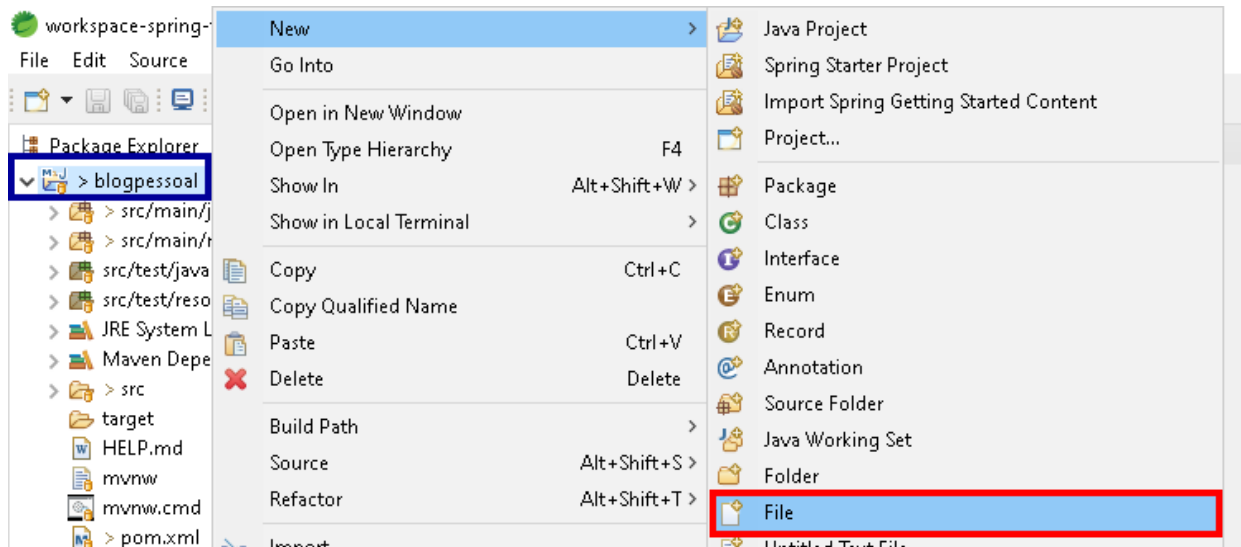
O arquivo **system.properties**, tem o objetivo de informar ao Heroku qual a versão a do Java ele deve utilizar para gerar o seu projeto e criar o arquivo executável (.jar).

1. Na **raiz do seu projeto**, na pasta **blogpessoal** (como mostra a figura abaixo), crie o arquivo **system.properties**.

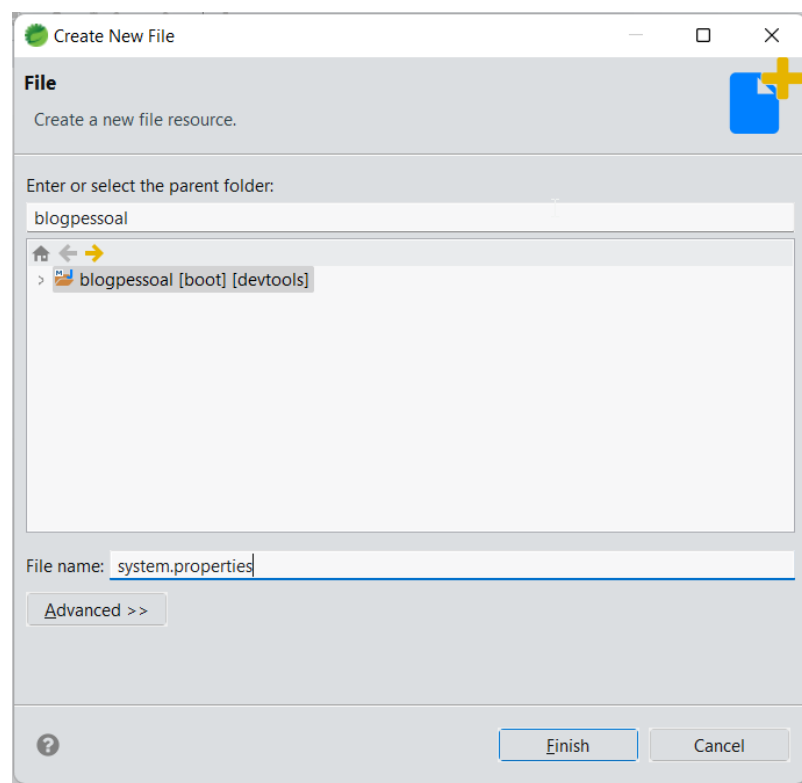


ALERTA DE BSM: *Mantenha a atenção aos detalhes ao criar o arquivo system.properties. Um erro muito comum é não criar o arquivo na pasta raiz do projeto. Outro erro comum é digitar o nome do arquivo errado.*

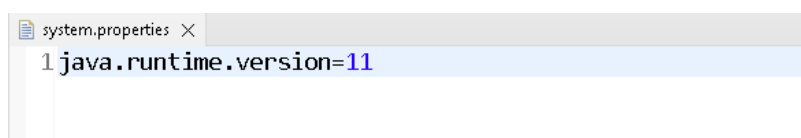
2. Na Guia **Package explorer**, clique com o botão direito do mouse sobre a pasta do projeto (indicada em azul) e clique na opção **New → File**.



3. Em **File name**, digite: **system.properties** e clique no botão **Finish**.



4. No arquivo **system.properties** indique a versão do Java que será utilizada pelo Heroku através da linha abaixo:



```
java.runtime.version=11
```



ATENÇÃO: A versão do Java informada no arquivo `system.properties` deve ser a mesma informada no arquivo `pom.xml`.



DICA: Durante o Deploy, caso o Heroku não reconheça a versão correta do Java (Exemplo: informei a versão 11 e o Heroku reconheceu a versão 1.8), apague o arquivo `system.properties`, recrie o arquivo na raiz do projeto e tente fazer o Deploy novamente.

Passo 07 - Adicionar a Dependência do PostgreSQL no pom.xml

O Heroku, na sua versão gratuita, utiliza o **PostgreSQL** como **SGBD** (Sistema Gerenciador de Banco de dados).

No Bloco 02 estamos utilizando o **MySQL** para desenvolver o Blog Pessoal. Ambos são Banco de dados Relacionais e graças ao **Spring Data JPA**, não será necessário realizar nenhuma alteração no código da nossa aplicação. A única mudança necessária, além de adicionar a **Dependência do PostgreSQL no pom.xml**, será necessário configurar a conexão com o Banco de dados PostgreSQL na nuvem.



[Site Oficial: PostgreSQL](https://www.postgresql.org/)

No arquivo, **pom.xml**, vamos adicionar as linhas abaixo, com a dependência do **PostgreSQL**:

```
<dependency>
  <groupId>org.postgresql</groupId>
  <artifactId>postgresql</artifactId>
</dependency>
```

Passo 08 - Configurar o Banco de Dados na Nuvem

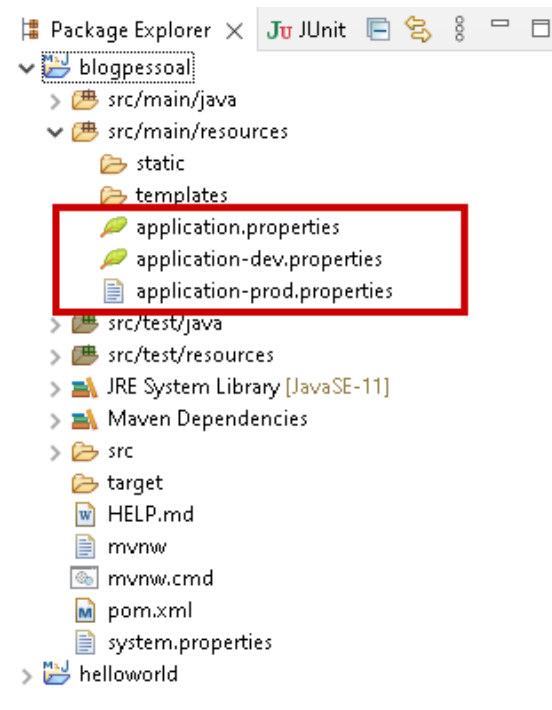
A Configuração do Banco de dados Local é diferente da configuração que será utilizada no Heroku.

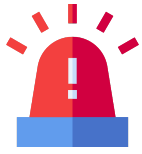
No passo anterior, adicionamos a Dependência do PostgreSQL no arquivo pom.xml, neste passo vamos configurar a aplicação para acessar o Banco de dados remoto no Heroku.

Para simplificar o processo, vamos utilizar um recurso do Spring chamado **Profiles** (perfis), que nada mais é do que criar um modelo de configuração para cada situação, ou seja, uma configuração para usar localmente (**application-dev.properties**) e outra para usar na nuvem (**application-prod.properties**).

O grande benefício dos Profiles é simplificar a troca entre a configuração Local para o Desenvolvimento da aplicação (**MySQL**) e a configuração Remota para o Deploy (**PostgreSQL**).

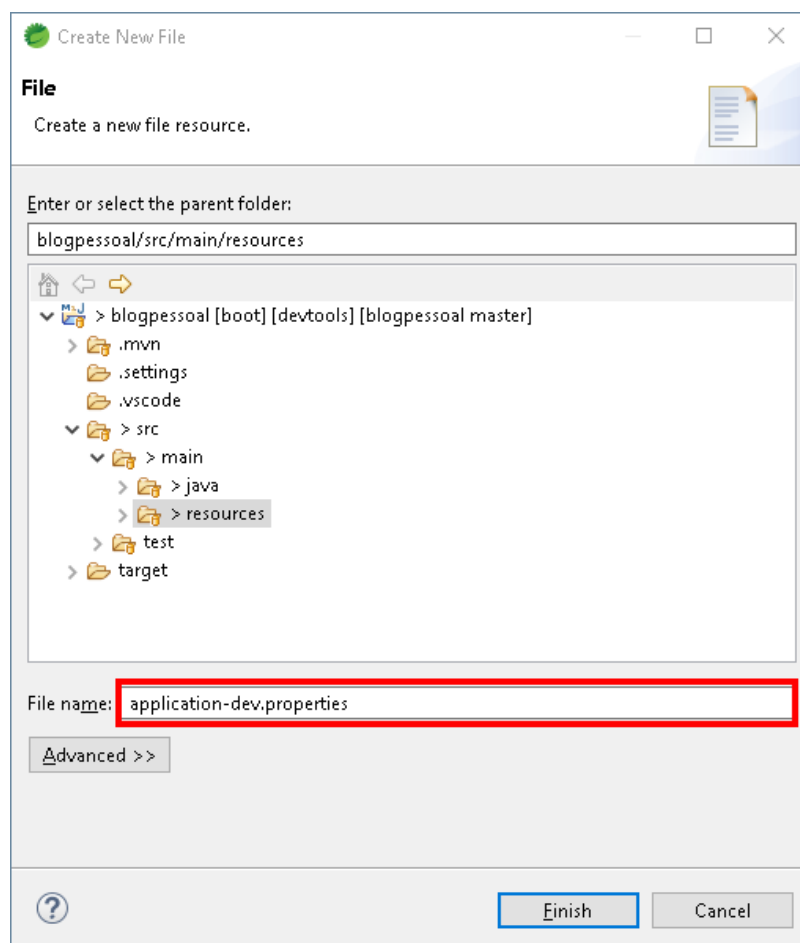
1. Na Source Folder **src/main/resources**, crie os arquivos **application-dev.properties** (Configuração do Banco de dados local) e **application-prod.properties** (Configuração do Banco de dados na nuvem).





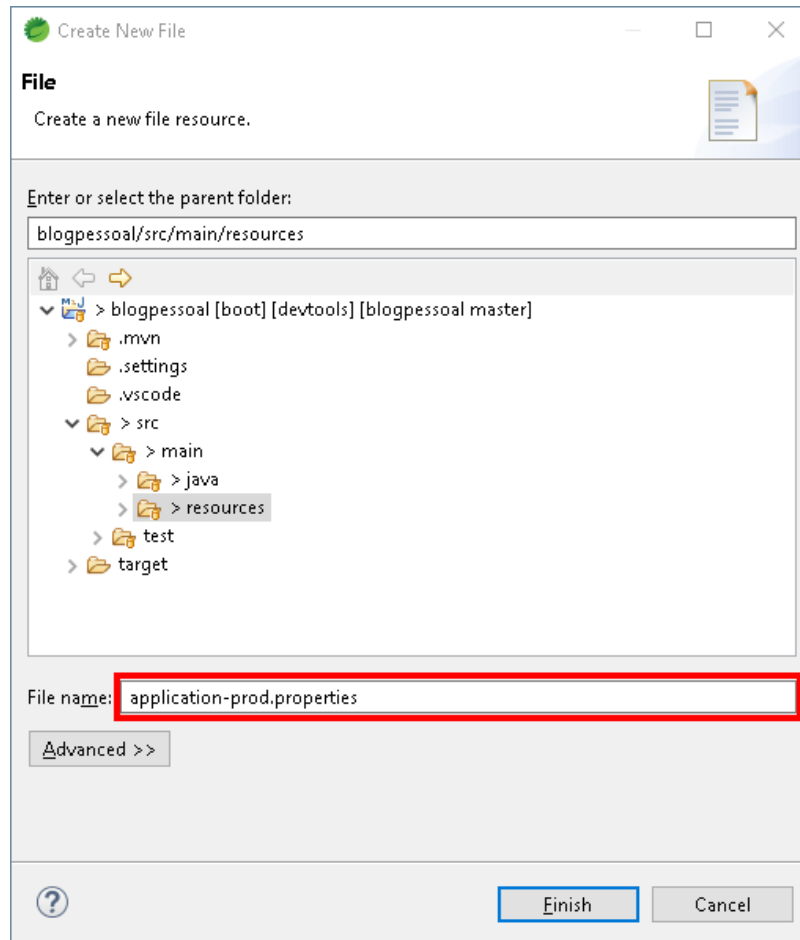
ALERTA DE BSM: Mantenha a atenção aos detalhes ao criar os arquivos *application-dev.properties* e *application-prod.properties*. Cuidado para não se equivocar ao nomear os arquivos ou criar em um pacote diferente.

2. Vamos criar o primeiro arquivo. No lado esquerdo superior, na Guia **Package explorer**, na Source Folder **src/main/resources**, clique com o botão direito do mouse e clique na opção **New** → **File**.
3. Em **File name**, digite o nome do primeiro arquivo (**application-dev.properties**) e clique no botão **Finish**.



4. Vamos criar o segundo arquivo. No lado esquerdo superior, na Guia **Package explorer**, na Source Folder **src/main/resources**, clique com o botão direito do mouse e clique na opção **New** → **File**.

5. Em **File name**, digite o nome do primeiro arquivo (**application-prod.properties**) e clique no botão **Finish**.



Agora vamos configurar os 3 arquivos:

8.1 Configuração do arquivo application.properties

1. Abra o arquivo **application.properties**, apague **todo o conteúdo da configuração do Banco de dados MySQL**, insira a linha: **spring.profiles.active=prod** e salve o arquivo. O arquivo application.properties ficará com o seguinte conteúdo abaixo:

```
spring.profiles.active=prod
```

```
springdoc.api-docs.path=/v3/api-docs
```

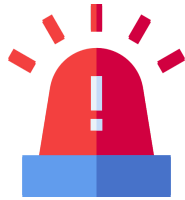
```
springdoc.swagger-ui.path=/swagger-ui.html
```

```
springdoc.swagger-ui.operationsSorter=method
```

```
springdoc.swagger-ui.disable-swagger-default-url=true
```

```
springdoc.swagger-ui.use-root-path=true
```

```
springdoc.packagesToScan=com.generation.blogpessoal.controller
```



ALERTA DE BSM: Mantenha a atenção aos detalhes ao configurar o arquivo `application.properties`. Cuidado para não apagar a configuração do Swagger (SpringDoc).

8.2 Configuração do arquivo `application-dev.properties`

1. Abra o arquivo **`application-dev.properties`**, insira as linhas abaixo (Configuração original do **`application.properties`**) e salve o arquivo. **Não esqueça de alterar a senha do usuário root caso a sua senha do MySQL não seja root.**

```
1 spring.jpa.hibernate.ddl-auto=update
2 spring.jpa.database=mysql
3 spring.datasource.url=jdbc:mysql://localhost/db_blogpessoal?
  createDatabaseIfNotExist=true&serverTimezone=America/Sao_Paulo&useSSL=false
4 spring.datasource.username=root
5 spring.datasource.password=root
6 spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect
7
8 spring.jpa.show-sql=true
9
10 spring.jackson.date-format=yyyy-MM-dd HH:mm:ss
11 spring.jackson.time-zone=Brazil/East
```

8.3 Configuração do arquivo `application-prod.properties`

1. No arquivo, **`application-prod.properties`**, insira as linhas abaixo e salve o arquivo:

```
spring.jpa.generate-ddl=true
spring.datasource.url=${JDBC_DATASOURCE_URL}
spring.jpa.show-sql=true

spring.jackson.date-format=yyyy-MM-dd HH:mm:ss
spring.jackson.time-zone=Brazil/East
```



ATENÇÃO: Depois de finalizar as configurações dos 3 arquivos, recomendamos executar o comando **Update Project** para atualizar as configurações do projeto.

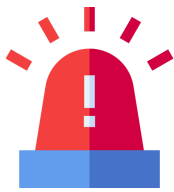
8.4 Alternando entre os perfis no arquivo `application.properties`

1. Para alternar entre as configurações Local e Remota, abra o arquivo **`application.properties`** e utilize uma das 2 opções abaixo:

`spring.profiles.active=dev` → O Spring executará a aplicação com a configuração do Banco de dados local (MySQL)

`spring.profiles.active=prod` → O Spring executará a aplicação com a configuração do Banco de dados na nuvem (Heroku)

Para o Deploy, devemos deixar a linha **`spring.profiles.active`** configurada com a opção **`prod`**.



ALERTA DE BSM: Mantenha a atenção aos detalhes ao criar os perfis do Banco de Dados. Um erro muito comum é tentar executar o seu projeto no STS com o Perfil `prod` habilitado no arquivo `application.properties`. Com o perfil `prod` habilitado, o projeto não será inicializado.



[Código fonte: Projeto finalizado](#)



Passo 09 - Deploy com o Git

Vamos preparar o nosso repositório local para subir a aplicação para o Heroku utilizando o Git.

1. Na pasta do projeto, clique com o botão direito do mouse e na sequência clique na opção: **Show in → System Explorer**
2. Será aberta a pasta Workspace onde o Eclipse/STS grava os seus projetos:
3. Copie a pasta da API: **`blogpessoal`**

Nome	Data de modificação	Tipo	Tamanho
.metadata	20/07/2021 00:26	Pasta de arquivos	
blogpessoal	02/08/2021 01:54	Pasta de arquivos	
helloworld	23/07/2021 09:19	Pasta de arquivos	
lojagames	30/07/2021 09:30	Pasta de arquivos	

4. Cole a pasta no mesmo diretório

Nome	Data de modificação	Tipo	Tamanho
.metadata	20/07/2021 00:26	Pasta de arquivos	
blogpessoal	20/07/2021 00:29	Pasta de arquivos	
blogpessoal - Copia	02/08/2021 01:54	Pasta de arquivos	
helloworld	23/07/2021 09:19	Pasta de arquivos	
lojagames	30/07/2021 09:30	Pasta de arquivos	

5. Renomeie a pasta para **deploy_blogpessoal**

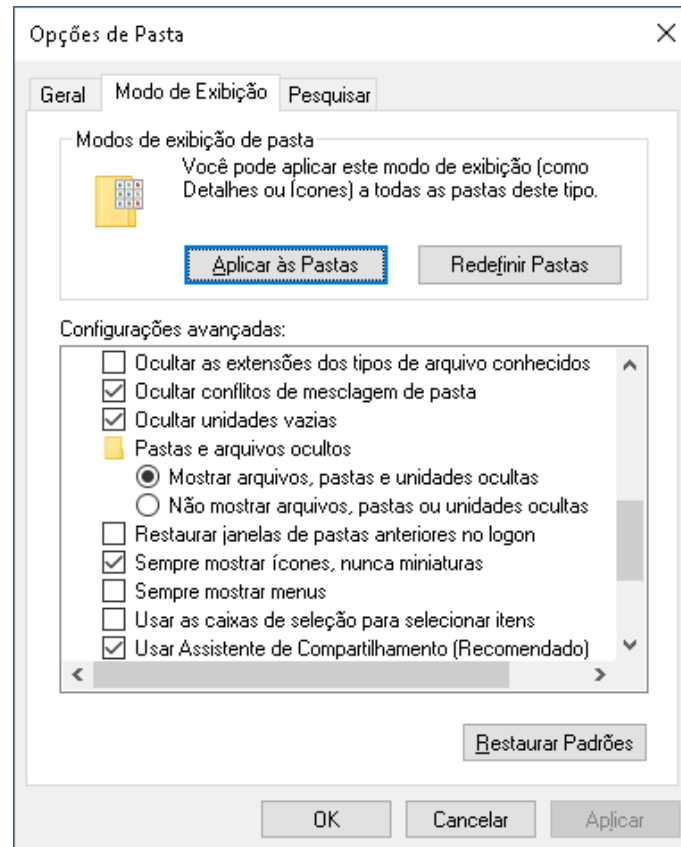
Nome	Data de modificação	Tipo	Tamanho
.metadata	20/07/2021 00:26	Pasta de arquivos	
blogpessoal	20/07/2021 00:29	Pasta de arquivos	
deploy_blogpessoal	02/08/2021 01:54	Pasta de arquivos	
helloworld	23/07/2021 09:19	Pasta de arquivos	
lojagames	30/07/2021 09:30	Pasta de arquivos	

6. Abra a pasta **deploy_blogpessoal** e verifique se existe uma pasta chamada **.git**.

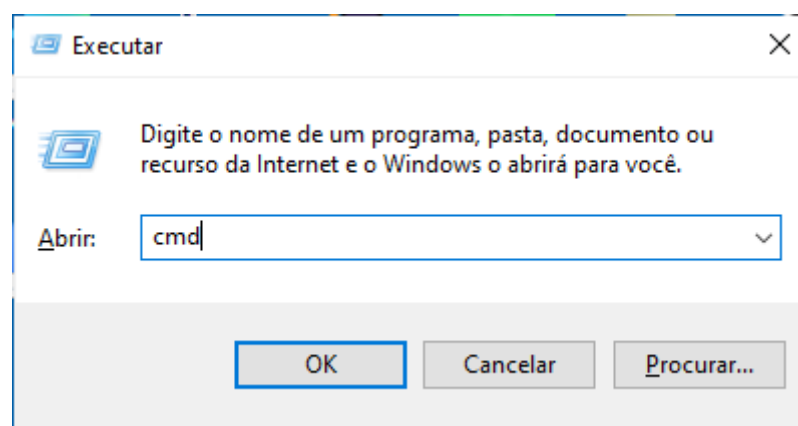
Caso ela exista, apague esta pasta. **Esta pasta estará presente APENAS se você inicializou o git dentro da pasta do projeto.**

Nome	Data de modificação	Tipo	Tamanho
.git	13/07/2021 16:50	Pasta de arquivos	
.mvn	13/07/2021 07:39	Pasta de arquivos	
.settings	13/07/2021 07:39	Pasta de arquivos	
.vscode	13/07/2021 07:39	Pasta de arquivos	
src	13/07/2021 07:39	Pasta de arquivos	
target	13/07/2021 07:39	Pasta de arquivos	
.classpath	13/07/2021 22:27	Arquivo CLASSPA...	3 KB
.gitignore	25/05/2021 22:47	Documento de Te...	1 KB
.project	27/05/2021 18:05	Arquivo PROJECT	1 KB
HELP.md	25/05/2021 22:47	Markdown File	2 KB
mvnw	25/05/2021 22:47	Arquivo	10 KB
mvnw.cmd	25/05/2021 22:47	Script de Comand...	7 KB
pom.xml	13/07/2021 22:27	Documento XML	3 KB
system.properties	13/07/2021 22:26	Arquivo PROPERTI...	1 KB

7. Caso esta pasta não esteja sendo exibida, na janela do Windows Explorer, clique na **Guia Exibir** e na sequência no botão **Opções**. Na janela **Opções de Pasta**, na **Guia Modo de Exibição**, no item **Configurações avançadas**, localize a opção: **Pastas e arquivos ocultos** e marque a opção **Mostrar arquivos, pastas e unidades ocultas** (como mostra a figura abaixo). Em seguida clique em **OK** para concluir.



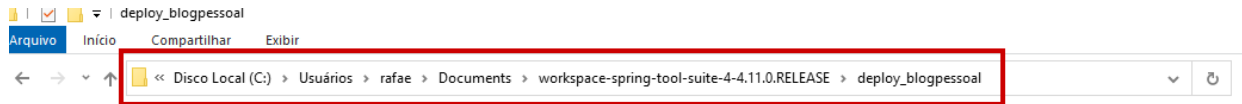
8. Execute o atalho  +  para abrir a janela Executar



9. Digite o comando abaixo para abrir o **Prompt de Comando do Windows**:

cmd

10. Na pasta **deploy_blogpessoal**, no **Windows Explorer**, copie o caminho da pasta conforme a figura abaixo:



11. No Prompt de comando do Windows digite o comando `cd` e cole na frente do comando o caminho copiado:

```
cd C:\Users\seu usuario\Documents\workspace-spring-tool-suite-4-4.11.0.RELEASE\deploy_blogpessoal
```

**o nome da pasta pode ser diferente*



ALERTA DE BSM: Mantenha a Atenção aos Detalhes ao iniciar o Deploy pelo Git. A partir deste ponto, todos os comandos do Git e do Heroku Client devem ser executados via Prompt de Comando do Windows (CMD), dentro da pasta *deploy-blogpessoal*.

12. Digite a sequência de comandos abaixo para inicializar o seu repositório local para efetuar o Deploy no Heroku:

```
git init
git add .
git commit -m "Deploy inicial - Blog Pessoal"
```

Passo 10 - Login no Heroku

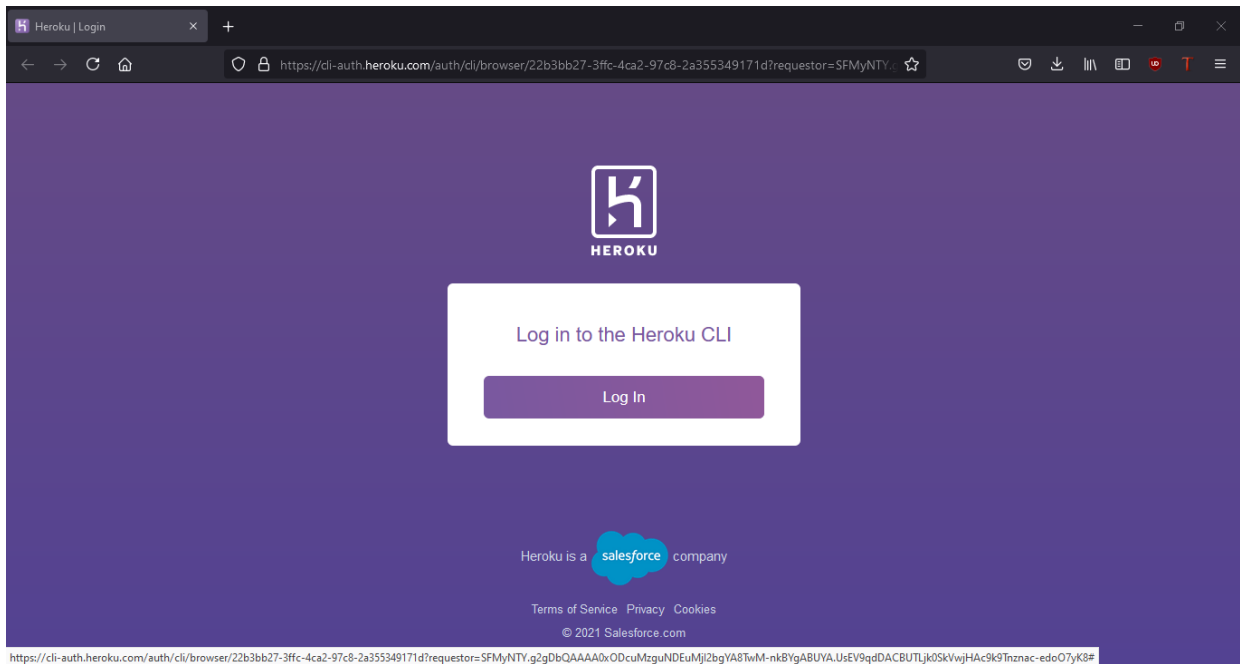
1. Digite o comando:

```
heroku login
```

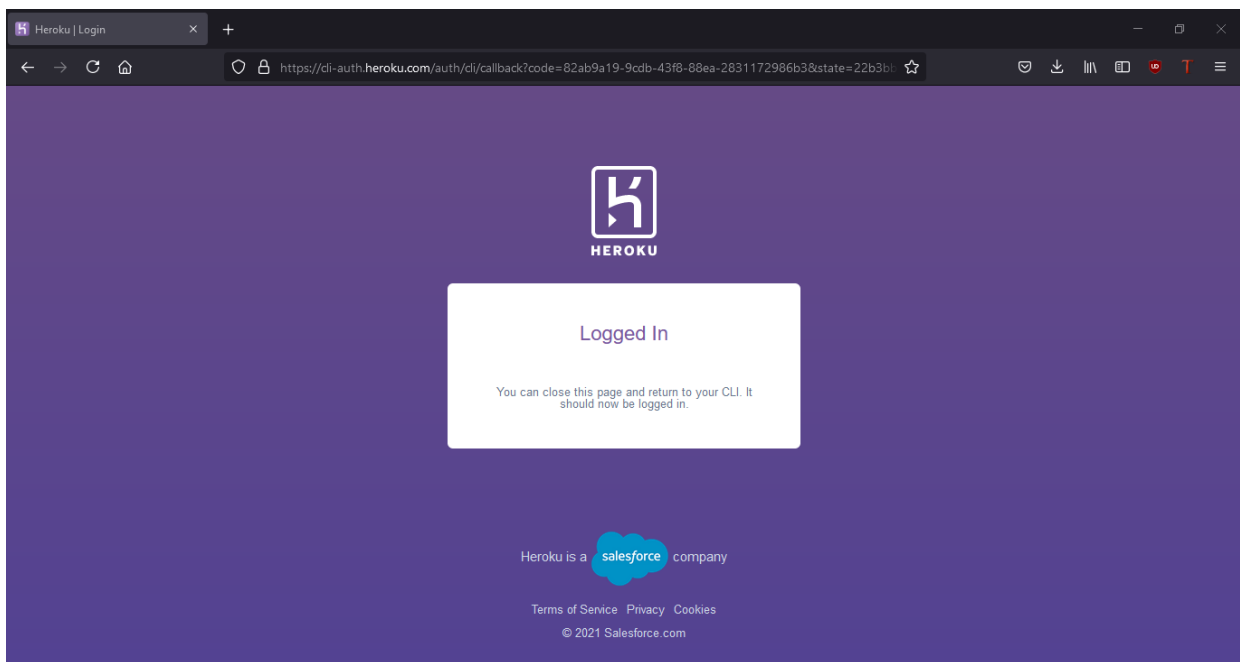
2. Será exibida a mensagem abaixo, solicitando que uma tecla do seu teclado seja pressionada.

```
>> Warning: Our terms of service have changed: https://dashboard.heroku.com/terms-of-service
heroku: Press any key to open up the browser to login or q to exit:
```

3. Na sequência, será aberta a janela abaixo. Clique no botão **Log in**



4. Após efetuar o login na sua conta, será exibida a janela abaixo.



5. Volte para o Prompt de comando para continuar o Deploy.

6. Será exibida a mensagem abaixo no prompt comando informando que você efetuou login no Heroku.

```
>> Warning: Our terms of service have changed: https://dashboard.heroku.com/terms-of-service
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/22b3bb27-3ffc-4ca2-97c8-2a355349171d?requestor=SFMyNTY.g
2gDbQAAAA0xODcuMzguNDEuMjI2bgYA8TwM-nkBYgABUYA.USeV9qdDACBUTLjk0SkVwjHAc9k9Tznac-edo07yK8
Logging in... done
Logged in as rafaelpinfo@gmail.com
```



Passo 11 - Criar um novo projeto no Heroku

1. Para criar um novo projeto na sua conta do Heroku, digite o comando abaixo, onde o **nomedoprojeto** deve ser substituído por um nome (escolhido por você), que esteja disponível no Heroku.

```
heroku create nomedoprojeto
```



ATENÇÃO: O NOME DO PROJETO NÃO PODE CONTER LETRAS MAIUSCULAS, NUMEROS OU CARACTERES ESPECIAIS. ALÉM DISSO ELE PRECISA SER ÚNICO DENTRO DA PLATAFORMA HEROKU.

2. Se o nome escolhido já existir, será exibida a mensagem abaixo:

```
Creating ☐ bloggen... !  
! Name bloggen is already taken
```

3. Se o nome escolhido for aceito, será exibida a mensagem abaixo:

```
Creating ☐ bprfp... done  
https://bprfp.herokuapp.com/ | https://git.heroku.com/bprfp.git
```



Passo 12 - Adicionar o Banco de dados no Heroku

1. Para adicionar um **Banco de Dados PostgreSQL** no seu projeto, digite o comando abaixo, onde o **nomedoprojeto** deve ser substituído pelo nome do projeto que foi criado no passo anterior.

```
heroku addons:create heroku-postgresql:hobby-dev -a nomedoprojeto
```

2. Se o Banco for adicionado corretamente, será exibida a mensagem abaixo:

```
Creating heroku-postgresql:hobby-dev on ☐ bprfp... free  
Database has been created and is available  
! This database is empty. If upgrading, you can transfer  
! data from another database with pg:copy  
Created postgresql-flexible-10004 as DATABASE_URL  
Use heroku addons:docs heroku-postgresql to view documentation
```



ATENÇÃO: *O processo do Deploy envlará apenas a sua aplicação para a nuvem, logo o Banco de dados que será criado nesta etapa estará vazio.

Passo 13 - Efetuar o Deploy

1. Para concluir o Deploy, digite o comando:

```
git push heroku master
```

2. Ao finalizar o Deploy, será exibida a mensagem **BUILD SUCESS** (destacado em verde na imagem) e será exibido o endereço (<https://nomedoprojeto.herokuapp.com>) para acessar a API na Internet (destacado em amarelo na imagem)

```
remote: [INFO] -----
remote: [INFO] BUILD SUCCESS
remote: [INFO] -----
remote: [INFO] Total time: 20.827 s
remote: [INFO] Finished at: 2021-06-11T07:59:26Z
remote: [INFO] -----
remote: ----> Discovering process types
remote: Procfile declares types -> (none)
remote: Default types for buildpack -> web
remote: ----> Compressing...
remote: Done: 108.9M
remote: ----> Launching...
remote: Released v5
remote: https://bprfp.herokuapp.com/ deployed to Heroku
remote: Verifying deploy... done.
To https://git.heroku.com/bprfp.git
 * [new branch]      master -> master
C:\Users\rafael\Documents\workspace-spring-tool-suite-4-4.10.0.RELEASE\blogpessoal>
```

Passo 14 - Configurar o fuso horário

1. Após concluir o Deploy (Passo 13 não apresentou nenhum problema), antes de testar a aplicação, digite o comando abaixo para configurar o fuso horário do servidor do Heroku:

```
heroku config:add TZ="America/Sao_Paulo" --app nomedoprojeto
```

Passo 15 - Abrir o link e Testar a aplicação

15.1 Abrir o link

1. Abra o navegador e digite o endereço a sua aplicação (<https://nomedoprojeto.herokuapp.com>), onde o nome do projeto deve ser substituído pelo nome do seu projeto, criado no Heroku.
2. Será solicitado o **Usuário e a Senha**. Digite **root** para ambos.
3. Sua aplicação abrirá o **Swagger**.
4. Utilize o **Checklist do Blog Pessoal** para verificar se o projeto está completo.

15.1 Testar a Aplicação no Insomnia

1. Abra o Insomnia e acesse a Workspace **Blog Pessoal**.
2. Crie uma pasta chamada **Blog Pessoal** e arraste as 3 pastas (Postagem, Tema e Usuario) para dentro dela.
3. Duplica a pasta **Blog Pessoal**.
4. Na próxima janela, defina o nome da nova pasta como **Blog Pessoal - Heroku**.



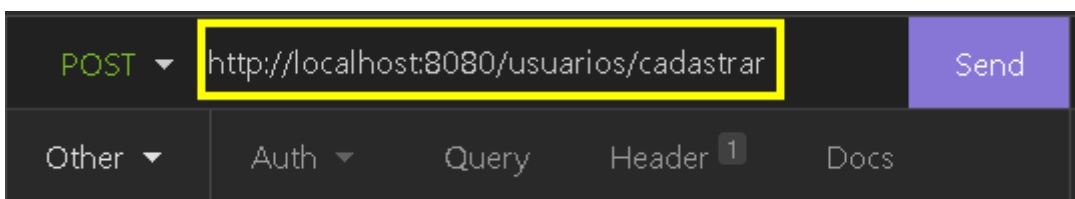
Duplicate Folder

New Name

Blog Pessoal - Heroku

Create

5. Abra a requisição Cadastrar Usuário na pasta **Blog Pessoal - Heroku**.
6. Altere o caminho atual: <http://localhost:8080/usuarios/cadastrar>



POST

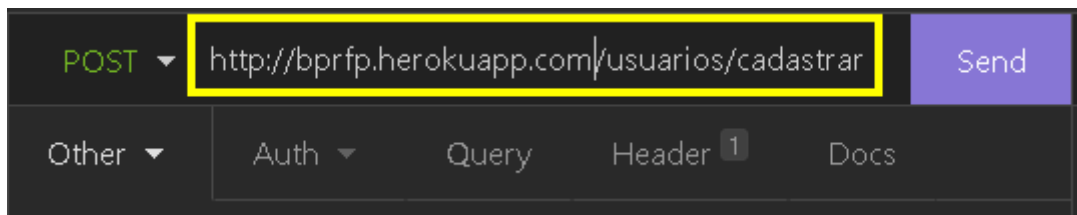
<http://localhost:8080/usuarios/cadastrar>

Send

Other Auth Query Header 1 Docs

7. Para o endereço do Heroku:
<https://meuprojeto.herokuapp.com/usuarios/cadastrar> (No exemplo abaixo:

<https://bprfp.herokuapp.com/usuarios/cadastrar>)



8. Execute a requisição e verifique se o Usuário foi criado corretamente.
9. Atualize o caminho de todas requisições da pasta **Blog Pessoal - Heroku**
10. Execute a requisição Login para acessar a API
11. Continue os testes conforme as orientações do **Checklist do Projeto Blog Pessoal**.

Atualizar o Deploy no Heroku



ALERTA DE BSM: *Mantenha a atenção aos detalhes e a persistência. Este item você utilizará apenas se você alterou alguma coisa no seu projeto Spring e necessite atualizar a aplicação na nuvem.*



1. Para fazer alterações no código do projeto e executar localmente, volte para o STS e altere a primeira linha do arquivo, **application.properties** conforme o código abaixo:

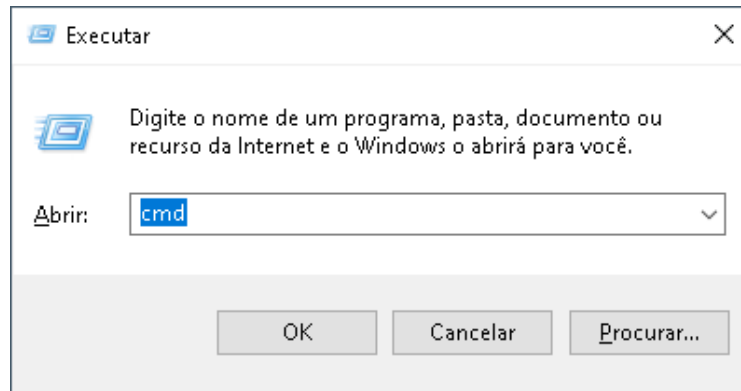
```
spring.profiles.active=dev
```

2. Faça as alterações necessárias no seu projeto, execute localmente e verifique se está tudo funcionando.
3. Antes de refazer o Deploy, altere novamente a primeira linha do arquivo, **application.properties** conforme o código abaixo:

```
spring.profiles.active=prod
```

4. Na pasta do projeto Blog Pessoal no STS, clique com o botão direito do mouse e na sequência clique na opção: **Show in → System Explorer**

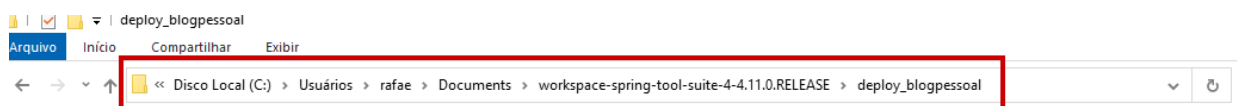
5. Será aberta a pasta **Workspace** onde o STS grava os seus projetos. Abra a pasta do projeto (**blogpessoal**).
6. Selecione todo o conteúdo da pasta do projeto (**exceto o arquivo .git**) e faça uma cópia do conteúdo (**CTRL + C**).
7. Abra a pasta do Deploy (**deploy_blogpessoal**) e cole o conteúdo (**CTRL + V**) dentro da pasta para atualizar a pasta do Deploy.
8. Execute o atalho  +  para abrir a janela Executar.



9. Digite o comando abaixo para abrir o **Prompt de Comando do Windows**:

cmd

10. Na pasta do Deploy do seu projeto, no Windows Explorer, copie o caminho da pasta **deploy_blogpessoal** conforme a figura abaixo:



11. No Prompt de comando do Windows digite o comando **cd** e cole na frente do comando o caminho copiado na pasta **deploy_blogpessoal**:

```
cd C:\Users\seu usuario\Documents\  
workspace-spring-tool-suite-4-4.11.0.RELEASE\deploy_blogpessoal
```

**o caminho da pasta pode ser diferente*

12. Atualize o Deploy utilizando a sequência de comandos abaixo:



ALERTA DE BSM: Mantenha a Atenção aos Detalhes ao atualizar o Deploy pelo Git. A partir deste ponto, todos os comandos do Git e do Heroku Client devem ser executados via Prompt de Comando do Windows (CMD), dentro da pasta *deploy-blogpessoal*.

```
git add .  
git commit -m "Atualização do Deploy - Blog Pessoal"  
git push heroku master
```

13. Ao finalizar a atualização do Deploy, será exibida a mensagem **BUILD SUCESS** e será exibido o endereço (<https://nomedoprojeto.herokuapp.com>) para acessar a API na Internet.

14. Caso ocorra algum erro de vinculação (link), verifique se a pasta está vinculada ao Heroku utilizando o comando abaixo:

```
git remote
```

15. Caso não apareça o resultado **heroku**, utilize o comando abaixo para vincular a pasta com o projeto no heroku.

```
heroku git:remote -a nomedoprojeto
```

16. Caso o comando acima falhe, inicialize o repositório git e refaça a vinculação.

```
git init  
heroku git:remote -a nomedoprojeto
```

17. Para atualizar o Deploy, utilize os comandos baixo:

```
git add .  
git commit -m "Atualização do Deploy - Blog Pessoal"  
git push heroku master
```

18. Caso o comando git push falhe, acrescente a opção **-f** para forçar o Deploy.

```
git push -f heroku master
```

19. Se todas as opções acima falharem, verifique se o erro não está na aplicação.