111-2進階程式設計課程(12)
Advanced Computer Programming

亞大資工系

課程大綱

- W1-課程介紹/Introduction
- W2-Python libraries
- W3-BeautifulSoup(1)
- W4-BeautifulSoup(2)
- W5-
- W6-Scrapy(1)
- W7-Scrapy(2)
- W8-Storing Data
- W9-Midterm project
- W10-Web & HTTP
- W11-Flask
- W12-Flask Routes
- W13-Jinja template
- W14-Flask-form
- W15-Flask-mail
- W16-REST API
- W17-Project development(2)
- W18-Final presentation

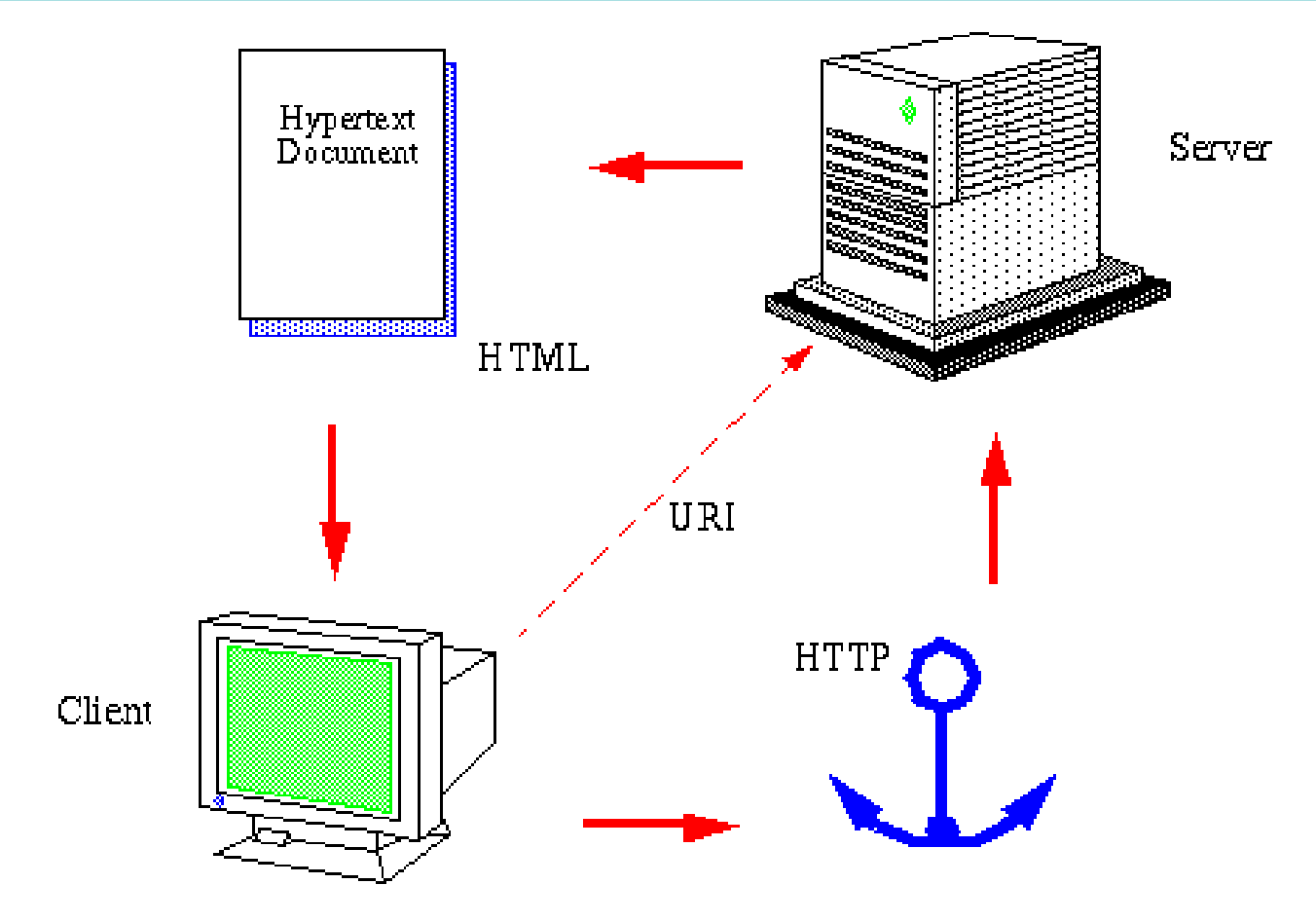
免費的線上 Python 執行環境 PythonAnywhere

- 免費帳號的功能受到如下限制 :
  - 只能建立一個 App (應用程式)
  - 網外存取 Internet 有限制
  - CPU 與儲存有限制 (一天 100 秒 CPU 時間, 512MB 儲存)
  - 不提供 Jupyter (但有 IPython)
  - 只能有兩個 Console (Bash 與 Python)

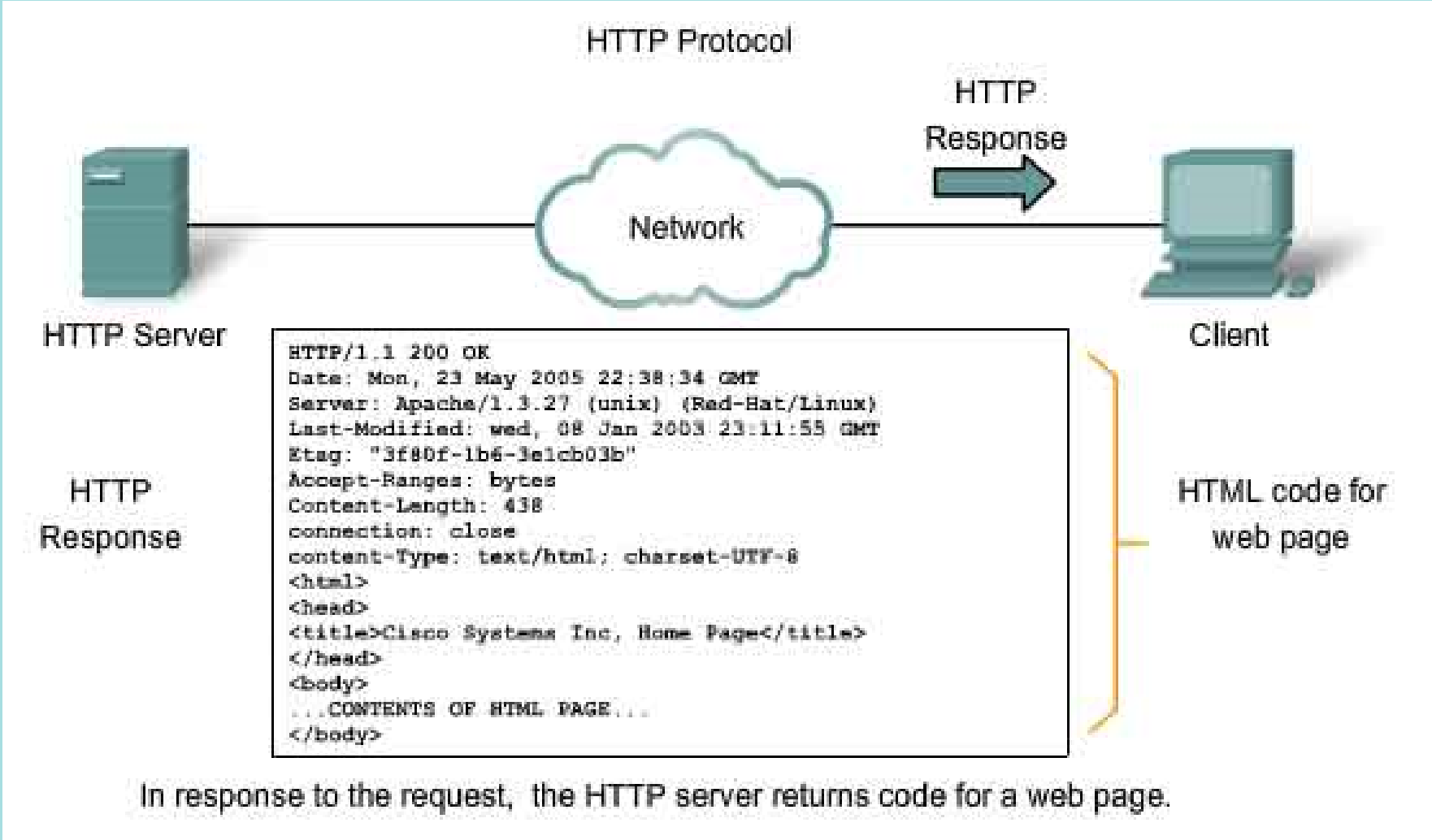
HTTP, URI, HTML

```
<!DOCTYPE html>
<html>
<!-- created 2010-01-01 -->
 <head>
  <title>sample</title>
 </head>
 <body>
  <p>Voluptatem accusantium
   totam rem aperiam.</p>
 </body>
</html>
```

HTML



HTTP


HTTP Protocol
HTTP
Response
Network
HTTP Server
Client
HTTP
Response
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Server: Apache/1.3.27 (unix) (Red-Hat/Linux)
Last-Modified: wed, 08 Jan 2003 23:11:55 GMT
Etag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Content-Length: 438
connection: close
content-Type: text/html; charset-UTF-8
<html>
<head>
<title>Cisco Systems Inc, Home Page</title>
</head>
<body>
...CONTENTS OF HTML PAGE...
</body>
HTML code for
web page
In response to the request, the HTTP server returns code for a web page.

Web application frameworks

- JavaScript
  - Express.js, React.js, Angular.js
- PHP
  - Laravel, CodeIgniter
- Ruby
  - Rails
- Python
  - Django, Flask , FastAPI
- Java
  - Spring Boot

Flask framework

- Required
  - Jinja-template engine
  - Werkzeug-WSGI toolkit
- Optional
  - sqlalchemy-SQL toolkit
  - marshmallow: simplified object serialization
  - Celery-task queue

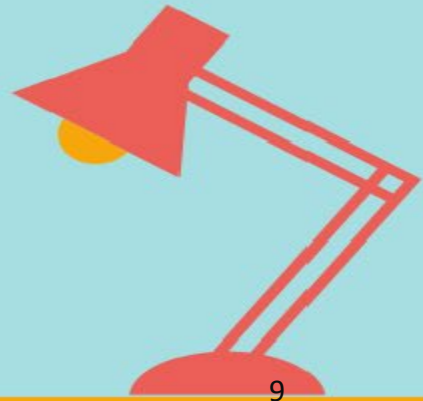
Flask extensions

- **Flask-Bootstrap**: Bootstrap
- **Flask-WTF**: WTForms including CSRF, file upload, and reCAPTCHA
- **Flask-Moment**: Localization of Dates and Times
- **Flask-Babel**: Internationalization and localization support
- **Flask-DebugToolbar**: In-browser debugging tools
- **Flask-Assets**: Integration of CSS and JavaScript assets
- **Flask-Session**: implementation of user sessions with server-side storage
- **Flask-SocketIO**: Socket.IO server implementation with support for WebSocket and long-polling

Flask

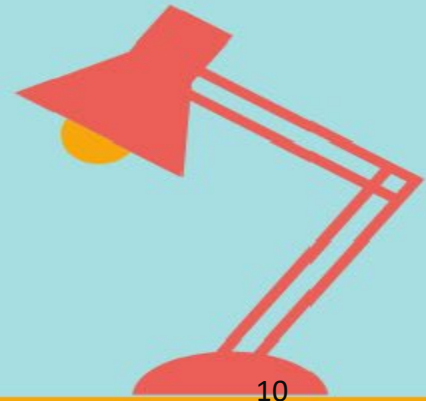
- Flask is a class with
  - run() function
  - route() functions

- Flask is a command
  - flask run
  - flask routes
  - flask shell

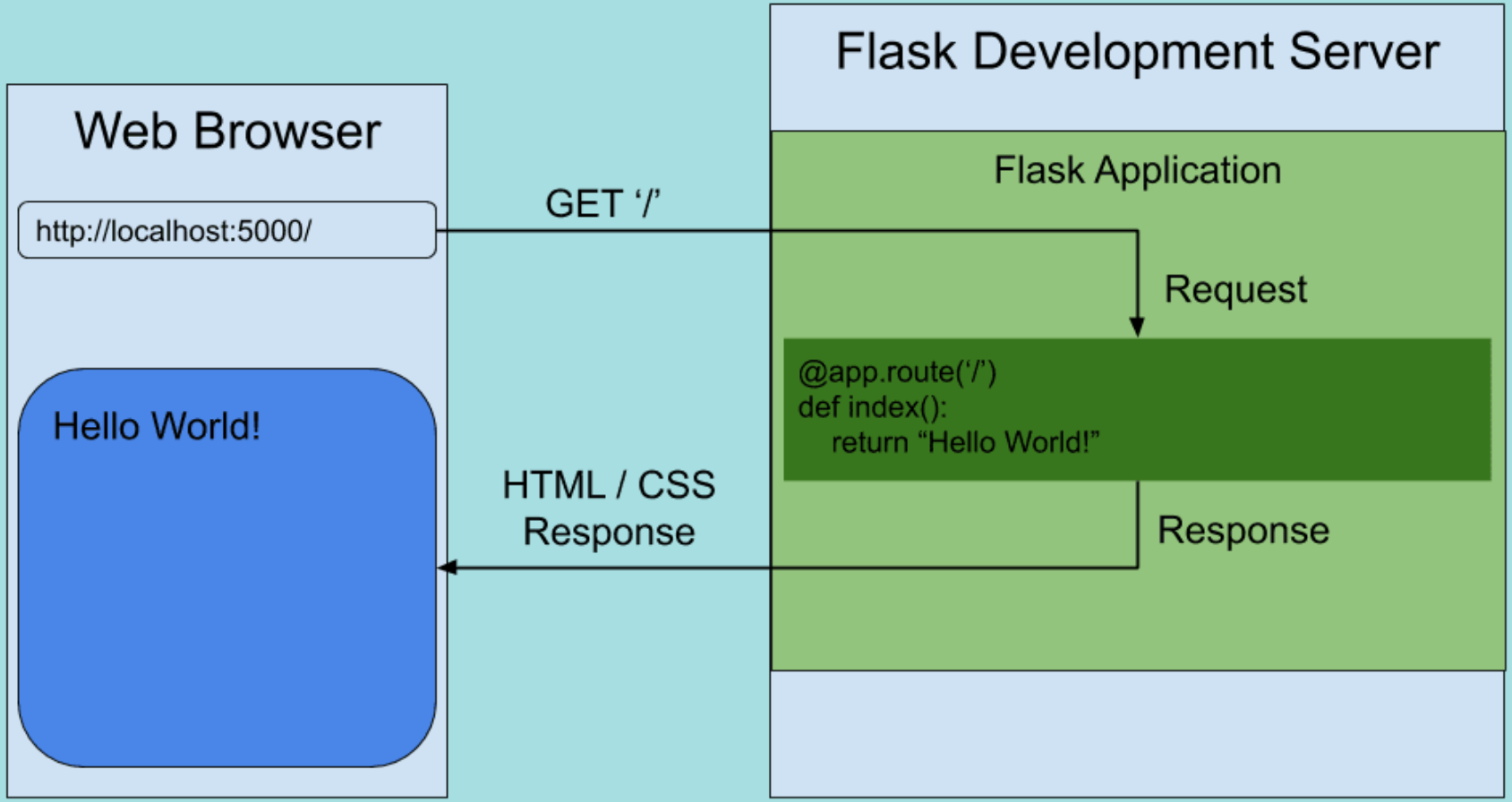


Basic Flask Application Structure

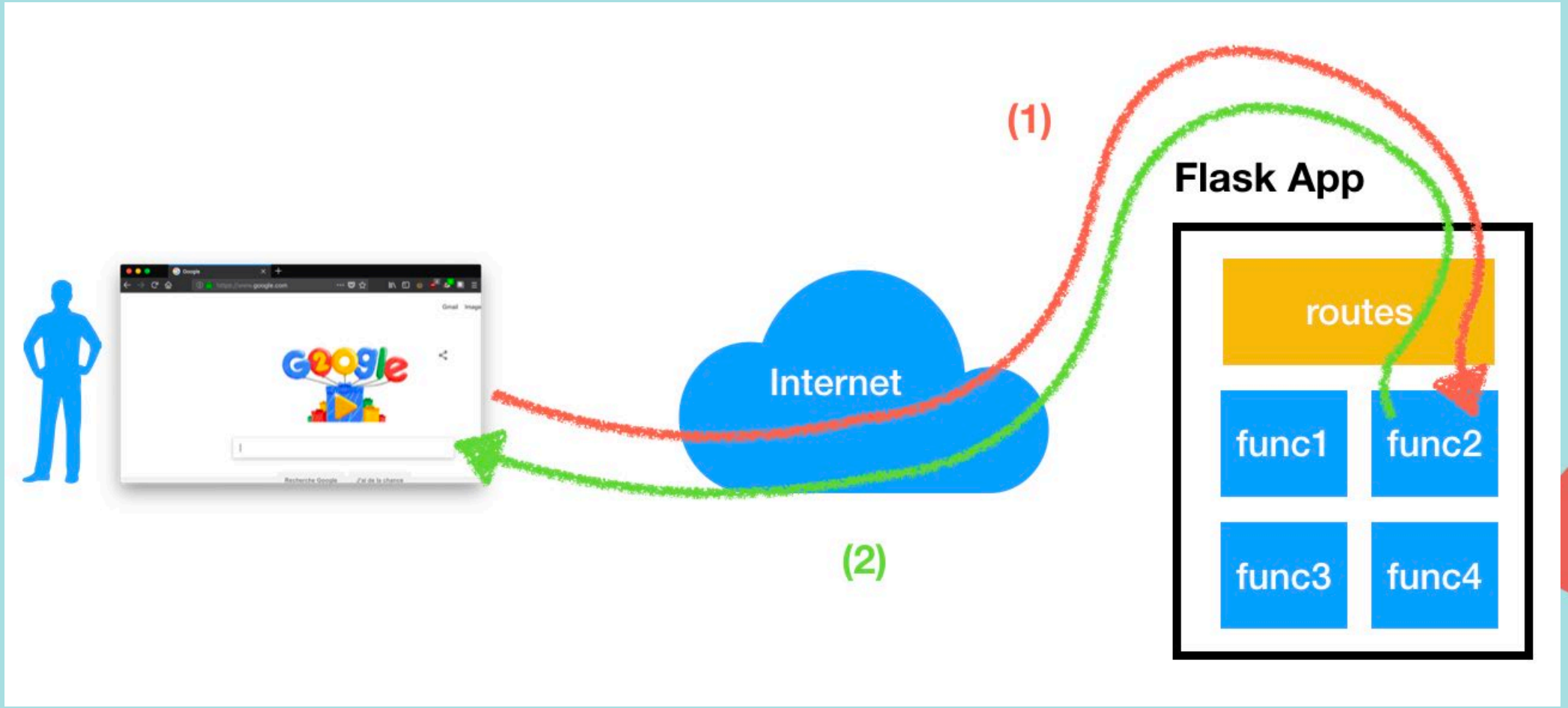
- **Routes and View Functions**
- **Dynamic Routes**
- **Application and Request Contexts**
- Flask context globals
- Flask request object
- Flask response object



Routes and View Functions

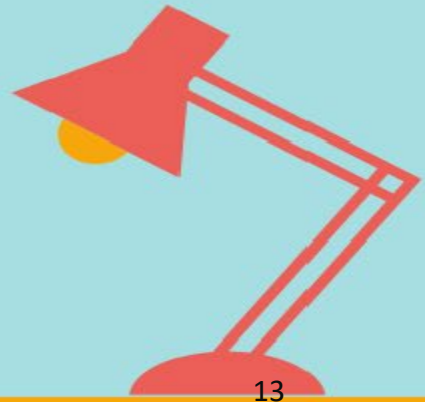


View functions



Dynamic Routes

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

@app.route('/')
def index():
    return '<h1>Hello World!</h1>'

@app.route('/user/<name>')
def user(name):
    return '<h1>Hello, {}!</h1>'.format(name)
```



Application and Request Contexts

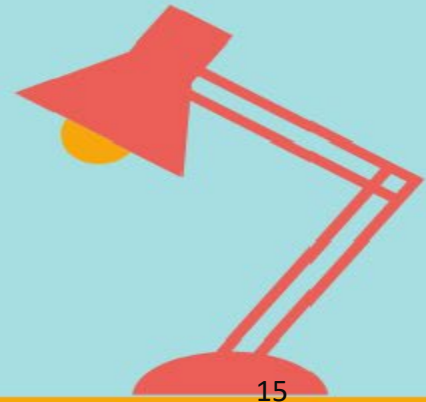
```
from flask import request

@app.route('/')
def index():
    user_agent = request.headers.get('User-
Agent')
    return '<p>Your browser is
{}</p>'.format(user_agent)
```

Table 2-1. Flask context globals

Variable name	Context	Description
`current_app`	Application context	The application instance for the active application.
`g`	Application context	An object that the application can use for temporary storage during the handling of a request. This variable is reset with e
`request`	Request context	The request object, which encapsulates the contents of an HTTP request sent by the client.
`session`	Request context	The user session, a dictionary that the application can use to store values that are "remembered" between requests.

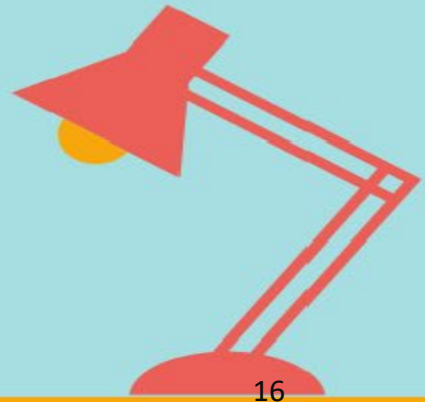
Flask context globals

form	A dictionary with all the form fields submitted with the request.
args	A dictionary with all the arguments passed in the query string of the URL.
values	A dictionary that combines the values in form and args.
cookies	A dictionary with all the cookies included in the request.
headers	A dictionary with all the HTTP headers included in the request.



Flask request object

status_code	The numeric HTTP status code
headers	A dictionary-like object with all the headers that will be sent with the response
set_cookie()	Adds a cookie to the response
delete_cookie()	Removes a cookie
content_length	The length of the response body
content_type	The media type of the response body
set_data()	Sets the response body as a string or bytes value
get_data()	Gets the response body

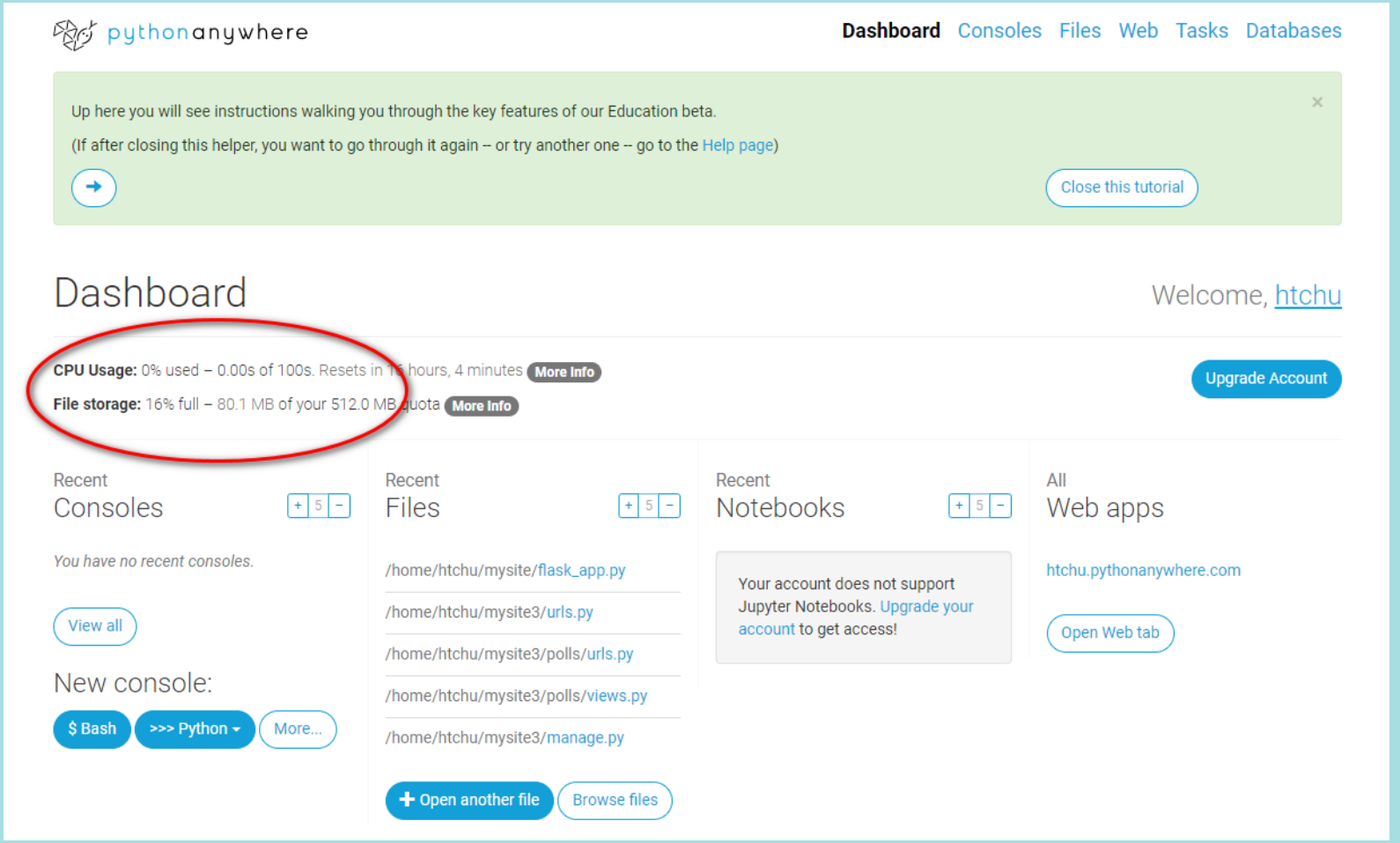


Flask response object

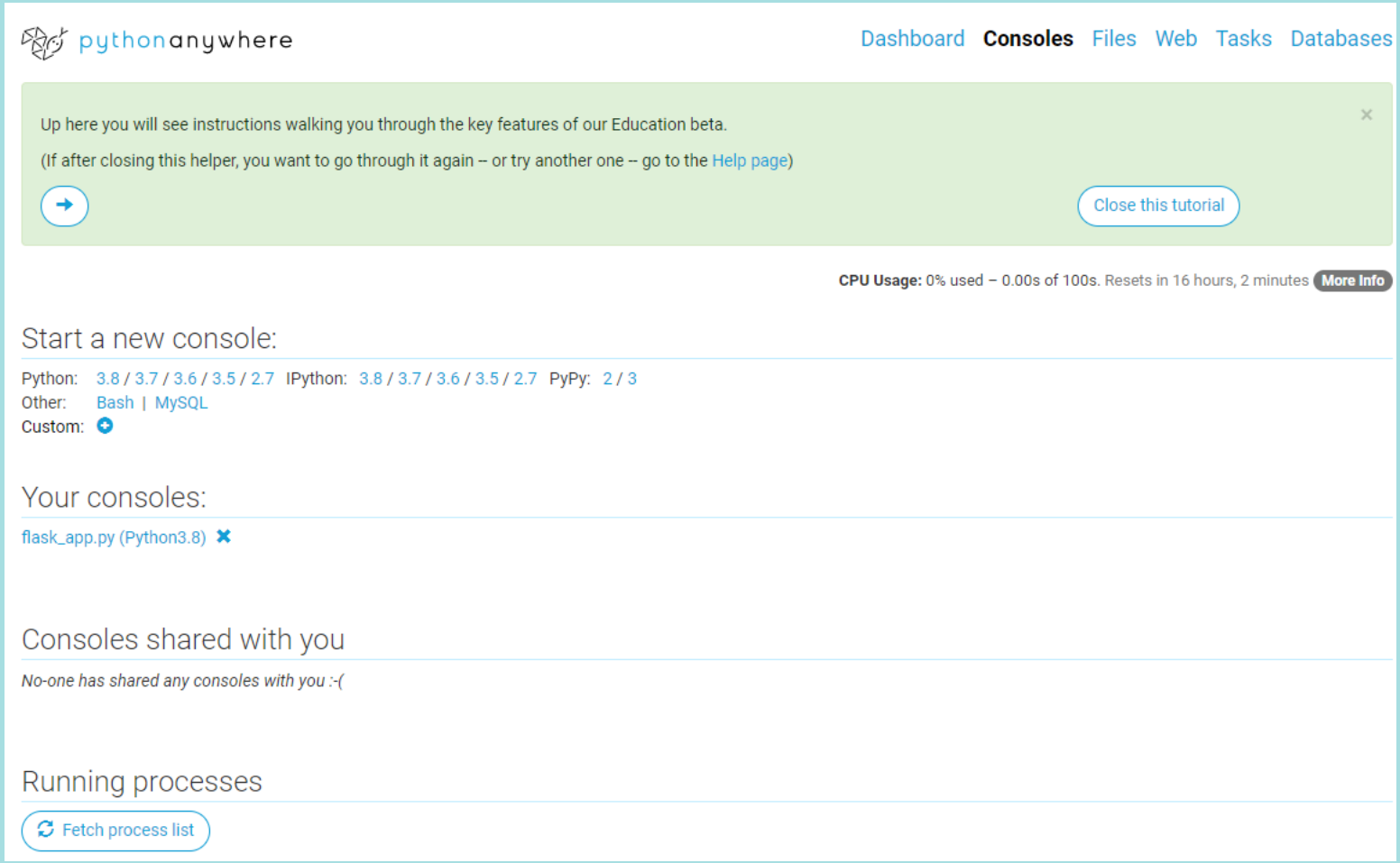
status_code	The numeric HTTP status code
headers	A dictionary-like object with all the headers that will be sent with the response
set_cookie()	Adds a cookie to the response
delete_cookie()	Removes a cookie
content_length	The length of the response body
content_type	The media type of the response body
set_data()	Sets the response body as a string or bytes value
get_data()	Gets the response body

```
from flask import redirect

@app.route('/')
def index():
    return redirect('http://www.example.com')
```

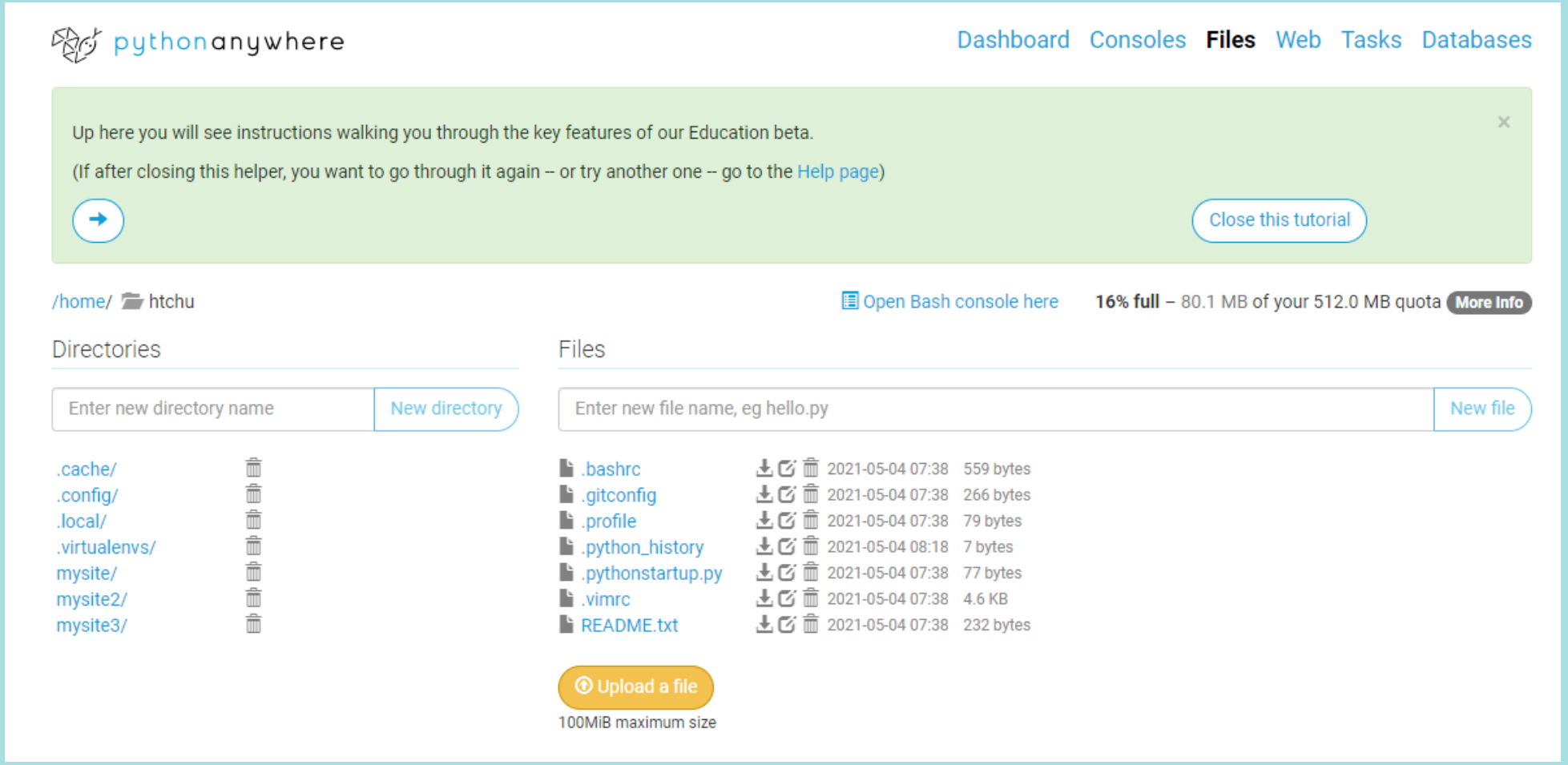
Dashboard儀表板



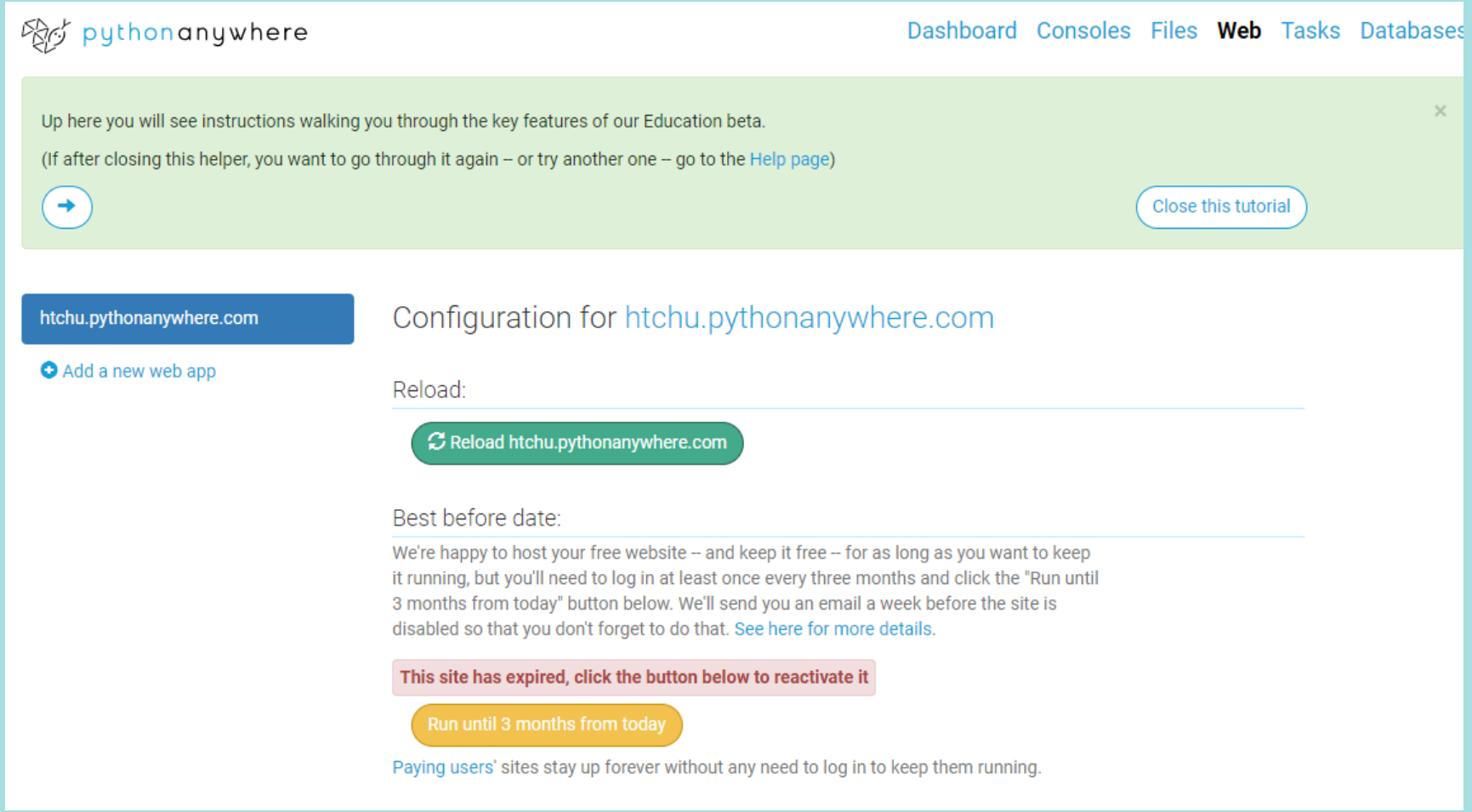
Command Console命令控制台



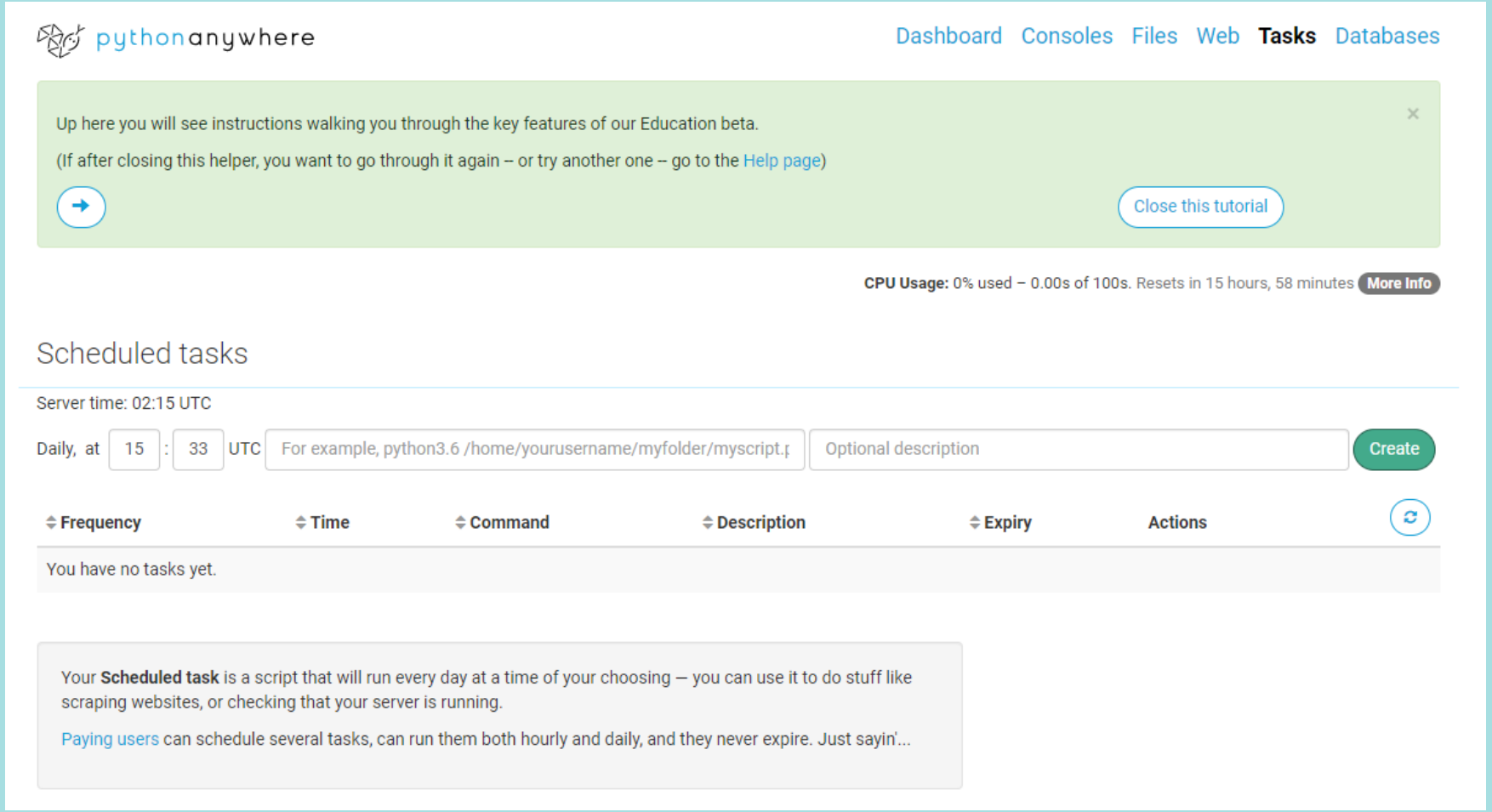
Files


pythonanywhere
Dashboard Consoles Files Web Tasks Databases
Up here you will see instructions walking you through the key features of our Education beta.
(If after closing this helper, you want to go through it again -- or try another one -- go to the Help page)
Close this tutorial
/home/ htchu
Open Bash console here
16% full – 80.1 MB of your 512.0 MB quota More Info
Directories
Enter new directory name
New directory
.cache/
.config/
.local/
.virtualenvs/
mysite/
mysite2/
mysite3/
Files
Enter new file name, eg hello.py
New file
.bashrc 2021-05-04 07:38 559 bytes
.gitconfig 2021-05-04 07:38 266 bytes
.profile 2021-05-04 07:38 79 bytes
.python_history 2021-05-04 08:18 7 bytes
.pythonstartup.py 2021-05-04 07:38 77 bytes
.vimrc 2021-05-04 07:38 4.6 KB
README.txt 2021-05-04 07:38 232 bytes
Upload a file
100MiB maximum size

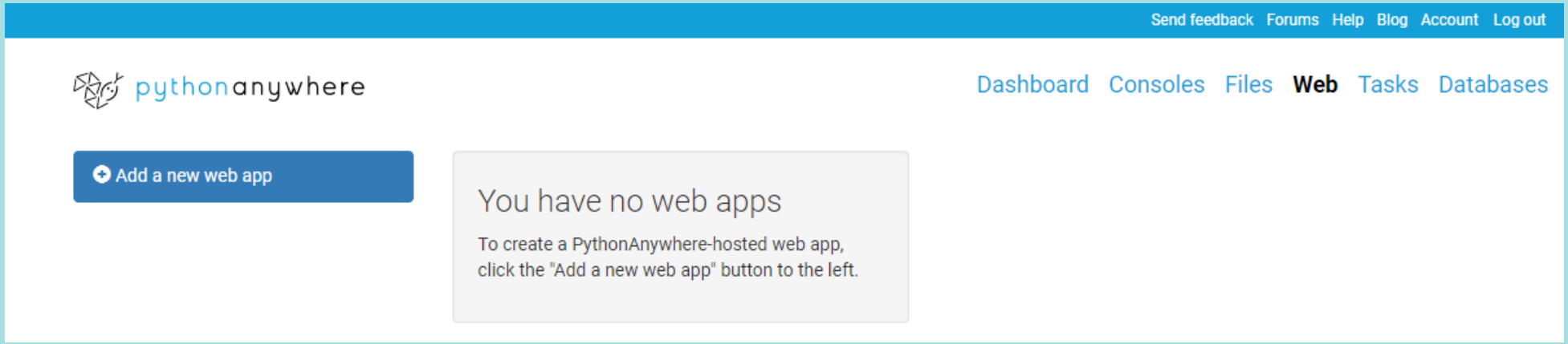
Web

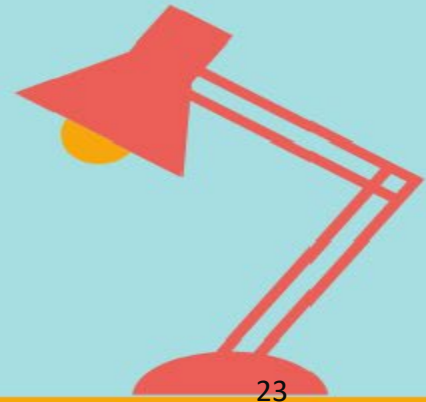

pythonanywhere
Dashboard Consoles Files Web Tasks Databases
Up here you will see instructions walking you through the key features of our Education beta.
(If after closing this helper, you want to go through it again -- or try another one -- go to the Help page)
Close this tutorial
htchu.pythonanywhere.com
Add a new web app
Configuration for htchu.pythonanywhere.com
Reload:
Reload htchu.pythonanywhere.com
Best 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 has expired, click the button below to reactivate it
Run until 3 months from today
Paying users' sites stay up forever without any need to log in to keep them running.

Tasks任務

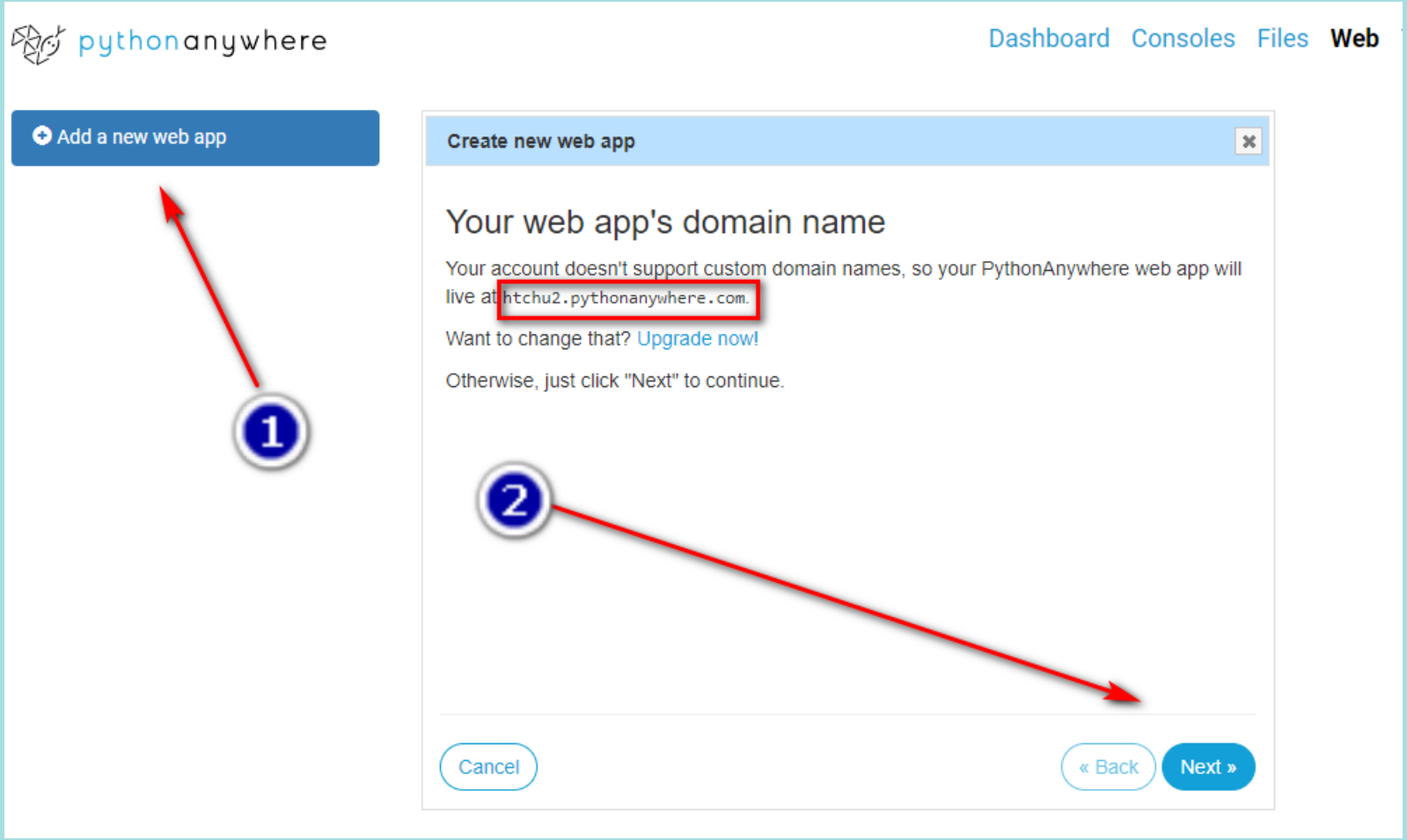

pythonanywhere
Dashboard Consoles Files Web Tasks Databases
Up here you will see instructions walking you through the key features of our Education beta.
(If after closing this helper, you want to go through it again – or try another one – go to the Help page)
Close this tutorial
CPU Usage: 0% used – 0.00s of 100s. Resets in 15 hours, 58 minutes More Info
Scheduled tasks
Server time: 02:15 UTC
Daily, at 15 : 33 UTC For example, python3.6 /home/yourusername/myfolder/myscript.p Optional description Create
Frequency Time Command Description Expiry Actions
You have no tasks yet.
Your Scheduled task is a script that will run every day at a time of your choosing — you can use it to do stuff like scraping websites, or checking that your server is running.
Paying users can schedule several tasks, can run them both hourly and daily, and they never expire. Just sayin'...

Step 1: Go to Web tab





Step 2: Add a new web app



Step 3: Select a Python Web framework and a Python version

Select a Python Web framework

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

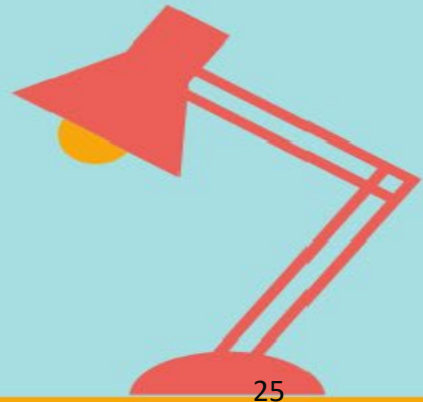
- » **Django**
- » **web2py**
- » **Flask**
- » **Bottle**
- » **Manual configuration** (including virtualenvs)

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

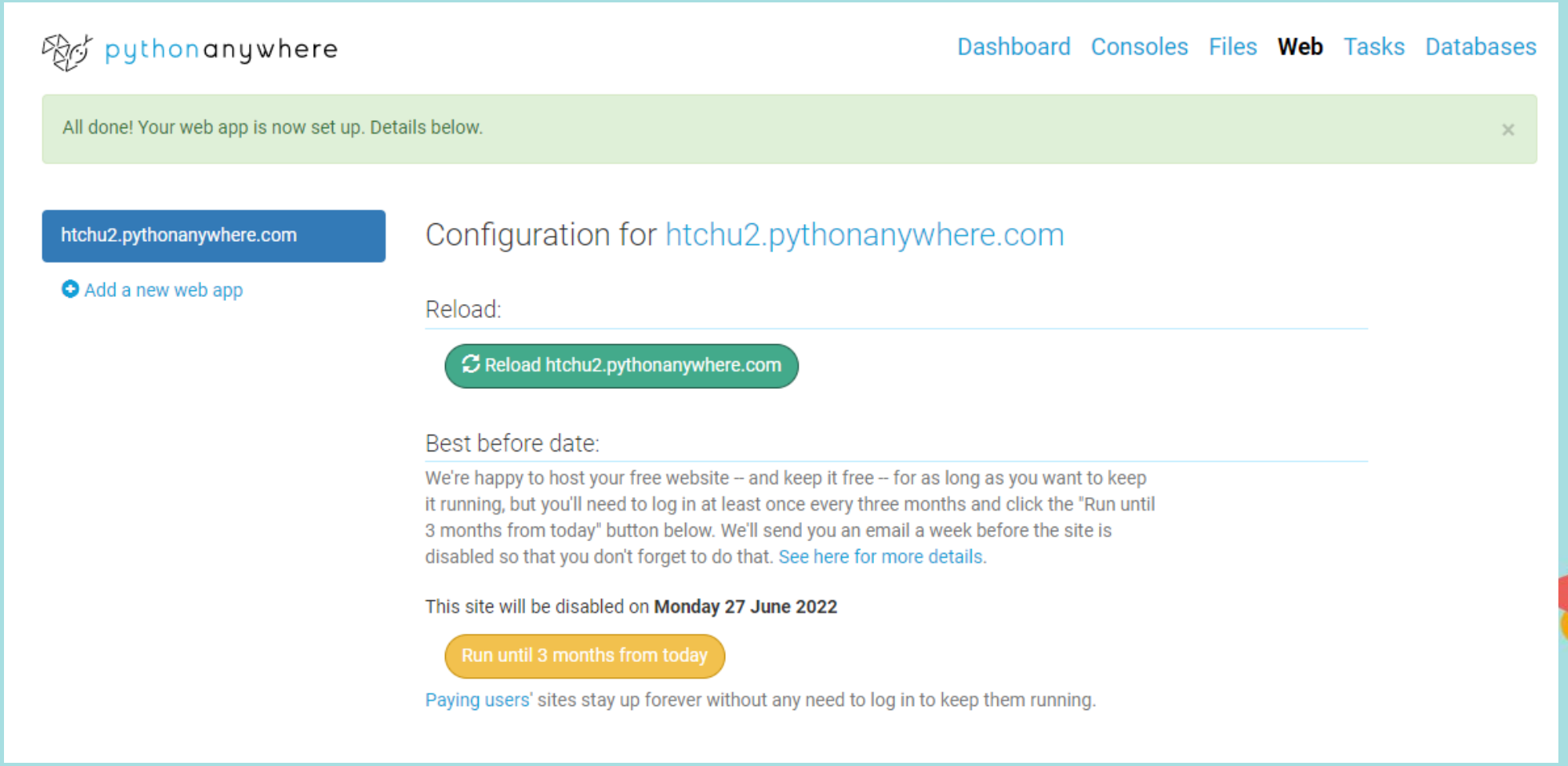
Select a Python version

- » **Python 3.6** (Flask 2.0.0)
- » **Python 3.7** (Flask 2.0.0)
- » **Python 3.8** (Flask 2.0.0)
- » **Python 3.9** (Flask 2.0.0)

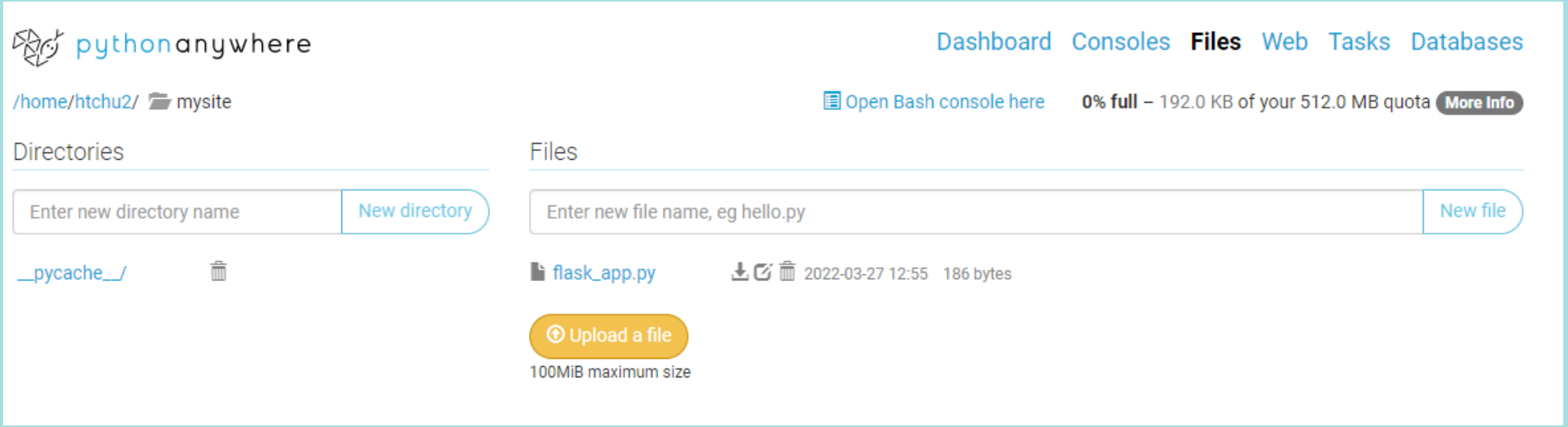
Note: If you'd like to use a different version of Flask to the default version, you can use a virtualenv for your web app. There are instructions here.



Step 4: Quick start new Flask project



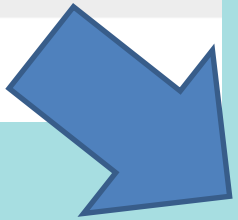
Step 5: check the files

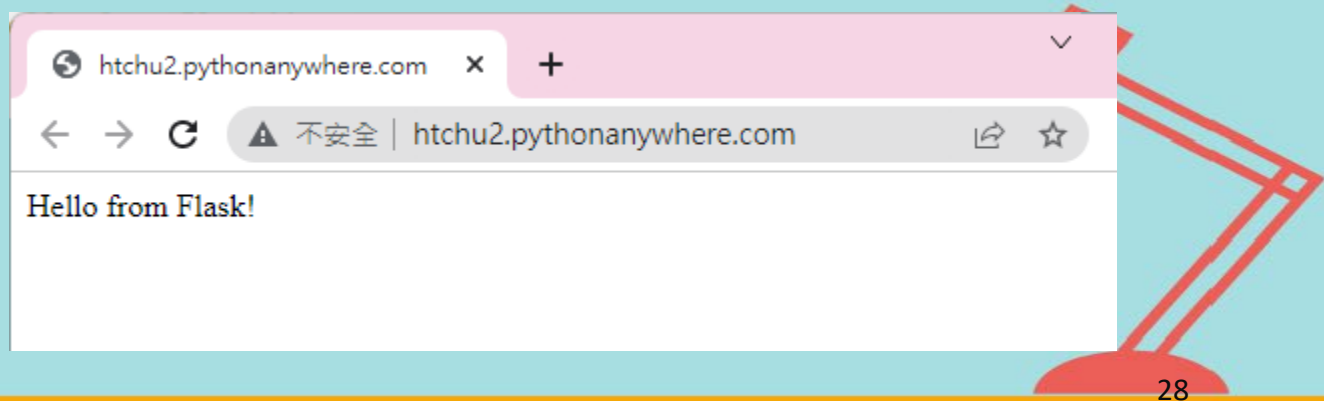


Step 6: check the program and the web app

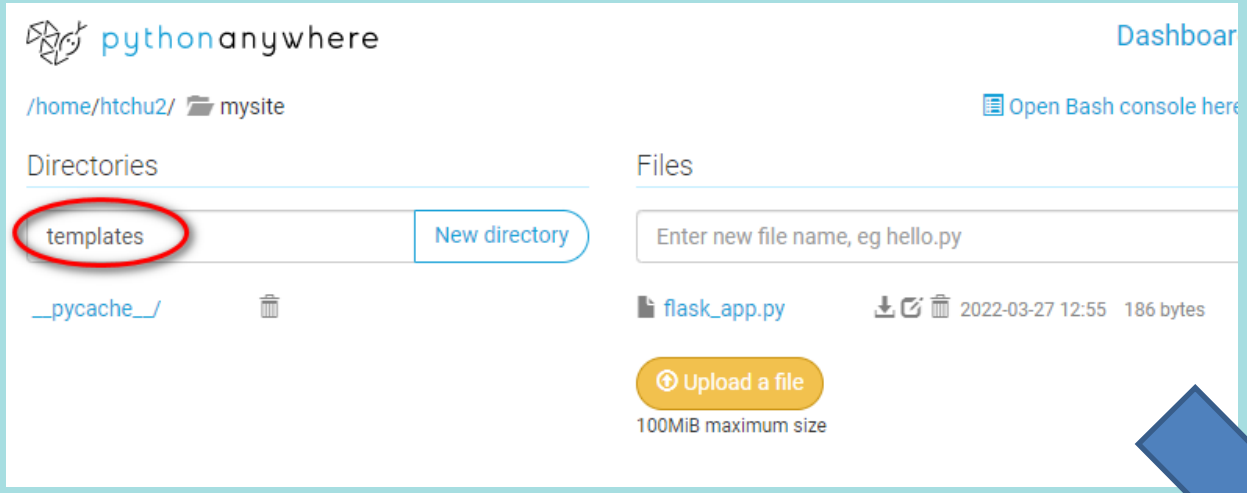
/home/htchu2/mysite/flask_app.py

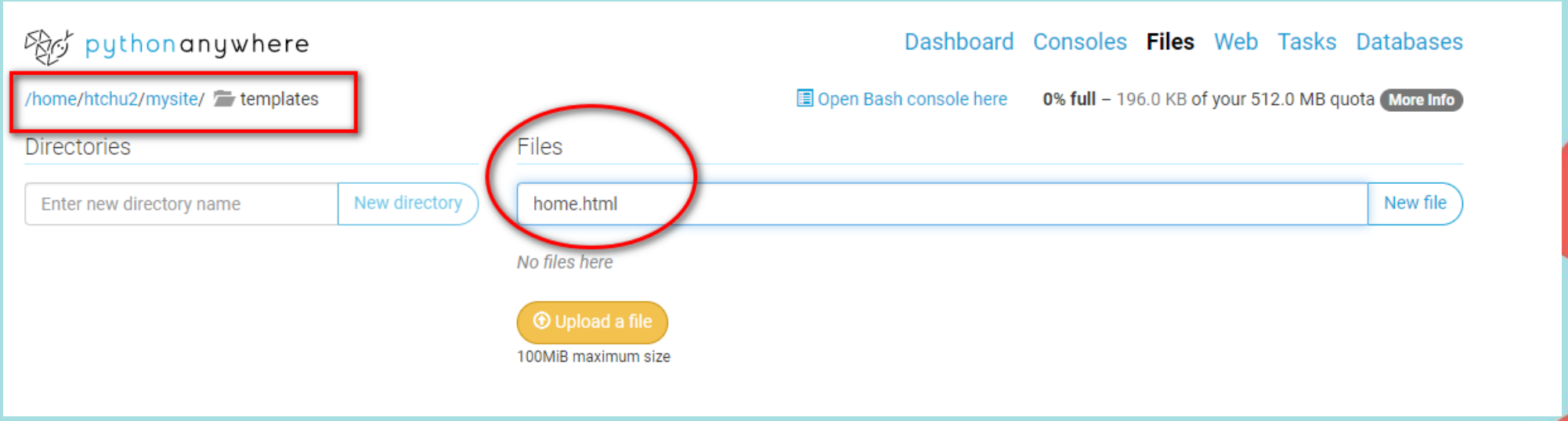
```
1
2  # A very simple Flask Hello World app for you to get started with...
3
4  from flask import Flask
5
6  app = Flask(__name__)
7
8  @app.route('/')
9  def hello_world():
10     return 'Hello from Flask!'
11
12
```



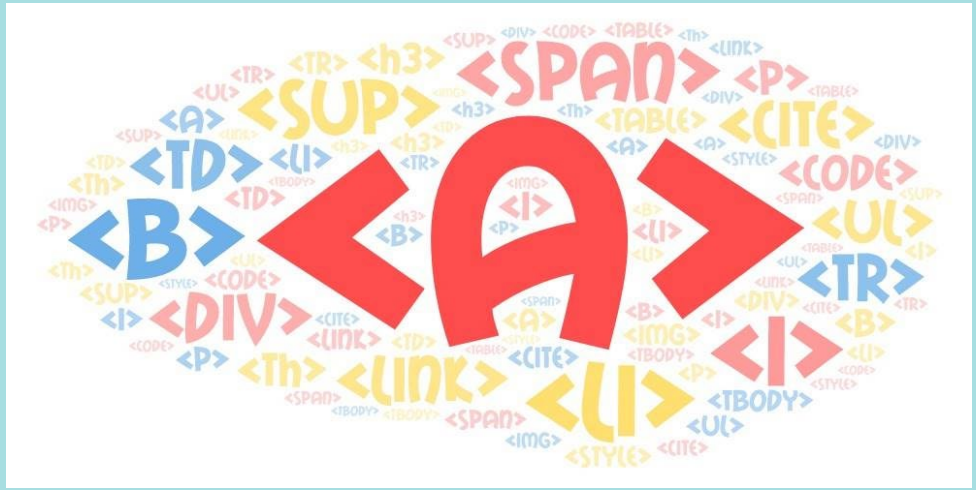


Step 7: add html templates





Step 8: edit home.html



- <h1>Top 10 HTML tags</h1>

In this article we will explain our list of top 10 HTML tags.

 <h2>First tag is </h2>Bold is all about making words more important.

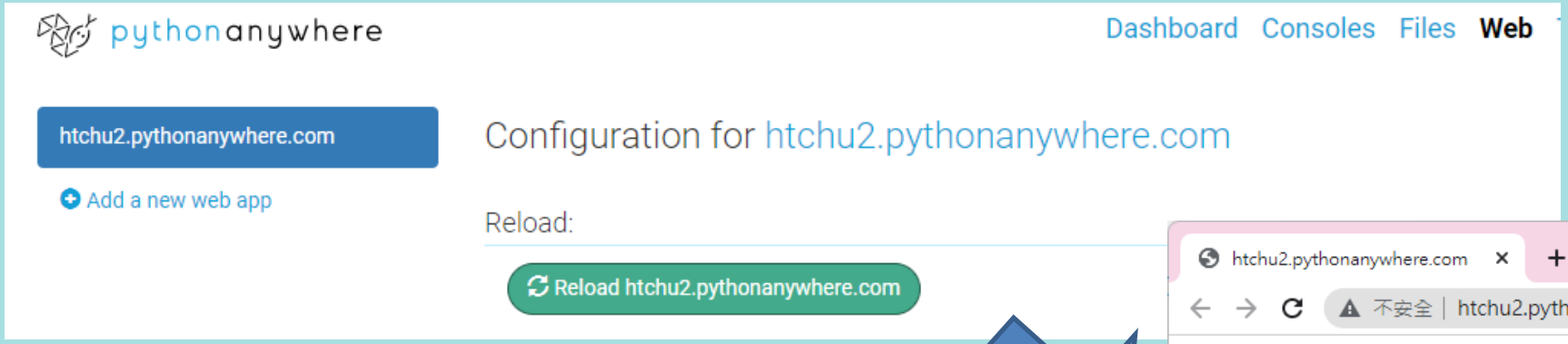
https://www.codebrainer.com/blog/top-10-html-tags

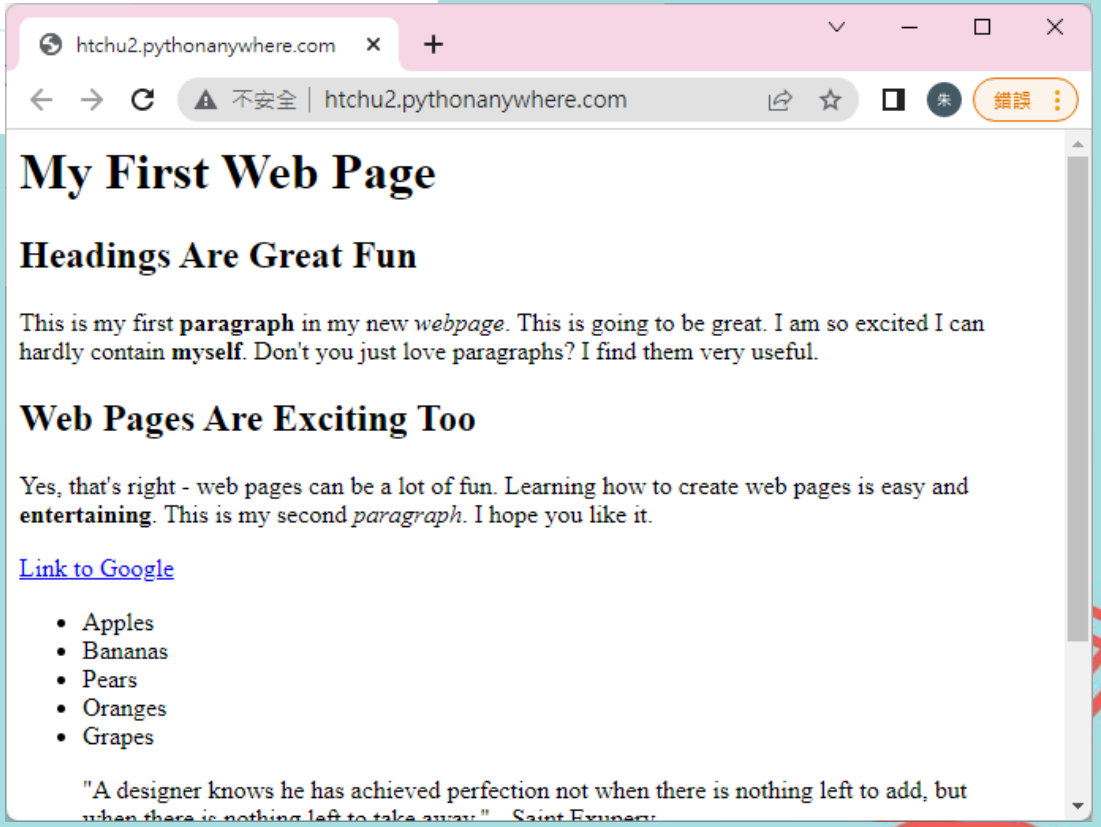
Step 9: edit flask_app.py

/home/htchu2/mysite/flask_app.py

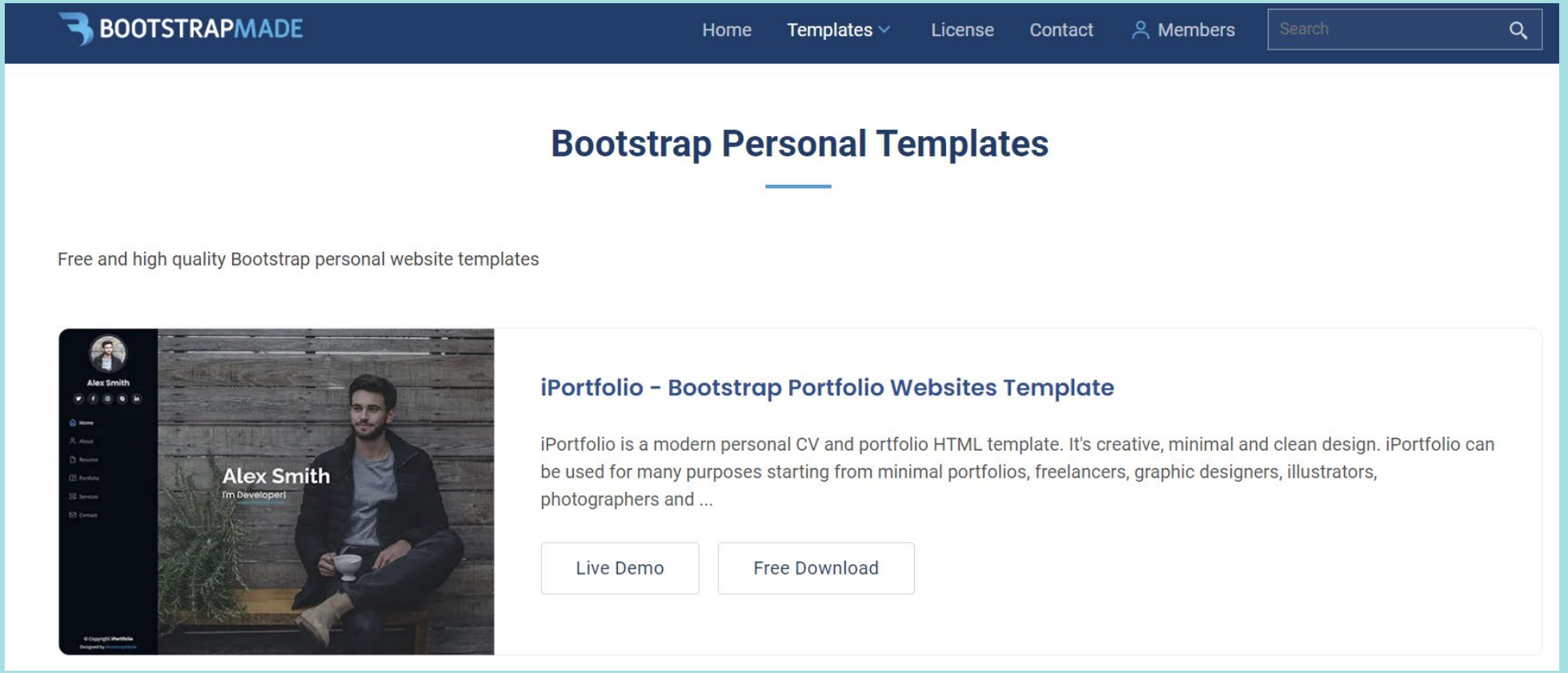
```
1
2  # A very simple Flask Hello World app for you to get started with...
3
4  from flask import Flask, render_template
5
6  app = Flask(__name__)
7
8  @app.route('/')
9  def home():
10     return render_template("home.html")
11
```

Step 10: Reload web



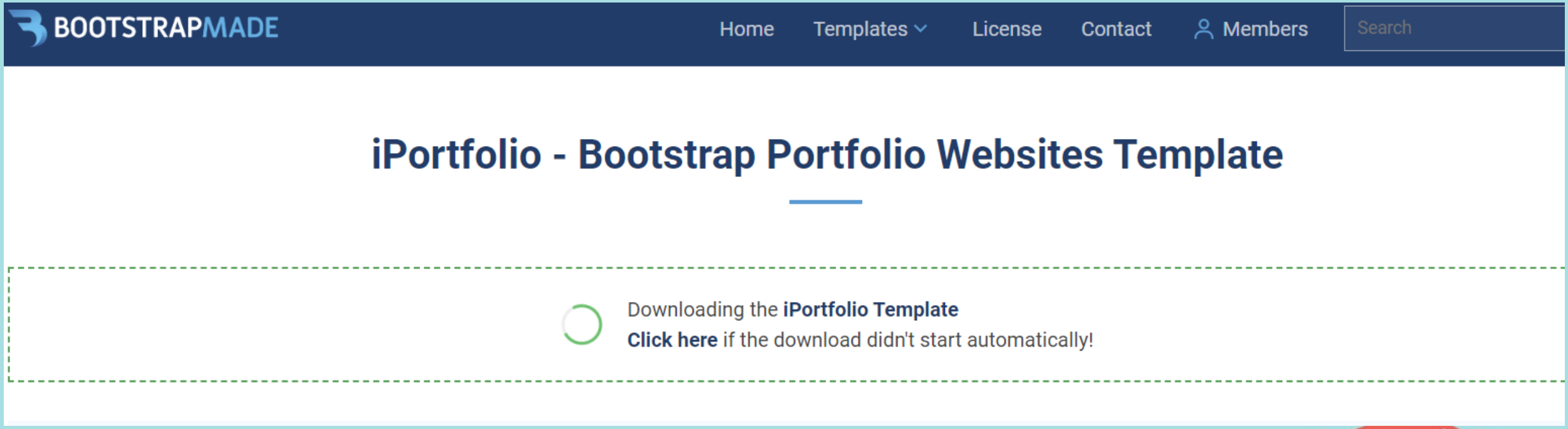


Step 11: Bootstrap Personal Templates Website

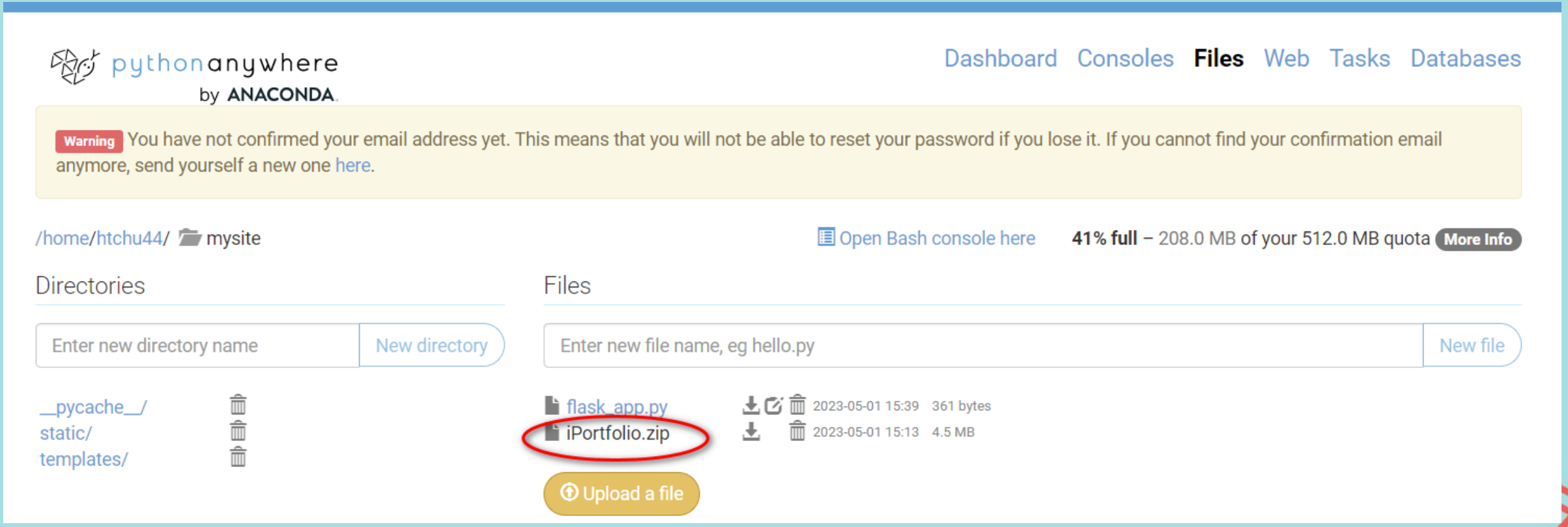


https://bootstrapmade.com/bootstrap-personal-templates/

Step 12: Downloading the iPortfolio Template



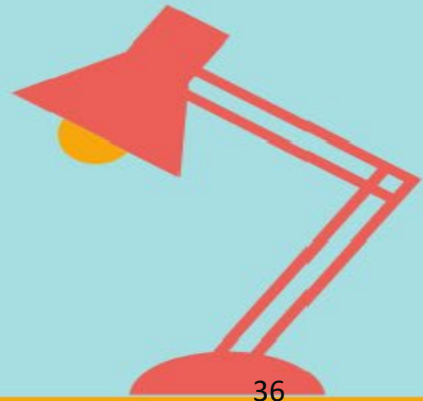
Step 13: Upload iPortfolio.zip to mysite



Step 14: Unzip iPortfolio.zip

Bash console 28503198

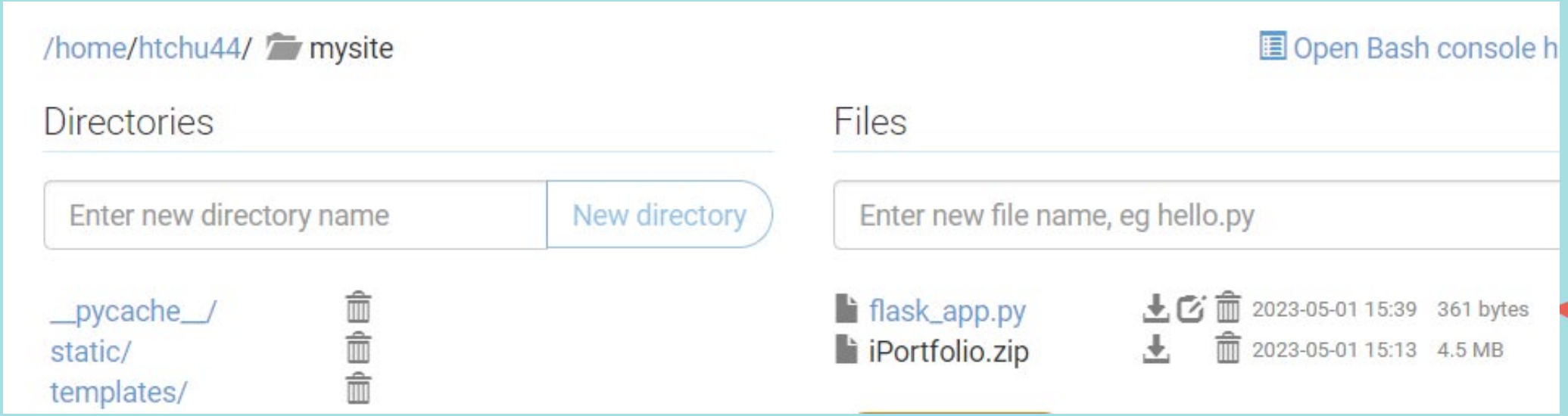
```
/home/htchu44
15:14 ~ $ cd mysite/
15:14 ~/mysite $ unzip iPortfolio.zip
Archive:  iPortfolio.zip
  inflating: iPortfolio/assets/css/style.css
 extracting: iPortfolio/assets/img/apple-touch-icon.png
 extracting: iPortfolio/assets/img/favicon.png
 extracting: iPortfolio/assets/img/hero-bg.jpg
 extracting: iPortfolio/assets/img/portfolio/portfolio-1.jpg
 extracting: iPortfolio/assets/img/portfolio/portfolio-2.jpg
```



Step 15: move mvfiles

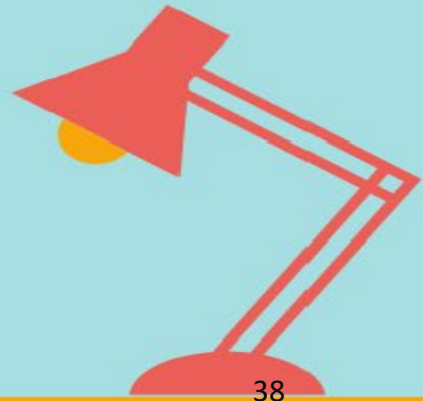
```
~/mysite $ mv iPortfolio/*.html templates/.
~/mysite $ mv iPortfolio static
```

mv iPortfolio/*.html templates/.
mv iPortfolio static

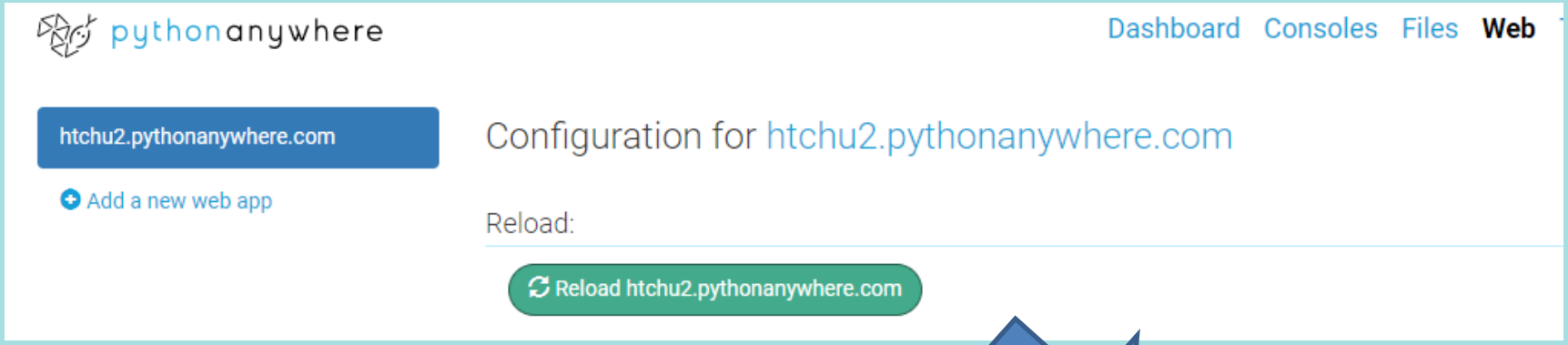


Step 16: edit flask_app.py

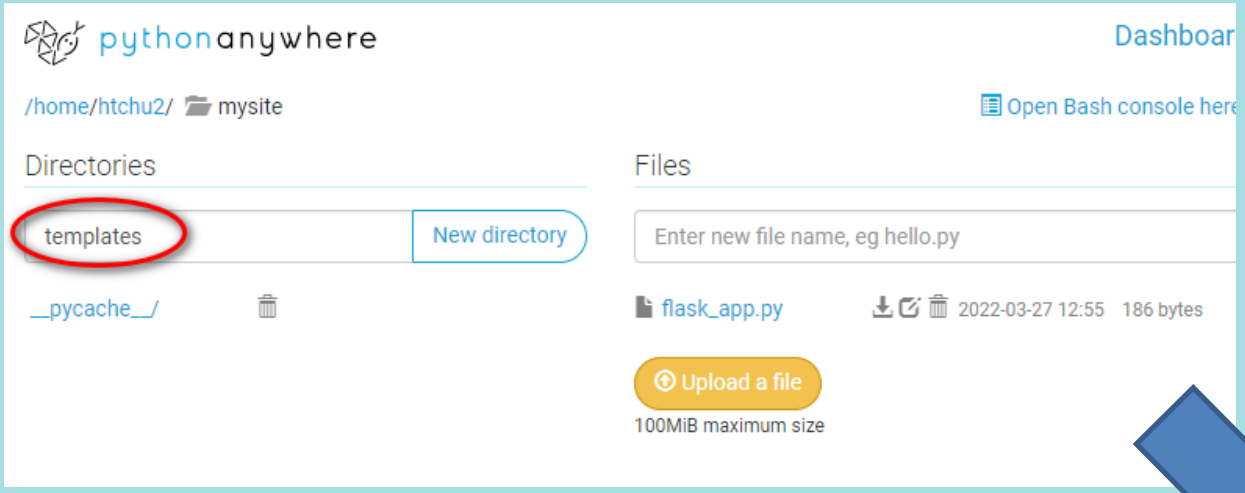
```
from flask import Flask, render_template
from flask import send_from_directory

app = Flask(__name__)

@app.route('/')
def hello_world():
    return render_template("index.html")

@app.route('/assets/<path:path>')
def send_report(path):
    return send_from_directory('static/assets', path)
```



Step 17: Reload web

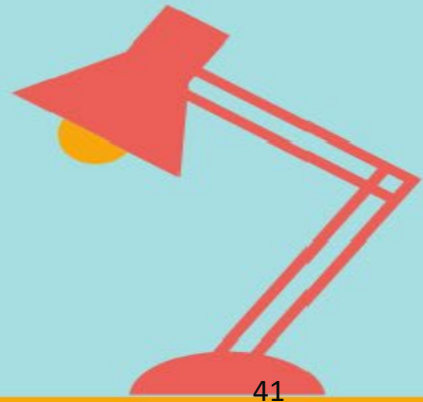


Step 17: add html templates



作業3

- 選擇一個Bootstrap Personal Templates
- 將網頁上的資料及照片換成你個人的資料
- 參考老師的教學步驟，重新上傳至pythonanywhere網站
- 繳交pythonanywhere網址



Thanks!

Q&A