

PROGRAMAÇÃO WEB II

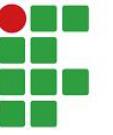
Curso Técnico Integrado em Informática
Lucas Sampaio Leite



Deploy

- Deploy de um sistema web é o processo de colocar uma aplicação pronta para funcionar em um servidor, tornando-a acessível para os usuários pela internet.
- É a etapa em que o sistema sai do ambiente de desenvolvimento e passa a rodar em um ambiente de produção.





Deploy

- Etapas envolvidas em um deploy:
 - Subir os arquivos para um servidor
 - Configurar o servidor
 - Configurar o banco de dados
 - Definir ambiente de produção
 - Executar a aplicação

Deploy

- Onde o deploy pode ser feito:
 - Servidores próprios (VPS: DigitalOcean, Linode, AWS EC2).
 - Hospedagens compartilhadas.
 - Plataformas de deploy automático, exemplo:
 - Heroku
 - Railway
 - Render
 - Vercel (mais comum para front-end)
 - Netlify
 - AWS / Azure / GCP

Plataforma Render

- O Render é uma plataforma moderna de hospedagem em nuvem (cloud hosting) que permite fazer deploy de aplicações web, APIs, bancos de dados e serviços estáticos de forma simples.
- O Render aceita aplicações desenvolvidas com Flask, permitindo fazer deploy de APIs e sistemas web em poucos passos.





Deploy

Debug your Render services in Claude Code and Cursor. [Try Render MCP >](#)

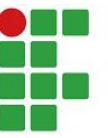
Render Product Pricing Customers Blog Docs Changelog Company Contact Sign In Get Started

Your fastest path to production

Build, deploy, and scale your apps with unparalleled ease – from your first user to your billionth

[Get Started for Free >](#) [Contact Sales >](#)

<https://render.com/>



Deploy

Lucas's workspace Projects

Search ⌘ K

+ New Upgrade ⚡ L

Create a new Service

1 Choose service > 2 Configure > 3 Deploy Which to use?

Static Sites

Static content served over a global CDN. Ideal for frontend, blogs, and content sites.

New Static Site →

Web Services

Dynamic web app. Ideal for full-stack apps, API servers, and mobile backends.

New Web Service →

Private Services

Web app hosted on a private network, accessible only from your other Render services.

New Private Service →

Background Workers

Long-lived services that process async tasks, usually from a job queue.

New Worker →

Cron Jobs

Short-lived tasks that run on a periodic schedule.

New Cron Job →

Postgres

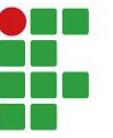
Relational data storage. Supports point-in-time recovery, read replicas, and high availability.

New Postgres →

KeyValue

Managed Redis®-compatible storage. Ideal for use as a shared cache, message broker, or job queue.

New Key Value Instance →



Deploy

Lucas's workspace Projects

Search ⌘ K

+ New Upgrade ⚡ L

Create a new Service

1 Choose service > 2 Configure > 3 Deploy Which to use?

Static Sites

Static content served over a global CDN. Ideal for frontend, blogs, and content sites.

New Static Site →

Web Services

Dynamic web app. Ideal for full-stack apps, API servers, and mobile backends.

New Web Service →

Private Services

Web app hosted on a private network, accessible only from your other Render services.

New Private Service →

Background Workers

Long-lived services that process async tasks, usually from a job queue.

New Worker →

Cron Jobs

Short-lived tasks that run on a periodic schedule.

New Cron Job →

Postgres

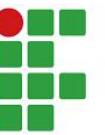
Relational data storage. Supports point-in-time recovery, read replicas, and high availability.

New Postgres → ←

KeyValue

Managed Redis®-compatible storage. Ideal for use as a shared cache, message broker, or job queue.

New Key Value Instance →



Deploy

Lucas's workspace ▾ New Postgres ▾

Search ⌘ K + New ⚡ Upgrade ⓧ L

Configure and deploy your new Database

1 Choose service > 2 Configure > 3 Deploy Need help? Docs ↗

Name
A unique name for your Postgres instance.
 

Database Optional
The Postgres dbname
 

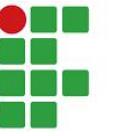
User Optional
 

Region
Your services in the same region can communicate over a private network.
 

PostgreSQL Version
 

Datadog API Key Optional
The API key to use for sending metrics to Datadog. Setting this will enable Datadog monitoring.

Datadog Region Optional
The region key to use for sending metrics to Datadog.
 



Deploy

Plan Options

Instance Type

Set your database's RAM and CPU. You can change your instance type later.



Free

For testing out Render Postgres

Free

\$0 / month

256 MB (RAM)

0.1 CPU

1 GB (Storage)

Basic

Reliability and performance for hobby projects. Starting at \$6 / month plus storage.

Pro

Perfect for production use cases at any scale. Starting at \$55 / month plus storage.

Accelerated

Memory-optimized plans offer significant performance improvements. Starting at \$160 / month plus storage.

Need a [custom instance type](#)? We support up to 1024 GB RAM, 128 CPUs, and 16 TB storage.

Deploy

Storage

Your database's capacity, in GB. You can increase storage later, but you can't decrease it.

Specify 1 GB or any multiple of 5 GB.

Storage Autoscaling

When your database is 90% full, automatically increase its storage by 50% (rounded up to the nearest 5 GB).

Max one increase per 12 hours. [Learn more](#)

High Availability

Run a standby instance of your database and automatically fail over to it if the primary encounters an issue. [Learn more](#)

1

GB

\$0 / month



Disabled

\$0 / month



Disabled

ⓘ Only available for **Pro** instances and higher.

Monthly Total

Database instances are billed and prorated by the second.

Instance: **Free**

\$0 / month

Total

\$0 / month

Create Database





Deploy

Lucas's workspace

My project / Production / flask-blog-pg

Search K + New ⚡ Upgrade L

← Environment flask-blog-pg POSTGRESQL Free Upgrade your instance → View docs Connect

Service ID: dpg-d4bo8bndiees73dc8cq0-a

Info

MONITOR

Logs Metrics

MANAGE

Apps Recovery

Info

Your database will expire on December 14, 2025. The database will be deleted unless you [upgrade to a paid instance type](#).

General

Name flask-blog-pg A unique name for your database. Edit

Creating since a few seconds ago

Status Creating

PostgreSQL Version 18

Region Oregon (US West)

Changelog

Invite a friend

Contact support

Render Status



Deploy

Lucas's workspace My project Production flask-blog-pg

Search + New ⚡ Upgrade ?

← Environment flask-blog-pg POSTGRESQL Upgrade your instance → View docs Connect

Service ID: dpg-d4bo8bndiees73dc8cq0-a

Info

MONITOR

Logs

Metrics

MANAGE

Apps

Recovery

Info

Your database will expire on December 14, 2025. The database will be deleted unless you [upgrade to a paid instance type](#).

General

Name: flask-blog-pg

A unique name for your database.

Created: 3 minutes ago

Status:

PostgreSQL Version: 18

Region: Oregon (US West)

Changelog

Invite a friend

Contact support

Render Status



Deploy

Connections

Hostname
An internal hostname used by your Render services.
dpg-d4csdpjjchc73dk046g-a

Port
5432

Database
blogdb_vudo

Username
blogadmin

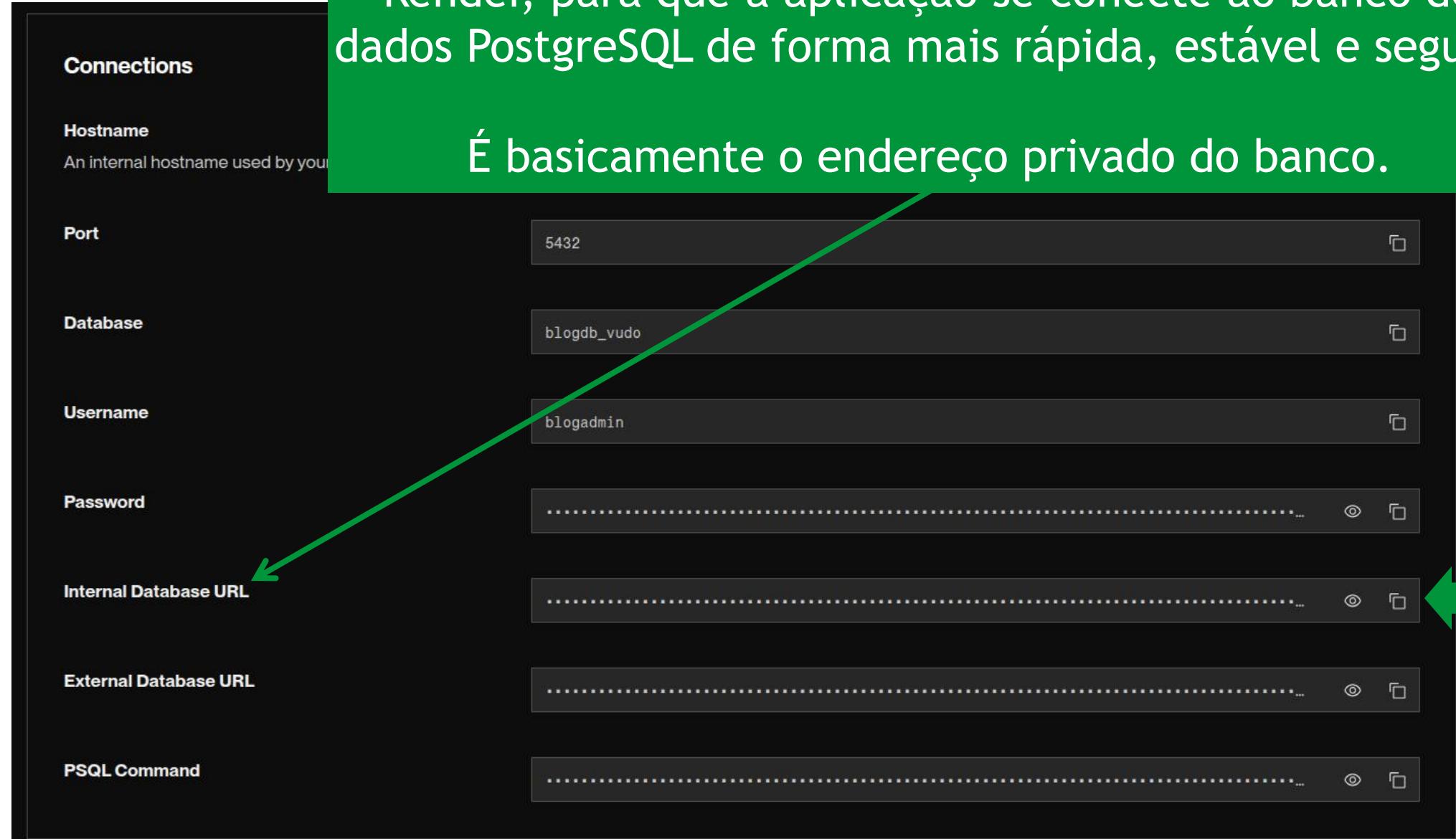
Password
.....

Internal Database URL
..... 

External Database URL
.....

PSQL Command
.....

Deploy



A Internal Database URL do Render é um endereço interno, usado somente dentro da infraestrutura do Render, para que a aplicação se conecte ao banco de dados PostgreSQL de forma mais rápida, estável e segura.

É basicamente o endereço privado do banco.

Hostname: An internal hostname used by your application to connect to the database.

Port: 5432

Database: blogdb_vudo

Username: blogadmin

Password: (redacted)

Internal Database URL: (redacted) (This field is highlighted with a green arrow pointing to it.)

External Database URL: (redacted)

PSQL Command: (redacted)



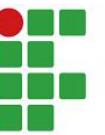
Deploy

The screenshot shows a web-based interface for managing cloud services. At the top, there's a navigation bar with 'Lucas's workspace', 'My project / Production / flask-blog-pg'. On the right side, there's a search bar, a '+ New' button, an 'Upgrade' button, and a help icon. A green arrow points from the 'INSTITUTO FEDERAL Baiano' logo to the '+ New' button.

The main area displays a 'POSTGRESQL' service named 'flask-blog-pg' which is 'Free'. It shows a 'Service ID: dpg-d4bo8bndiees73dc8cq0-a'. Below this, there's an 'Info' section with a warning message: 'Your database will expire on December 14, 2025. The database will be deleted unless you upgrade to a paid instance type.' A green arrow points from this message to the 'Web Service' option in the sidebar menu.

The sidebar on the left includes sections for 'Environment', 'flask-blog-pg' (selected), 'Info' (highlighted with a purple background), 'MONITOR', 'Logs', 'Metrics', 'MANAGE', 'Apps', and 'Recovery'. The 'Info' section contains a 'General' tab with a 'Name' field set to 'flask-blog-pg' and an 'Edit' button. The 'Created' field shows '8 minutes ago'.

The sidebar on the right lists service types: 'Static Site', 'Web Service' (highlighted with a purple background and has a green arrow pointing to it), 'Private Service', 'Background Worker', 'Cron Job', 'Postgres', 'Key Value', 'Project', and 'Blueprint'. There's also a 'Connect' button.



Deploy

L Lucas's workspace ⚙️ New Web Service ⚙️ Q Search ⚙️ K + New ⚙️ Upgrade ⚙️ L

New Web Service

Source Code

Git Provider Public Git Repository Existing Image

Connect Git provider

Connect your Git provider to deploy from your existing repositories.

GitHub GitLab Bitbucket



Name

A unique name for your web service.

example-service-name

Project Optional

Add this web service to a project once it's created.

Select a project... / Select an environment...

Language

Choose the runtime environment for this service.

Node



Deploy

L Lucas's workspace ⚙️ New Web Service ⚙️ Q Search ⚙️ K + New ⚙️ Upgrade ⚙️ L

New Web Service

Source Code

Git Provider Public Git Repository Existing Image

Connect Git provider

Connect your Git provider to deploy from your existing repositories.

GitHub GitLab Bitbucket

Name

A unique name for your web service.

example-service-name

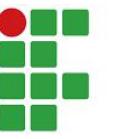
Project Optional

Add this web service to a project once it's created.

Language

Choose the runtime environment for this service.

O Git Provider é o local onde o código está hospedado e de onde o Render vai buscar o repositório para fazer o deploy da aplicação.



Deploy

github.com/login/oauth/authorize?client_id=lv1.558507eb9b186639&redirect_uri=https%3A%... 🔍 | 🌐 | ⚙️ | 🔍 | + New | ⚡ Upgrade | ? | L

Render by **Render** would like permission to:

- Verify your GitHub identity (lucassampaioleite)
- Know which resources you can access
- Act on your behalf
 - Learn more

Resources on your account

- Email addresses (read)
View your email addresses

Learn more about Render

Cancel Authorize Render ←

Authorizing will redirect to <https://dash.render.com/auth/github>

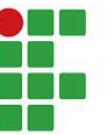
repository Existing Image

Connect Git provider

Connect your Git provider to deploy from your existing repositories.

GitHub GitLab Bitbucket

Select an environment...



Deploy

github.com/apps/render/installations/new/permissions?target_id=13689528

Search + New Upgrade ? L

Install Render

Install on your personal account lucassampaioleite 

for these repositories:

All repositories 

This applies to all current and future repositories owned by the resource owner. Also includes public repositories (read-only).

Only select repositories

Select at least one repository. Also includes public repositories (read-only).

with these permissions:

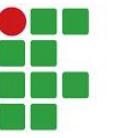
repository Existing Image

Connect Git provider

Connect your Git provider to deploy from your existing repositories.

 GitHub  GitLab  Bitbucket

Select an environment... 



Deploy

L Lucas's workspace ▾

New Web Service ▾

Search ⌘ K

+ New ⌘ U Upgrade ⌘ ⓘ L

New Web Service

Source Code

Git Provider Public Git Repository Existing Image

Search

Credentials ⓘ

lucassampaioleite /

Name example-service-name

Project Optional Select a project... / Select an environment...

Add this web service to a project once it's created.

A screenshot of a web application interface for deploying a new web service. At the top, there are navigation links for 'Lucas's workspace' and 'New Web Service'. On the right, there are buttons for 'Search' (with a keyboard shortcut ⌘ K), '+ New' (with ⌘ U), 'Upgrade' (with ⌘ U), and a help icon (ⓘ). Below the header, the title 'New Web Service' is displayed. Underneath, there are tabs for 'Source Code' (selected), 'Git Provider' (highlighted with a green arrow pointing to it), 'Public Git Repository', and 'Existing Image'. A search bar with the placeholder 'Search' is present. To the right of the search bar is a 'Credentials' dropdown with an 'ⓘ' icon. Below the tabs, a list of repository names is shown, all starting with 'lucassampaioleite /'. The 'Name' field is populated with 'example-service-name'. The 'Project' section includes a dropdown labeled 'Select a project...' and a dropdown labeled 'Select an environment...'. A note at the bottom says 'Add this web service to a project once it's created.'

Deploy

1 General

Owner * lucassampaioleite / Repository name *

flask-blog 

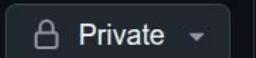
flask-blog is available.

Great repository names are short and memorable. How about [fuzzy-garbanzo](#)?

Description

0 / 350 characters

2 Configuration

Choose visibility * 

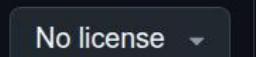
Choose who can see and commit to this repository

Add README 

READMEs can be used as longer descriptions. [About READMEs](#)

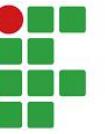
Add .gitignore 

.gitignore tells git which files not to track. [About ignoring files](#)

Add license 

Licenses explain how others can use your code. [About licenses](#)

Create repository 



Deploy

lucassampaioleite / flask-blog

Type to search

Code Issues Pull requests Actions Projects Security Insights Settings

Watch 0 Fork 0 Star 0

flask-blog Private

Set up GitHub Copilot
Use GitHub's AI pair programmer to autocomplete suggestions as you code.
[Get started with GitHub Copilot](#)

Add collaborators to this repository
Search for people using their GitHub username or email address.
[Invite collaborators](#)

Quick setup — if you've done this kind of thing before

HTTPS SSH <git@github.com:lucassampaioleite/flask-blog.git>

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# flask-blog" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:lucassampaioleite/flask-blog.git
git push -u origin main
```



Deploy

Lucas's workspace ▾ New Web Service ▾ Search ⌘ K + New ⚡ Upgrade ? L

New Web Service

Source Code

Git Provider Public Git Repository Existing Image

Search Credentials (1) ▾

View repo ↗

lucassampaioleite / flask-blog 2m ago ←

lucassampaioleite /

lucassampaioleite /

lucassampaioleite /

lucassampaioleite /

lucassampaioleite /

lucassampaioleite /

lucassampaioleite / concourse_it Aug 30, 2021

Name

A unique name for your web service.

example-service-name

Project Optional

Add this web service to a project once it's created.

Select a project... / Select an environment...



Deploy

Lucas's workspace ▾ New Web Service ▾ Q Search ⌘ K + New ⚡ Upgrade ? L

New Web Service

Source Code lucassampaioleite / flask-blog • 2m ago Edit

Name flask-blog

Project Optional Select a project... / Select an environment...

Language Python 3 (highlighted with a green arrow)

Branch main

Region Oregon (US West) 1 existing service Deploy in a new region +

Root Directory Optional e.g. src

If set, Render runs commands from this directory instead of the repository root. Additionally, code changes outside of this directory do not trigger an auto-deploy. Most commonly used with a monorepo.

Deploy

Build Command

Render runs this command to build your app before each deploy.

```
$ pip install -r requirements.txt
```



Start Command

Render runs this command to start your app with each deploy.

```
$ gunicorn src.wsgi:app
```



Instance Type



For hobby projects

Free	512 MB (RAM)
\$0 / month	0.1 CPU

⚠ Upgrade to enable more features

Free instances spin down after periods of inactivity. They do not support SSH access, scaling, one-off jobs, or persistent disks. Select any paid instance type to enable these features.

For professional use

For more power and to get the most out of Render, we recommend using one of our paid instance types. All paid instances support:

- Zero Downtime
- SSH Access
- Scaling
- One-off jobs
- Support for persistent disks

Starter	512 MB (RAM)
\$7 / month	0.5 CPU

Standard	2 GB (RAM)
\$25 / month	1 CPU

Pro	4 GB (RAM)
\$85 / month	2 CPU

Pro Plus	8 GB (RAM)
\$175 / month	4 CPU

Pro Max	16 GB (RAM)
\$225 / month	4 CPU

Pro Ultra	32 GB (RAM)
\$450 / month	8 CPU

Need a custom instance type? We support up to 512 GB RAM and 64 CPUs.

Deploy

O gunicorn é um servidor web de produção usado para executar aplicações Python, como Flask ou Django, em ambientes reais de produção.

Build Command

Render runs this command to build your app before each deploy.

Start Command

Render runs this command to start your app with each deploy.

Instance Type

For hobby projects

```
$ pip install -r requirements.txt
```

```
$ gunicorn src.wsgi:app
```

Free
\$0 / month

For professional use

For more power and to get the most out of Render, we recommend using one of our paid instance types. All paid instances support:

- Zero Downtime
- SSH Access
- Scaling
- One-off jobs
- Support for persistent disks

O wsgi.py é necessário porque ele funciona como o ponto de entrada padrão para servidores web Python

Starter \$7 / month	512 MB (RAM) 0.5 CPU	Standard \$25 / month	2 GB (RAM) 1 CPU
Pro \$85 / month	4 GB (RAM) 2 CPU	Pro Plus \$175 / month	8 GB (RAM) 4 CPU
Pro Max \$225 / month	16 GB (RAM) 4 CPU	Pro Ultra \$450 / month	32 GB (RAM) 8 CPU

Deploy

Build Command
Render runs this command to build your app before each deploy.

Start Command
Render runs this command to start your app with each deploy.

Instance Type

For hobby projects	Free \$0 / month	Starter \$7 / month	Pro \$85 / month	Pro Max \$225 / month	16 GB (RAM) 4 CPU	Pro Ultra \$450 / month	32 GB (RAM) 8 CPU
--------------------	---------------------	------------------------	---------------------	--------------------------	----------------------	----------------------------	----------------------

O build command (comando de build) é o comando que o Render executa para preparar o projeto antes de rodá-lo.

Ele é essencial para: instalar dependências, compilar arquivos, gerar artefatos necessários para a execução e preparar o ambiente da aplicação

\$ pip install -r requirements.txt

\$ gunicorn src.wsgi:app

For hobby projects

Free
\$0 / month

Starter
\$7 / month

Pro
\$85 / month

Pro Max
\$225 / month

16 GB (RAM)
4 CPU

Pro Ultra
\$450 / month

32 GB (RAM)
8 CPU

For professional use

For more power and to get the most out of Render, we recommend using one of our paid instance types. All paid instances support:

- Zero Downtime
- SSH Access
- Scaling
- One-off jobs
- Support for persistent disks

Need a custom instance type? We support up to 512 GB RAM and 64 CPUs.



Deploy

L Lucas's workspace ▾ New Web Service ▾ Search K + New ⚡ Upgrade ? L

- Zero Downtime
- SSH Access
- Scaling
- One-off jobs
- Support for persistent disks

Pro \$85 / month	4 GB (RAM) 2 CPU
Pro Max \$225 / month	16 GB (RAM) 4 CPU
Pro Plus \$175 / month	8 GB (RAM) 4 CPU
Pro Ultra \$450 / month	32 GB (RAM) 8 CPU

Need a custom instance type? We support up to 512 GB RAM and 64 CPUs.

Environment Variables

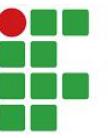
Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more](#).

DATABASE_URL 

+ Add Environment Variable Add from .env

> Advanced

Deploy Web Service 



Deploy

Lucas's workspace

flask-blog

WEB SERVICE

flask-blog Python 3 Free Upgrade your instance →

Service ID: srv-d4bp2dogjchc73ctsh5g

lucassampaiolete / flask-blog main

<https://flask-blog-iokt.onrender.com>

Connect Manual Deploy

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more. [Upgrade now](#)

Your service has not been deployed because the GitHub repository is empty. Make a commit before retrying.

Filter events 31

No events from the past 90 days match the selected filters.

Events

Settings

MONITOR

Logs

Metrics

MANAGE

Environment

Shell

Scaling

Previews

Disks

Jobs

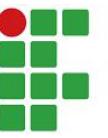
Deploy

...or create a new repository on the command line

```
echo "# flask-blog" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:lcassampaioleite/flask-blog.git
git push -u origin main
```



```
Initialized empty Git repository in /home/lucas/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog/.git/
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ git add README.md
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ git commit -m "first commit"
[master (root-commit) 770ec88] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ git branch -M main
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ git remote add origin git@github.com:lcassampaioleite/flask-blog.git
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 226 bytes | 226.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:lcassampaioleite/flask-blog.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
```



Deploy

lucassampaioleite / flask-blog

Type / to search

Code Issues Pull requests Actions Projects Security Insights Settings

flask-blog Private Watch 0 Fork 0 Star 0

main 1 Branch 0 Tags Go to file Add file Code

lucassampaioleite first commit 770ec88 · 6 minutes ago 1 Commit

README.md first commit 6 minutes ago

README

flask-blog

About

No description, website, or topics provided.

Readme Activity 0 stars 0 watching 0 forks

Releases

No releases published Create a new release

Packages

No packages published Publish your first package

Deploy

```
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ git status
On branch main
Your branch is up to date with 'origin/main'.
```

Untracked files:

(use "git add <file>..." to include in what will be committed)

.gitignore
Pipfile
Pipfile.lock
→ __pycache__/
→ application.py
controllers/
extensions/
→ instance/
models/
views/

nothing added to commit but untracked files present (use "git add" to track)

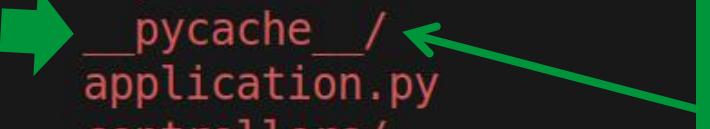
```
○ lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ █
```

Deploy

```
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ git status
On branch main
Your branch is up to date with 'origin/main'.
```

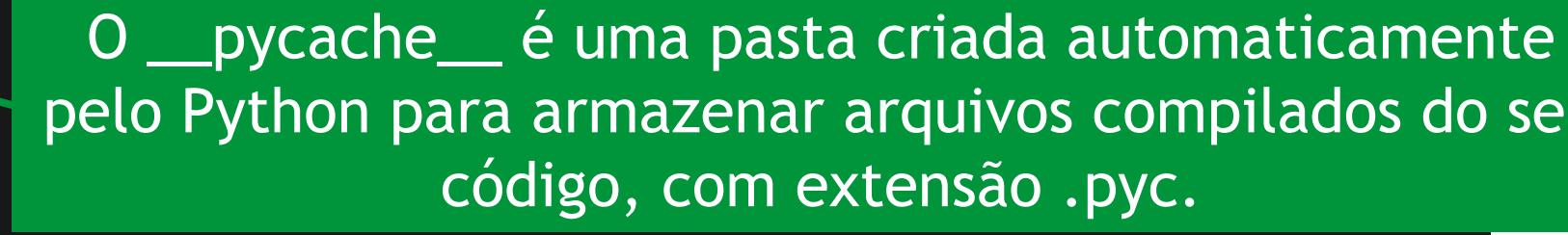
Untracked files:

(use "git add <file>..." to include in what will be committed)

.gitignore
Pipfile
Pipfile.lock
 __pycache__/
application.py
controllers/
extensions/
instance/
models/
views/

nothing added to commit but untracked files present (use "git add" to track)

```
○ lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ █
```

 O `__pycache__` é uma pasta criada automaticamente pelo Python para armazenar arquivos compilados do seu código, com extensão `.pyc`.

Deploy

- O `.gitignore` é um arquivo especial usado pelo Git para dizer quais arquivos ou pastas devem ser ignorados – ou seja, não serão adicionados, versionados ou enviados para o repositório.
- `gitignore.io` é uma ferramenta online gratuita que cria automaticamente arquivos `.gitignore` completos e atualizados para qualquer linguagem, framework ou IDE.

Deploy



gitignore.io

Crie arquivos .gitignore úteis para o seu projeto.

Criar

[Código-fonte](#) | [Documentação](#)

gitignore.io

Deploy



gitignore.io

Crie arquivos .gitignore úteis para o seu projeto.



[Código-fonte](#) | [Documentação](#)

Deploy

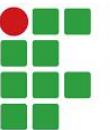
```
# Created by https://www.toptal.com/developers/gitignore/api/python,flask
# Edit at https://www.toptal.com/developers/gitignore?templates=python,flask

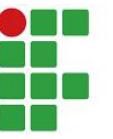
### Flask ####
instance/*
!instance/.gitignore
.webassets-cache
.env

### Flask.Python Stack ####
# Byte-compiled / optimized / DLL files
__pycache__
*.py[cod]
*$py.class

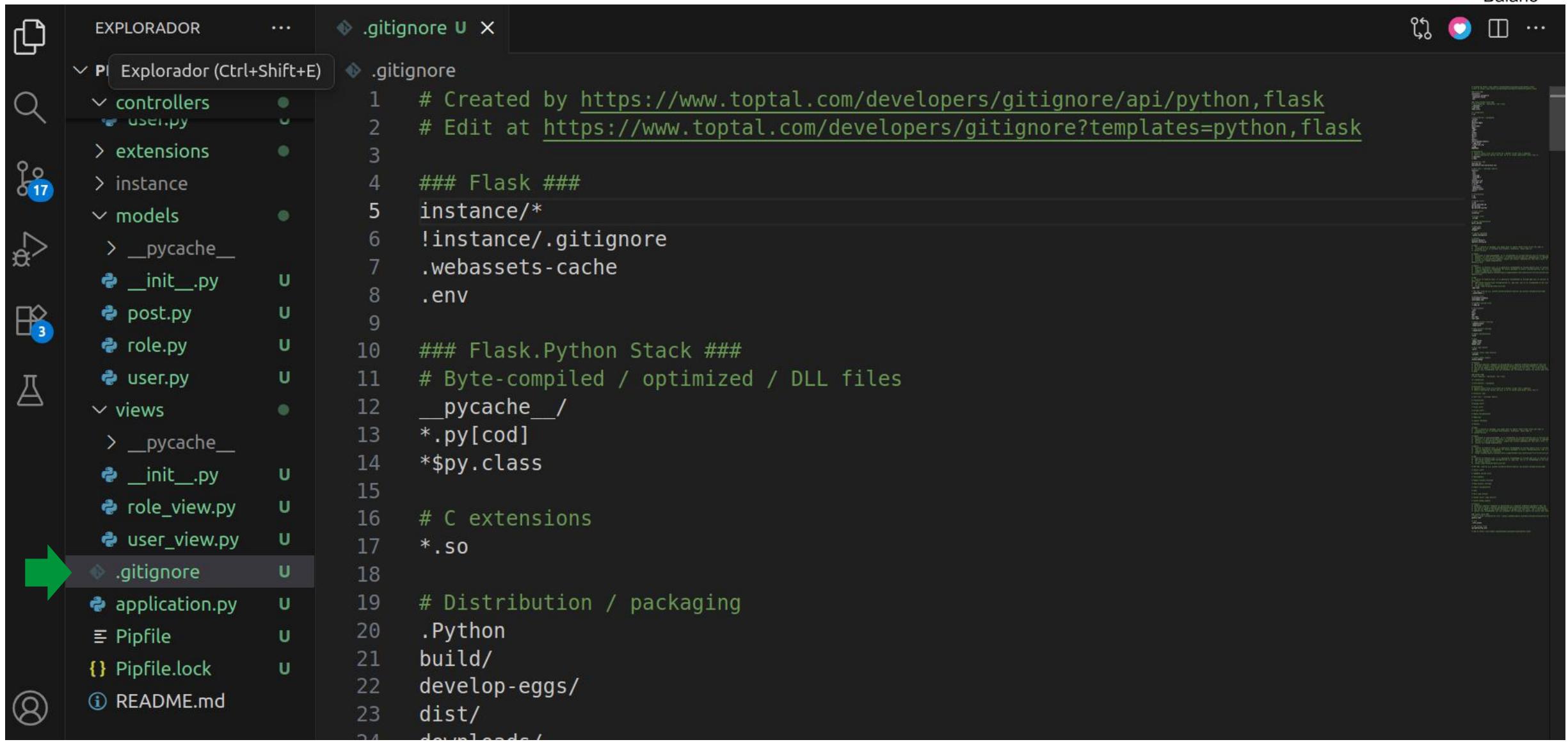
# C extensions
*.so

# Distribution / packaging
.Python
build/
develop-eggs/
dist/
downloads/
eggs/
.eggs/
lib/
lib64/
parts/
sdist/
var/
wheels/
share/python-wheels/
*.egg-info/
.installed.cfg
*.egg
MANIFEST
```





Deploy



```
EXPLORADOR ... .gitignore X
P Explorador (Ctrl+Shift+E) .gitignore
controllers user.py
> extensions
> instance
models __pycache__ __init__.py post.py role.py user.py
views __pycache__ __init__.py role_view.py user_view.py
.gitignore application.py Pipfile Pipfile.lock README.md

1 # Created by https://www.toptal.com/developers/gitignore/api/python,flask
2 # Edit at https://www.toptal.com/developers/gitignore?templates=python,flask
3
4 ### Flask ####
5 instance/*
6 !instance/.gitignore
7 .webassets-cache
8 .env
9
10 ### Flask.Python Stack ###
11 # Byte-compiled / optimized / DLL files
12 __pycache__/
13 *.py[cod]
14 *$py.class
15
16 # C extensions
17 *.so
18
19 # Distribution / packaging
20 .Python
21 build/
22 develop-eggs/
23 dist/
24 downloads/
```

Deploy

```
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    → .gitignore
      Pipfile
      Pipfile.lock
      application.py
      controllers/
      extensions/
      models/
      views/
```

Deploy

- O psycopg2-binary é o driver que permite que sua aplicação Python (Flask) se conecte a um banco PostgreSQL, que é o banco padrão oferecido pelo Render.
 - Em outras palavras, é o "conector" que permite ao Flask conversar com o PostgreSQL.
- O gunicorn é o servidor web recomendado para rodar aplicações Flask em produção.

Deploy

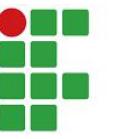
- pipenv install psycopg2-binary

```
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ pipenv install psycopg2-binary
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
Installing psycopg2-binary...
✓ Installation Succeeded
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
Installing dependencies from Pipfile.lock (659eb1)...
All dependencies are now up-to-date!
Upgrading psycopg2-binary in dependencies.
Building requirements...
Resolving dependencies...
✓ Success!
```

Deploy

- pipenv install gunicorn

```
● lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_Blog$ pipenv install gunicorn
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
Installing gunicorn...
✓ Installation Succeeded
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
Installing dependencies from Pipfile.lock (3c5272)...
All dependencies are now up-to-date!
Upgrading gunicorn in dependencies.
Building requirements...
Resolving dependencies...
✓ Success!
```



Deploy

The screenshot shows a terminal window with the following text:

```
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/we
Blog$
```

Three green arrows point from the left towards the terminal window, highlighting specific parts of the code and the terminal output.

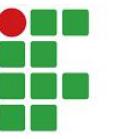
Code Editor View:

- The code editor shows the file `wsgi.py` with the following content:

```
1  from src.application import create_app
2
3  app = create_app()
```
- A green arrow points to the first line of code (`from src.application import create_app`).
- A green arrow points to the third line of code (`app = create_app()`).

Terminal View:

- The terminal tab is active at the bottom of the interface.
- The terminal window displays the command prompt: `(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/we
Blog$`.
- A green arrow points to the terminal window.



Deploy

```
EXPLORADOR ... wsgi.py U X
PR Explorador (Ctrl+Shift+E) src > wsgi.py > ...
    _pycache_
    application.cpython-37.pyc
> instance
    src
        _pycache_
        controllers
        extensions
        instance
        models
        views
    application.py U
    wsgi.py U
    .gitignore A
    Pipfile M
    Pipfile.lock M
    README.md

1     from src.application import create_app
2
3     app = create_app()
4
5 |
```

WSGI significa Web Server Gateway Interface.
É um padrão que define como servidores web se comunicam com aplicações Python.

```
PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL
> v TERMINAL
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/we
Blog$
```

Deploy

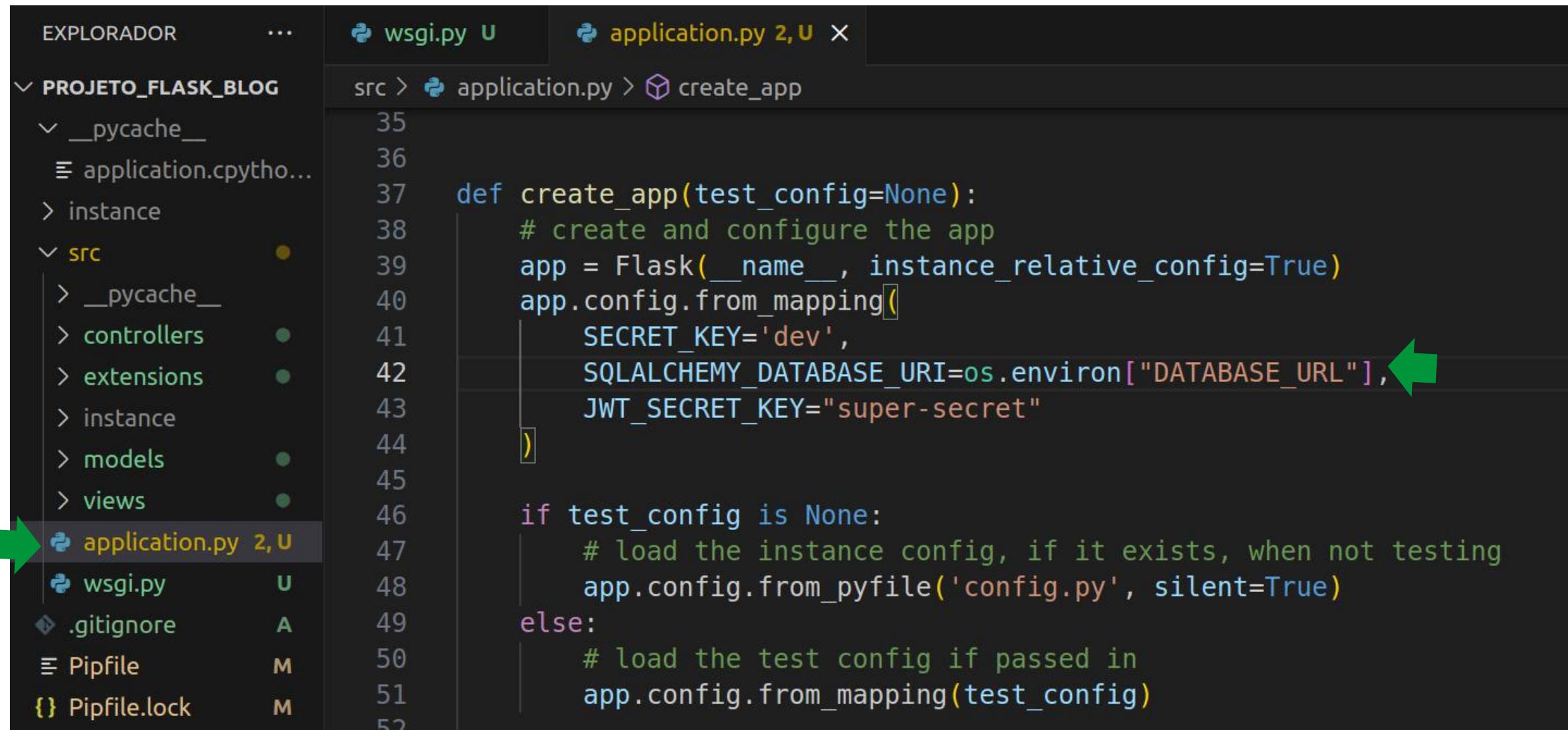
```
from src.extensions import db
from src.extensions import jwt
```

Corrigir erros de import:
src.models
src.controllers
src.views
src.extensions



```
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ flask --app src.application run --debug
 * Serving Flask app 'src.application'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 110-182-683
```

Deploy



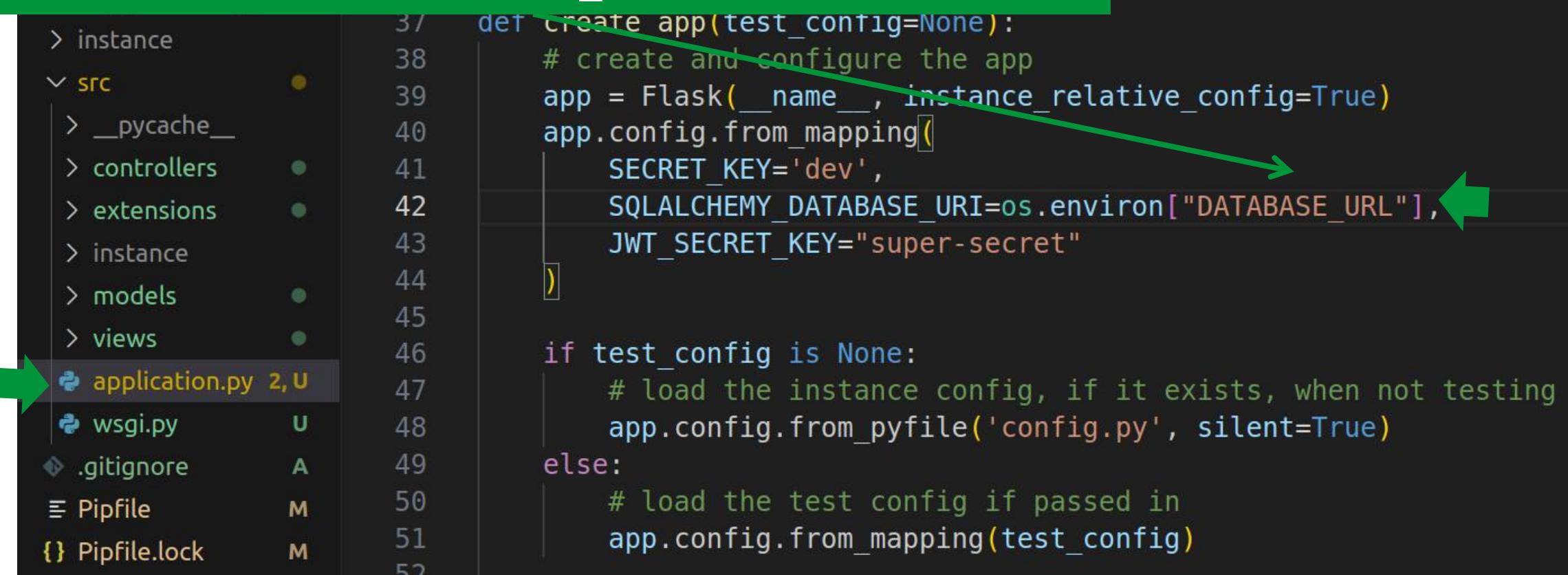
The screenshot shows a code editor interface with the following details:

- EXPLORADOR** (Explorer) sidebar on the left lists the project structure:
 - PROJETO_FLASK_BLOG
 - __pycache__
 - application.cpython-37.pyc
 - > instance
 - src
 - > __pycache__
 - > controllers
 - > extensions
 - > instance
 - > models
 - > views
 - application.py 2, U (highlighted)
 - wsgi.py U
 - .gitignore A
 - Pipfile M
 - { } Pipfile.lock M
- wsgi.py U** tab is selected in the top bar.
- application.py 2, U** tab is also present in the top bar.
- The main editor area displays the `application.py` code:

```
src > application.py > create_app
35
36
37 def create_app(test_config=None):
38     # create and configure the app
39     app = Flask(__name__, instance_relative_config=True)
40     app.config.from_mapping(
41         SECRET_KEY='dev',
42         SQLALCHEMY_DATABASE_URI=os.environ["DATABASE_URL"], ←
43         JWT_SECRET_KEY="super-secret"
44     )
45
46     if test_config is None:
47         # load the instance config, if it exists, when not testing
48         app.config.from_pyfile('config.py', silent=True)
49     else:
50         # load the test config if passed in
51         app.config.from_mapping(test_config)
52
```

Deploy

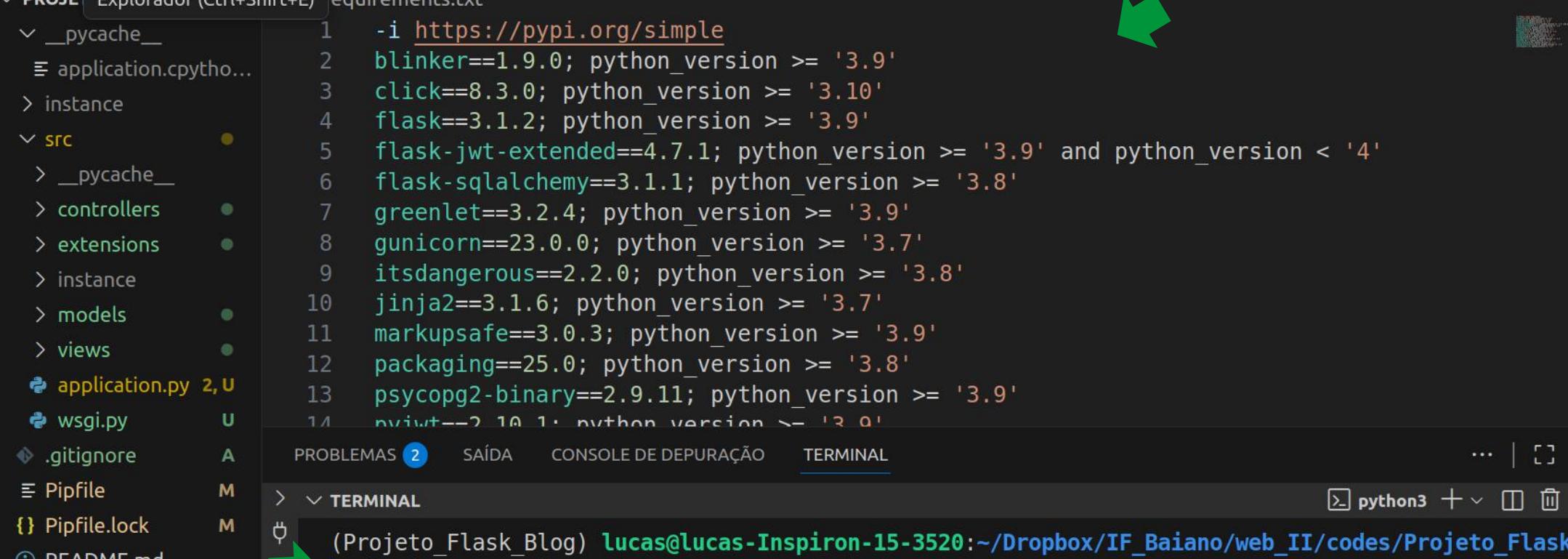
A URL de conexão do banco de dados está sendo carregada de uma variável de ambiente chamada **DATABASE_URL**



```
> instance
└─ src
    └─ __pycache__
    └─ controllers
    └─ extensions
    └─ instance
    └─ models
    └─ views
    └─ application.py 2, U
    └─ wsgi.py      U
    └─ .gitignore   A
    └─ Pipfile      M
    └─ Pipfile.lock M
37     def create_app(test_config=None):
38         # create and configure the app
39         app = Flask(__name__, instance_relative_config=True)
40         app.config.from_mapping(
41             SECRET_KEY='dev',
42             SQLALCHEMY_DATABASE_URI=os.environ["DATABASE_URL"], ←
43             JWT_SECRET_KEY="super-secret"
44         )
45
46         if test_config is None:
47             # load the instance config, if it exists, when not testing
48             app.config.from_pyfile('config.py', silent=True)
49         else:
50             # load the test config if passed in
51             app.config.from_mapping(test_config)
52
```

Deploy

- pipenv requirements > requirements.txt



The screenshot shows the VS Code interface with the following details:

- Project Explorer:** Shows files like `wsgi.py`, `application.py`, and `requirements.txt`.
- Editor:** The `requirements.txt` file is open, displaying dependency specifications. A green arrow points from the terminal output at the bottom to this file.
- Terminal:** Shows the command `pipenv requirements > requirements.txt` being run, with the output matching the contents of the `requirements.txt` file.

Deploy

```
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ git add .
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ git commit -m "added init_db"
[main e1ab334] added init_db
 1 file changed, 8 insertions(+)
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 398 bytes | 398.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:lucassampaioleite/flask-blog.git
 748c06f..e1ab334 main -> main
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ █
```



Deploy

The screenshot shows the Render.com dashboard interface. On the left, there's a sidebar with navigation links like Dashboard, flask-blog (which is selected and highlighted in purple), Events (also highlighted in purple with a green arrow pointing to it), Settings, MONITOR, Logs, Metrics, MANAGE, Environment, Shell, Scaling, Previews, Disks, and Jobs. The main content area has tabs for flask-blog and https://flask-blog-iokt.onrender.com. A search bar at the top right includes 'Search' and 'K' keys, and buttons for '+ New', 'Upgrade', and help. A message banner at the top says 'Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.' with a 'Upgrade now' link. Below the banner, the 'Events' section shows a log entry for November 15, 2025, at 8:49 AM, indicating the application was added. The 'Logs' section shows a detailed log of service startup and initial traffic. Three green arrows highlight the 'Events' section, the 'Live' status indicator in the log entry, and the log entry itself.

Service ID: srv-d4bp2dogjhc73ctsh5g

lucassampaiolete / flask-blog main

https://flask-blog-iokt.onrender.com

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more. Upgrade now

November 15, 2025 at 8:49 AM ✓ Live

748c06f added application

All logs Search

Nov 15 08:52:47 AM 127.0.0.1 - [15/Nov/2025:11:52:47 +0000] "HEAD / HTTP/1.1" 404 0 "-" "Go-http-client/1.1"

Nov 15 08:52:53 AM 127.0.0.1 - => Your service is live 🎉

Nov 15 08:52:53 AM 127.0.0.1 - =>

Nov 15 08:52:53 AM 127.0.0.1 - =>

Nov 15 08:52:53 AM 127.0.0.1 - => Available at your primary URL <https://flask-blog-iokt.onrender.com>

Nov 15 08:52:53 AM 127.0.0.1 - =>

Nov 15 08:52:53 AM 127.0.0.1 - =>

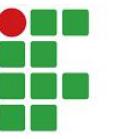
Nov 15 08:52:54 AM 127.0.0.1 - [15/Nov/2025:11:52:54 +0000] "GET / HTTP/1.1" 404 207 "-" "Go-http-client/2.0"

Nov 15 08:53:09 AM 127.0.0.1 - [15/Nov/2025:11:53:09 +0000] "GET / HTTP/1.1" 404 207 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36"

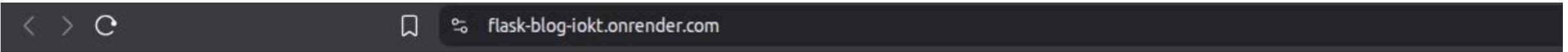
Nov 15 08:53:10 AM 127.0.0.1 - [15/Nov/2025:11:53:10 +0000] "GET /favicon.ico HTTP/1.1" 404 207 "https://flask-blog-iokt.onrender.com/" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36"

Nov 15 08:53:17 AM 127.0.0.1 - [15/Nov/2025:11:53:17 +0000] "GET /users HTTP/1.1" 308 273 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36"

Nov 15 08:53:17 AM 127.0.0.1 - [15/Nov/2025:11:53:17 +0000] "GET /users/ HTTP/1.1" 401 39 "-" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36"



Deploy



Not Found

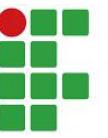
The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.



Deploy

A screenshot of a web browser window. The address bar shows the URL `flask-blog-iokt.onrender.com/users/`. The main content area displays a JSON response with a single key-value pair:

```
{  
    "msg": "Missing Authorization Header"  
}
```



Deploy

Team Workspace New Import

POST login GET list_users POST logout POST login Copy

HTTP Blog / auth / login Copy

Save Share

Send

Collections Environments Flows APIs History

+

Search collections

Blog user post auth

POST login

POST login Copy (selected)

POST logout

roles credit-application-system opencep.com RESTful API with Java 17 and Springboot 3 udemy_spring_boot_course User_API

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

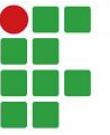
none form-data x-www-form-urlencoded raw binary GraphQL JSON Schema Beautify

```
1 {  
2   "username": "admin",  
3   "password": "admin123"  
4 }
```

Body Cookies Headers (12) Test Results 500 Internal Server Error 345 ms 531 B Save Response

HTML Preview Debug with AI

```
1 <!doctype html>  
2 <html lang=en>  
3 <title>500 Internal Server Error</title>  
4 <h1>Internal Server Error</h1>  
5 <p>The server encountered an internal error and was unable to complete your request. Either the server is  
overloaded or  
there is an error in the application.</p>
```



Deploy

The screenshot shows the Render platform interface. On the left is a sidebar with navigation links: Dashboard, flask-blog (selected), Events, Settings, MONITOR, Logs (selected), Metrics, MANAGE, Environment, Shell, Scaling, Previews, Disks, and Jobs. At the bottom of the sidebar are Changelog, Invite a friend, and Contact support. The main area has tabs for All logs and a search bar. A purple banner at the top says "Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more." with an "Upgrade now" button. The log stream shows several log entries from a flask-blog application, with one entry highlighted in red containing a link to a detailed error page.

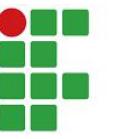
```
Nov 15 08:58:24 AM i zsd7m File "/opt/render/project/src/.venv/lib/python3.13/site-packages/sqlalchemy/engine/default.py", line 951, in do_execute
Nov 15 08:58:24 AM i zsd7m cursor.execute(statement, parameters)
Nov 15 08:58:24 AM i zsd7m -----
Nov 15 08:58:24 AM i zsd7m sqlalchemy.exc.ProgrammingError: (psycopg2.errors.UndefinedTable) relation "user" does not exist
Nov 15 08:58:24 AM i zsd7m LINE 2: FROM "user"
Nov 15 08:58:24 AM i zsd7m ^
Nov 15 08:58:24 AM i zsd7m
Nov 15 08:58:24 AM i zsd7m [SQL: SELECT "user".id AS user_id, "user".username AS user_username, "user".email AS user_email, "user".password_hash AS user_password_hash, "user".role_id AS user_role_id
Nov 15 08:58:24 AM i zsd7m FROM "user"
Nov 15 08:58:24 AM i zsd7m WHERE "user".username = %(username_1)s
Nov 15 08:58:24 AM i zsd7m LIMIT %(param_1)s]
Nov 15 08:58:24 AM i zsd7m [parameters: {'username_1': 'admin', 'param_1': 1}]
Nov 15 08:58:24 AM x zsd7m (Background on this error at: https://sqlalche.me/e/20/f405)
Nov 15 08:58:24 AM i zsd7m 127.0.0.1 - - [15/Nov/2025:11:58:24 +0000] "POST /auth/login HTTP/1.1" 500 265 "-" "PostmanRuntime/7.49.0"
Nov 15 09:14:25 AM i zsd7m [2025-11-15 12:14:25 +0000] [56] [INFO] Handling signal: term
Nov 15 09:14:25 AM i zsd7m [2025-11-15 12:14:25 +0000] [57] [INFO] Worker exiting (pid: 57)
Nov 15 09:14:26 AM i zsd7m [2025-11-15 12:14:26 +0000] [56] [INFO] Shutting down: Master
Nov 15 09:14:25 AM i zsd7m [2025-11-15 12:14:25 +0000] [56] [INFO] Handling signal: term
Nov 15 09:14:25 AM i zsd7m [2025-11-15 12:14:25 +0000] [57] [INFO] Worker exiting (pid: 57)
Nov 15 09:14:26 AM i zsd7m [2025-11-15 12:14:26 +0000] [56] [INFO] Shutting down: Master
Nov 15 09:14:25 AM i zsd7m [2025-11-15 12:14:25 +0000] [56] [INFO] Handling signal: term
Nov 15 09:14:25 AM i zsd7m [2025-11-15 12:14:25 +0000] [57] [INFO] Worker exiting (pid: 57)
Nov 15 09:14:26 AM i zsd7m [2025-11-15 12:14:26 +0000] [56] [INFO] Shutting down: Master
```

Deploy

```
class User(db.Model):
    id: Mapped[int] = mapped_column(primary_key=True)
    username: Mapped[str] = mapped_column(
        db.String(80), unique=True, nullable=False)
    email: Mapped[str] = mapped_column(db.String(120), nullable=True)
    password_hash: Mapped[str] = mapped_column(db.String(300), nullable=False)
```



Aumentar o tamanho do campo da hash



Deploy

```
@click.command("init-db")
def init_db_command():
    with current_app.app_context():
        init_db()

def init_db():
    with current_app.app_context():
        db.create_all()

    # cria a role "admin" se não existir
    role_admin = Role.query.filter_by(name="admin").first()
    if not role_admin:
        role_admin = Role(name="admin")
        db.session.add(role_admin)
        db.session.commit()
        click.echo("Role 'admin' criada!")

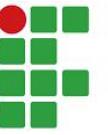
    # cria um usuário "admin" se não existir
    if not User.query.filter_by(username="admin").first():
        user = User(username="admin",
```

Em application.py,
função create_app()

```
with app.app_context():
    init_db()
```

Deploy

```
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ git add .
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ git commit -m "added init_db"
[main e1ab334] added init_db
 1 file changed, 8 insertions(+)
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 398 bytes | 398.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:lucassampaioleite/flask-blog.git
 748c06f..e1ab334 main -> main
(Projeto_Flask_Blog) lucas@lucas-Inspiron-15-3520:~/Dropbox/IF_Baiano/web_II/codes/Projeto_Flask_B
log$ █
```



Deploy

Home Workspaces API Network

Search Postman Ctrl K

Invite 1 Upgrade

Team Workspace New Import

POST login GET list_users POST logout POST login Copy

Blog / auth / login Copy

HTTP POST https://flask-blog-1-g0qq.onrender.com/auth/login

Save Share

Send

Collection: Blog

- > user
- > post
- > auth
 - POST login
 - POST login Copy
 - POST logout
- > roles

Environments

History

APIs

Flows

+

POST Headers (9) Body (1)

Body (1)

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {  
2   "username": "admin",  
3   "password": "admin123"  
4 }
```

Cookies

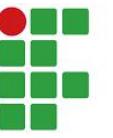
Schema Beautify

Body Cookies Headers (12) Test Results

200 OK 5.22 s 627 B Save Response

{ } JSON Preview Visualize

```
1 {  
2   "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJpcmVzaCI6ZmFsc2UsIm1hdCI6MTc2MzQxMTgxMSwianRpIjoiMzYzNzZhNzQtNGY5OS00YjI1LThmNWEtYTdmMjQzNzVmYTc3IiwidHlwZSI6ImFjY2VzcyI  
sInN1YiI6IjEiLCJuYmYiOjE3NjM0MTE4MTEsImNzcmYiOiI3MDk2MDJkYy00NDIwLTQ0YTgtODZmOS1iMjI5MzhhZmM4NzUiLCJleHAIoJ3E3NjM0MTI3MTF9.  
-GH1JAERiEr2dn18T4SeyHQ8JPyD3QqwVGzpQYz1D7A"  
3 }
```



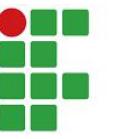
Deploy

The screenshot shows the CircleCI Team Workspace interface. On the left, there's a sidebar with icons for Collections, Environments (highlighted with a green arrow), History, APIs, and Flows. The main area shows a 'BLOG' project with several environment configurations.

Environments:

- BLOG**: Selected environment. It has two variables: `SPRING_JAVA_DEFAULT` (Type: default, Value: `...
...`) and `SECRET` (Type: secret, Value: `...
...`). A modal window titled 'What's changed for variables' is open over this section, containing three items:
 - Local by default:** All variable values you input stay local unless shared explicitly.
 - Initial Shared values:** Share variable values with your teammates with click of a button.
 - Mark as sensitive:** Caution your teammates about the sensitive fields and mask their values.
- REMOTE**: Variable `SECRET` has a value of `https://flask-blog-1-g0gq.onrender.com`.
- LOCAL**: Variable `SECRET` has a value of `localhost:5000`.

Global Environment: A 'Globals' section is visible at the top left of the main workspace area.



Deploy

The screenshot shows the Postman application interface for testing a RESTful API. The left sidebar lists collections and environments, with 'Blog' expanded to show 'user', 'post', and 'auth' collections. The 'auth' collection is currently selected, and its 'POST login' endpoint is highlighted.

The main workspace displays the 'login' endpoint details:

- Method: POST
- URL: {{REMOTE}}/auth/login
- Body tab is selected, showing raw JSON data:

```
1 {  
2   "username": "admin",  
3   "password": "admin123"  
4 }
```
- Headers tab shows 9 items.
- Cookies tab is present.

At the bottom, the response is shown as a 200 OK status with a response time of 1m 7.56s and a size of 626 B. The response body is displayed in JSON format:

```
{ } JSON ▾
```

```
1 {  
2   "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJmc...  
7o7Pe_LzH0xqY7hnluW0kfVfmTEdQLIpjBtmomW8N2k"
```

Dúvidas



PROGRAMAÇÃO WEB II

Curso Técnico Integrado em Informática
Lucas Sampaio Leite

