1. Gerar Certificados Locais com OpenSSL

Primeiro, crie um certificado e uma chave privada com o OpenSSL para testar HTTPS localmente.

No terminal, execute:

```
bash
# Gerar chave privada
openssl genrsa -out local_private.key 2048

# Gerar certificado usando a chave privada
openssl req -new -x509 -key local_private.key -out
local_certificate.crt -days 365
```

Isso criará dois arquivos: `local_private.key` (chave privada) e `local_certificate.crt` (certificado), que usaremos para configurar o HTTPS.

2. Configurar Middleware para Forçar HTTPS nas Rotas Específicas

No seu projeto Django, crie uma middleware para redirecionar as rotas `/login` e `/cadastro` para HTTPS.

1. Crie um arquivo `force_https.py` dentro de uma pasta chamada `middleware` (ou outra pasta de sua escolha) no seu projeto:

```
# middleware/force_https.py

from django.http import HttpResponseRedirect

class ForceHTTPSMiddleware:
    def __init__(self, get_response):
        self.get_response = get_response

    def __call__(self, request):
        # Forçar HTTPS nas rotas de login e cadastro
        if request.path in ["/login/", "/cadastro/"] and
not request.is_secure():
            return HttpResponseRedirect(f"https://
{request.get_host()}{request.path}")
        return self.get_response(request)

```

14. Adicione a middleware no seu `settings.py`:
    python

```
MIDDLEWARE = [
```

```
    # Outras middlewares...
    'seu_projeto.middleware.force_https.ForceHTTPSMiddleware',
]
```

3. Rodar o Servidor Django com HTTPS Localmente

Agora, para rodar o Django localmente com HTTPS nas rotas necessárias, você precisará de um pacote chamado `django-extensions`, que fornece o comando `runserver_plus` com suporte a certificados.

1. Instale o `django-extensions`:

   ```
   pip install django-extensions
   ```

2.

3. Adicione `django-extensions` ao seu `INSTALLED_APPS` em `settings.py`:
   `python`

   ```
   INSTALLED_APPS = [
       # Outros apps...
       'django_extensions',
   ]
   ```

4.

5. Execute o servidor com o comando `runserver_plus`, usando os certificados que você gerou:
   `bash`

   ```
   python manage.py runserver_plus --cert-file local_certificate.crt --key-file local_private.key
   ```

6.

7. Agora, ao acessar as páginas de **login** e **cadastro**, a middleware redirecionará automaticamente para HTTPS.

4. Testar o Redirecionamento

- Acesse `http://127.0.0.1:8000/login` e `http://127.0.0.1:8000/cadastro`. O Django deve redirecionar para `https://127.0.0.1:8000/login` e `https://127.0.0.1:8000/cadastro`.
- Outras páginas continuarão acessíveis apenas por HTTP, sem redirecionamento para HTTPS.

Essa configuração funcionará localmente e permitirá que você teste a implementação de HTTPS somente nas rotas sensíveis, sem afetar o restante do projeto.