PABX IP Documentation

Versão 0.1

Fabio Hiroki

05/03/2012

Sumário

CAPÍTULO 1

Introdução

Esta documentação tem o objetivo de servir de base para auxiliar o entendimento do código-fonte do projeto PABX-IP, facilitando as futuras extensões de funcionalidades a serem desenvolvidas.

CAPÍTULO 2

Recomendações

Antes de começar, é importante que o leitor tenha um conhecimento geral de programação orientada a objetos, banco de dados e desenvolvimento web. É indispensável um conhecimento prévio no framework Django.

Conteúdo

3.1 Para começar

3.1.1 1. Introdução

Nessa seção será mostrado as ferramentas que são pré-requisitos para desenvolvimento do projeto. Também mostraremos como instalá-las e configurá-las corretamente, para que o desenvolvedor já consiga ao menos rodar o PABX-IP em sua máquina local.

3.1.2 2. Ambiente de Desenvolvimento

- Sistema Operacional (recomendável): Ubuntu
- Python 2.7 (a versão 3 é incompatível com outras bibliotecas)
- Framework Django + Pinax
- Banco de dados SQLite

3.1.3 3. Instalação do ambiente de desenvolvimento

Esse tutorial foi feito no Ubuntu 11.10, usando o repositório git onde o código estava sendo versionado na época em que essa documentação foi feita. Caso o desenvolvedor já possua o código fonte, pule a parte 1.

3.1. Instalação do git e clone do repositório:

```
$ sudo apt-get install git
```

```
$ git clone https://github.com/fabiothiroki/pabx_ip.git
```

3.2. Instalação e ativação do virtualenv:

O virtualenv é uma ferramenta de criação de ambientes python isolados. Isso visa facilitar o deploy da aplicação.

Toda vez que o desenvolvedor quiser rodar o servidor local de desenvolvimento do django é necessário ativar esse ambiente, pois é nele que estarão instalados o Pinax e as bibliotecas python auxiliares.

```
$ sudo apt-get install python-pip

$ sudo pip install virtualenv

$ virtualenv pabx-env

$ source pabx-env/bin/activate
```

3.3. Instalação do Pinax e outros apps:

Pinax é uma plataforma baseada no Django que contém diversos apps pré-instalados.

Para mais informações consulte: http://pinaxproject.com/

```
(pabx-env)$ sudo pip install Pinax django_compressor
(pabx-env)$ sudo pip install django_debug_toolbar
(pabx-env)$ sudo pip install django_compressor
(pabx-env)$ sudo pip install django_staticfiles
(pabx-env)$ sudo pip install pinax_theme_bootstrap
```

3.4. Instalação do Django:

Django é o framework web utilizado no projeto.

Para mais informações consulte: http://djangoproject.com/

```
(pabx-env)$ sudo pip install Django
```

3.5. Instalação do Django South:

O South é utilizado para implementar o controle da estrutura e migração do banco de dados. Seus arquivos estão na pasta 'south', no diretório raiz do projeto.

Para mais informações consulte: http://south.aeracode.org/

```
(pabx-env)$ sudo pip install south
```

3.6. Conclusão:

Por fim utilize o seguinte comando no diretório raiz do projeto para ligar o servidor de desenvolvimento:

```
$ python manage.py runserver
```

A seguinte mensagem deverá ser retornada no terminal em caso de sucesso:

```
Validating models...

0 errors found
Django version 1.3.1, using settings 'pabx_ip.settings'
Development server is running at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Entre com o endereço http://127.0.0.1:8000/ no seu navegador para acessar a interface web do projeto.

O banco de dados padrão do projeto vem com o usuário super-admin com login *root* e senha *1234*.

3.2 Visão geral do código

Para que o desenvolvedor não fique confuso com a quantidade de arquivos do projeto, será listado aqui os apps desenvolvidos de fato (cada app corresponde a uma pasta na raiz do projeto):

- **Accounts**
- **Groups**
- **Skypelist**
- **Smtp**

Além dos arquivos *urls.py* e *settings.py* nenhum dos outros arquivos foi codificado de fato anteriormente, mas foi gerado automaticamente pelo framework e suas ferramentas.

O código desenvolvido no projeto está seguindo a estrutura de apps do Django, assim como a documentação relativa a essa parte.

Cada App documentado contém seu arquivos numa pasta de mesmo nome na raiz principal do projeto. Assim, a estrutura da d

- Formulários: relativo ao arquivo *forms.py*
- Modelos: relativo ao arquivo *models.py*
- Views: relativo ao arquivo *views.py*
- Decorators: relativo ao arquivo *decorators.py* (opcional)
- Templates: relativo aos arquivos dentro da pasta "templates" (opcional)

Eventualmente algum App possa precisar de arquivos templates adicionais que estarão contidos na pasta "templates" dentro da pasta do App. Além desses arquivos templates, os outros templates padrão estarão contidos na pasta "templates" dentro da pasta raiz.

3.3 App Accounts

Este app é responsável por:

- Definir privilégios de administrador a usuários e limitar acesso a usuários comuns
- Cadastro, edição, listagem e remoção de usuários
- Autenticação e recuperação de senha

3.3.1 Modelos (arquivo models.py)

No arquivo models.py estão as classes definidas para armazenar informações extras do usuário. Podemos observar que a classe UserProfile tem uma chave estrangeira na classe User, que é a classe padrão para usuários do Django. Desta maneira, podemos extender os atributos da classe User sem alterar a estrutura de usuário do Django, bastando apenas fazer essa pequena extensão.

Os atributos definidos na classe UserProfile estão detalhados a seguir:

class accounts.models.**UserProfile**

- **profile**: Chave estrangeira que associa um UserProfile a um Usuário. Pode haver apenas um UserProfile por User.
- **ramal**: Ramal associado a um usuário. Esse número é utilizado para fazer ligações.
- **passw**: É a cópia da senha do usuário salva encriptadamente. É utilizada para recuperação de senha.
- **admin**: Indica se o usuário possui o privilégio de administrador.
- **group**: Indica o grupo o qual o usuário pertence.

3.3.2 Formulários (arquivo forms.py)

3.4 Bibliotecas externas

3.4.1 1. Introdução

Nessa seção listaremos as bibliotecas auxiliares ao projeto, que já vem previamente instaladas e configuradas. O desenvolvedor deve ter conhecimento delas caso precise utilizá-las futuramente.

Cada biblioteca utilizada possui também uma documentação própria, que pode elucidar dúvidas mais específicas.

3.4.2 2. Twitter Bootstrap

http://twitter.github.com/bootstrap/

Utilizado como base da identidade visual do projeto. Embora o Pinax já venha com um template padrão usando o Twitter Bootstrap, foi preferível começar um novo template para fins de customização.

3.4.3 3. jQuery

http://jquery.com/

É o framework Javascript utilizado no projeto.

3.5 Futuras implementações

Índice de Módulos do Python

a

Índice