Desenvolvimento de Software para a Web II - Roteiro 1

Prof. Thiago Cavalcante

1. Instale o editor de texto*
2. Instale os programas necessários*
3. Crie a pasta de projetos*
4. Crie um app Rails chamado **primeiro_app** usando o comando **rails <versão> new <nome_app>***
5. Atualize o arquivo **Gemfile** e reinstalar os pacotes com o **Bundler***
6. Veja o app funcionando com o comando **rails server****
7. **Exercícios**

   (a) Verifique a **versão do Ruby** na página inicial do seu app e confirme rodando o comando **ruby -v** no terminal

   (b) Verifique a **versão do Rails** na página inicial do seu app; verifique que é a mesma versão que consta no **Gemfile**

8. Crie uma ação **ola** no controlador da aplicação

   app/controllers/application_controller.rb

```
class ApplicationController < ActionController::Base
  def ola
    render html: "olá, mundo!"
  end
end
```

*Olhar o **tutorial**

Esperar aparecer a mensagem sobre **Ctrl-C

9. Adicione uma rota para direcionar a ação anterior para a página inicial do app

config/routes.rb

```
Rails.application.routes.draw do
  root 'application#ola'
end
```

10. **Exercícios**

    (a) Modifique a ação **ola** para exibir a mensagem **"olá, pessoal!"**

    (b) Crie uma segunda ação chamada **tchau** que exiba o texto **"tchau, mundo!"** e modifique a rota da página inicial para esta nova ação

11. Faça a configuração inicial do **git**

```
git config --global user.name "Seu Nome"
git config --global user.email seu.email@email.com
git config --global credential.helper "cache --timeout=86400"
```

12. Adicione todos os arquivos ao repositório e faça o primeiro **commit**

```
git status # checando arquivos antes de adicionar
git add -A
git status # checando arquivos depois de adicionar
git commit -m "Inicializar repositório"
git log # checando histórico de commits
```

13. Crie um repositório no **GitHub**, associe-o ao seu repositório local e envie o código

```
git remote add origin <link https>
git push -u origin master
```

14. Crie um novo **branch** no repositório para alterar o **README** com a descrição do projeto

```
git checkout -b modificar-README
git branch # checando o branch atual
git status # checando as modificações atuais
git commit -a -m "Melhorar o README"
```

15. Insira as modificações do **branch** criado no **branch** principal do projeto (**master**)

```
git checkout master
git merge modificar-README
git branch -d modificar-README # deletando o branch
git push # enviando o código para o GitHub
```

16. Adicione a **gem** do **PostgreSQL** no **Gemfile**, no ambiente de produção, para assegurar compatibilidade com o **Heroku**

Gemfile

```
group :production do
  gem 'pg', '0.20.0'
end
```

17. Modifique o **Gemfile** de forma que a **gem** do **SQLite** seja usada apenas nos ambientes de desenvolvimento e teste

Gemfile

```
group :development, :test do
  gem 'sqlite3', '1.4.1'
  ...
end
```

18. Atualize os pacotes com o **Bundler** e salve as alterações no **git**

```
bundle update
bundle install --without production
git commit -a -m "Atualizar Gemfile para o Heroku"
```

19. Faça login no **Heroku** e crie o app

```
heroku login --interactive
heroku create # dentro da pasta do projeto
```

20. Envie o código para o **Heroku**

```
git push heroku master
```

21. **Exercícios**

    (a) repita os mesmos exercícios do tópico **10**, dessa vez fazendo o **deployment** para o **Heroku**

    (b) Rode **heroku help** para ver a lista de comandos do **Heroku** e descubra qual deles exibe os registros da atividade do seu app (**logs**)

    (c) Use o comando anterior para ver o **evento mais recente**