1 Oque é Django?

Abstract O desenvolvimento Web é uma necessidade de grande nos dias atuais, e será futu- ramente, com isso, visa-se a necessidade da criação e utilização de ferramentas que proporcionem uma melhor qualidade, velocidade e adaptabilidade dos projetos. Com isso, o web framework Django foi uma das soluções que utiliza da linguagem python para fazer o devesenvolvimento de aplicações Web. 1 Oque é Django? O Django é um web framework que segue a arquitetura MVT, e com ele podemos criar aplicações web com maior facilidade e rapidez, pelo fato de ele cuidar da maioria dos detalhes intrısecos do desenvolvimento web. Como uma das suas caracterısticas, é que ele é de uso livre, e possui código open- source. Dentre algumas das ferramentas que o Django inclui, podemos listar al- gumas como: • Site maps • Authentication • Context Administrator • RSS Feeds Além disso, possui ferramentas que possibilitam uma maior segurança da aplicação, e muitas outras ferramentas voltadas para o desenvolvedor. 2 Instalação e inicialização do framework Dentro do python, é comum a utilização de ambientes virtuais, pois ele facilita e organiza o uso de bibliotecas que podem ser baixadas para o uso da nossa aplicação. Com isso, para inicializarmos nosso ambiente virtual e instalar o framework no ambiente virtual, podemos usar as seguintes instruções bash $python-mvenvnome_ambiente_virtualAdependerdaplataformadedesenvolvimentoescolhida, devemosativa$ source ./nome$_ambiente_virtual/bin/activate$ pip install django Posteriormente a isso, temos no nosso terminal, uma das ferramentas que instalamos junto com o Django, que é sua CLI, a qual permite iniciarmos o projeto e criar automaticamente sua estrutura e os arquivos necessários para cosntruirmos uma aplicação Django. Para criarmos a aplicação, devemos usar o documando: $django-adminstartprojectnomeProjetoNapartefinaldocomandotemosporexemploaopçãodeadicionaroprojetoder$ manage.py runserver Com isso, estaremos rodando nossa aplicação, e ela pode ser acessada dentro do localhost, na porta 8000, que são definidos por padrão, mas caso necessário, podemos ser modificados nas configurações. Posteriormente a isso, temos que nosso projeto foi criado com uma esrtu- tura especıfica, que vai ser destrinchada a seguir: 2.1 Estrutura dos arquivos do Django Os arquivos que foram criados dentro do nosso projeto possuem caracterısticas distintas dos outros devido á funcionalidade que lhe é atribuida, eles se car- acterizam por serem: • manage.py: Esse é o arquivo utilitário da CLI do django, que é usado como django-admin. Ele nos permite interagir com o projeto de várias maneiras.2 Instalação e inicialização do framework 3 • db.sqlite3: É nossa base de dados, por configuração o SQLite é a nossa base de dados por ser leve e ser integrado com o python por natureza, não necessitando de configurações para que funcione desde o inıcio. Outras bases de dados podem substituir a mesma posterior- mente, mas para teste, ela é recomendada. • init .py: Arquivo de inicialização do pacote. Nosso projeto é desenvolvido como um pacote python, sendo assim, quando o mesmo é chamado, é necessário um arquivo de inicialização para que o mesmo seja executado na chamada do nosso pacote. • wsgi.py: Um entry-point para servidores web que sã ocompatıveis com WSGI (Web Server Gateway Interface) • asgi.py: Funciona como o WSGI, mas de forma assıncrona • settings.py: Ar-

quivo de configurações do nosso projeto, nos permite configurar até a forma como as variáveis de ambiente do projeto se comportam, até mesmo em qual pasta os arquivos estáticos se local- izam. • urls.py: As declarações de URL do nosso projeto Django funcionam como uma tabela, que permite especificarmos as URLs que podem ser acessadas no nosso site. Cada um desses arquivos possuem configurações padrões, que são mınimas para o funcionamento do Django. 2.2 Django.settings e funcionamento do framework O funcionamento do framework está intrısecamento ligado ao seu arquivo de configurações, pois é ele que faz o "meio de campo" entre todas as funcional- idades da ferramenta, por isso merece uma atenção em especial. Dentro dele teremos logo de inıcio, a nossa ligação com o banco de dados, que é feito por meio da biblioteca pathlib, que define a pasta a qual se localiza nossa base de dados que foi criada junto com o projeto. Temos que o django funciona por meio da ideia de "apps", como se cada parte da nossa aplicação fosse um app dedicado, ou ela fosse um app como um todo, e isso pode ser definido por meio de subprojetos dentro do qual estamos atualmente — esse tópico será tocado com mais detalhes e calma posteriormente — temos também que o funcionamento do Django permite a presença de middlewares2 Instalação e inicialização do framework 2.3 4 Criando as primeiras urls no Django Após criarmos nossa primeira aplicação, e vermos que sua instalação foi feita de maneira correta e completa, podemos assim criar nossas primeiras URLs, para que elas sejam acessadas dno navegador. Devemos nos direcionar ao arquivo de urls.py, dentro dela temos o seguinte: from django.contrib import admin from django.urls import path urlpatterns = [path('admin/', admin.site.urls),] Esse código permite que nós nos comuniquemos com os servidores por meio do protocolo HTTP(Hyper Text Transfer Protocol), que nos permite fazer requisições dos dados ao servidor, e ele nos responder com as informações desejada. Esse protocolo possui uma variação que é o HTTPS, que é sua versão que possui uma camada extra de segurança, que permite uma maior responsabilidade do lado do desenvolvedor e uma maior segurança do lado do cliente. Com isso, podemos assim, implementar uma URL customizada da seguinte forma: from django.contrib import admin from django.urls import path, HttpResponse def view(request): return HttpResponse('Voltando código') urlspatterns = [path('blog/', view)] Assim, quando criarmos essa função e essa url, vamos poder acessar essa url no nosso navegador, e ele vai prover o conteúdo disponibilizado pela função para que nós possamos usar ele para disponibilizar conteúdo para a requisição.3 Criando nossas primeiras aplicações Web 2.4 5 Criando apps com o manage.py Os apps dentro do Django, são como uma subcategoria do projeto, que podem ser criadas várias dentro de um mesmo projeto, e isso é muito útil quando temos por exemplo, um projeto que deve ter duas formas de aplicação com- pletamente diferentes, como um e-commerce que deve ter a parte de compra e venda de produtos feita pelos usuários, e a parte administrativa das con- tas que não tem acesso as informações de compra e venda (necessariamente diretamente), mas que é usada para ver as estatıticas de acesso e uso do site. Tendo esse exemplo em mente, visa-se a necessidade e a utilidade do uso de apps dentro do Django, para criarmos o nosso, devemos usar o comando: $python3 manage.py startapp nome_do_app Quando criarmos nosso app, teremos numa nova pasta que come later -$

$emosassimumanovaformadecriarnossosite, eorganizareledamaneiramaisfácil.3Criandonossasprimeir$
$mitirumfuncionamentomaiscleanemelhordanossaaplicação, alémdofatodequenoarquivodeurls, organiz$
$projeto/urls.pyfromdjango.contribimportadminfromdjango.urlsimportpath, includeurlpatterns =$
$[path('/', include('home.urls')), path('blog/', include('blog.urls')), 3Criandonossasprimeirasaplicações$
$returnHttpResponse("retornandodapáginainicialdoblog")blog/urls.pyfromdjango.urlsimportpathfrom$
$[path('', views.blog),]3.2Renderizandotemplates, html, econfigurandooINSTALLEDAPPSAinternet$
$gramação.Comisso, paraservimosessestiposdearquivos, precisamoses-pecificaraexistência, eaentregad$
$fromdjango.shortcutsimportrenderdefhome(request) : returnrender(request,' nome-$
$do - arquivo.html')$

1 Abstract O desenvolvimento Web é uma necessidade de grande nos dias atuais, e será futu- ramente, com isso, visa-se a necessidade da criação e utilização de ferramentas que proporcionem uma melhor qualidade, velocidade e adaptabilidade dos projetos. Com isso, o web framework Django foi uma das soluções que utiliza da linguagem python para fazer o devesenvolvimento de aplicações Web. 1 Oque é Django? O Django é um web framework que segue a arquitetura MVT, e com ele podemos criar aplicações web com maior facilidade e rapidez, pelo fato de ele cuidar da maioria dos detalhes intrısecos do desenvolvimento web. Como uma das suas caracterısticas, é que ele é de uso livre, e possui código open- source. Dentre algumas das ferramentas que o Django inclui, podemos listar al- gumas como: • Site maps • Authenticati on • Context Administrator • RSS Feeds Além disso, possui ferramentas que possibilitam uma maior segurança da aplicação, e muitas outras ferramentas voltadas para o desenvolvedor. 2 Instalação e inicialização do framework Dentro do python, é comum a utilização de ambientes virtuais, pois ele facilita e organiza o uso de bibliotecas que podem ser baixadas para o uso da nossa aplicação. Com isso, para inicializarmos nosso ambiente virtual e instalar o framework no ambiente virtual, podemos usar as seguintes instruções bash $python-mvenvnome_ambiente_virtualAdependerdaplafatormadedesenvolvimentoescolhida, devemosativo$ source ./nome$_a$mbiente$_v$irtual/bin/activate pip install django Posteriormente a isso, temos no nosso terminal, uma das ferramentas que instalamos junto com o Django, que é sua CLI, a qual permite iniciarmos o projeto e criar automaticamente sua estrutura e os arquivos necessários para cosntruirmos uma aplicação Django. Para criarmos a aplicação, devemos usar o documando: $django-adminstartprojectnomeProjetoNapartefinaldocomandotemosporexemploaopçãodeadicionaroprojetoder$ manage.py runserver Com isso, estaremos rodando nossa aplicação, e ela pode ser acessada dentro do localhost, na porta 8000, que são definidos por padrão, mas caso necessário, podemos ser modificados nas configurações. Posteriormente a isso, temos que nosso projeto foi criado com uma esrtu- tura especıfica, que vai ser destrinchada a seguir: 2.1 Estrutura dos arquivos do Django Os arquivos que foram criados dentro do nosso projeto possuem caracterısticas distintas dos outros devido á funcionalidade que lhe é atribuida, eles se car- acterizam por serem: • manage.py: Esse é o arquivo utilitário da CLI do django, que é usado como django-admin. Ele nos permite interagir com o projeto de várias maneiras.2 Instalação e inicialização do framework 3 • db.sqlite3: É nossa base de dados, por configuração o SQLite é a nossa base de dados por ser leve e ser integrado com o python por natureza, não necessitando de configurações para que funcione desde

o inıcio. Outras bases de dados podem substituir a mesma posterior- mente, mas para teste, ela é recomendada. • init .py: Arquivo de inicialização do pacote. Nosso projeto é desenvolvido como um pacote python, sendo assim, quando o mesmo é chamado, é necessário um arquivo de inicialização para que o mesmo seja executado na chamada do nosso pacote. • wsgi.py: Um entry-point para servidores web que sã ocompatıveis com WSGI (Web Server Gateway Interface) • asgi.py: Funciona como o WSGI, mas de forma assıncrona • settings.py: Arquivo de configurações do nosso projeto, nos permite configurar até a forma como as variáveis de ambiente do projeto se comportam, até mesmo em qual pasta os arquivos estáticos se local- izam. • urls.py: As declarações de URL do nosso projeto Django funcionam como uma tabela, que permite especificarmos as URLs que podem ser acessadas no nosso site. Cada um desses arquivos possuem configurações padrões, que são mınimas para o funcionamento do Django. 2.2 Django.settings e funcionamento do framework O funcionamento do framework está intrısecamento ligado ao seu arquivo de configurações, pois é ele que faz o "meio de campo" entre todas as funcional- idades da ferramenta, por isso merece uma atenção em especial. Dentro dele teremos logo de inıcio, a nossa ligação com o banco de dados, que é feito por meio da biblioteca pathlib, que define a pasta a qual se localiza nossa base de dados que foi criada junto com o projeto. Temos que o django funciona por meio da ideia de "apps", como se cada parte da nossa aplicação fosse um app dedicado, ou ela fosse um app como um todo, e isso pode ser definido por meio de subprojetos dentro do qual estamos atualmente — esse tópico será tocado com mais detalhes e calma posteriormente — temos também que o funcionamento do Django permite a presença de middlewares2 Instalação e inicialização do framework 2.3 4 Criando as primeiras urls no Django Após criarmos nossa primeira aplicação, e vermos que sua instalação foi feita de maneira correta e completa, podemos assim criar nossas primeiras URLs, para que elas sejam acessadas dno navegador. Devemos nos direcionar ao arquivo de urls.py, dentro dela temos o seguinte: from django.contrib import admin from django.urls import path urlpatterns = [path('admin/', admin.site.urls),] Esse código permite que nós nos comuniquemos com os servidores por meio do protocolo HTTP(Hyper Text Transfer Protocol), que nos permite fazer requisições dos dados ao servidor, e ele nos responder com as informações desejada. Esse protocolo possui uma variação que é o HTTPS, que é sua versão que possui uma camada extra de segurança, que permite uma maior responsabilidade do lado do desenvolvedor e uma maior segurança do lado do cliente. Com isso, podemos assim, implementar uma URL customizada da seguinte forma: from django.contrib import admin from django.urls import path, HttpResponse def view(request): return HttpResponse('Voltando código') urlspatterns = [path('blog/', view)] Assim, quando criarmos essa função e essa url, vamos poder acessar essa url no nosso navegador, e ele vai prover o conteúdo disponibilizado pela função para que nós possamos usar ele para disponibilizar conteúdo para a requisição.3 Criando nossas primeiras aplicações Web 2.4 5 Criando apps com o manage.py Os apps dentro do Django, são como uma subcategoria do projeto, que podem ser criadas várias dentro de um mesmo projeto, e isso é muito útil quando temos por exemplo, um projeto que deve ter duas

formas de aplicação com- pletamente diferentes, como um e-commerce que deve ter a parte de compra e venda de produtos feita pelos usuários, e a parte administrativa das con- tas que não tem acesso as informações de compra e venda (necessariamente diretamente), mas que é usada para ver as estatıticas de acesso e uso do site. Tendo esse exemplo em mente, visa-se a necessidade e a utilidade do uso de apps dentro do Django, para criarmos o nosso, devemos usar o comando:
$python3manage.pystartappnome_do_appQuandocriarmosnossoapp, teremosnumanovapastaquecomelater-$
$emosassimumanovaformadecriarnossosite, eorganizareledamaneiramaisfácil.3Criandonossasprimeir$
$mitirumfuncionamentomaiscleanemelhordanossaaplicação, alémdofatodequenoarquivodeurls, organiz$
$projeto/urls.pyfromdjango.contribimportadminfromdjango.urlsimportpath, includeurlpatterns =$
$[path('/', include('home.urls')), path('blog/', include('blog.urls')), 3CriandonossasprimeirasaplicaçõesW$
$returnHttpResponse("retornandodapáginainicialdoblog")blog/urls.pyfromdjango.urlsimportpathfrom$
$[path('', views.blog),]3.2Renderizandotemplates, html, econfigurandooINSTALLEDAPPSAinternette$
$gramação.Comisso, paraservimosessestiposdearquivos, precisamoses-pecificaraexistência, eaentregad$
$fromdjango.shortcutsimportrenderdefhome(request) : returnrender(request,' nome-$
$do - arquivo.html')$