

Web Application Programming

Jennifer Helsby, Eric Potash

Computation for Public Policy

Lecture 15: February 23, 2016

computationforpolicy.github.io

Announcements

- Homework 5 will be the last homework to give you more time to work on final projects
- Next time:
 - We will go over Homework 4 in-class
 - More on web application development

Today

- Homework 4 hints
- Web application programming with Flask

Homework 4

- Part a is now extra credit
- You can download a csv containing the DataFrame with the links you should click from the course website
- Just do parts b-e for the HW
- We will go through *in class after the assignment is due* how to go about part a

Challenge #1: Date Separator Changes in the Document

- March 28 – **ČSA Flight 511**, an Ilyushin Il-18, crashes in Gräfenberg, West Germany. All 52 passengers and crew on board are killed.
- April 3 - **LAN Chile Flight 210**, a Douglas DC-3, crashes in the Andes, killing all 24 on board including footballers and coaching staff from the CD Green Cross Chilean football team.

- March 28  **ČSA Flight 511**, an Ilyu
- April 3  **LAN Chile Flight 210**, a Do
Chilean football team.

Challenge #2: Multiple crashes under a single bullet

- September 11 – **September 11 attacks**

- **American Airlines Flight 11**, a Boeing 767-200ER with 92 people on board, is hijacked after taking off from Boston, and is flown into the north tower of the World Trade Center in New York City; all on board are killed as well as others on the ground and in the building.
- **United Airlines Flight 175**, a Boeing 767-200 with 65 people on board, is hijacked after taking off from Boston and is flown into the south tower of the World Trade Center in New York City; all on board are killed as well as others on the ground and in the building; the collapse of both towers brings the total death toll from the two crashes to at least 2,759, **the worst disaster involving commercial aircraft**.
- **American Airlines Flight 77**, a Boeing 757-200 with 64 people on board, is hijacked after taking off from Dulles International Airport and is flown into The Pentagon; all on board are killed as well as 125 people in the building and on the ground.
- **United Airlines Flight 93**, a Boeing 757-200 with 44 people on board, is hijacked after taking off from Newark, New Jersey; passengers struggle with the hijackers, and the aircraft crashes in a field near Shanksville, Pennsylvania, killing all on board.

- **1950 Air France multiple Douglas DC-4 accidents:**

- June 12 – An Air France Douglas DC-4 (F-BBDE) on a flight from Saigon to Paris crashes in the Arabian Sea while on approach to Bahrain Airport, killing 46 of 52 on board.
- June 14 – An Air France Douglas DC-4, F-BBDM, crashes in the Arabian Sea while on approach to Bahrain Airport, killing 40 of 53 on board. This aircraft was operating on the same flight route as F-BBDE.

- August 24 – **2004 Russian aircraft bombings:**

- Siberia Airlines Flight 1047, a Tupolev Tu-154, explodes in mid-air while flying over Rostov Oblast, Russia, killing all 38 passengers and 8 crew members on board.
- Volga-AviaExpress Flight 1303, a Tupolev Tu-134, explodes in mid-air while flying over Tula Oblast, Russia, killing all 34 passengers and 9 crew members on board.

Debris left of Kish Air Flight 7170



Challenge #3: Years are not stored with the month, day

2016 [[edit](#)]

- January 8 – **West Air Sweden Flight 294**, a Bombardier CRJ200 cargo flight, crashes while in cruise near Akkajaure in Sweden. Both crew members on board are killed.
- February 2 - **Daallo Airlines Flight 159**, an Airbus A321, encountered an explosion shortly after taking off from Mogadishu's International Airport. 3 people were injured and one, the suspected suicide bomber, was killed.


Details Page

1919 Verona Caproni Ca.48 crash



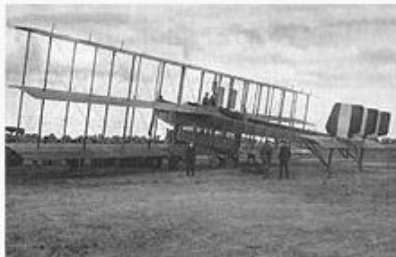
A Caproni Ca.48 airliner.

Accident summary

Date	August 2, 1919
Summary	Possible structural failure
Site	Verona, Italy  45°23′47″N 10°53′17″E
Passengers	12, ^[1] 13, or 15 (sources vary)
Crew	2 ^[1]
Injuries (non-fatal)	0
Fatalities	all on board (14, ^[1] 15, ^[2] or 17, ^[3] sources vary)
Survivors	0
Aircraft type	Caproni Ca.48
Operator	Caproni
Flight origin	Venice, Italy
Destination	Taliedo, Milan, Italy


Details Page

1919 Verona Caproni Ca.48 crash



A Caproni Ca.48 airliner.

Accident summary

Date	August 2, 1919
Summary	Possible structural failure
Site	Verona, Italy  45°23′47″N 10°53′17″E
Passengers	12, ^[1] 13, or 15 (sources vary)
Crew	2 ^[1]
Injuries (non-fatal)	0
Fatalities	all on board (14, ^[1] 15, ^[2] or 17, ^[3] sources vary)
Survivors	0
Aircraft type	Caproni Ca.48
Operator	Caproni
Flight origin	Venice, Italy
Destination	Taliedo, Milan, Italy

Web Development with Flask

Developing Dynamic Webapps with Python

- Web frameworks simplify the writing of web applications by providing basic functionality for common tasks, e.g. setting cookies, mapping incoming HTTP requests to the appropriate page

Developing Dynamic Webapps with Python

- Web frameworks simplify the writing of web applications by providing basic functionality for common tasks, e.g. setting cookies, mapping incoming HTTP requests to the appropriate page
- Two major web application frameworks in Python: Django and Flask
- Django:
 - Most common framework for Python-based web applications
 - Read more at: <https://www.djangoproject.com/>
 - Best for complex applications
- Flask:
 - A “microframework” for Python-based web applications
 - Very quick to get started
 - Best for simpler applications

Example

What makes up a framework

- Templating engine to dynamically produce HTML:
 - Enables the programmer to separate basic functions (written in Python) from the “look” of the website with HTML
 - Jinja2 in Flask

What makes up a framework

- Templating engine to dynamically produce HTML:
 - Enables the programmer to separate basic functions (written in Python) from the “look” of the website with HTML
 - Jinja2 in Flask
- Routing:
 - Maps paths and incoming HTTP requests to Python code
 - e.g. whenever a user goes to /user/dailystats a piece of code is invoked that computes some statistics

What makes up a framework

- Templating engine to dynamically produce HTML:
 - Enables the programmer to separate basic functions (written in Python) from the “look” of the website with HTML
 - Jinja2 in Flask
- Routing:
 - Maps paths and incoming HTTP requests to Python code
 - e.g. whenever a user goes to /user/dailystats a piece of code is invoked that computes some statistics
- Tools for setting cookies, etc.

Let's start building a simple web application together
Feel free to re-use parts of this code in your project

What do we want our application to do?

For now, let's just have it display a dataset that we've been using (we'll use DataFrames for extra pandas practice)

Potential features:

- Click on a row and take us to a detail page with more info about that row

- Sort by certain columns

- Show subsets of the data based on user selections

- Generate plots / visualizations based on user selections

First we set up the appropriate directory structure

```
webapp/  
    templates/  
    static/  
    dfdisplay.py
```

This structure and code in the `lecture_examples` github repository

Writing the basic application: dfdisplay.py

```
from flask import Flask import Flask
import pandas as pd
app = Flask(__name__)

@app.route('/')
def load_dataframe():
    filename = '../Current_Employee_Names__Salaries__and_Position_Titles.csv'
    df = pd.read_csv(filename)
    return df.to_string()

if __name__ == '__main__':
    app.run()
```

Writing the basic application: dfdisplay.py

```
from flask import Flask
import pandas as pd
app = Flask(__name__)
```

create a new app

```
@app.route('/')
def load_dataframe():
    filename = '../Current_Employee_Names__Salaries__and_Position_Titles.csv'
    df = pd.read_csv(filename)
    return df.to_string()

if __name__ == '__main__':
    app.run()
```

Writing the basic application: dfdisplay.py

```
from flask import Flask
import pandas as pd
app = Flask(__name__)
```

```
@app.route('/') tell the app to map incoming HTTP requests to this piece of Python code
def load_dataframe():
    filename = '../Current_Employee_Names__Salaries__and_Position_Titles.csv'
    df = pd.read_csv(filename)
    return df.to_string()

if __name__ == '__main__':
    app.run()
```

Writing the basic application: dfdisplay.py

```
from flask import Flask
import pandas as pd
app = Flask(__name__)

@app.route('/')
def load_dataframe():
    filename = '../Current_Employee_Names__Salaries__and_Position_Titles.csv'
    df = pd.read_csv(filename)
    return df.to_string() display this in the browser

if __name__ == '__main__':
    app.run()
```

Firing up the Flask development webserver

- Run the code with `python3 dfdisplay.py`:

```
Tue Feb 23 03:25 webapp ($ 7) $ python3 dfdisplay.py
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [23/Feb/2016 03:51:54] "GET / HTTP/1.1" 200 -
```

127.0.0.1 is the loopback address, a special IP address that denotes your local machine

If you mess something up you'll see:

Internal Server Error

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.

Success!

← → ↻ 127.0.0.1:5000

Name	Position	Title	Department	Employee	Annual Salary	
0 AARON, ELVIA J	WATER RATE TAKER	WATER MGMNT	\$88968.00	1 AARON, KARINA	POLICE OFFICER POLICE \$80778.00	
2 AARON, KIMBERLEI R	CHIEF COM	GENERAL SERVICES	\$84780.00	4 ABAD JR, VICENTE M	CIVIL ENGINEER IV WATER MGMNT \$104736.00	
5 ABARCA, ANABEL A	ALDERMAN CITY COUNCIL	\$70764.00	6 ABARCA, EMMANUEL	GENERAL LABORER - DSS STREETS & SAN	\$40560.00	
7 ABBATE, TERRY	ELECTRICAL MECHANIC AVIATION	\$91520.00	8 ABBATEMARCO, JAMES J	FIRE ENGINEER FIRE	\$90456.00	
9 ABBATE, TERRY	POLICE	\$86520.00	10 ABBOTT, BETTY L	FOSTER GRANDPARENT FAMILY & SUPPORT	\$2756.00	
11 ABBOTT, LYNISE M	CLERK		12 ABBRUZZESE, WILLIAM J	INVESTIGATOR - IPRA II IPRA	\$72468.00	
13 ABDALLAH, ZAID	POLICE OFFICER POLICE	\$69684.00	14 ABDALMAHD, POLICE OFFICER POLICE	\$80778.00	15 ABDELLATIF, AREF R	FIREFIGHTER (PER ARBITRATORS AWARD)-PARA
16 ABDELMAJEID, AZIZ	POLICE OFFICER POLICE	\$80778.00	17 ABDOLLAHZADEH, ALI	FIREFIGHTER/PARAMEDIC		
18 ABDUL-KARIM, MUHAMMAD A	ENGINEERING TECHNICIAN VI WATER MGMNT	\$106104.00	19 ABDULLAH, DANIEL N	FIREFI	\$91764.00	
20 ABDULLAH, KEVIN	LIEUTENANT FIRE	\$110370.00	21 ABDULLAH, LAKENYA N	CROSSING GUARD POLICE	\$16692	
22 RASHAD J	ELECTRICAL MECHANIC-AUTO-POLICE MTR MNT	GENERAL SERVICES	\$91520.00	23 ABDUL-SHAKUR, TAHIR	GEN	
24 ABEJERO, JASON V	POLICE OFFICER POLICE	\$86520.00	25 ABERCROMBIE IV, EARL S	PARA	\$54114.00	
26 ABERCROMBIE, TIMOTHY	MOTOR TRUCK DRIVER STREETS & SAN	\$71780.80	27 ABIOYE, ADEWOLE A	LIBRARY		
28 ABOUELKHEIR, HASSAN A	SENIOR PROGRAMMER/ANALYST DoIT	\$104736.00	29 ABI	CIVIL ENGINEER IV WATER MGMNT	\$104736.00	
30 ABRAHAM, GODWIN K	ENGINEERING TECHNICIAN V BUSINESS AFFAIRS		31 ABRAHAM, NANCY A	POLICE OFFICER POLICE	\$46206.00	
32 ABRAHAMAVICIUS, ANNA A	SUPERVISING TRAFFIC CONTROL AID		33 ABRAMS, DANIELLE T	SANITATION LABORER STREETS & SAN	\$72384.00	
34 ABRAMS, HENRY L	POLICE OFFICER POLICE	\$92	35 ABRAMSKI, JOHN E	AMBULANCE COMMANDER FIRE	\$123948.00	
36 ABRAMS, SAMUEL A	POOL MOTOR TRUCK DRIVER STR	\$16151.20	37 ABRATANSKI, MARK A	FIREFIGHTER-EMT FIRE	\$85680.00	
38 ABREU, DILAN	SEWER BRICKLAYER WATER MGM		39 ABREU, EDWIN	TREE TRIMMER STREETS & SAN	\$74464.00	
40 ABREU, ROBERTO J	TRAFFIC SIGNAL REPAIRMAN TRANSPORT		41 ABREU, ROSITA	PERSONAL COMPUTER OPERATOR III HEALTH	\$66684.00	
42 ABREU, VICTOR	SENIOR COMPANION FAMILY		43 ABREU, VICTOR	FIREFIGHTER-EMT FIRE	\$95460.00	
44 ABRON, FLOYD	POLICE OFFICER POLICE	\$86520.00	45 ABRONS, KEN			

Success!

But this looks really awful...

← → ↻ 127.0.0.1:5000

Name	Position	Title	Department	Employee	Annual Salary
0 AARON, ELVIA J	WATER RATE TAKER	WATER MGMNT	\$88968.00	1 AARON, KARINA	POLICE OFFICER POLICE \$80778.00
2 AARON, KIMBERLEI R	CHIEF COM	GENERAL SERVICES	\$84780.00	4 ABAD JR, VICENTE M	CIVIL ENGINEER IV WATER MGMNT \$104736.00
5 ABARCA, ANABEL A	ALDERMAN CITY COUNCIL	\$70764.00	6 ABARCA, EMMANUEL	GENERAL LABORER - DSS STREETS & SAN	\$40560.00
7 ABBATE, TERRY	ELECTRICAL MECHANIC AVIATION	\$91520.00	8 ABBATEMARCO, JAMES J	FIRE ENGINEER FIRE	\$90456.00
9 ABBOTT, BETTY L	FOSTER GRANDPARENT FAMILY & SUPPORT	\$2756.00	11 ABBOTT, LYNISE M	CLERK	
12 ABBRUZZESE, WILLIAM J	INVESTIGATOR - IPRA II	IPRA	\$72468.00	13 ABDALLAH, ZAID	POLICE OFFICER POLICE \$69684.00
14 ABDALMAHD, AREF R	FIREFIGHTER (PER ARBITRATORS AWARD)-PARA	\$98244.00	16 ABDELMAJEID, AZIZ	POLICE OFFICER POLICE	\$80778.00
17 ABDOLLAHZADEH, ALI	FIREFIGHTER/PARAMEDIC		19 ABDULLAH, DANIEL N	FIREFI	\$91764.00
20 ABDULLAH, KEVIN	LIEUTENANT FIRE	\$110370.00	21 ABDULLAH, LAKENYA N	CROSSING GUARD POLICE	\$16692.00
22 RASHAD J	ELECTRICAL MECHANIC-AUTO-POLICE MTR MNT	GENERAL SERVICES	\$91520.00	23 ABDUL-SHAKUR, TAHIR	GEN
24 ABEJERO, JASON V	POLICE OFFICER POLICE	\$86520.00	25 ABERCROMBIE IV, EARL S	PARA	\$54114.00
26 ABERCROMBIE, TIMOTHY	MOTOR TRUCK DRIVER STREETS & SAN	\$71780.80	27 ABIOYE, ADEWOLE A	LIBRARY	
28 ABOUELKHEIR, HASSAN A	SENIOR PROGRAMMER/ANALYST DoIT	\$104736.00	29 ABI	CIVIL ENGINEER IV WATER MGMNT	\$104736.00
30 ABRAHAM, GODWIN K	ENGINEERING TECHNICIAN V BUSINESS AFFAIRS		32 ABRAHAM, NANCY A	POLICE OFFICER POLICE	\$46206.00
33 ABRAMAVICIUS, ANNA A	SUPERVISING TRAFFIC CONTROL AID		34 ABRAMS, DANIELLE T	SANITATION LABORER STREETS & SAN	\$72384.00
35 ABRAMS, HENRY L	POLICE OFFICER POLICE	\$92	36 ABRAMSKI, JOHN E	AMBULANCE COMMANDER FIRE	\$123948.00
37 ABRAMS, SAMUEL A	POOL MOTOR TRUCK DRIVER STR	\$16151.20	38 ABREU, DILAN	SEWER BRICKLAYER WATER MGM	
39 ABREU, EDWIN	TREE TRIMMER STREETS & SAN	\$74464.00	40 ABREU, ROBERTO J	TRAFFIC SIGNAL REPAIRMAN TRANSPOR	
41 ABREU, ROSITA	PERSONAL COMPUTER OPERATOR III HEALTH	\$66684.00	42 ABREU, VICTOR	SENIOR COMPANION FAMILY	
43 ABREU, VICTOR	FIREFIGHTER-EMT FIRE	\$95460.00	44 ABRON, FLOYD	POLICE OFFICER POLICE	\$86520.00
45 ABRONS, KEN					

Protip #1: pandas DataFrames to_html()

```
from flask import Flask
import pandas as pd
app = Flask(__name__)

@app.route('/')
def load_dataframe():
    filename = '../Current_Employee_Names__Salaries__and_Position_Titles.csv'
    df = pd.read_csv(filename)
    return df.to_html()

if __name__ == '__main__':
    app.run()
```

Bam!

← → ↺ 127.0.0.1:5000



	Name	Position Title	Department	Employee Annual Salary
0	AARON, ELVIA J	WATER RATE TAKER	WATER MGMNT	\$88968.00
1	AARON, JEFFERY M	POLICE OFFICER	POLICE	\$80778.00
2	AARON, KARINA	POLICE OFFICER	POLICE	\$80778.00
3	AARON, KIMBERLEI R	CHIEF CONTRACT EXPEDITER	GENERAL SERVICES	\$84780.00
4	ABAD JR, VICENTE M	CIVIL ENGINEER IV	WATER MGMNT	\$104736.00
5	ABARCA, ANABEL	ASST TO THE ALDERMAN	CITY COUNCIL	\$70764.00
6	ABARCA, EMMANUEL	GENERAL LABORER - DSS	STREETS & SAN	\$40560.00
7	ABBATACOLA, ROBERT J	ELECTRICAL MECHANIC	AVIATION	\$91520.00
8	ABBATEMARCO, JAMES J	FIRE ENGINEER	FIRE	\$90456.00
9	ABBATE, TERRY M	POLICE OFFICER	POLICE	\$86520.00
10	ABBOTT, BETTY L	FOSTER GRANDPARENT	FAMILY & SUPPORT	\$2756.00
11	ABBOTT, LYNISE M	CLERK III	POLICE	\$43920.00
12	ABBRUZZESE, WILLIAM J	INVESTIGATOR - IPRA II	IPRA	\$72468.00
13	ABDALLAH, ZAID	POLICE OFFICER	POLICE	\$69684.00
14	ABDELHADI, ABDALMAHD	POLICE OFFICER	POLICE	\$80778.00
15	ABDELLATIF, AREF R	FIREFIGHTER (PER ARBITRATORS AWARD)-PARAMEDIC	FIRE	\$98244.00
16	ABDELMAJEID, AZIZ	POLICE OFFICER	POLICE	\$80778.00
17	ABDOLLAHZADEH, ALI	FIREFIGHTER/PARAMEDIC	FIRE	\$87720.00
18	ABDUL-KARIM, MUHAMMAD A	ENGINEERING TECHNICIAN VI	WATER MGMNT	\$106104.00

Getting feedback on errors

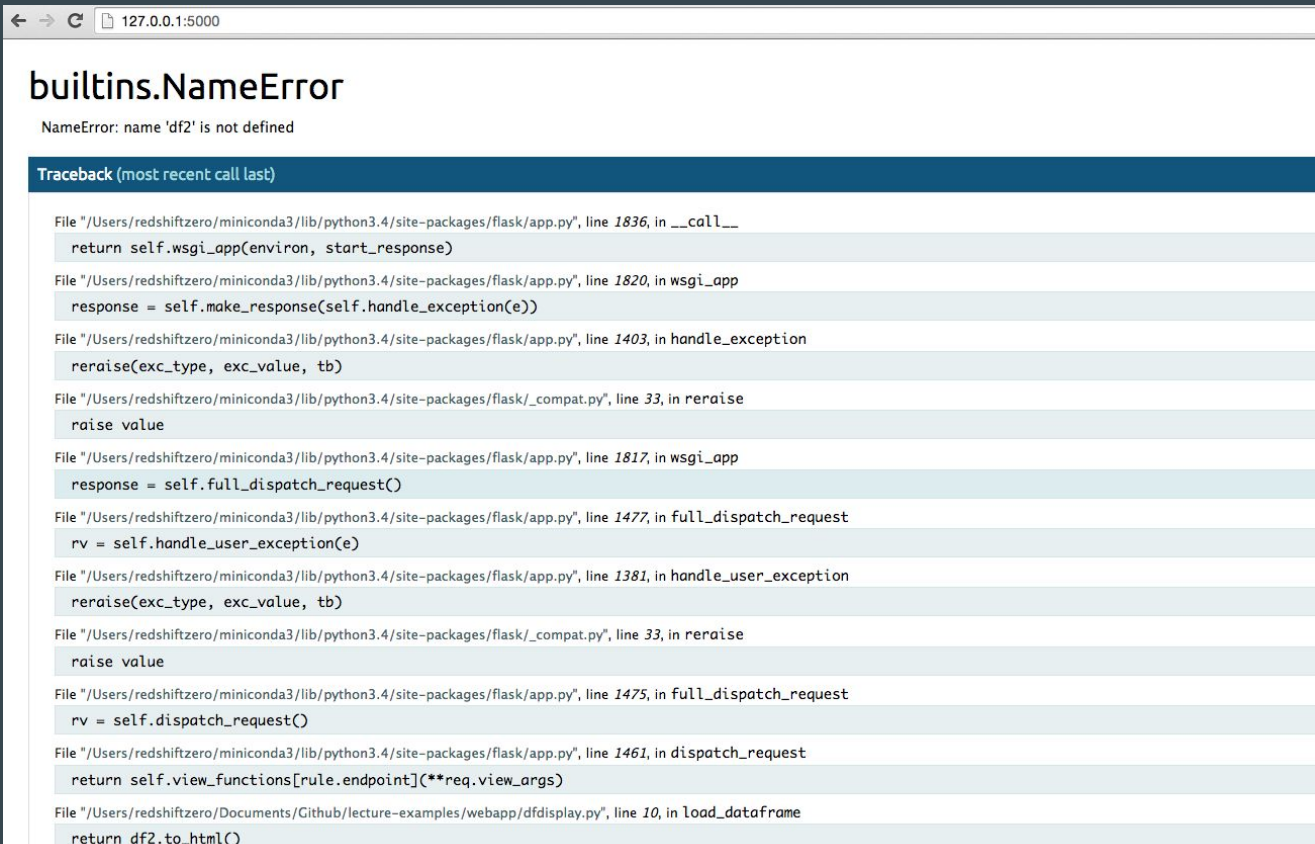
```
from flask import Flask
import pandas as pd
app = Flask(__name__)

@app.route('/')
def load_dataframe():
    filename = '../Current_Employee_Names__Salaries__and_Position_Titles.csv'
    df = pd.read_csv(filename)
    return df.to_html()

if __name__ == '__main__':
    app.run(debug=True)
```

Use Flask's debugging mode

Getting feedback on errors



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000'. The main content area displays a Python error message: 'builtins.NameError' followed by 'NameError: name 'df2' is not defined'. Below this, a blue header reads 'Traceback (most recent call last)'. The traceback itself is a list of code snippets from various files, including Flask's app.py and a local file 'load_dataframe.py', showing the sequence of calls that led to the error. The final line of the traceback is 'return df2.to_html()' in 'load_dataframe.py', line 10.

```
builtins.NameError
NameError: name 'df2' is not defined

Traceback (most recent call last)
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/app.py, line 1836, in __call__
    return self.wsgi_app(environ, start_response)
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/app.py, line 1820, in wsgi_app
    response = self.make_response(self.handle_exception(e))
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/app.py, line 1403, in handle_exception
    reraise(exc_type, exc_value, tb)
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/_compat.py, line 33, in reraise
    raise value
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/app.py, line 1817, in wsgi_app
    response = self.full_dispatch_request()
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/app.py, line 1477, in full_dispatch_request
    rv = self.handle_user_exception(e)
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/app.py, line 1381, in handle_user_exception
    reraise(exc_type, exc_value, tb)
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/_compat.py, line 33, in reraise
    raise value
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/app.py, line 1475, in full_dispatch_request
    rv = self.dispatch_request()
File ~/Users/redshiftzero/miniconda3/lib/python3.4/site-packages/flask/app.py, line 1461, in dispatch_request
    return self.view_functions[rule.endpoint](**req.view_args)
File ~/Users/redshiftzero/Documents/Github/lecture-examples/webapp/dfdisplay.py, line 10, in load_dataframe
    return df2.to_html()
```

Using HTML Templates

- Generate HTML dynamically based on Python code
- Cleanly separates design and functionality

Bootstrap

<https://getbootstrap.com>

- Framework for HTML and CSS
- Fast way of making pages look nice



Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

Download Bootstrap

Currently v3.3.6

Project name New Open Close

Bootstrap starter template

Use this document as a way to quickly start any new project.
All you get is this text and a mostly barebones HTML document.

Product name

Price

Buy

Hello, world!

This is a template for a simple marketing or informational website. It includes a large callout called a **jumbotron** and three **responsive** blocks of content. Use it as a starting point to make something new and unique.

Learn more

Heading

Donec id elit non mi porta gravida at eget metus. Maecenas nec odio et magna tincidunt aliquam sed eu lacinia odio.

Learn more >

Heading

Donec id elit non mi porta gravida at eget metus. Maecenas nec odio et magna tincidunt aliquam sed eu lacinia odio.

Learn more >

Heading

Donec id elit non mi porta gravida at eget metus. Maecenas nec odio et magna tincidunt aliquam sed eu lacinia odio.

Learn more >

© Company, 2012

[illegible]

Bootstrap grid examples

Basic grid layouts to get you started with building within the Bootstrap grid system.

Three equal columns

Three equal width columns starting at desktops and scaling to large desktops.

```
<div class="row">
  <div class="col-md-4">...</div>
  <div class="col-md-4">...</div>
  <div class="col-md-4">...</div>
</div>
```

Three unequal columns

Desktops columns starting at desktops and scaling to large desktops (col-md-8, col-md-4). Tablets columns start at tablets (col-sm-6, col-sm-6). Mobiles columns start at mobile phones (col-sm-12).

```
<div class="row">
  <div class="col-md-8">...</div>
  <div class="col-md-4">...</div>
</div>
```

Two columns

Desktops starting at desktops and scaling to large desktops.

```
<div class="row">
  <div class="col-md-6">...</div>
  <div class="col-md-6">...</div>
</div>
```

Full width, single column

The grid columns are expanding to fill the width available.

```
<div class="row">
  <div class="col">...</div>
</div>
```

Two columns with two nested columns

The columns are expanding, making a total of four columns of varying widths. The first row is nested starting at desktops and scaling to large desktops, while the second row is nested starting at tablets and scaling to large desktops.

```
<div class="row">
  <div class="col-md-8">...</div>
  <div class="col-md-4">...</div>
  <div class="col-sm-6">...</div>
  <div class="col-sm-6">...</div>
</div>
```

Multiple examples of grid layouts with all four tiers, nesting, and more.

Using HTML Templates

- Generate HTML dynamically based on Python code
- Cleanly separates design and functionality
- Use by:

1. Adding an *.html template to `templates/`

2. Use `render_template()` to pass the template (first argument) and then any data that should be passed to the template as keyword arguments

Using render_template

```
from flask import Flask
from flask import render_template
import pandas as pd
app = Flask(__name__)

@app.route('/')
def load_dataframe():
    filename = '../Current_Employee_Names__Salaries__and_Position_Titles.csv'
    df = pd.read_csv(filename)
    return render_template('unfiltered.html', data=df.to_html())

if __name__ == '__main__':
    app.run(debug=True)
```

Our HTML Template

...

```
<title>DataFrame Display</title>
```

```
    <link href="{{ url_for('static', filename='static/template.css') }}" rel="
stylesheet">
```

```
</head>
```

```
<body>
```

...

```
{{ data|safe }}
```

...

```
</body>
```

...

Our HTML Template

...

```
<title>DataFrame Display</title>
```

```
    <link href="{{ url_for('static', filename='static/template.css') }}" rel="
stylesheet">
```

```
</head>
```

```
<body>
```

...

```
{{ data|safe }}
```

...

```
</body>
```

...

Our HTML Template

...

```
<title>DataFrame Display</title>
```

```
    <link href="{{ url_for('static', filename='static/template.css') }}" rel="
stylesheet">
```

```
</head>
```

```
<body>
```

...

```
{{ data|safe }}
```

...

```
</body>
```

...

Rendering using HTML templates as well as CSS styling

127.0.0.1:5000

City of Chicago Employee roster and salaries

	Name	Position Title	Department	Employee Annual Salary
0	AARON, ELVIA J	WATER RATE TAKER	WATER MGMNT	\$88968.00
1	AARON, JEFFERY M	POLICE OFFICER	POLICE	\$80778.00
2	AARON, KARINA	POLICE OFFICER	POLICE	\$80778.00
3	AARON, KIMBERLEI R	CHIEF CONTRACT EXPEDITER	GENERAL SERVICES	\$84780.00
4	ABAD JR, VICENTE M	CIVIL ENGINEER IV	WATER MGMNT	\$104736.00
5	ABARCA, ANABEL	ASST TO THE ALDERMAN	CITY COUNCIL	\$70764.00
6	ABARCA, EMMANUEL	GENERAL LABORER - DSS	STREETS & SAN	\$40560.00
7	ABBATACOLA, ROBERT J	ELECTRICAL MECHANIC	AVIATION	\$91520.00
8	ABBATEMARCO, JAMES J	FIRE ENGINEER	FIRE	\$90456.00
9	ABBATE, TERRY M	POLICE OFFICER	POLICE	\$86520.00
10	ABBOTT, BETTY L	FOSTER GRANDPARENT	FAMILY & SUPPORT	\$2756.00
11	ABBOTT, LYNISE M	CLERK III	POLICE	\$43920.00
12	ABBRUZZESE, WILLIAM J	INVESTIGATOR - IPRA II	IPRA	\$72468.00
13	ABDALLAH, ZAID	POLICE OFFICER	POLICE	\$69684.00
14	ABDELHADI, ABDALMAHD	POLICE OFFICER	POLICE	\$80778.00
15	ABDELLATIF, AREF R	FIREFIGHTER (PER ARBITRATORS AWARD)-PARAMEDIC	FIRE	\$98244.00
16	ABDELMAJEID, AZIZ	POLICE OFFICER	POLICE	\$80778.00
17	ABDOLLAHZADEH, ALI	FIREFIGHTER/PARAMEDIC	FIRE	\$87720.00
18	ABDUL-KARIM, MUHAMMAD A	ENGINEERING TECHNICIAN VI	WATER MGMNT	\$106104.00
19	ABDULLAH, DANIEL N	FIREFIGHTER-EMT	FIRE	\$91764.00

CSS: Cascading Style Sheets

- Language that formats HTML presentation
- We will store content like CSS and JavaScript files in `static/`

CSS: Cascading Style Sheets

- Language that formats HTML presentation
- We will store content like CSS and JavaScript files in `static/`

...

```
body {  
  padding-top: 50px;  
}
```

...

City of Chicago Employee roster and salaries

	Name	Position Title	Department	Employee Annual Salary
0	AARON, ELVIA J	WATER RATE TAKER	WATER MGMNT	\$88968.00
1	AARON, JEFFERY M	POLICE OFFICER	POLICE	\$80778.00
2	AARON, KARINA	POLICE OFFICER	POLICE	\$80778.00
3	AARON, KIMBERLEI R	CHIEF CONTRACT EXPEDITER	GENERAL SERVICES	\$84780.00
4	ABAD JR, VICENTE M	CIVIL ENGINEER IV	WATER MGMNT	\$104736.00
5	ABARCA, ANABEL	ASST TO THE ALDERMAN	CITY COUNCIL	\$70764.00
6	ABARCA, EMMANUEL	GENERAL LABORER - DSS	STREETS & SAN	\$40560.00
7	ABBATACOLA, ROBERT J	ELECTRICAL MECHANIC	AVIATION	\$91520.00
8	ABBATEMARCO, JAMES J	FIRE ENGINEER	FIRE	\$90456.00
9	ABBATE, TERRY M	POLICE OFFICER	POLICE	\$86520.00
10	ABBOTT, BETTY L	FOSTER GRANDPARENT	FAMILY & SUPPORT	\$2756.00
11	ABBOTT, LYNISE M	CLERK III	POLICE	\$43920.00
12	ABBRUZZESE, WILLIAM J	INVESTIGATOR - IPRA II	IPRA	\$72468.00
13	ABDALLAH, ZAID	POLICE OFFICER	POLICE	\$69684.00
14	ABDELHADI, ABDALMAHD	POLICE OFFICER	POLICE	\$80778.00
15	ABDELLATIF, AREF R	FIREFIGHTER (PER ARBITRATORS AWARD)-PARAMEDIC	FIRE	\$98244.00
16	ABDELMAJEID, AZIZ	POLICE OFFICER	POLICE	\$80778.00
17	ABDOLLAHZADEH, ALI	FIREFIGHTER/PARAMEDIC	FIRE	\$87720.00
18	ABDUL-KARIM, MUHAMMAD A	ENGINEERING TECHNICIAN VI	WATER MGMNT	\$106104.00
19	ABDULLAH, DANIEL N	FIREFIGHTER-EMT	FIRE	\$91764.00
20	ABDULLAH, KEVIN	LIEUTENANT	FIRE	\$110370.00
21	ABDULLAH, LAKENYA N	CROSSING GUARD	POLICE	\$16692.00
22	ABDULLAH, RASHAD J	ELECTRICAL MECHANIC-AUTO-POLICE MTR MNT	GENERAL SERVICES	\$91520.00
23	ABDUL-SHAKUR, TAHIR	GENERAL LABORER - DSS	STREETS & SAN	\$40560.00

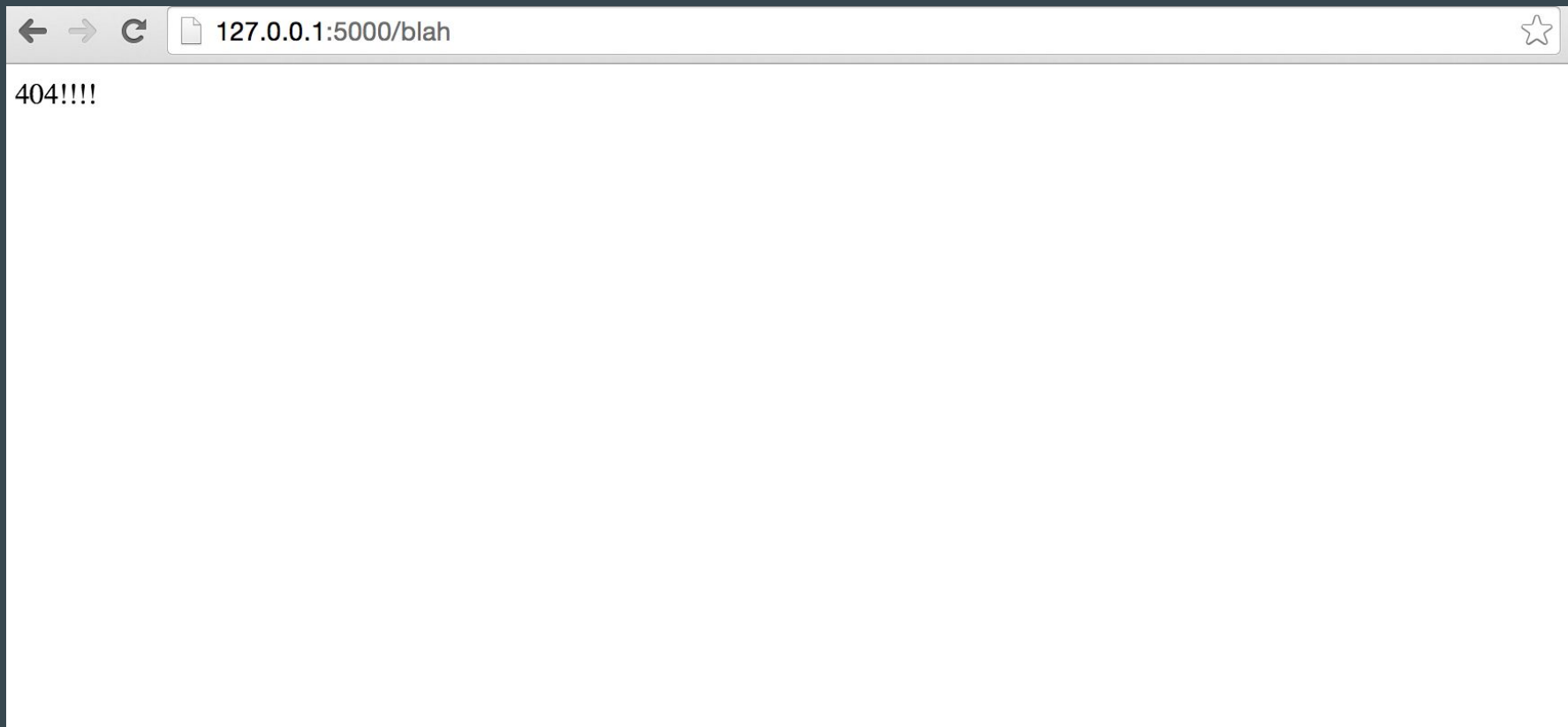
Handling errors: 404

```
@app.errorhandler(404)
```

```
def not_found(error):
```

```
    return '404!!!!'
```

Routes the app doesn't know about produce 404



Summary

- Made a basic flask web application
- Nicely formatted it
- Handled errors e.g. 404 File not found

Next Time

- We will go over the scraping homework
- We will add additional features to our web application