



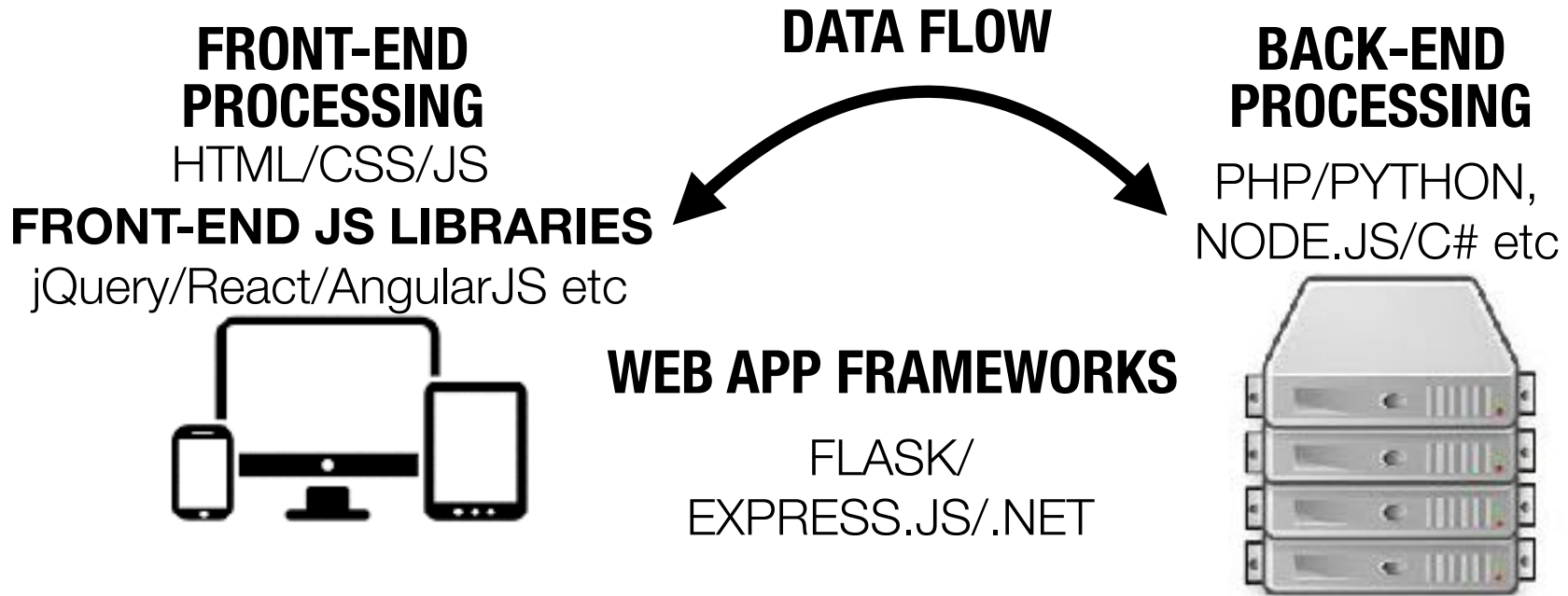
Source: Jose Aragonés

Integrated Data-Driven Story Telling: Lesson 3 - Introduction to the Python Flask Framework

Today's Agenda

- 1. What is Python Flask?**
- 2. Setup of Python Flask on JupyterNotebook**
- 3. Setup of Python Flask on PythonAnywhere**
- 4. Running “Hello World”**
- 5. Setting up your Bootstrap codes inside Flask
Jinja 2 template engine**
- 6. Calling the Airtable API to request data**

POPULAR FULL STACK FRAMEWORK FOR SOFTWARE DEVELOPMENT



Which functions should be processed on the front-end and which should be on the back-end?

Digital Infrastructure for Platform Development

云计算的三层架构



Public(公共)/Private(私人)/Hybrid Clouds(混合)

(Full Stack Developer
由全栈工程到全栈创业)

API (e.g. RestFUL)



BACK-END TECHNOLOGY

后台

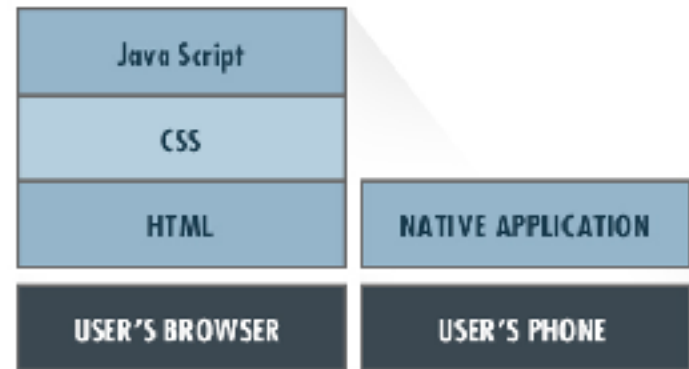
WHAT IS YOUR “CLOUD” AND “STACK” STRATEGY?

云架构及前端和后端的全栈策略

{ JSON }



THE INTERNET



FRONT-END TECHNOLOGY

前台

Setting up MySQL on MAMP

Server: localhost:3306

Database | SQL | Status | Users | Export | Import | Settings | Functions | Variables | Charsets | Engines

General Settings

☐ serveradmin@localhost (root) utf8mb4_unicode_ci

Appearance Settings

Language: English
Theme: orange
Font size: 12px
[More settings](#)

Database server

- Server location: localhost
- Server type: MySQL
- Server version: 5.7.32 - Source distribution
- PHP version: 7.4
- PHP extension: mysqli
- Server charset: UTF-8 Unicode (utf8)

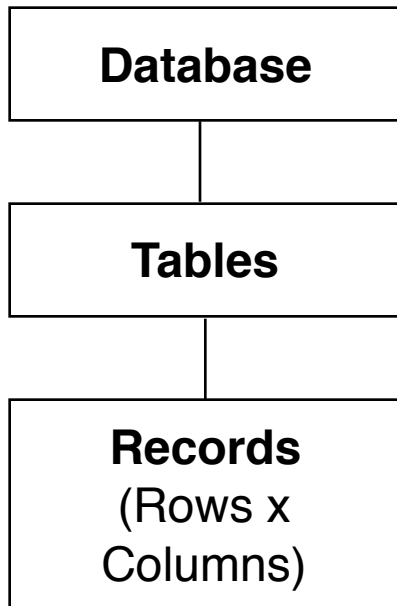
Web server

- Apache/2.4.18 (Ubuntu) PHP/5.6.35 (Ubuntu) MySQL/5.7.32 (Ubuntu)
- Database client version: libmysql - compiled 5.7.32 - 20180505 - 5.7.32
- Server version: 5.7.32 - 20180505 - 5.7.32
- PHP version: 5.6.35

phpMyAdmin

- Version information: 4.8.0
- Documentation
- Wiki
- Official Help page
- Community
- Get support
- List of changes

phpMyAdmin



“A **database** is a collection of **tables** which relate to each other for storing and processing **records** that are defined by rows and columns (**fields**).”

INTRODUCTION TO AIRTABLE

Create, your way

Part spreadsheet, part database, and entirely flexible, teams use Airtable to organize their work, their way.

[Get started](#)

Already using Airtable? [Sign in](#)



Interacting with Airtable & MySQL in Jupyter Notebook

Install Essential Python Modules

In []:

```
1 import sys
2 !pip install Werkzeug
```

In []:

```
1 !pip install --upgrade tensorflow # for python 3.*
```

In []:

```
1 !pip install PyMySQL
```

Testing Airtable API

Testing MySQL Connection

In []:

```
1 import pymysql.cursors
2 connection = pymysql.connect(host='localhost',
3                               user='<Your database username>',
4                               passwd='<Your database password>',
5                               db='<your database name>',
6                               charset='utf8mb4',
7                               cursorclass=pymysql.cursors.DictCursor)
8 with connection.cursor() as cursor:
9     sql = 'SELECT * FROM temperature'
10    cursor.execute(sql)
11    result = cursor.fetchall()
12    dataset = []
13    temp = {}
14    i = 0
15    for row in result:
16        temp[i] = row
17        dataset.append(temp[i])
18        i = i + 1
19    print(dataset)
```


Testing Airtable API

```
In [ ]: 1 import requests
2 import pandas as pd
3 headers = {
4     'Authorization': 'Bearer <Your API Key>',
5 }
6
7 params = {
8     ('view', 'Main View'),
9 }
10
11 def test():
12     r = requests.get('https://api.airtable.com/v0/<Your App Id>/Tasks?api_key=<Your API Key>&
13                     sortField=_createdTime&sortDirection=desc', headers=headers, params=params)
14     dict = r.json()
15     a_dict = dict['records']
16     a_list = []
17     b_dict = {}
18     name_list = []
19     stage_list = []
20
21     df = pd.DataFrame.from_dict(a_dict)
22     for i in a_dicts:
23         dict = i['fields']
24         del dict['Design_Projects']
25         del dict['Assignee']
26         del dict['Stage']
27         b_dict['Name'] = dict['Name']
28         b_dict['Stage_Desc'] = dict['Stage_Desc']
29         b_dict['Completed'] = dict['Completed']
30         b_dict['Time_Estimate'] = dict['Time_Estimate']
31         b_dict['Weight_Factor'] = dict['Weight_Factor']
32         b_dict['converted'] = dict['converted']
33         name_list.append(dict['Name'])
34         stage_list.append(dict['Stage_Desc'])
35         a_list.append(b_dict)
36     # print(name_list)
37     # print(stage_list)
38     return a_list
39
40 list_a = []
41 list_a = test()
42 print(list_a)
```

Project Tracker

HELP

Design Projects

Tasks

Clients

Stage

Venues

STARTS

BLUZZ

Main View

Hide fields

Filter

Group

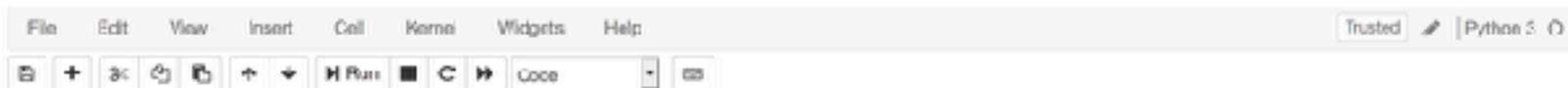
Sort

Color

	Name	Design Projects	Time Estimate	Weight Factor	converted	Notes	Stage	Stage Desc
1	Design clamshell case hinge	Mass Notebook Computer	10.0	2.0	20		Research	Research
2	Source battery supplier	Mass Notebook Computer	1.5	3.0	5		Problem Definition	Problem Definition
3	Research other tea packaging	packaging	2.5	5.0	13	Tea packaging varies alot...	Ideation	Ideation
4	Testing Airtable integration	Airtble 2260 Laptop	1.5	4.0	6		Prototyping	Prototyping
5	Im showing the team about Slack L...	Airtble 2260 Laptop	2.0	5.0	10		Prototyping	Prototyping
6	A demo for 741 students	Hygiene System	5.0	7.0	21		Problem Definition	Problem Definition
7	Another demo for EPIN students	ed Home Brand Identity	3.0	8.0	24		Prototyping	Prototyping
8	Prepare for class on Saturday 13th...	Hygiene System	3.0	9.0	27		Testing	Testing
9	Testing Airtable integration	ed Home Brand Identity	2.0	10.0	20		Research	Research
10	check availability of room	A Brand Identity	2.0	11.0	22		Problem Definition	Problem Definition
11	Have EPIN 1020	Greenwich Brand Identity	2.0	12.0	24		Prototyping	Prototyping
12	Row 12	Greenwich Brand Identity	5.0	11.0	55		Problem Definition	Problem Definition
13	Row 13	Greenwich Brand Identity	6.0	3.0	18		Prototyping	Prototyping
+								

Python for Web Development Using Flask

Running Flask in Local Mode



```
In [ ]: 1 import sys
        2 !pip3 install Werkzeug
```

```
In [ ]: 1 import os
        2 from werkzeug.wsgi import SharedDataMiddleware
        3
        4 from flask import Flask
        5
        6 app = Flask(__name__)
        7
        8 @app.route('/')
        9 def hello_world():
        10     return 'Hello from Flask!'
        11
        12 if __name__ == '__main__':
        13     from werkzeug.serving import run_simple
        14     run_simple('localhost', 9002, app)
```

```
import os
from werkzeug.wsgi import SharedDataMiddleware

from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello from Flask!'

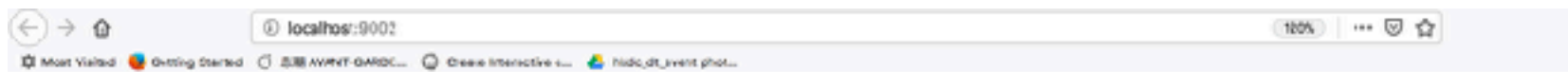
if __name__ == '__main__':
    from werkzeug.serving import run_simple
    run_simple('localhost', 9002, app)
```

```
In [ ]: 1 import sys
        2 !pip3 install Werkzeug
```

```
In [*]: 1 import os
        2 from werkzeug.wsgi import SharedDataMiddleware
        3
        4 from flask import Flask
        5
        6 app = Flask(__name__)
        7
        8 @app.route('/')
        9 def hello_world():
        10     return 'Hello from Flask!'
        11
        12 if __name__ == '__main__':
        13     from werkzeug.serving import run_simple
        14     run_simple('localhost', 9002, app)

* Running on http://localhost:9002/ (Press CTRL+C to quit)
```

```
In [ ]: 1
```



Hello from Flask!

Bootstrap

Build responsive, mobile-first projects on the web with the world's most popular front-end component library.

Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery.

[Get started](#)

[Download](#)

Currently v4.3.1




Power-up the lifetime value of your app users with rewarded video ads via Google

Examples

Quickly get a project started with any of our examples ranging from using parts of the framework to custom components and layouts.

Download source code

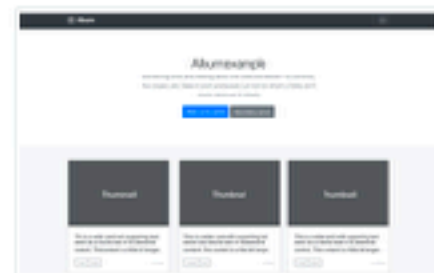


Grow your app revenue with quality demand from Facebook's global advertisers ads via Gelboon

Get Started

Custom components

Brand new components and templates to help folks quickly get started with Bootstrap and demonstrate best practices for adding onto the framework.



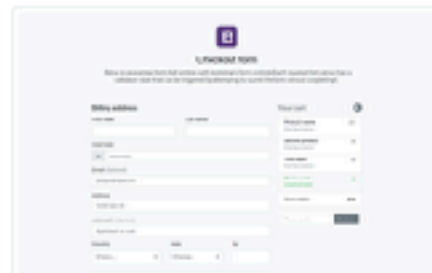
Album

Simple one-page template for photo galleries, portfolios, and more.



Pricing

Example pricing page built with Cards and featuring a custom header and footer.



Checkout

Custom checkout form showing our form components and their validation features.



Product

Lean product-focused marketing page with extensive grid and image work.

Album example

Something short and leading about the collection below—its contents, the creator, etc. Make it short and sweet, but not too short so folks don't simply skip over it entirely.

[Main call to action](#)[Secondary action](#)

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

[View item](#)

[View item](#)

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

[View item](#)

[View item](#)

[View item](#)

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

[View item](#)

[View item](#)

[View item](#)

[illegible]

```

1 import os
2 from werkzeug.wsgi import SharedDataMiddleware
3
4 from flask import Flask, render_template
5
6 app = Flask(__name__)
7
8 # The following line is only needed for running inside Jupyter Notebook
9 root_path = os.path.sep.join(app.instance_path.split(os.path.sep)[-1])
10
11 @app.route("/")
12 def home():
13     return render_template('myhome.html')
14
15 if __name__ == '__main__':
16     from werkzeug.serving import run_simple
17     app.wsgi_app = SharedDataMiddleware(app.wsgi_app, {
18         '/static': root_path+'/flask_apps/static',
19         '/templates': root_path+'/flask_apps/templates'
20     })
21     run_simple('localhost', 9003, app)

```

```

import os
from werkzeug.wsgi import SharedDataMiddleware

```

```

from flask import Flask, render_template

```

```

app = Flask(__name__)

```

```

# The following line is only needed for running inside Jupyter Notebook
root_path = os.path.sep.join(app.instance_path.split(os.path.sep)[-1])

```

```

@app.route("/")
def home():
    return render_template('myhome.html')

```

```

if __name__ == '__main__':
    from werkzeug.serving import run_simple
    app.wsgi_app = SharedDataMiddleware(app.wsgi_app, {
        '/static': root_path+'/flask_apps/static',
        '/templates': root_path+'/flask_apps/templates'
    })
    run_simple('localhost', 9003, app)

```

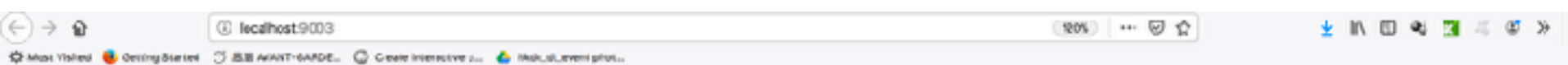
```
12 if name == '__main__':
13     from werkzeug.serving import run_simple
14     run_simple('localhost', 9002, app)
```

* Running on <http://localhost:9002/> (Press CTRL+C to quit)
127.0.0.1 - - [17/Sep/2019 17:59:51] "GET / HTTP/1.1" 200 -

```
In [4]: 1 import os
2 from werkzeug.wsgi import SharedDataMiddleware
3
4 from flask import Flask, render_template
5
6 app = Flask(__name__)
7
8 # The following line is only needed for running inside Jupyter Notebook
9 root_path = os.path.sep.join(app.instance_path.split(os.path.sep)[-1])
10
11 @app.route("/")
12 def home():
13     return render_template('myhome.html')
14
15 if __name__ == '__main__':
16     from werkzeug.serving import run_simple
17     app.wsgi_app = SharedDataMiddleware(app.wsgi_app, {
18         '/static': root_path + '/flask_apps/static',
19         '/templates': root_path + '/flask_apps/templates'
20     })
21     run_simple('localhost', 9003, app)
```

* Running on <http://localhost:9003/> (Press CTRL+C to quit)

```
In [ ]: 1
```



About

Add some information about the album below, the author, or any other background context. Make it a few sentences long so folks can pick up some informative tidbits. Then, link them off to some social networking sites or contact information.

Contact

- [Follow on Twitter](#)
- [Like on Facebook](#)
- [Email me](#)

Album

Album example

Something short and leading about the collection below — its contents, the creator, etc. Make it short and sweet, but not too short so folks don't simply skip over it entirely.

[Main call to action](#) [Secondary action](#)

Thumbnail

This is a wider card with supporting text below as a natural lead in to additional content. This content is a little bit longer.

[View](#) [Edit](#)

9 mins

Album example

Something short and leading about the collection below—its contents, the creator, etc. Make it short and sweet, but not too short so folks don't simply skip over it entirely.

Main call to action

Secondary action

Thumbnail

Thumbnail

Thumbnail

- ▶ **Web template development in Python: Introducing Flask Jinja2 (i.e. render_template library)**
- ▶ **Retrieving information from an Airtable base into a Python Flask program through the Airtable API**

MVC Framework Using the Python Flask

Source: commons.wikimedia.org

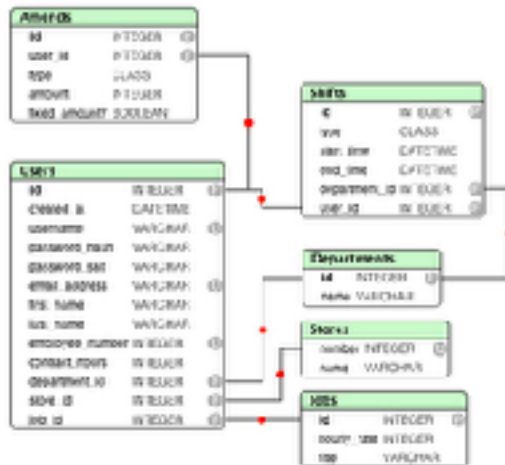


Source: [Caius Durling \(Flickr\)](https://www.flickr.com/photos/caiusdurling/)

Source: [pexels.com](https://www.pexels.com)

V(iew)
(Flask templates)

**Routing control to
pre-defined functions**



M(odel)
(e.g. Python functions)



C(ontroller)
(e.g. Routes)

URL Routing in Flask

```
from flask import Flask, render_template
@app.route('/')
def main():
    :
```

return render_template('index.html')

```
@app.route('/about_us')
def about_us():
    :
```

return render_template('aboutus.html')

Python Code (control flow and
data model)



Html Templates (view)

```
<!DOCTYPE html>
<head>
<meta charset="utf8">
<title>Main Page</title> </head>
<body>
<h1>This is the main page</h1>
    :
    :
</body> </html>
```

index.html

```
<!DOCTYPE html>
<head>
<meta charset="utf8">
<title>About Us</title> </head>
<body>
<h1>This is the about us page</h1>
    :
    :
</body> </html>
```

aboutus.html

Template Tags

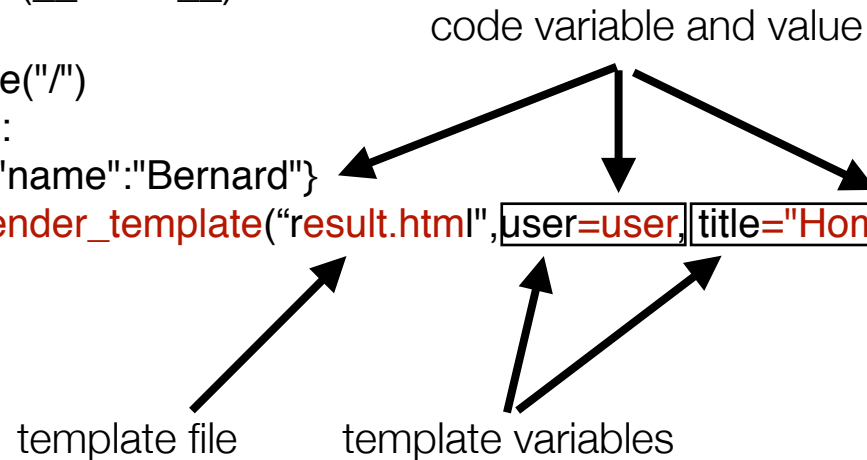
**(i.e. comments, expressions,
statements)**

- `{#... comment#}` : template tag for comments
- `{{... expression...}}` : template tag for expressions
- `{%...statement...}}` : template tag for statements

Passing Values to Template Variables

```
from flask import Flask, render_template
app = Flask(__name__)
```

```
@app.route("/")
def index():
    user = {"name": "Bernard"}
    return render_template("result.html", user=user, title="Home Page")
```



Passing values to template variables.


```

44- <button class='navbar-toggler' type='button' data-toggle='collapse' data-target='#navbarHeader' aria-controls='navbarHeader' aria-expanded='false'
45-   <span class='navbar-toggler-icon'></span>
46- </button>
47- </div>
48- </div>
49- </header>
50-
51- <main role='main'>
52-
53- <section class='jumbotron text-center'>
54-   <div class='container'>
55-     <!--h1 class='jumbotron-heading'>Album</h1-->
56-     <h1 class='jumbotron-heading'>[[user.name]]'s [[title]]</h1>
57-     <p class='lead text-muted'>Something short and leading about the collection below—its contents, the creator, etc. Make it short and sweet, but n
58-     </p>
59-     <a href='https://getbootstrap.com/docs/4.0/examples/album/#' class='btn btn-primary my-2'>Main call to action</a>
60-     <a href='https://getbootstrap.com/docs/4.0/examples/album/#' class='btn btn-secondary my-2'>Secondary action</a>
61-   </div>
62- </section>
63-
64-
65- <div class='album py-5 bg-light'>
66-   <div class='container'>
67-
68-     <div class='row'>
69-       <div class='col-md-4'>
70-         <div class='card mb-4 box-shadow'>
71-           <img class='card-img-top' data-src='holder.js/180px225?theme=thumb&amp;bg=55595c&amp;fg=eceef&amp;text=Thumbnail' alt='Thumbnail [180x225]'>
72-           <div class='card-body'>
73-             <p class='card-text'>This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little
74-             <div class='c-flex justify-content-between align-items-center'>
75-               <div class='btn-group'>
76-                 <button type='button' class='btn btn-sm btn-outline-secondary'>View</button>
77-                 <button type='button' class='btn btn-sm btn-outline-secondary'>Edit</button>
78-               </div>
79-               <small class='text-muted'>9 mins</small>
80-             </div>
81-           </div>
82-         </div>
83-       </div>

```

template variables

Bernard's Home Page

Something short and leading about the collection below—its contents, the creator, etc. Make it short and sweet, but not too short so folks don't simply skip over it entirely.

[Main call to action](#)[Secondary action](#)

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

[View](#) [Edit](#)

9 mins

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

[View](#) [Edit](#)

5 mins

Thumbnail

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

[View](#) [Edit](#)

9 mins

Thumbnail

Thumbnail

Thumbnail

Running Flask Codes on PythonAnywhere



Host, run, and code Python in the cloud!

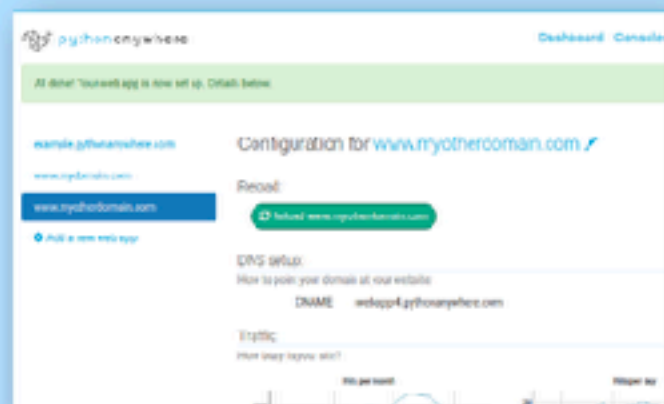
Get started for free. Our basic plan gives you access to machines with a [full Python environment](#) already installed. You can develop and host your website or any other code directly from your browser without having to install software or manage your own server.

Need more power? Upgraded plans start at \$5/month.

[Start running Python online in less than a minute! »](#)

[Watch our one-minute video »](#)

Not convinced? [Read what our users are saying!](#)



Start hosting quickly

Just write your application. No need to configure or maintain a web server — everything is set up and ready to go.

[More »](#)

Develop anywhere

Take your development environment with you! If you have a browser and an Internet connection, you've got everything you need.

[More »](#)

Teach and learn

PythonAnywhere is a fully-fledged Python environment, ready to go, for students and teachers — concentrate on teaching, not on installation hassles.

Amazing support

Need help with PythonAnywhere? If you get in touch, you can talk directly with the development team. Help for developers, from developers.

[More »](#)



Plans and pricing

Beginner: Free!

A **limited account** with one webapp at your username.pythonanywhere.com, restricted outbound internet access from your app, low CPU/bandwidth, no Python/Shell/SQL auto-backup/updates. It works and it's a great way to get started!

[Create a Beginner account](#)

Education accounts

Are you a teacher looking for a place your students can make Python? You're not alone. Click through to find out more about our [Education beta](#).

All of our paid plans come with a non-refundable 30-day money-back guarantee — you're billed monthly and you can cancel at any time. The minimum contract length is just one month. You get unrestricted internet access from your applications, unlimited in-browser Python, Bash and database consoles, and full SSH access to your account. All accounts (including free ones) have screen-sharing with other PythonAnywhere accounts, and free SSL support (though you'll need to get a certificate for your own domains).

Hacker	\$5/month	Web dev	\$12/month	Startup	\$99/month	Custom	\$5 to \$500/month
Run your Python code in the cloud from one web app and the console		If you want to host small Python-based websites for yourself or your clients		Start a business and don't worry about having to scale to handle traffic spikes		Wants a combination that's not on the list? Create your own! All custom plans have:	
A Python IDE in your browser with unlimited Python/Bash consoles		A Python IDE in your browser with unlimited Python/Bash consoles		A Python IDE in your browser with unlimited Python/Bash consoles		A Python IDE in your browser with unlimited Python/Bash consoles	
One web app on a custom domain or your-username.pythonanywhere.com		Up to 2 web apps on custom domains or your-username.pythonanywhere.com		Up to 3 web apps or custom domains or your-username.pythonanywhere.com		Up to 25 web apps, on custom domains or your-username.pythonanywhere.com	
Enough power to run a typical 100,000 hit/day website. (more info)		Enough power to run a typical 150,000 hit/day website on each web app. (more info)		Enough power to run a typical 1,000,000 hit/day website on each web app. (more info)		As many web workers as you need to scale your site's capacity. (more info)	
2,800 CPU seconds per day free							



Create your account

Username:

This username is already taken. Please choose another.

Email:

Password:

Password (repeat):

☒ I agree to the [Terms and Conditions](#) and the [Privacy and Cookies Policy](#), and confirm that I am at least 13 years old.

[Register](#)

We promise not to spam or pass your details on to anyone else.

Warning You have not confirmed your email address yet. This means that you will not be able to reset your password if you lose it. If you cannot find your confirmation email anymore, send yourself a new one [here](#).

Dashboard

Welcome, [bssuen](#)

CPU Usage: 0% used - 8.008 of 1026. Resets in 23 HOURS, 59 minutes [More info](#)

[Upgrade Account](#)

File storage: 0% full - 48.0 KB of your 512.8 MB quota

Recent Consoles



You have no recent consoles.

New console:

[Bash](#)

[Python](#)

[More...](#)

Recent Files

You have no recently added files.

[+ Open another file](#)

[Browse files](#)



All Web apps

You don't have any web apps.

[Open Web tab](#)

Welcome to PythonAnywhere

This tour will give you a quick overview of how the site works. It only lasts 7 steps. Promise!

[Prev](#) [Next](#) [End tour](#)

Dashboard

Welcome, [hissuer](#)

CPU Usage: 0% used — 3.00s of 100s. Resets in 22 hours, 59 minutes. [More info](#)

[Upgrade Account](#)

File storage: 0% full — 43.0 KB of your 512.0 MB quota

Recent Consoles



You have no recent consoles.

New console:

\$ Bash

>>> Python

[More...](#)

Recent Files



You have no recently edited files.

[+ Open a new file](#)

[Browse files](#)

Recent Notebooks



Your account does not support
Jupyter Notebooks. [Upgrade your
account](#) to get access!

All Web apps

You don't have any web apps.

[Open Web app](#)



[Add a new web app](#)

You have no web apps

To create a PythonAnywhere-hosted web app, click the "Add a new web app" button to the left.

[+ Add a new web app](#)

Create new web app

Your web app's domain name

Your account doesn't support custom domain names, so your PythonAnywhere web app will live at `username.pythonanywhere.com`.

Want to change that? [Upgrade now!](#)

Otherwise, just click "Next" to continue.

[Cancel](#)[< Back](#)[Next >](#)

[Add a new web app](#)

Create new web app

Select a Python Web framework

...or select "Manual configuration" if you want detailed control.

- » Django
- » web2py
- » Flask
- » Bottle
- » Manual configuration (including virtualenvs)

What other frameworks should we have here? Send us some feedback using the link at the top of the page!

[Cancel](#)

[« Back](#)

[Next »](#)

 Add a new web app

Create new web app

Select a Python version

- **Python 2.7** (Flask 1.0.2)
- **Python 3.4** (Flask 1.0.2)
- **Python 3.5** (Flask 1.0.2)
- **Python 3.6** (Flask 1.0.2)
- **Python 3.7** (Flask 1.0.2)

note: if you'd like to use a different version of Flask to the default version, you can use a [virtualenv](#) for your web app. There are [instructions here](#).

Cancel

◀ Back

Next ▶

[Add a new web app](#)

Create new web app

Quickstart new Flask project

Enter a path for a Python file you wish to use to hold your Flask app. If this file already exists, its contents will be overwritten with the new app.

**Path**[Cancel](#)[Back](#)[Next](#)

All done! Your web app is now set up. Details below.

30

bssuan.pythonanywhere.com

Add a new web app

Configuration for `bssuan.pythonanywhere.com`

Reload:

Reload `bssuan.pythonanywhere.com`

Bes: before date:

We're happy to host your free website – and keep it free – for as long as you want to keep it running, but you'll need to log in at least once every three months and click the "Run until 3 months from today" button below. We'll send you an email a week before the site is disabled so that you don't forget to do that. [See here for more details.](#)

This site will be disabled on **Tuesday 17 December 2019**

Run until 3 months from today

Paying users' sites stay up forever without any need to log in to keep them running.

Traffic:

How busy is your site?

This month (previous month)	1	(0)
Today (yesterday)	1	(0)
Hour (previous hour)	1	(0)

Want some more data? [Paying accounts](#) get pretty charts:-)

[/home/](#) [bssuen](#)[Open Bash console here](#) **0% full** – 128.0 KB of your 512.0 MB quota

Directories

[New directory](#)

[local/](#)

[.virtualenvs/](#)

[mysite/](#)

Files

[New file](#)

.bashrc	2011-06-13 09:03	519 bytes
.gitconfig	2011-06-13 09:03	616 bytes
.profile	2011-06-13 09:03	71 bytes
.pythonstartup.py	2011-06-13 09:03	71 bytes
vimrc	2011-06-13 09:03	4.6 KB
README.txt	2011-06-13 09:03	242 bytes

[Upload a file](#)

100MB maximum size

[Dashboard](#) [Consoles](#) [Files](#) [Web](#) [Tasks](#) [Databases](#)[/home/](#) **bsuen** [Open Bash console here](#) **0% full** - 132.0 KB of your 512.0 MB quota

Directories

[New directory](#)

[./local/](#)

[./virtualenvs/](#)

[mysite](#)

Files

[New file](#)

baselrc	2019-09-17 09:03	504 bytes
gitconfig	2019-09-17 09:03	264 bytes
profile	2019-09-17 09:03	79 bytes
pythonstartegy.py	2019-09-17 09:03	77 bytes
.vimrc	2019-09-17 09:03	4.6 KB
README.txt	2019-09-17 09:03	230 bytes

[Upload a file](#)

100MB maximum size



/home/bssuer/ @mysite

[Dashboard](#) [Consoles](#) [Files](#) [Web](#) [Tasks](#) [Databases](#)

[Open Bash console here](#) **0% full** ~ 28.0 KB of your 51.20 MB quota

Directories

static

[New directory](#)

[_private_](#)



Files

Enter new file name, eg hello.py

[New file](#)

task_app.py

2011-09-17 09:27 180 bytes

[Upload a file](#)

100MB maximum size



/home/bssuer/ myste

Directories

templates/

New directory

__pycache__/
static/



[Dashboard](#) [Consoles](#) **Files** [Web](#) [Tasks](#) [Databases](#)

[Open Bash console here](#)

0% full - 132.0 KB of your 5120 MB quota

Files

Enter new file name, eg hello.py

New file

flask_app.py

1011/06/13 08:27 156 bytes

Upload a file

100MB maximum size



[/home/boone/](#) [my site](#)

[Open Bash console here](#) **2% full** ~ 336.0 KB of your 512.0 MB quota

Directories

[New directory](#)

[__pycache__](#)

[static/](#)

[winput.txt](#)

Files

[New file](#)

[flask_app.py](#) 2019-09-17 09:27 180 bytes

[Upload a file](#)

100MB maximum size



/home/bcsuen/ mysite

[Dashboard](#) [Consoles](#) **Files** [Web](#) [Tasks](#) [Databases](#)

[Open Bash console here](#) **0% full** – 136.0 KB of your 512.0 MB quota

Directories

[New directory](#)

[__pycache__/](#)

[static/](#)

[templates/](#)



Files

[New file](#)

[flask_app.py](#)

2116.00.17 00:17 136 bytes

[Upload a file](#)

100MB maximum size







Setting MySQL on PythonAnywhere

Your MySQL password has been changed



MySQL

Postgres

MySQL settings

Connecting:

Use these settings in your web applications.

Database host address: `localhost.mysql.pythonanywhere-services.com`
Username: `localhost`

Your databases:

Click a database's name to start a MySQL console logged into it.

Start a console on: [localhost](#)

Create a database

Your database names always start with your username + '@'. There's no need to type that prefix in below, though PythonAnywhere will automatically add it.

Database name:

Create

MySQL password:

This should be different to your main PythonAnywhere password, because it is likely to appear in plain text in any web applications you write.

New password:

Confirm password:

Set MySQL password

Your MySQL password has been changed.



MySQL

Postgres

MySQL settings

Connecting:

Use these settings in your web applications:

Database host address:	bernardays.mysql.pythonanywhere-services.com
Username:	bernardays

Your databases:

Click a database's name to start a MySQL console logged in to it.

Start a console on: [bernardays\\$default:](#)

Create a database

Your database names always start with your username + "\$". There's no need to type that prefix in below, though: PythonAnywhere will automatically add it.

Database name:

Create

MySQL password:

This should be different to your main PythonAnywhere password, because it is likely to appear in plain text in any web applications you write.

New password:

Confirm password:

Set MySQL password

MySQL

[Postgres](#)

MySQL settings

Connecting.

Use these settings in your web applications.

Database host address: `bernardays.mysql.pythonanywhere-services.com`
Username: `bernardays`

Your databases:

Click a database's name to start a MySQL console logged in to it.

Start a console on: [bernardaysdefault](#)
Start a console on: [bernardaysdemo](#)

Create a database

Your database names always start with your username + "\$". There's no need to type that prefix in below, though PythonAnywhere will automatically add it.

Database name:

[Create](#)

MySQL password:

This should be different to your main PythonAnywhere password, because it is likely to appear in plain text in any web applications you write.

New password:

Confirm password:

[Set MySQL password](#)

MySQL

[Postgres](#)

MySQL settings

Connecting:

Use these settings in your web applications.

Database host address: `bernardays.mysql.pythonanywhere-services.com`
 Username: `bernardays`

Your databases

Click a database's name to start a MySQL console logged in to it.

Start a console on: [bernardayside!ault](#)

Start a console on: [bernardayside!ault](#)

Create a database

Your database names always start with your username, so you don't have to type that prefix in below, though: `pythonanywhereusername`

Database name:

Create

MySQL password:

This should be different to your main PythonAnywhere password, because it is likely to appear in plain text in any web applications you write.

New password:

confirm password:

Set MySQL password

- Open Link in New Tab
- Open Link in New Window
- Download Linked File
- Download Linked File As...
- Add Link to Bookmarks...
- Add Link to Reading List
- Copy Link
- Share
- Inspect Element
- Services



MySQL banana:sys\$lsnu

Share with others



```
loading console...  
Preparing execution environment... OK  
Reticulating splines... OK  
Loading Mysql interpreter...|
```



```
Welcome to the MySQL monitor.  Commands end with ; or \g.  
Your MySQL connection id is 13715638  
Server version: 5.7.21-log MySQL Community Server (GPL)
```

```
Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.
```

```
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
mysql> 
```

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 13215739

Server version: 5.7.21-log MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
mysql> show databases;
```

Database
information_schema
bernardsysdemo

3 rows in set (0.21 sec)

```
mysql> use bernardsysdemo;
```

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 13215739

Server version: 5.7.21-log MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;

Database
information_schema
bernardsysdemo
bernardsysdemo

3 rows in set (0.21 sec)

mysql> use bernardsysdemo;

Database changed

mysql> █


```
mysql> SELECT * FROM EMPLOYEE;
```

name	address	rank	salary
Sam	Harding	beginner	100000
Bernie	Third	advanced	100000
Meenere	Third	advanced	150000
Jake	Harding	beginner	50000

```
4 rows in set (0.00 sec)
```

```
mysql> SELECT name  
-> FROM Employee;
```

name
Sam
Bernie
Meenere
Jake

```
4 rows in set (0.00 sec)
```

```
mysql> SELECT name  
-> FROM Employee  
-> WHERE salary = 100000;
```

name
Sam
Bernie

```
2 rows in set (0.00 sec)
```

```
mysql> _
```

```
information_schema
bernardsys$default
bernardsys$demo
```

3 rows in set (0.21 sec)

```
mysql> show tables;
empty set (0.00 sec)
```

```
mysql> status;
```

mysql Ver 14.14 Distrib 5.7.23, for Linux (x86_64) using Editline wrapper

```
Connection id:      13215739
Current database:    bernardsys$demo
Current user:        bernardsys@10.0.0.174
SSL:                 Cipher in use is DHE-RSA-AES256-SHA
Current pager:       stdout
Using outfile:       ''
Using delimiter:     ;
Server version:      5.7.23-log MySQL Community Server (GPL)
Protocol version:    10
Connection:          bernardsys.mysql.pythonanywhere-services.com via TCP/IP
Server characterset: utf8
Db. characteraset:  utf8
Client characterset: utf8
Conn. characterset:  utf8
TCP port:            3306
Uptime:              63 days 17 hours 7 min 29 sec
```

Threads: 32 Questions: 41502778 Slow queries: 159 Opens: 7510728 Flush tables: 1 Open tables: 992 Queries per second avg: 70.177

```
mysql> CREATE TABLE customer;
```

ERROR 1113 (42000): A table must have at least 1 column

```
mysql> CREATE TABLE customer (
```

```
  -> Id INT NOT NULL AUTO INCREMENT,
  -> FirstName VARCHAR (20) NOT NULL,
  -> LastName VARCHAR (20) NOT NULL,
  -> City VARCHAR (30) NOT NULL,
  -> Country VARCHAR (30) NOT NULL,
  -> Phone VARCHAR (15) NOT NULL,
  -> PRIMARY KEY ( Id )
  -> );
```

Query OK, 0 rows affected (0.02 sec)

```
information_schema
bernardsys$default
bernardsys$demo
```

3 rows in set (0.21 sec)

```
mysql> INSERT customer (Firstname, Lastname, City, Country, Phone)
      => VALUES ('David', 'Zhang', 'Shanghai', 'PRC', '13756789');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from customers;
```

Id	Firstname	Lastname	City	Country	Phone
1	David	Zhang	Shanghai	PRC	13756789

1 row in set (0.01 sec)

```
mysql> INSERT customer (Firstname, Lastname, City, Country, Phone) VALUES ('Jean', 'Hu', 'Shenzhen', 'PRC', '13756789');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from customers;
```

Id	Firstname	Lastname	City	Country	Phone
1	David	Zhang	Shanghai	PRC	13756789
2	Jean	Hu	Shenzhen	PRC	13756789

2 rows in set (0.00 sec)

THANK YOU FOR YOUR TIME!