

BÀI THỰC HÀNH FLASK

Nội dung

BÀI THỰC HÀNH FLASK	1
Bài thực hành Flask	2
Bài 1: Kiểm tra cài đặt gói flask trên máy và cài đặt gói Flask (nếu cần)	2
Bài 2: Xây dựng trang web “Xin chào!”	3
Bài 4: Điều chỉnh chữ ‘Xin chào!’ và điều chuyển về trang Đại học Văn Lang	4
Bài 5: Thay đổi cổng web và thêm định tuyến trang con.....	5
Bài 6: Định tuyến đến một môn học	6
Bài 7: Web templating: truyền biến đơn	9
Bài 8: Responses	11
Bài 9: Web templating: Truyền biến kiểu danh sách	12
Bài 10: Các tập tin tĩnh (static files)	12
Bài 11: Định tuyến nội bộ url_for.....	13
Bài 12: Triển khai web theo mô hình MVC.....	13
Bài 13: Gói request và GET/POST	14
Bài 14: Cookies.....	15
Bài 15: Bootstrap Integration with Flask-Bootstrap	16
Bài 16: Xử lý lỗi không tìm được trang 404	17
Bài 17: Session.....	18
Bài 18: Về form	19
Bài 19: Về form classes	20
Bài 20: Message Flashing	20
Bài tập:	21
PHỤ LỤC: ĐỌC DỮ LIỆU COVID CÁC NƯỚC VÀ VÙNG LÃNH THỔ	22

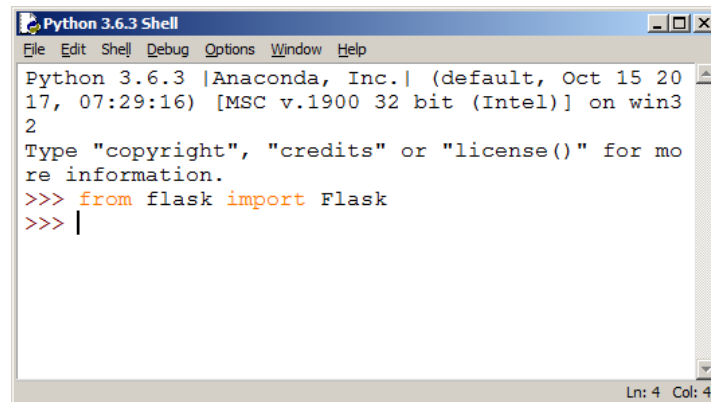
Bài thực hành Flask

Bài 1: Kiểm tra cài đặt gói flask trên máy và cài đặt gói Flask (nếu cần)

Thực hiện:

Trong IDLE, chúng ta dễ dàng kiểm tra thông qua lệnh

`from flask import Flask`

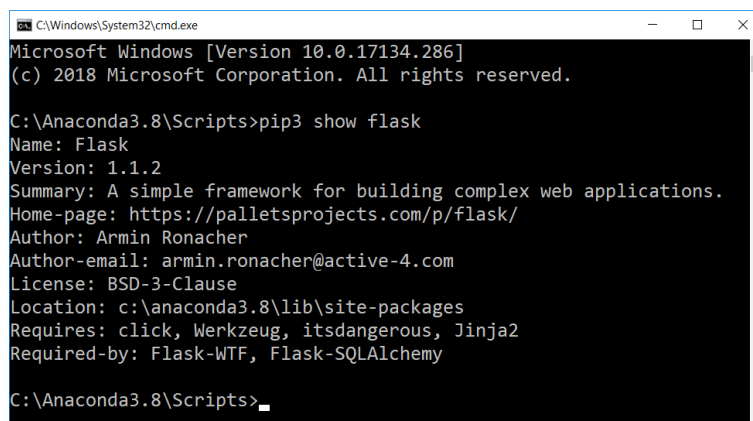


```
Python 3.6.3 Shell
File Edit Shell Debug Options Window Help
Python 3.6.3 |Anaconda, Inc.| (default, Oct 15 2017, 07:29:16) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> from flask import Flask
>>> |
```

Sinh viên hãy cho biết thông tin trên máy của các bạn:.....

.....

Ngoài ra, chúng ta có thể sử dụng lệnh: **pip3 show flask** để kiểm tra các thông tin của gói Flask trên hệ thống khi đã cài đặt.



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.17134.286]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Anaconda3.8\Scripts>pip3 show flask
Name: Flask
Version: 1.1.2
Summary: A simple framework for building complex web applications.
Home-page: https://palletsprojects.com/p/flask/
Author: Armin Ronacher
Author-email: armin.ronacher@active-4.com
License: BSD-3-Clause
Location: c:\anaconda3.8\lib\site-packages
Requires: click, Werkzeug, itsdangerous, Jinja2
Required-by: Flask-WTF, Flask-SQLAlchemy

C:\Anaconda3.8\Scripts>
```

Sinh viên hãy cho biết thông tin trên máy của các bạn:.....

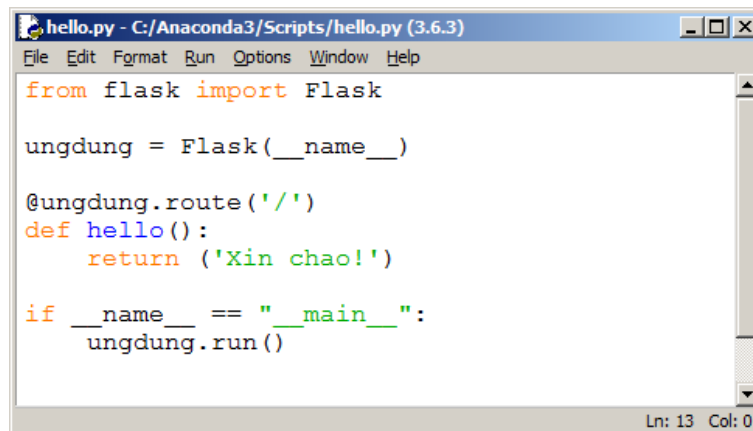
.....

Trong trường hợp gói Flask chưa được cài đặt, chúng ta có thể cài đặt thông qua lệnh:

pip install flask (tại dấu nhắc hệ điều hành).

Bài 2: Xây dựng trang web “Xin chào!”

Sinh viên sử dụng IDLE có thể tạo mới 1 tập tin đặt tên là hello.py như sau:



```
hello.py - C:/Anaconda3/Scripts/hello.py (3.6.3)
File Edit Format Run Options Window Help
from flask import Flask
ungdung = Flask(__name__)
@ungdung.route('/')
def hello():
    return ('Xin chào!')
if __name__ == '__main__':
    ungdung.run()
Ln: 13 Col: 0
```

Thực hiện: Viết các đoạn code như sau:

```
from flask import Flask
```

```
ungdung = Flask(__name__)
```

```
@ungdung.route('/')
```

```
def hello():
```

```
    return ('Xin chào!')
```

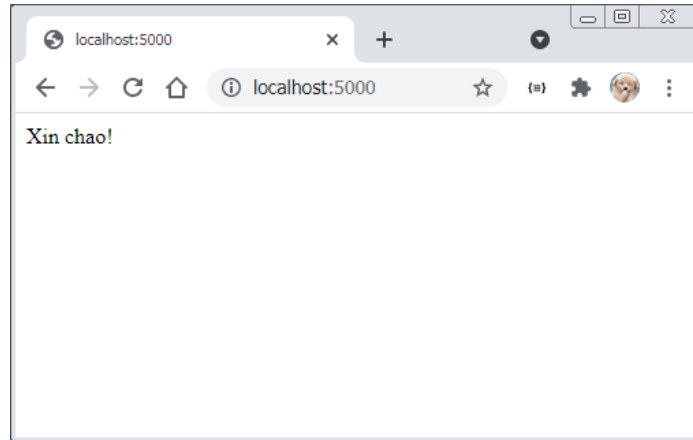
```
if __name__ == "__main__":    # (lưu ý là 2 dấu gạch dưới _ ở mỗi vị trí name và main)
```

```
    ungdung.run()
```

Sau khi tạo xong ứng dụng, sinh viên bấm F5 để chạy ứng dụng web.

```
>>>
===== RESTART: C:/Anaconda3/Scripts/hello.py =====
* Running on http://127.0.0.1: ???? / (Press CTRL+C to quit)
```

Cổng mặc định của trang web Flask là: Và kết quả trang web là:



Lưu ý: về unicode, đối với phiên bản Python 2.x, unicode được sử dụng thông qua 2 lệnh sau:

```
import sys
```

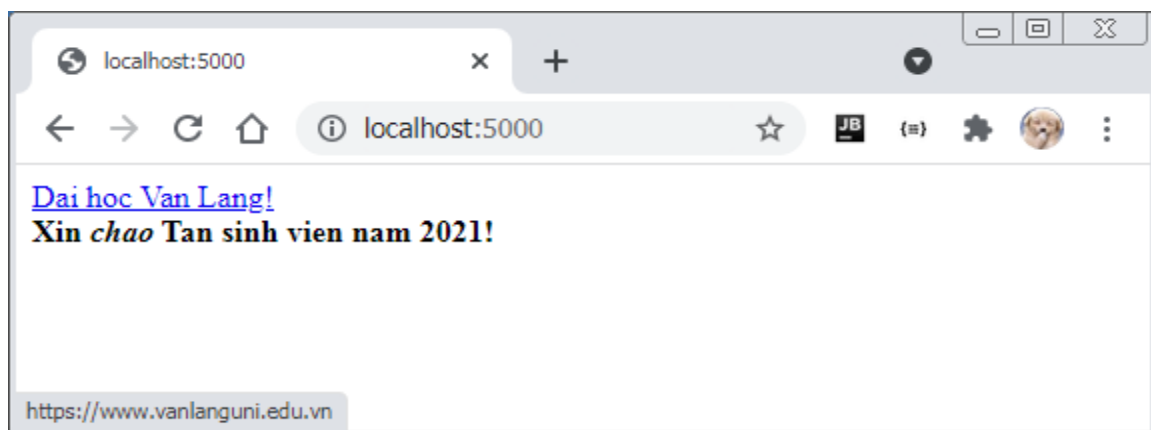
```
reload(sys)
```

```
sys.setdefaultencoding("utf-8")
```

Tuy nhiên, đối với các phiên bản Python 3.8, hệ thống sẽ tự nhận dạng kí tự Unicode (theo mã UTF-8)

Bài 4: Điều chỉnh chữ ‘Xin chào!’ và điều chuyển về trang Đại học Văn Lang

Yêu cầu: Trang web được viết như hình bên dưới:



Cụ thể:

- Thêm chuỗi Đại học Văn Lang và tạo liên kết với trang web của trường Đại học Văn Lang.
- Điều chỉnh chữ ‘Xin chào!’ sang in đậm chữ ‘Xin’, in nghiêng và đậm chữ ‘chào!’

- Sau chữ 'Xin chào' bổ sung cụm 'Tan sinh viên 2021'. Yêu cầu: trong đó, số 2021 là con số được tạo từ ngày hiện tại của web server Flask.

Lưu ý: Địa chỉ trang web của Trường Đại học Văn Lang là: <https://www.vanlanguni.edu.vn>

Thực hiện: Chương trình được điều chỉnh phần hàm hello() của trang chủ ('/') như sau:

```
from flask import Flask
```

```
ungdung = Flask(__name__)
```

```
@ungdung.route('/')
def hello():
```

```
    tentruong = ' Dai hoc Van Lang!'
```

```
    lienket = '<a href="https://www.vanlanguni.edu.vn">' +tentruong+' </a> <br>'
```

```
    chuoii = lienket
```

```
    import datetime
```

```
    nam = datetime.date.today().year
```

```
    chuoii = chuoii + ' <b>Xin <i>chào</i> Tan sinh viên năm ' + str(nam) + '!</b> '
```

```
    return chuoii
```

```
if __name__ == "__main__":
```

```
    ungdung.run()
```

Nhận xét: Như vậy, phần kết xuất trả về của hàm hello() chính là một trang HTML

Bài 5: Thay đổi cổng web và thêm định tuyến trang con

Yêu cầu: Từ ứng dụng bài 4 hãy:

- Thay đổi cổng mặc định của trang hiện hành thành cổng 5050.
- Tạo thêm 1 trang con có tên là **monhoc**, nghĩa là trang <http://localhost:5050/monhoc>

Thực hiện:

Trong hàm main(), chúng ta điều chỉnh cổng trang web bằng cách xác định cổng trên tham số port như sau:

```
if __name__ == "__main__":  
    ugdung.run(port=5050)
```

Nghĩa là bổ sung: `ungdung.run(port=5050)`

Sau đó, chúng ta thêm 1 trang mới bằng điều trang route như sau:

@ungdung.route('/monhoc')

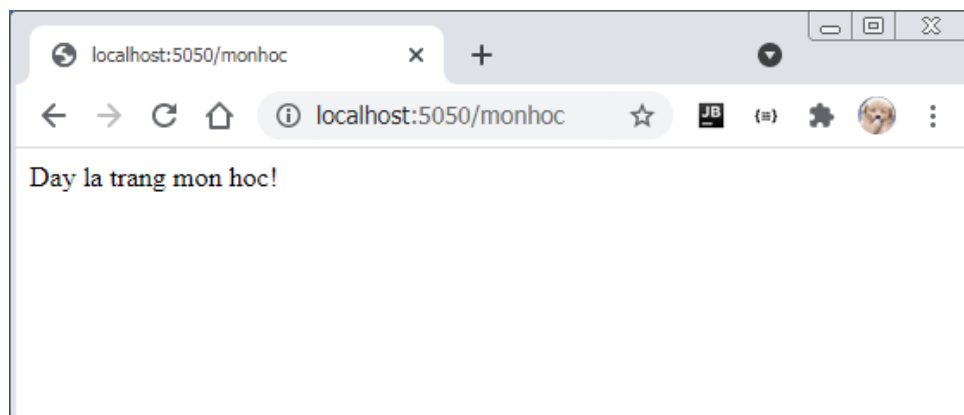
def learn():

return "Day la trang mon hoc!"

Như hình bên dưới đây

```
from flask import Flask, render_template  
lienkett = '<a href="https://www.vanlang  
chuoi = lienket  
import datetime  
nam = datetime.date.today().year  
chuoi = chuoi + ' <b>Xin <i>chao</i> Ta  
return chuoi  
  
@ungdung.route('/monhoc')  
def learn():  
    return "Day la trang mon hoc!"  
  
if __name__ == "__main__":  
    ugdung.run(port=5050)
```

Và thực thi chương trình bằng F5 để kiểm tra trang web:



Bài 6: Định tuyến đến một môn học

Yêu cầu:

- Yêu cầu 1: Bổ sung /monhoc/<tên-môn> cụ thể. Ví dụ: /monhoc/python thì trang web trả về môn học đang xem là 'python', viết hoa tên môn học.
- Yêu cầu 2: Tương tự, viết 1 liên kết /sinhvien/<khóa> sẽ ra thông báo liên kết với sinh viên khóa đó. Ví dụ: <http://localhost:5050/sinhvien/2021> sẽ ra trang web của sinh viên 2021.

Thực hiện:

Yêu cầu 1: Thay đổi bài trước như sau:

```
@ungdung.route('/monhoc/')
def learn():
    chuoai = "Day la trang mon hoc!"
    return chuoai

@ungdung.route('/monhoc/<tenmon>')
def subjects(tenmon):
    chuoai = "Day la trang mon hoc"
    monhoc = str(tenmon).upper()
    if monhoc == "":
        chuoai = chuoai + "!"
    else:
        chuoai = chuoai + " " + monhoc
    return chuoai
```

@ungdung.route('/monhoc/')

def learn():

chuoai = "Day la trang mon hoc!"

return chuoai

@ungdung.route('/monhoc/<tenmon>')

def subjects(tenmon):

chuoai = "Day la trang mon hoc"

monhoc = str(tenmon).upper()

if monhoc == "":

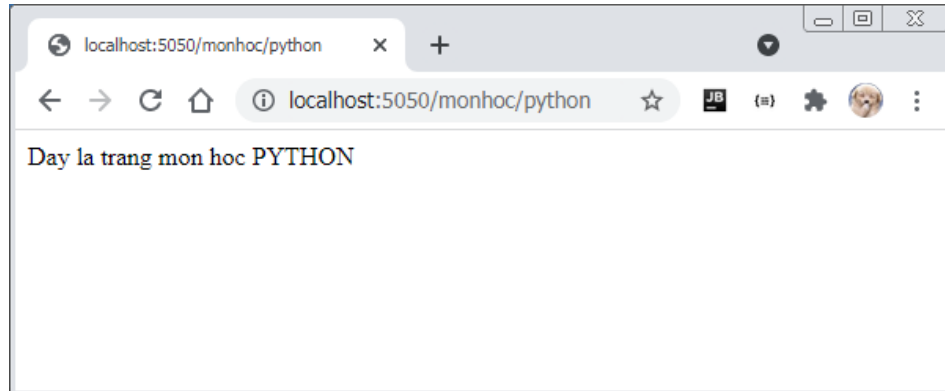
chuoai = chuoai + "!"

else:

chuoi = chuoi + " " + monhoc

return chuoi

Và thực thi ứng dụng bằng F5:



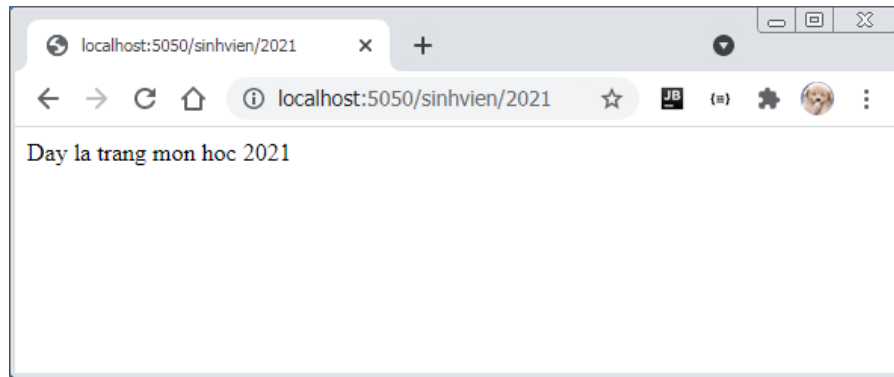
Yêu cầu 2: Bổ sung như sau:

Bổ sung 2 **route** và 2 hàm **def** để xác định trang sinh viên và năm học của sinh viên đó

```
@ungdung.route('/sinhvien/')
def students():
    chuoi = "Day la trang cac khoa hoc!"
    return chuoi

@ungdung.route('/sinhvien/<int:namhoc>')
def school_year(namhoc):
    chuoi = "Day la nam hoc"
    nam = str(namhoc).upper()
    if nam == "":
        chuoi = chuoi + "!"
    else:
        chuoi = chuoi + " " + nam
    return chuoi
```

Kết quả là:



Lưu ý: Các kiểu dữ liệu được hỗ trợ:

<i>string</i>	Chuỗi kí tự, không có dấu /
<i>int</i>	Giá trị nguyên
<i>float</i>	Giá trị thực
<i>path</i>	Như mặc định nhưng cho phép dấu / (slashes)
<i>any</i>	cho phép giống với 1 trong những cái trong danh sách
<i>uuid</i>	Chấp nhận chuỗi UUID

Bài 7: Web templating: truyền biến đơn

Web templating là truyền giá trị vào một tập tin html có sẵn giao diện để thể hiện các giá trị. Để chúng ta thực hiện như sau:

- Khai báo: **from flask import render_template**
- Tạo thư mục templates (viết chữ thường) sau đó sao chép tập tin html vào thư mục **templates** (lưu ý: phải đúng luôn cả hoa thường)

Lưu ý: có hai phương án để đặt file html để sử dụng được lệnh **render_template** là:

Case 1: a module:

```
/application.py  
/templates  
    /hello.html
```

Case 2: a package:

```
/application  
    /__init__.py  
    /templates  
        /hello.html
```

- Phần return của các trang sẽ trả về bằng lệnh `render_template` với các biến đơn theo sau, ví dụ:
`return render_template('abc.html', bienweb = bienpython)`
Trong đó, biến `bienpython` là 1 biến xử lý trong python và biến `bienweb` được đặt trong tập tin HTML (trường hợp này là `abc.html`) với cú pháp: `{{ bienweb }}`

Minh họa:

- Đoạn lệnh Flask:

```
from flask import render_template  
  
@app.route('/hello/')  
@app.route('/hello/<name>')  
def hello(name=None):  
    return render_template('hello.html', name=name)
```

- Đoạn lệnh tập tin `hello.html`:

```

<!doctype html>

<title>Hello from Flask</title>

{% if name %}

    <h1>Hello {{ name }}!</h1>

{% else %}

    <h1>Hello, World!</h1>

{% endif %}

```

Lưu ý: phải có dấu cách giữa các chữ {{, }} với chữ name.

Lưu ý: chúng ta có thể truyền nhiều biến, mỗi biến cách nhau 1 dấu phẩy (.). Ví dụ:

```
return render_template('abc.html', bienweb1 = bienpython1, bienweb2 = bienpython2,...)
```

Bài 8: Responses

Khi một chức năng dạng xem cần phản hồi bằng mã trạng thái khác, nó có thể thêm mã số làm giá trị trả về thứ hai sau văn bản phản hồi.

Ví dụ: Hàm following view trả về mã trạng thái 400, mã cho lỗi yêu cầu không hợp lệ:

```

@app.route('/')
def index():
    return '<h1>Bad Request</h1>', 400

```

Chương trình sau tạo một đối tượng phản hồi và sau đó đặt một cookie trong đó:

```

from flask import make_response

@app.route('/')
def index():
    response = make_response('<h1>This document carries a
    cookie!</h1>') response.set_cookie('answer', '42')
    return response

```

Tương tự, hãy viết chương trình tạo ra phản hồi dùng hàm mà Flask đã hỗ trợ **redirect()**.

Bài 9: Web templating: Truyền biến kiểu danh sách

Để truyền một biến dạng danh sách (list) hoặc các bộ dữ liệu tuple, chúng ta sử dụng kỹ thuật `render_template` với tham số truyền danh sách tương tự truyền biến đơn.

Giả định, chúng ta có 1 danh sách các ngôn ngữ:

```
languages = [ {'STT':1, 'ten': "Python"}, {'STT':2, 'ten': "Java"} , {'STT':3, 'ten': "C++"}]
```

```
languages.append({'STT':4, 'ten': ".NET" })
```

```
languages.append({'STT':5, 'ten': "Matlab" })
```

và thực hiện lệnh truyền như sau:

```
return render_template('abc.html', ngon_ngu = languages)
```

Khi đó, các lệnh dưới đây minh họa việc sử dụng như sau trong tập tin HTML `abc.html`:

```
<table>
{% for ketqua in ngon_ngu %}
    <tr>
        <td> {{ ketqua.STT }} </td>
        <td> {{ ketqua.ten }} </td>
    </tr>
{% endfor %}
</table>
```

Bài 10: Các tập tin tĩnh (static files)

Các tập tin CSS và Javascripts cần được khai báo trong thư mục tĩnh (static folder) để tham chiếu và liên kết. Chúng ta chỉ cần bổ sung tham số `static_folder` với thư mục tĩnh trên ổ đĩa. Ví dụ:

```
app = Flask(__name__, static_folder='C:\\flask_covid\\templates\\static')
```

Khi đó, các tập tin ảnh, liên kết trong file html đều trỏ đến thư mục tĩnh bên trên.

Ví dụ: Khi khai báo như sau thì chúng ta có thể sử dụng tương ứng như sau:

Cách khai báo cho cả thư mục:

```
app = Flask(__name__, static_folder='C:\\web\\templates\\static' )
```

Lúc này cả thư mục `'C:\\web\\templates\\static'` sẽ được gọi là thư mục chứa các tập tin static.

Khi đó, trong trang web, muốn sử dụng hình ảnh, tài liệu thì chỉ việc “trỏ” vào thư mục static:

```
<link rel="icon" href="static/img/icon.png">
<link rel="stylesheet" href="static/css/bootstrap.min.css">
```

Bài 11: Định tuyến nội bộ url_for

Để định tuyến cục bộ, gói url_for được sử dụng:

```
from flask import abort, redirect, url_for

@app.route('/')
def index():
    return redirect(url_for('login'))

@app.route('/login')
def login():
    abort(401)
```

Lưu ý: abort là để đưa ra thông báo lỗi theo mã lỗi.

Bài 12: Triển khai web theo mô hình MVC

Kinh nghiệm để triển khai web theo mô hình MVC, chúng ta xây dựng một số tập tin Python như sau:

Một kiến trúc trang web Flask gồm các tập tin Python như sau:

+ Thư mục gốc của Flask sẽ chứa các tập tin như sau:

application.py (hoặc tên khác): để quản lý chung

config.py (hoặc tên khác): để quản lý cấu hình

ví dụ: cần kết nối CSDL

database.py: nơi chứa các mô tả, các xử lý của CSDL

services.py: nơi chứa các hàm xử lý (control)

`models.py`: (như file `database.py`) thường chứa mô tả csdl

Thư mục tên là **templates** (tên chính xác) để chứa

template về web, javascript, css,...

Để Python hiểu các tập tin cùng thư mục, một lệnh được thực thi

```
if __name__ == '__main__' and __package__ is None:
```

```
    from os import sys, path
```

```
    sys.path.append(path.dirname(path.dirname(path.abspath(__file__))))
```

Sau lệnh này, từ tập tin `application.py`, chúng ta hoàn toàn có thể import các tập tin hoặc thư viện hàm cùng thư mục. Ví dụ:

```
>>> from services import ham1
```

Trong đó `ham1` là 1 function trong tập tin `services.py`

Bài 13: Gối request và GET/POST

Lưu ý: Để trang thực hiện các phương thức GET/POST (lấy dữ liệu từ form web), chúng ta thêm tham số `methods` vào trong định nghĩa của route trang. Minh họa như sau.

```
@app.route('/upload', methods=['GET', 'POST'])
```

Khi sử dụng, chúng ta cần lưu ý:

- Bổ sung `from flask import request` này vào bên trên của trang.
- Nhận diện các yêu cầu GET/POST như: `if request.method == 'POST':`

Ví dụ:

```

from flask import request

@app.route('/upload', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        f = request.files['the_file']
        f.save('/var/www/uploads/uploaded_file.txt')
    ...

```

Lưu ý: Để xử lý những trang web có nhận và chuyển thông tin. Ví dụ: trang web có nút Tìm kiếm → người sử dụng phải điền thông tin đi: tìm kiếm hoặc Thêm mới, Cập nhật, Xóa... (tóm lại là trang web có tương tác) thì trong khai báo chúng ta phải thêm phương thức GET và POST cho nó. Ví dụ: tạo trang /timkiem

```
@app.route('/timkiem', methods = ['GET', 'POST'])
```

```
def ham_nay_cho_trang_tim_kiem():
```

```
    #....
```

```
    noidung_cantim = request.args.get('txtnoidung', '')
```

Lưu ý: **'txtnoidung'** là 1 thẻ text trong tập tin html

Bài 14: Cookies

Đọc và xử lý Cookie thông qua gói request như sau:

- Đọc:

```

from flask import request

@app.route('/')
def index():
    username = request.cookies.get('username')

    # use cookies.get(key) instead of cookies[key] to not get a
    # KeyError if the cookie is missing.

```

- Ghi:

```

from flask import make_response

@app.route('/')
def index():
    resp = make_response(render_template(...))
    resp.set_cookie('username', 'the username')
    return resp

```

Bài 15: Bootstrap Integration with Flask-Bootstrap

Tải gói bootstrap bằng cách (câu lệnh pip install flask-bootstrap)

Ví dụ về khởi tạo Flask-Bootstrap

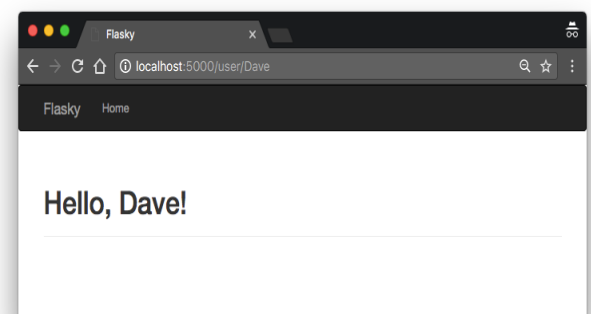
Example 3-4. hello.py: Flask-Bootstrap initialization

```

from flask_bootstrap import Bootstrap
# ...
bootstrap = Bootstrap(app)

```

Chương trình sử dụng Flask-Bootstrap




```

{% extends "bootstrap/base.html" %}

{% block title %}Flasky{% endblock %}

{% block navbar %}
<div class="navbar navbar-inverse" role="navigation">
  <div class="container">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle"
        data-toggle="collapse" data-target=".navbar-collapse">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="/">Flasky</a>
    </div>
    <div class="navbar-collapse collapse">
      <ul class="nav navbar-nav">
        <li><a href="/">Home</a></li>
      </ul>
    </div>
  </div>
</div>
{% endblock %}

{% block content %}
<div class="container">
  <div class="page-header">
    <h1>Hello, {{ name }}!</h1>
  </div>
</div>
{% endblock %}

```

Bài 16: Xử lý lỗi không tìm được trang 404

Để xử lý lỗi 404, chúng ta có thể định nghĩa một lỗi và sử dụng kỹ thuật chuyển về trang báo lỗi như sau:

```

from flask import render_template

@app.errorhandler(404)
def page_not_found(error):
    return render_template('page_not_found.html'), 404

```

Một cách khác là:

Example 3-6. hello.py: custom error pages

```
@app.errorhandler(404)
def page_not_found(e):
    return render_template('404.html'), 404

@app.errorhandler(500)
def internal_server_error(e):
    return render_template('500.html'), 500
```

Bài 17: Session

Đoạn code sau mô tả cách thực session hoạt động:

```
from flask import Flask, session, redirect, url_for, escape, request

app = Flask(__name__)

@app.route('/')
def index():
    if 'username' in session:
        return 'Logged in as %s' % escape(session['username'])
    return 'You are not logged in'
```

Cụ thể hơn:

```

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        session['username'] = request.form['username']
        return redirect(url_for('index'))
    return '''
    <form method="post">
        <p><input type="text" name="username">
        <p><input type="submit" value="Login">
    </form>
    '''

@app.route('/logout')
def logout():
    # remove the username from the session if it's there
    session.pop('username', None)
    return redirect(url_for('index'))

# set the secret key. keep this really secret:
app.secret_key = 'A0Zr98j/3yX R~XHH\jmN]LWX/,?RT'

```

Bài 18: Về form

Web form của flask do gói **flask-wtf** quản lý. Gói flask_wtf có thể cần được cài thêm (bằng lệnh `pip install flask-wtf`).

Các đối tượng của flask_wtf bao gồm: Form, TextField, PasswordField,... và csrf là “dấu” riêng của trang web (do người xây dựng tự đặt, như tên công ty,...).

Ví dụ:

```
from flask.ext.wtf import Form
```

```

from wtforms import StringField, SubmitField

from wtforms.validators import Required

class DienTen(Form):

    ten = StringField('Ten ban la gi?',
                      validators=[Required()])

    nut_guilenserver = SubmitField('Gui')

```

Lưu ý: SubmitField sẽ đại diện cho 1 phần tử trong trang web có type = “submit”

Với các gói về form như gói **flask-wtf** hoặc gói sqlalchemy, chúng ta phải cài đặt bổ sung (bằng lệnh pip vì bản thân gói flask) không có các gói này.

Bài 19: Về form classes

Viết chương trình định nghĩa form class hello.py

Example 4-2. hello.py: form class definition

```

from flask_wtf import FlaskForm
from wtforms import StringField, SubmitField
from wtforms.validators import DataRequired

class NameForm(FlaskForm):
    name = StringField('What is your name?',
                       validators=[DataRequired()])
    submit = SubmitField('Submit')

```

Bài 20: Message Flashing

Hãy viết chương trình thể hiện chức năng của flash trong flashed messages

Example 4-6. hello.py: flashed messages

```
from flask import Flask, render_template, session, redirect,
url_for, flash
@app.route('/', methods=['GET', 'POST'])
def index():
    form = NameForm()
    if form.validate_on_submit():
        old_name = session.get('name')
        if old_name is not None and old_name != form.name.data:
            flash('Looks like you have changed your name!')
            session['name'] = form.name.data
            return redirect(url_for('index'))
    return render_template('index.html',
        form = form, name = session.get('name'))
```

Bài tập:

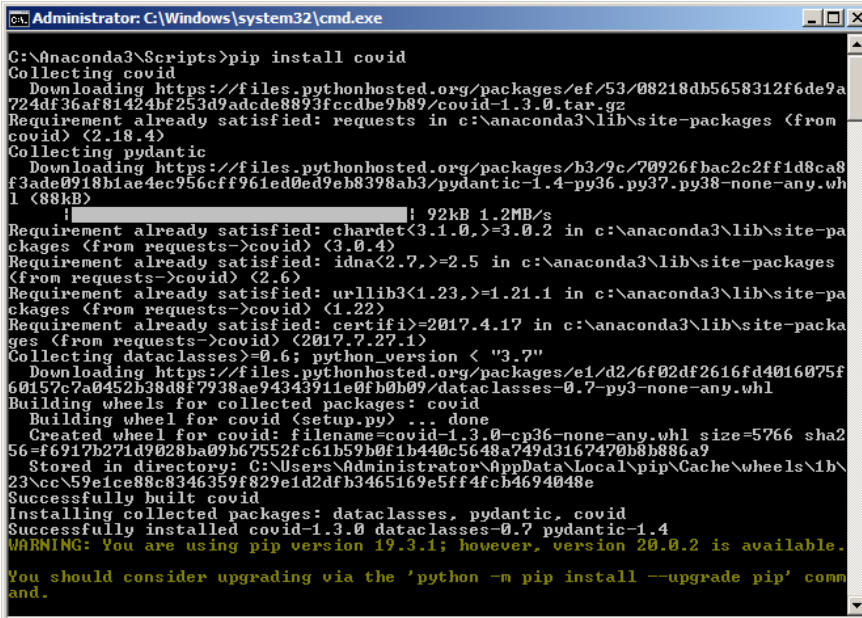
1. Triển khai xây dựng một web Flask trên nền tảng Heroku.
Tham khảo: <https://realpython.com/flask-by-example-part-1-project-setup/>
2. Nghiên cứu 1 trong 7 trang web cơ bản minh họa về Flask:
<https://pythonistaplanet.com/python-flask-project-ideas/>
3. Viết ứng dụng web kết nối dịch vụ hiển thị số ca covid trên thế giới.
4. Cài đặt trang web trên Heroku. Tham khảo: <https://realpython.com/flask-by-example-part-1-project-setup/#creating-the-python-flask-example-application>
5. Nghiên cứu dự án này (A great starting point to build your SaaS in Flask & Python, with Stripe subscription billing): <https://github.com/alectrocute/flaskSaaS>
6. SQLAlchemy: <https://www.fullstackpython.com/sqlalchemy.html>
7. Nghiên cứu full stack web dev trong Python: <https://www.fullstackpython.com/web-development.html>
8. Fullstack về Flask: <https://www.fullstackpython.com/flask-code-examples.html>
9. Nghiên cứu: <https://github.com/gothinkster/flask-realworld-example-app>
10. Flask RESTFUL: <https://flask-restful.readthedocs.io/en/latest/> hoặc ví dụ khác: <https://www.kdnuggets.com/2021/05/building-restful-apis-flask.html>

PHỤ LỤC: ĐỌC DỮ LIỆU COVID CÁC NƯỚC VÀ VÙNG LÃNH THỔ

Gói Python để lấy dữ liệu các nước bị nhiễm Covid19. Dữ liệu có nguồn từ đại học Johns Hopkins. Lưu ý: Nguồn dữ liệu chúng ta có thể tải về trực tiếp tại đường liên kết sau: <https://github.com/CSSEGISandData/COVID-19>

Cài đặt gói covid: thực hiện tại dấu nhắc hệ điều hành (bằng cmd) hoặc môi trường Anaconda Prompt: **pip install covid**

Yêu cầu cài đặt: sử dụng Python phiên bản từ 3.6 trở lên.



```
Administrator: C:\Windows\system32\cmd.exe
C:\Anaconda3\Scripts>pip install covid
Collecting covid
  Downloading https://files.pythonhosted.org/packages/ef/53/08218db5658312f6de9a724df36af81424bf253d9adcde8893fccdbe9b89/covid-1.3.0.tar.gz
Requirement already satisfied: requests in c:\anaconda3\lib\site-packages (from covid) (2.18.4)
Collecting pydantic
  Downloading https://files.pythonhosted.org/packages/b3/9c/709226fbac2c2fffd8ca8f3ade0918b1ae4ec956cff961ed0ed9eb8398ab3/pydantic-1.4-py36.py37.py38-none-any.whl (88kB)
    | 92kB 1.2MB/s
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\anaconda3\lib\site-packages (from requests->covid) (3.0.4)
Requirement already satisfied: idna<2.7,>=2.5 in c:\anaconda3\lib\site-packages (from requests->covid) (2.6)
Requirement already satisfied: urllib3<1.23,>=1.21.1 in c:\anaconda3\lib\site-packages (from requests->covid) (1.22)
Requirement already satisfied: certifi=2017.4.17 in c:\anaconda3\lib\site-packages (from requests->covid) (2017.7.27.1)
Collecting dataclasses=0.6; python_version < "3.7"
  Downloading https://files.pythonhosted.org/packages/e1/d2/6f02df2616fd4016075f60157c7a0452b38d8f7938ae94343911e0fb0b09/dataclasses-0.7-py3-none-any.whl
Building wheels for collected packages: covid
  Building wheel for covid (setup.py) ... done
  Created wheel for covid: filename=covid-1.3.0-cp36-none-any.whl size=5766 sha256=f6917b271d9028ba09b67552fc61b57b0f1b440c5648a749d3167470b8b886a9
  Stored in directory: C:\Users\Administrator\AppData\Local\pip\Cache\wheels\1b\23\cc\59\ce\88c8346359f827e1d2dfb3465167e5ff4fcb4694048e
Successfully built covid
Installing collected packages: dataclasses, pydantic, covid
Successfully installed covid-1.3.0 dataclasses-0.7 pydantic-1.4
WARNING: You are using pip version 19.3.1; however, version 20.0.2 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Lưu ý: Thông thường thì một số gói sẽ được yêu cầu cài đặt trước như:

- Gói pydantic
- Gói requests

Sử dụng: Sau khi cài đặt hoàn tất, chúng ta có thể lấy dữ liệu covid19 (trả về theo chuỗi JSON) bằng các lệnh sau:

```
>>> from covid import Covid
```

```
>>> covid = Covid()
```

```
>>> covid.get_data()
```

Minh họa các màn hình:

```
Python 3.6.3 Shell
File Edit Shell Debug Options Window Help
Python 3.6.3 [Anaconda, Inc.] (default, Oct 15 2017, 07:29:16) [MS
C v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> from covid import Covid
>>> covid = Covid()
>>> covid.get_data()
[{'id': '59', 'country': 'China', 'confirmed': 81101, 'active': 81
42, 'deaths': 3241, 'recovered': 69718, 'latitude': 30.5928, 'long
itude': 114.3055, 'last_update': 1584097775000}, {'id': '121', 'co
untry': 'Italy', 'confirmed': 31506, 'active': 26062, 'deaths': 25
03, 'recovered': 2941, 'latitude': 41.8719, 'longitude': 12.5674,
'last_update': 1584469982000}, {'id': '90', 'country': 'Iran', 'co
nfirmed': 16169, 'active': 9792, 'deaths': 988, 'recovered': 5389,
'latitude': 32.4279, 'longitude': 53.688, 'last_update': 158445798
9000}, {'id': '19', 'country': 'Spain', 'confirmed': 11826, 'activ
e': 10265, 'deaths': 533, 'recovered': 1028, 'latitude': 40.4637,
'longitude': -3.7492, 'last_update': 1584498786000}, {'id': '18',
'country': 'Germany', 'confirmed': 9360, 'active': 9263, 'deaths':
2000, 'recovered': 5357, 'latitude': 52.5200, 'longitude': 13.4050,
'last_update': 1584498786000}]
```

```
IDLE Shell 3.8.8
File Edit Shell Debug Options Window Help
Python 3.8.8 (default, Apr 13 2021, 15:08:03) [MSC v.1916 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import covid
>>> from covid import Covid
>>> covid = Covid()
>>> covid.get_data()
Squeezed text (446 lines).
>>> covid.get_status_by_country_name("VietNam")
{'id': '191', 'country': 'Vietnam', 'confirmed': 797712, 'active': None, 'death
s': 19437, 'recovered': None, 'latitude': 14.058324, 'longitude': 108.277199,
'last_update': 1633159283000}
>>>
```

Tài liệu tham khảo:

https://ahmednafies.github.io/covid/?fbclid=IwAR3EXNdu0mdhgkX5TaCCtZ6pzKwuQdenid1CswSucZS5sVR4-NP0l_Xq520

Chọn nguồn dữ liệu:

```
from covid import Covid

# by default data source is "john_hopkins"
covid = Covid()

# or
covid = Covid(source="john_hopkins")

# to get data from worldometers.info
covid = Covid(source="worldometers")
```

```
# get all data  
covid.get_data()
```