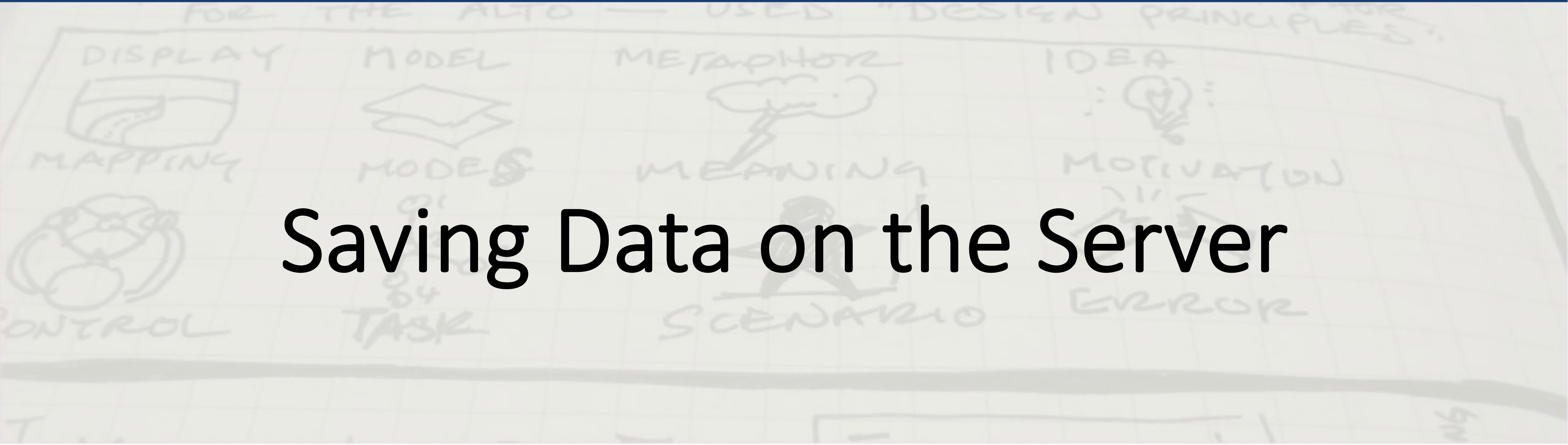
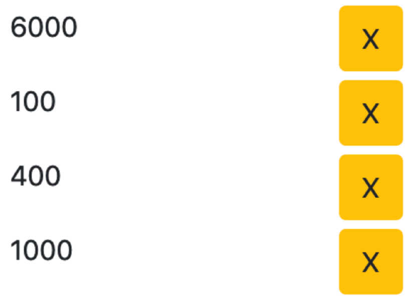


Saving Data on the Server

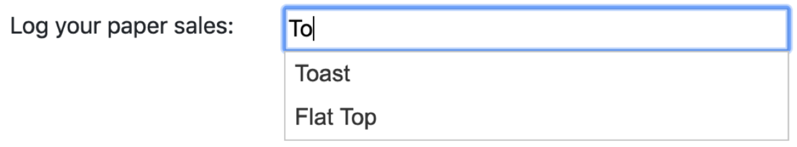
Prof. Lydia Chilton
COMS 4170
14 February 2024

In HW4, you dynamically created widgets

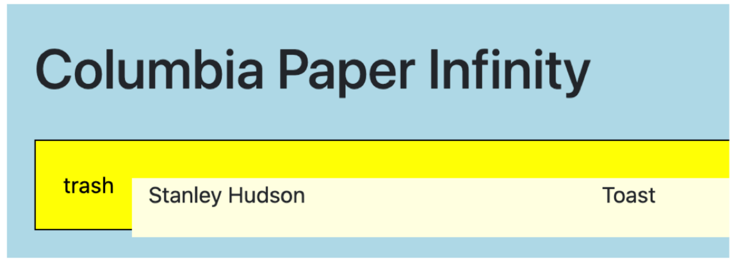
Buttons



Autocomplete

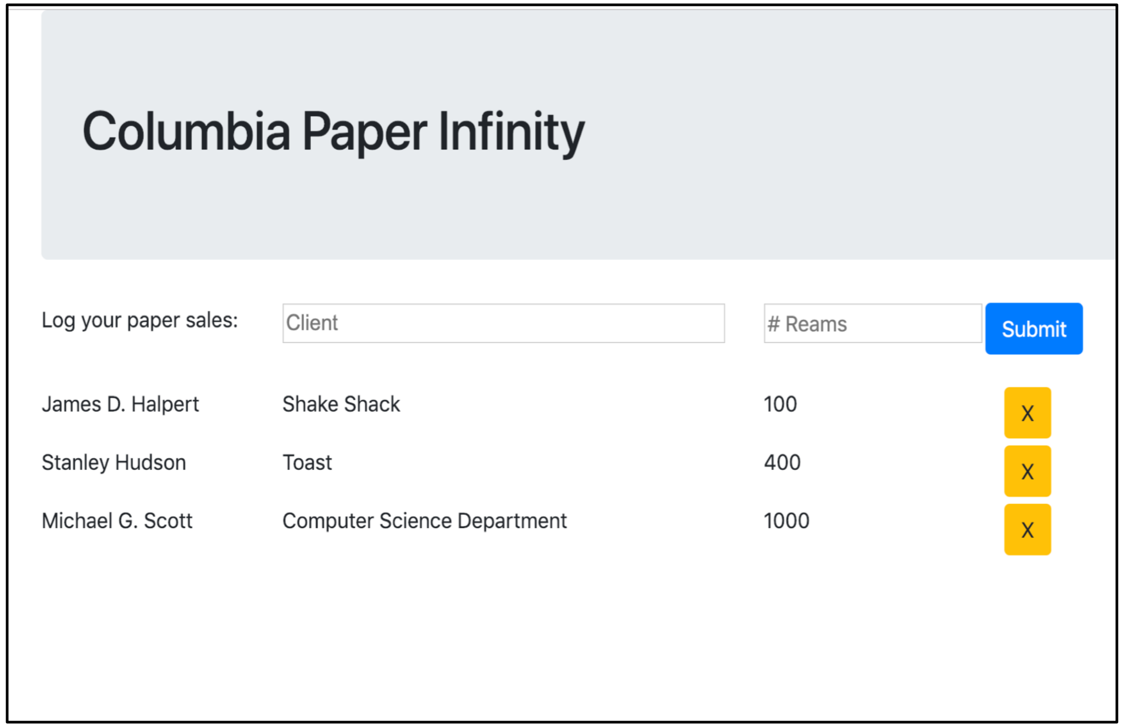


Drag and Drop



Added customization
(hovering and drop target feedback)

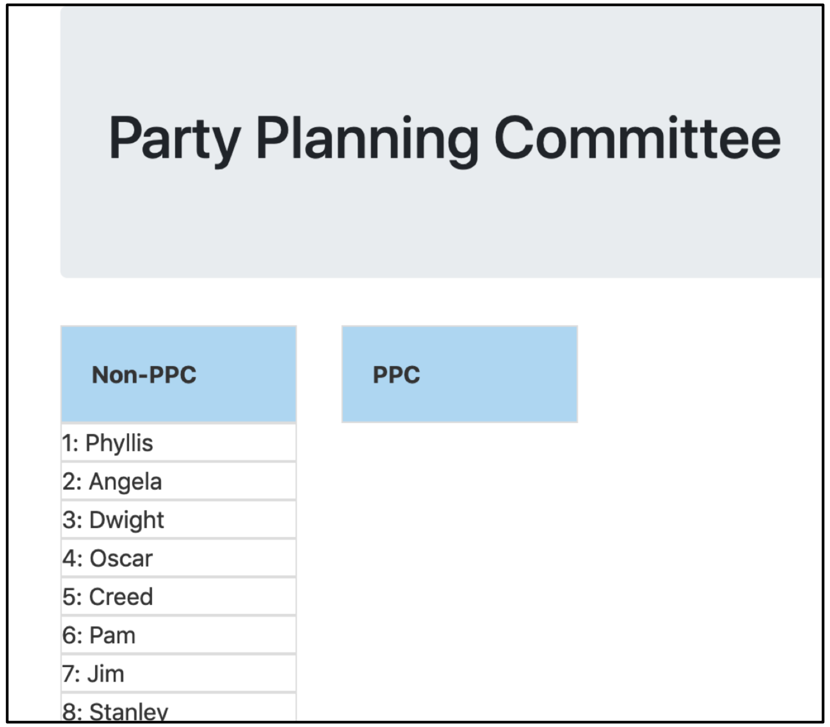
You allowed users to interact with data



Columbia Paper Infinity

Log your paper sales: Client # Reams Submit

James D. Halpert	Shake Shack	100	X
Stanley Hudson	Toast	400	X
Michael G. Scott	Computer Science Department	1000	X



Party Planning Committee

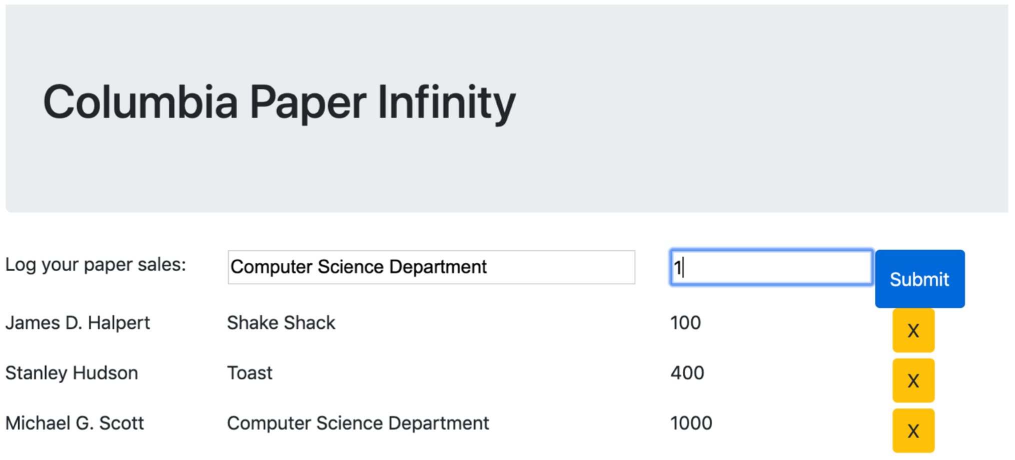
Non-PPC	PPC
1: Phyllis	
2: Angela	
3: Dwight	
4: Oscar	
5: Creed	
6: Pam	
7: Jim	
8: Stanley	

Create / Delete data

Update data

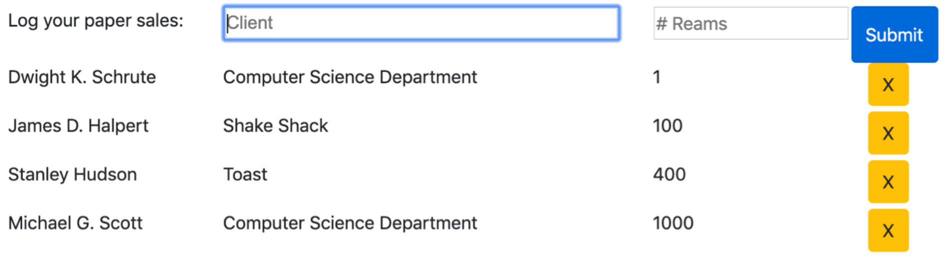
But there's a big problem:

Add data


Columbia Paper Infinity

Log your paper sales:	Computer Science Department	1	Submit
James D. Halpert	Shake Shack	100	X
Stanley Hudson	Toast	400	X
Michael G. Scott	Computer Science Department	1000	X

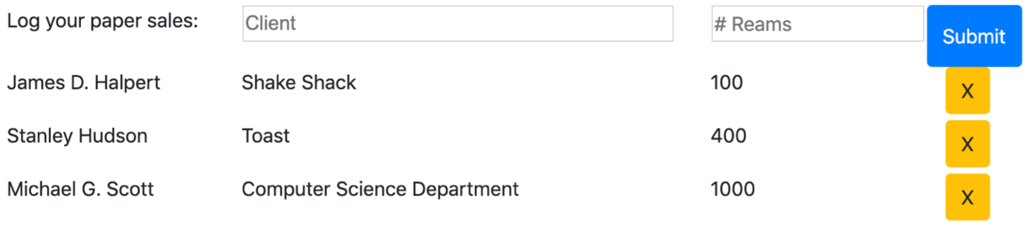
Data appears



Log your paper sales:	Client	# Reams	Submit
Dwight K. Schrute	Computer Science Department	1	X
James D. Halpert	Shake Shack	100	X
Stanley Hudson	Toast	400	X
Michael G. Scott	Computer Science Department	1000	X

REFRESH PAGE

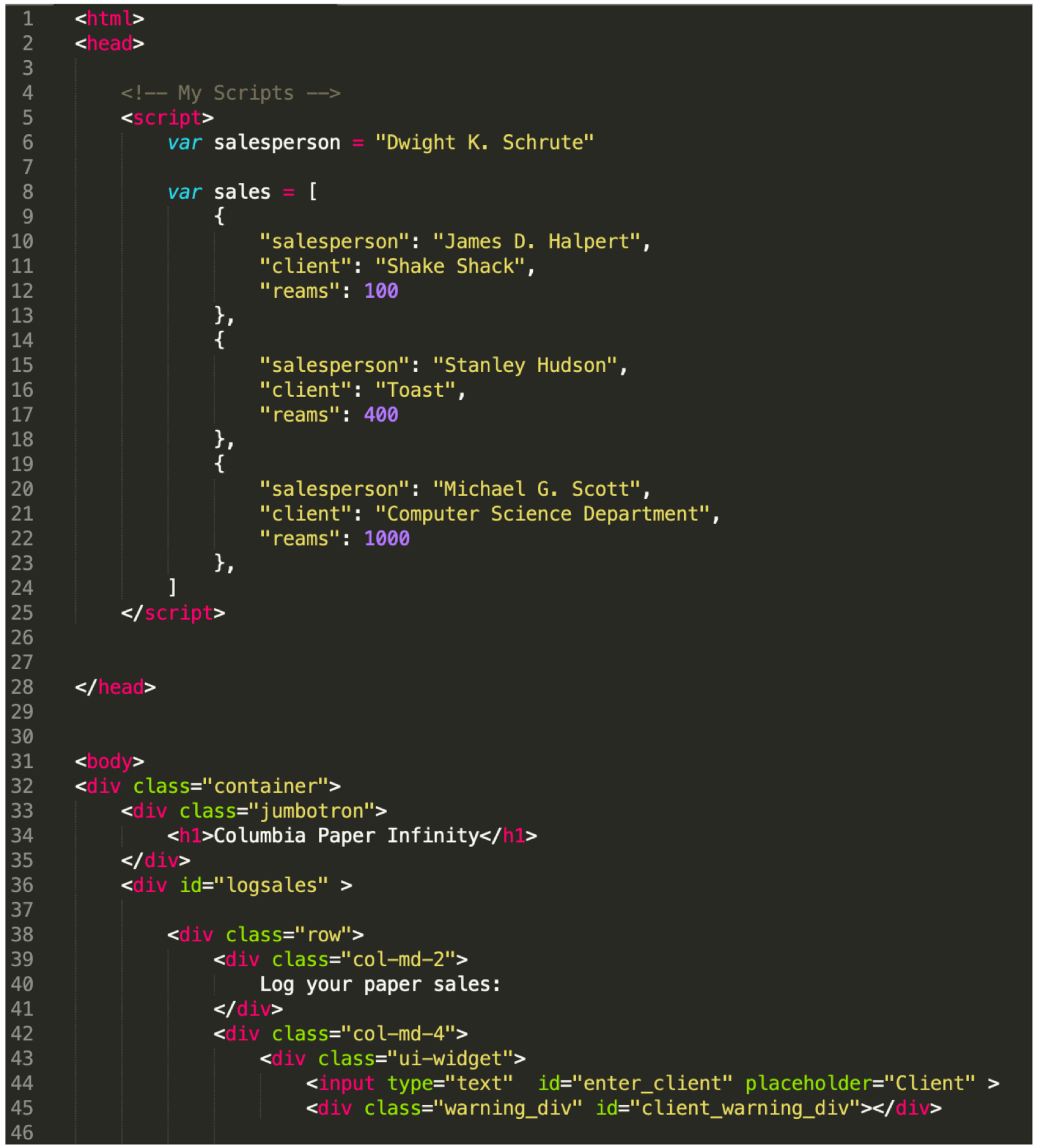
Data is gone!



Log your paper sales:	Client	# Reams	Submit
James D. Halpert	Shake Shack	100	X
Stanley Hudson	Toast	400	X
Michael G. Scott	Computer Science Department	1000	X

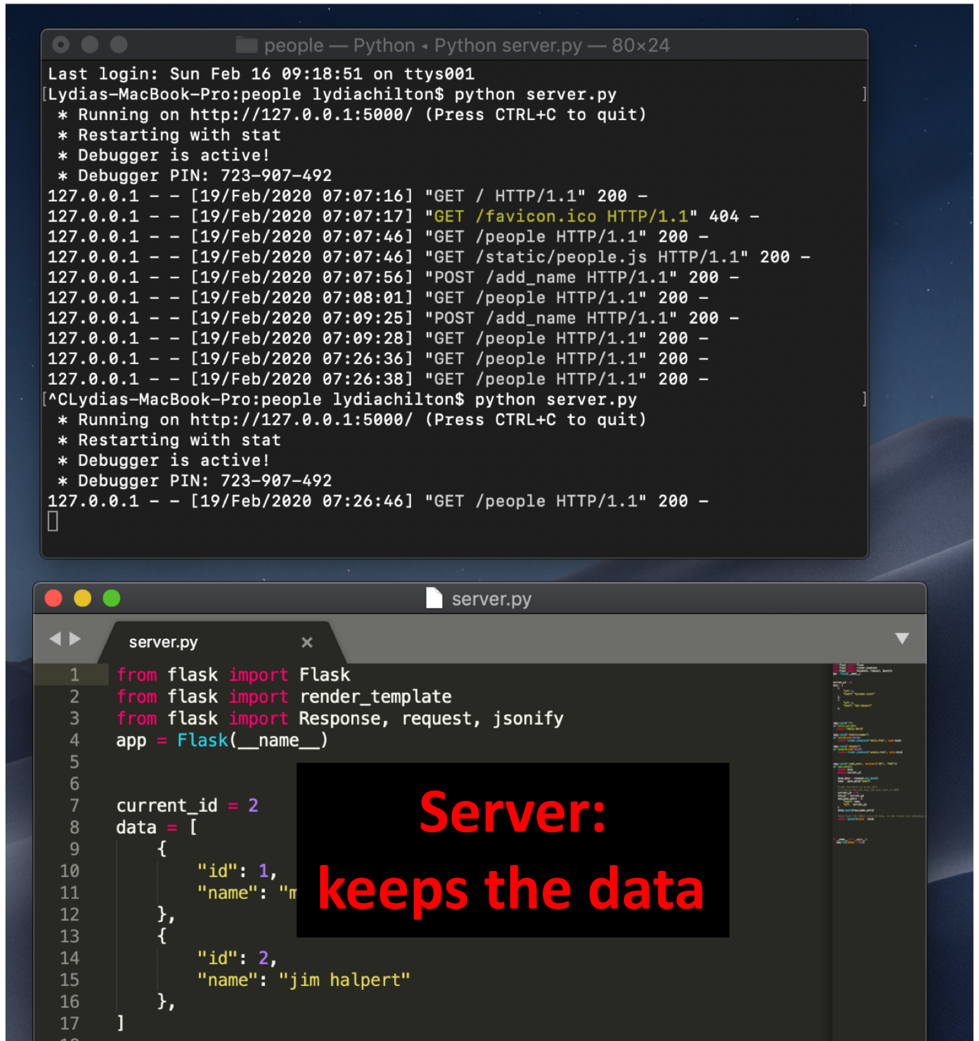
The data doesn't save

In HW4, the data is only stored in the browser



```
<html>
<head>

    <!-- My Scripts -->
    <script>
        var salesperson = "Dwight K. Schrute"

        var sales = [
            {
                "salesperson": "James D. Halpert",
                "client": "Shake Shack",
                "reams": 100
            },
            {
                "salesperson": "Stanley Hudson",
                "client": "Toast",
                "reams": 400
            },
            {
                "salesperson": "Michael G. Scott",
                "client": "Computer Science Department",
                "reams": 1000
            },
        ]
    </script>


</head>


<body>
<div class="container">
    <div class="jumbotron">
        <h1>Columbia Paper Infinity</h1>
    </div>
    <div id="logsales" >

        <div class="row">
            <div class="col-md-2">
                Log your paper sales:
            </div>
            <div class="col-md-4">
                <div class="ui-widget">
                    <input type="text"  id="enter_client" placeholder="Client" >
                    <div class="warning_div" id="client_warning_div"></div>

```

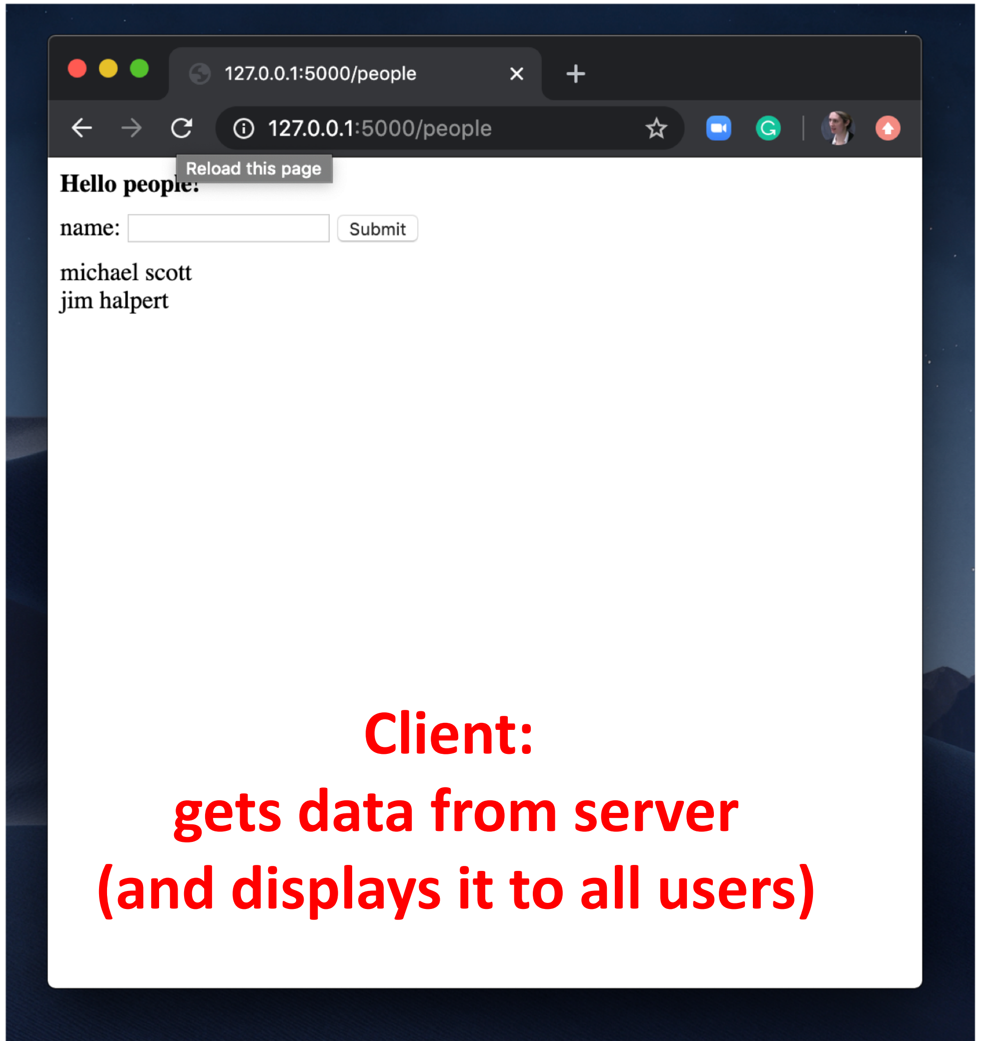
To keep data around, we need to store it somewhere else – another computer that will never get turned off.



```
Last login: Sun Feb 16 09:18:51 on ttys001
Lydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
127.0.0.1 - - [19/Feb/2020 07:07:16] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:17] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [19/Feb/2020 07:07:46] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:46] "GET /static/people.js HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:56] "POST /add_name HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:08:01] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:09:25] "POST /add_name HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:09:28] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:26:36] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:26:38] "GET /people HTTP/1.1" 200 -
^CLydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
127.0.0.1 - - [19/Feb/2020 07:26:46] "GET /people HTTP/1.1" 200 -
```

```python
from flask import Flask
from flask import render_template
from flask import Response, request, jsonify
app = Flask(__name__)


current_id = 2
data = [
    {
        "id": 1,
        "name": "m
    },
    {
        "id": 2,
        "name": "jim halpert"
    },
]
```

Server:
keeps the data



Hello people:

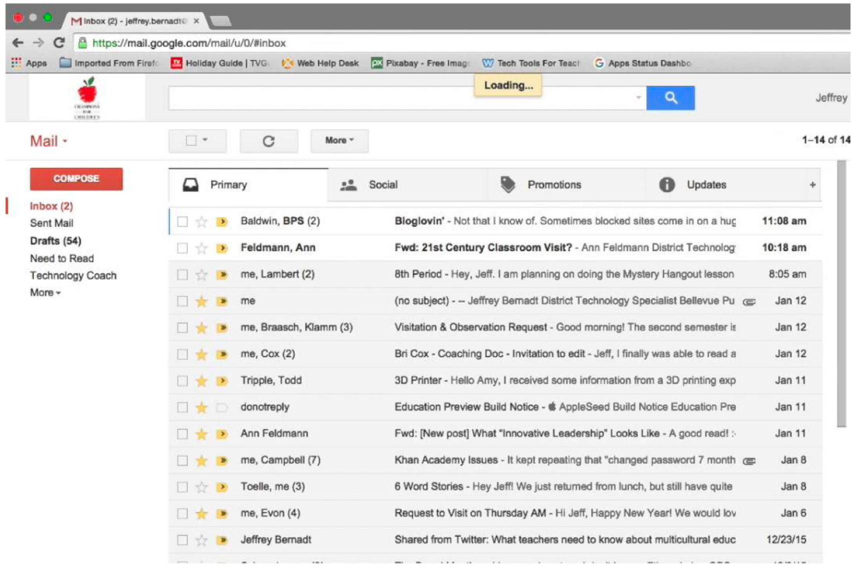
name:

michael scott
jim halpert

Client:
gets data from server
(and displays it to all users)

What data does the server keep?

```
emails = [
    {
    “id”: 9374384320,
    “from”: “bollinger”,
    “to”: “chilton”,
    “subject”: “4170 is awesome!”
    },
    {
    “id”: 038347438,
    “from”: “obama”,
    “to”: “chilton”,
    “subject”: “belated medal of freedom”
    },
]
```



https://mail.google.com/mail/u/0/#inbox

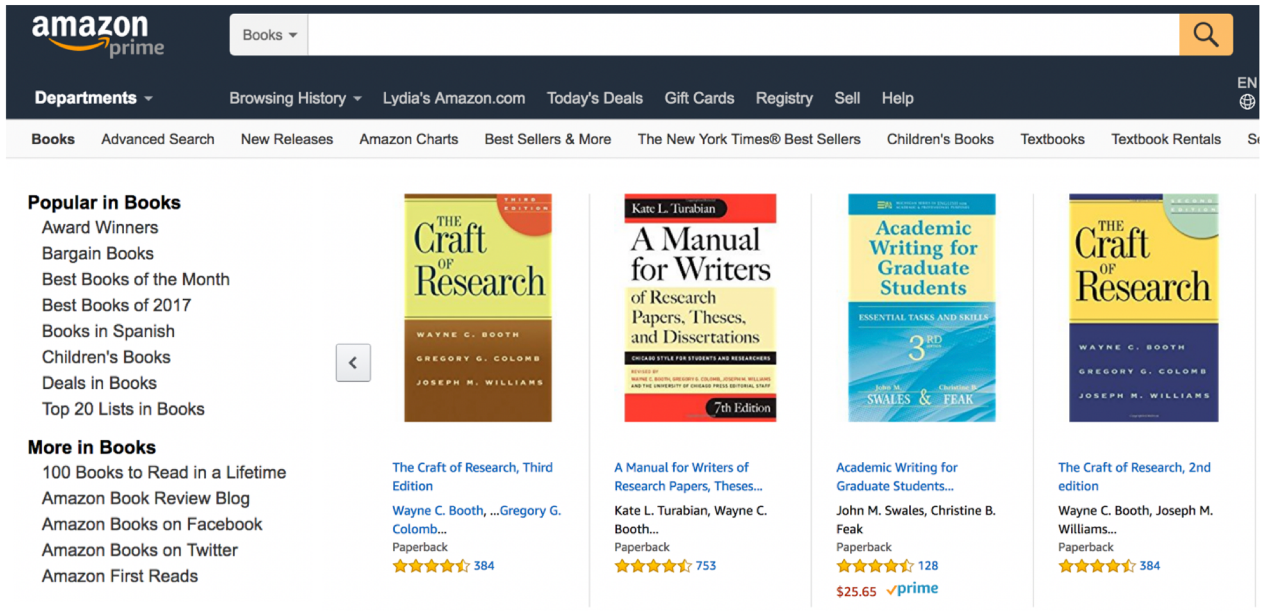
Baldwin, BPS (2)	**Bloglovin'** - Not that I know of. Sometimes blocked sites come in on a hug	11:08 am
Feldmann, Ann	**Fwd: 21st Century Classroom Visit?** - Ann Feldmann District Technolog	10:18 am
me, Lambert (2)	8th Period - Hey, Jeff. I am planning on doing the Mystery Hangout lesson	8:05 am
me	(no subject) - -- Jeffrey Bernadt District Technology Specialist Bellevue Pu	Jan 12
me, Braasch, Klamm (3)	Visitation & Observation Request - Good morning! The second semester is	Jan 12
me, Cox (2)	Bri Cox - Coaching Doc - Invitation to edit - Jeff, I finally was able to read a	Jan 12
Tripple, Todd	3D Printer - Hello Amy, I received some information from a 3D printing exp	Jan 11
donotreply	Education Preview Build Notice - AppleSeed Build Notice Education Pre	Jan 11
Ann Feldmann	Fwd: [New post] What "Innovative Leadership" Looks Like - A good read! :-	Jan 11
me, Campbell (7)	Khan Academy Issues - It kept repeating that "changed password 7 month	Jan 8
Toelle, me (3)	6 Word Stories - Hey Jeff! We just returned from lunch, but still have quite	Jan 8
me, Evon (4)	Request to Visit on Thursday AM - Hi Jeff, Happy New Year! We would lov	Jan 6
Jeffrey Bernadt	Shared from Twitter: What teachers need to know about multicultural educ	12/23/15

Server:
keeps the data

Client:
gets data from server
(and displays it to all users)

What data does the server keep?

```
products = [
    {
    “id”: 694274583,
    “title”: “Ivy League Web Design”,
    “author”: “chilton”,
    “stars”: “5”
    },
    {
    “id”: 28447430033,
    “title”: “JavaScript and You”,
    “author”: “chilton”,
    “stars”: “6”
    },
]
```



Popular in Books
- Award Winners
- Bargain Books
- Best Books of the Month
- Best Books of 2017
- Books in Spanish
- Children's Books
- Deals in Books
- Top 20 Lists in Books

More in Books
- 100 Books to Read in a Lifetime
- Amazon Book Review Blog
- Amazon Books on Facebook
- Amazon Books on Twitter
- Amazon First Reads

The Craft of Research, Third Edition
Wayne C. Booth, ...Gregory G. Colomb...
Paperback
384

A Manual for Writers of Research Papers, Theses...
Kate L. Turabian, Wayne C. Booth...
Paperback
753

Academic Writing for Graduate Students...
John M. Swales, Christine B. Feak
Paperback
128
$25.65 prime

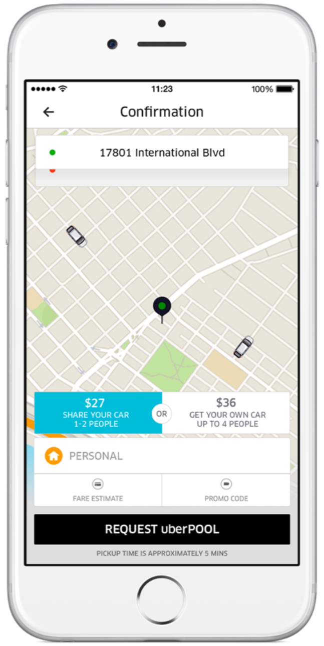
The Craft of Research, 2nd edition
Wayne C. Booth, Joseph M. Williams...
Paperback
384

Server:
keeps the data

Client:
gets data from server
(and displays it to all users)

What data does the server keep?

```
cars = [
    {
    “id”: 847434714,
    “location”: “116 and broadway”,
    “driver”: “michael roger”,
    “car type”: “uber XL”
    },
    {
    “id”: 55429181,
    “location”: “times square”,
    “driver”: “grace li”,
    “car type”: “normal”
    },
]
```



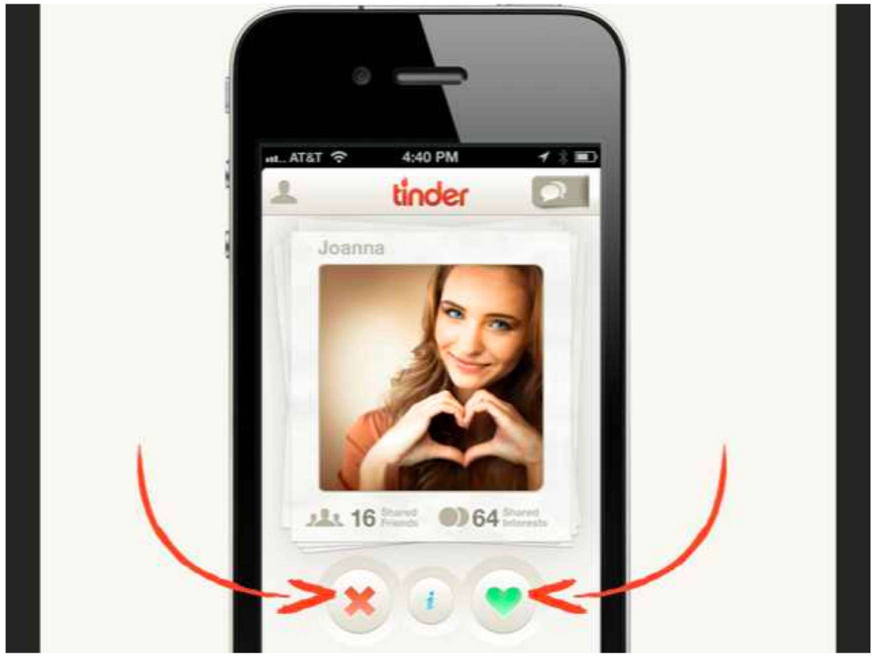
Server:
keeps the data

Client:
gets data from server
(and displays it to all users)

What data does the server keep?

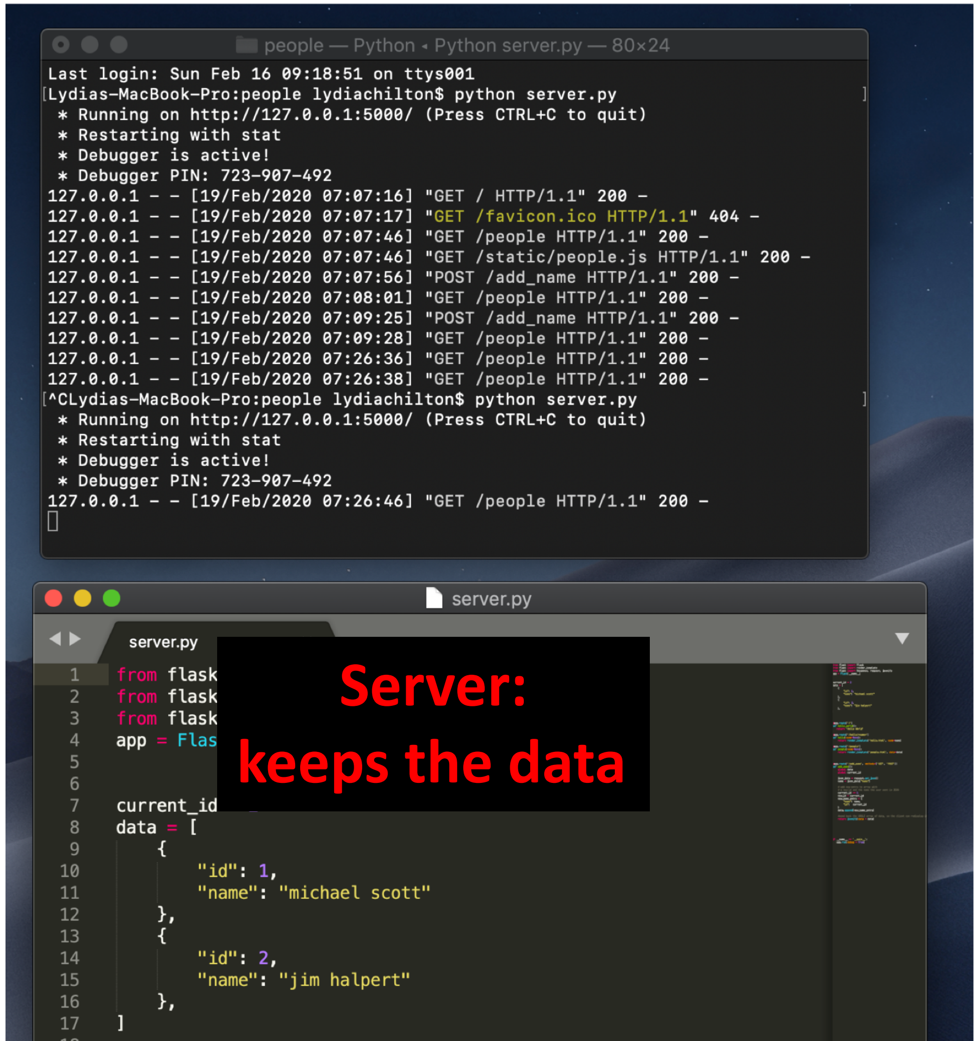
```
profiles = [
    {
    “id”: 707072343,
    “name”: “nigel”,
    “image”: “./nigel.png”,
    “likes”: “1000”,
    ”dislikes”: 0,
    },
    {
    “id”: 821212134,
    “name”: “lidia”,
    “image”: “./lidia.png”,
    “likes”: “1000”,
    “dislikes”: 0,
    },
]
```

Server:
keeps the data



Client:
gets data from server
(and displays it to all users)

We need to have another computer **store and serve the data.**

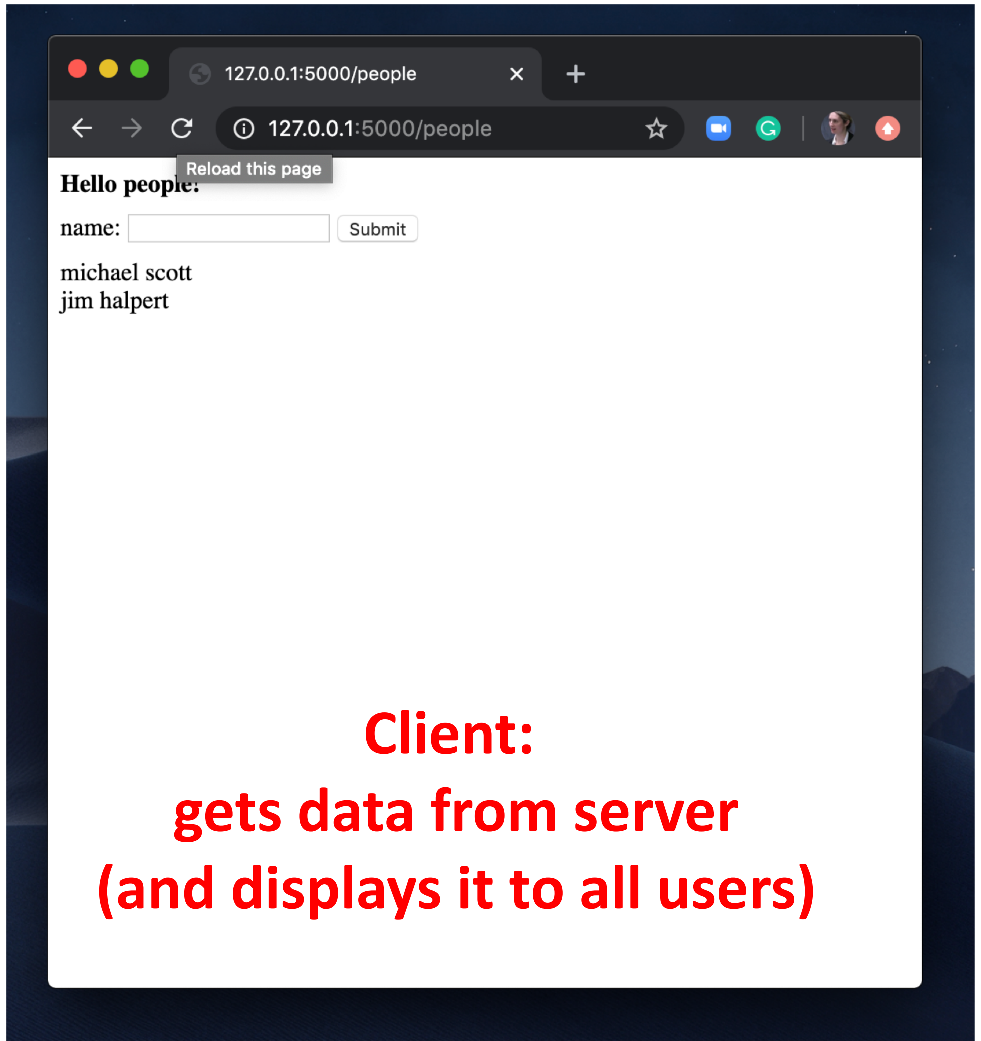


```
people — Python ◂ Python server.py — 80×24
Last login: Sun Feb 16 09:18:51 on ttys001
Lydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
127.0.0.1 - - [19/Feb/2020 07:07:16] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:17] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [19/Feb/2020 07:07:46] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:46] "GET /static/people.js HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:56] "POST /add_name HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:08:01] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:09:25] "POST /add_name HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:09:28] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:26:36] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:26:38] "GET /people HTTP/1.1" 200 -
^CLydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
127.0.0.1 - - [19/Feb/2020 07:26:46] "GET /people HTTP/1.1" 200 -
```

server.py

```python
from flask
from flask
from flask
app = Flas


current_id
data = [
    {
        "id": 1,
        "name": "michael scott"
    },
    {
        "id": 2,
        "name": "jim halpert"
    },
]
```

Server:
keeps the data



127.0.0.1:5000/people

Reload this page

Hello people:

name: [ ] Submit

michael scott
jim halpert

Client:
gets data from server
(and displays it to all users)

Example application:

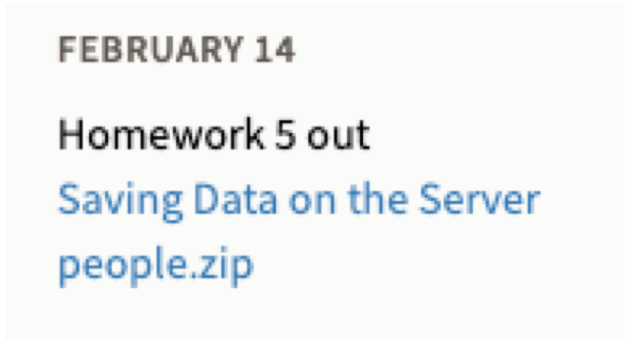
Storing and Serving data in Flask

We will use Flask web framework to server our applications. It’s in python.

Flask

web development,
one drop at a time

The HW5 warm up is to download a flask application and run it.

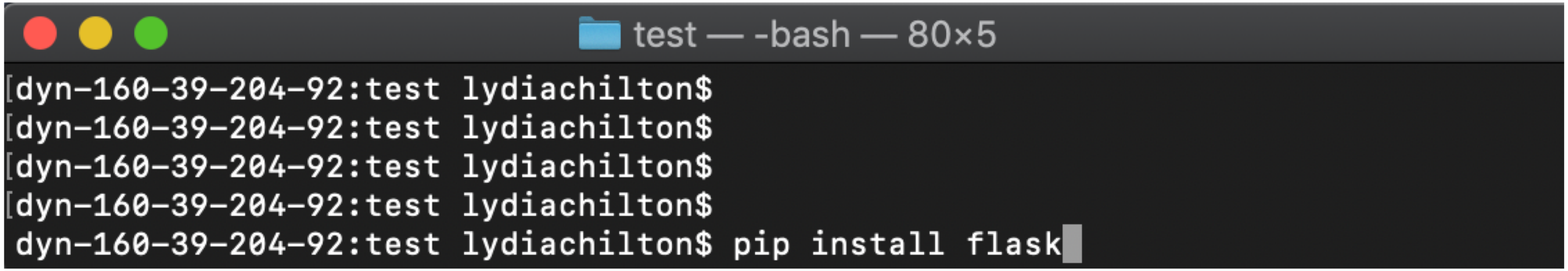

FEBRUARY 14

Homework 5 out

Saving Data on the Server

people.zip

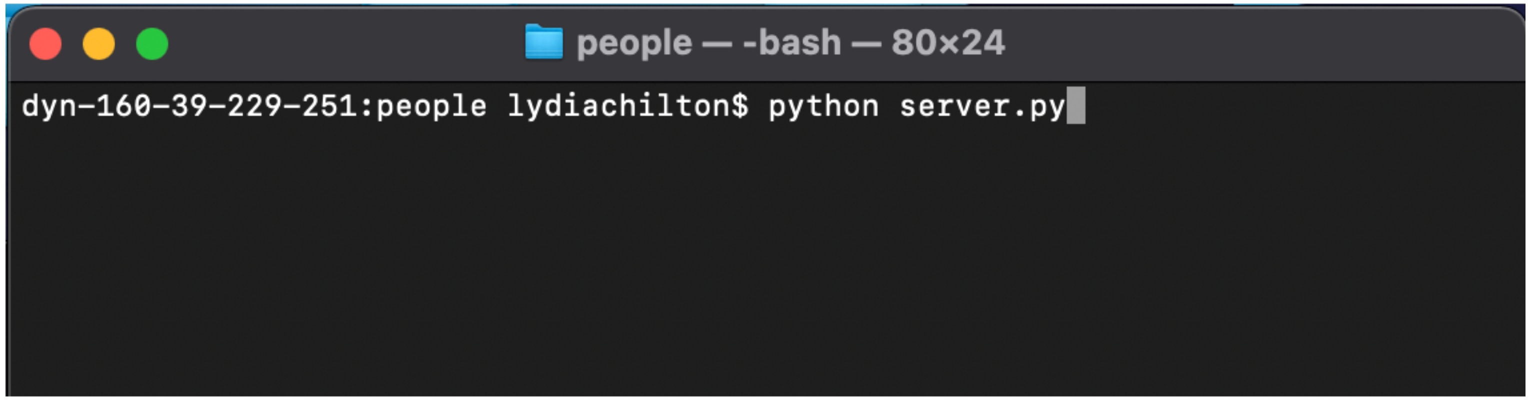
You must first install Flask



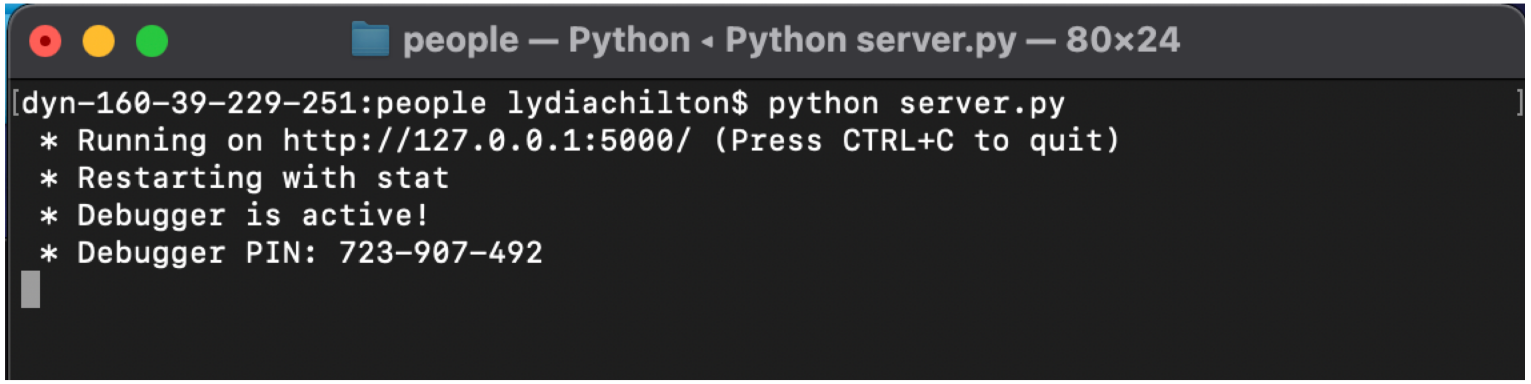
```
[dyn-160-39-204-92:test lydiachilton$
[dyn-160-39-204-92:test lydiachilton$
[dyn-160-39-204-92:test lydiachilton$
[dyn-160-39-204-92:test lydiachilton$
 dyn-160-39-204-92:test lydiachilton$ pip install flask
```

Then run the server.py file.

Type “python server.py” in the terminal inside the project folder or “python3 server.py”

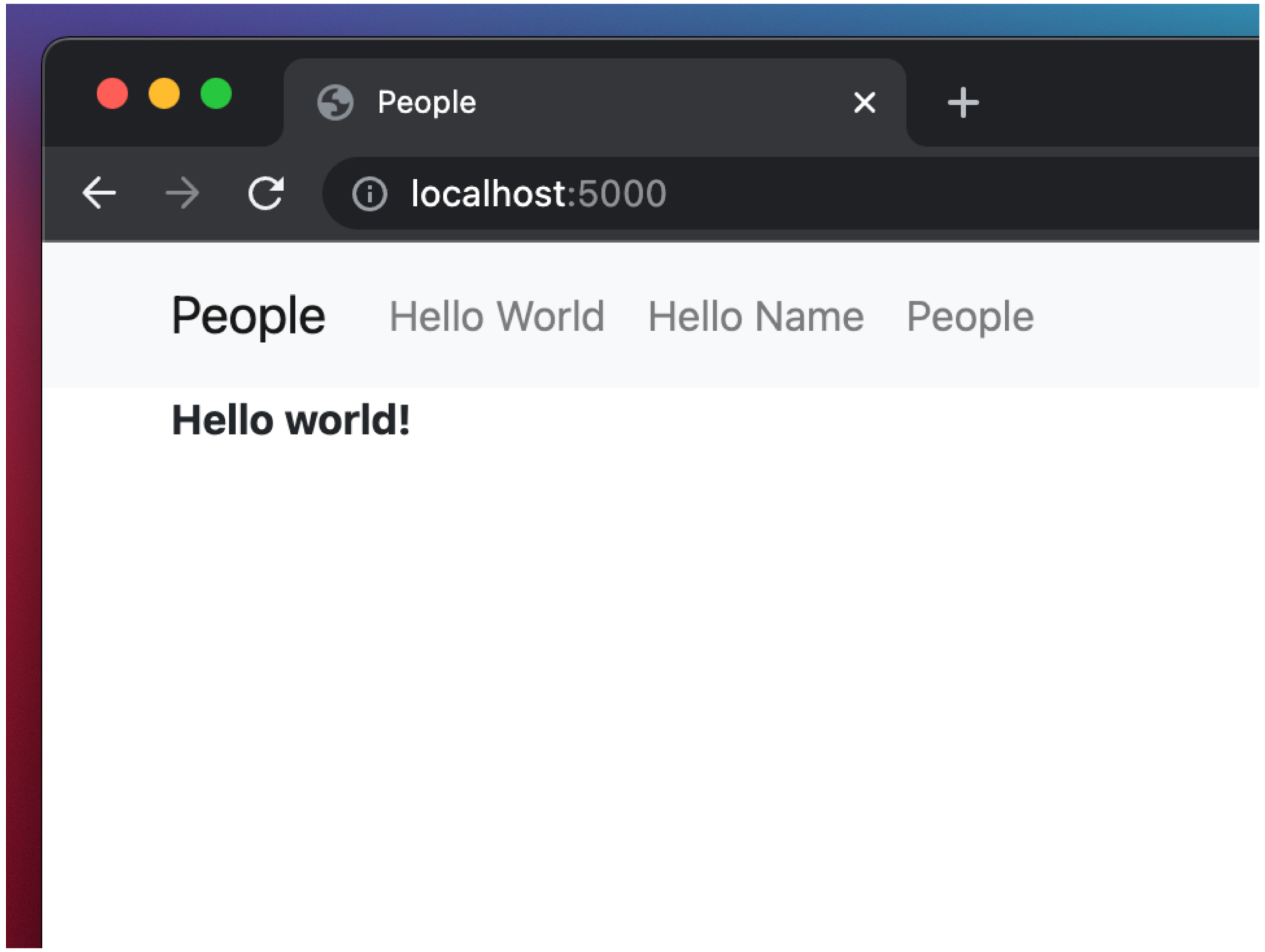


```
dyn-160-39-229-251:people lydiachilton$ python server.py
```



```
[dyn-160-39-229-251:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
```

See you site at: http://localhost:5000/


People

localhost:5000

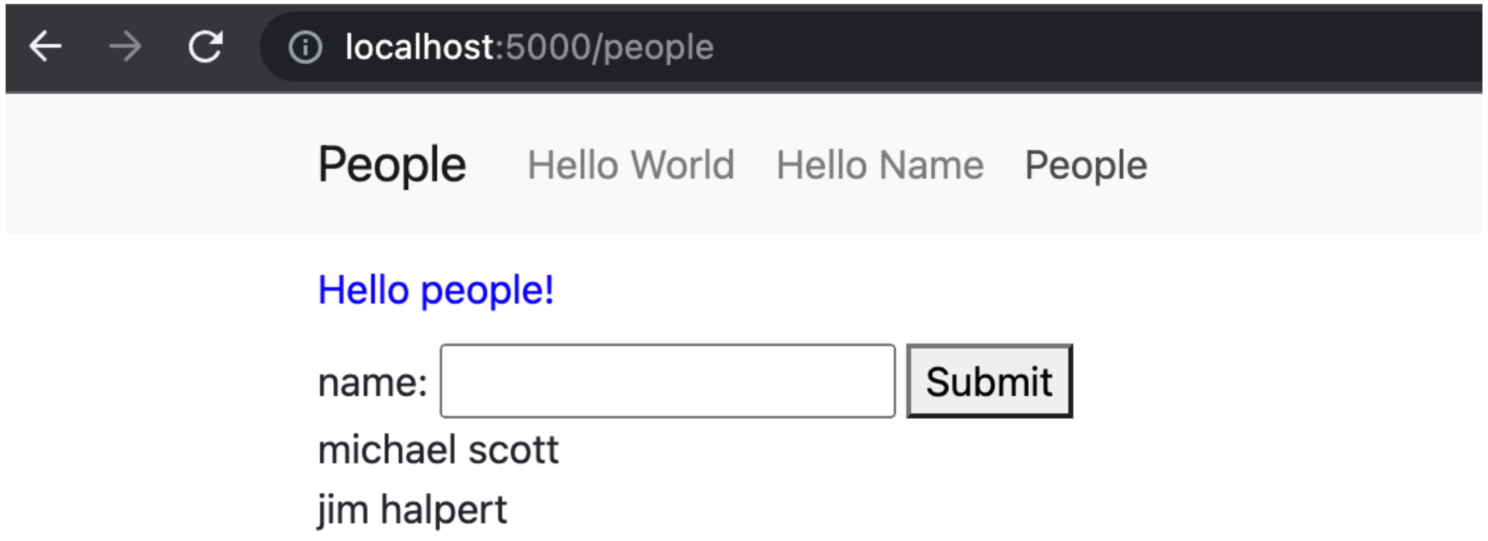
People Hello World Hello Name People

Hello world!

Navbar!

Content block!

http://localhost:5000/people
lets you create a list of names (look familiar?)


localhost:5000/people

People Hello World Hello Name People

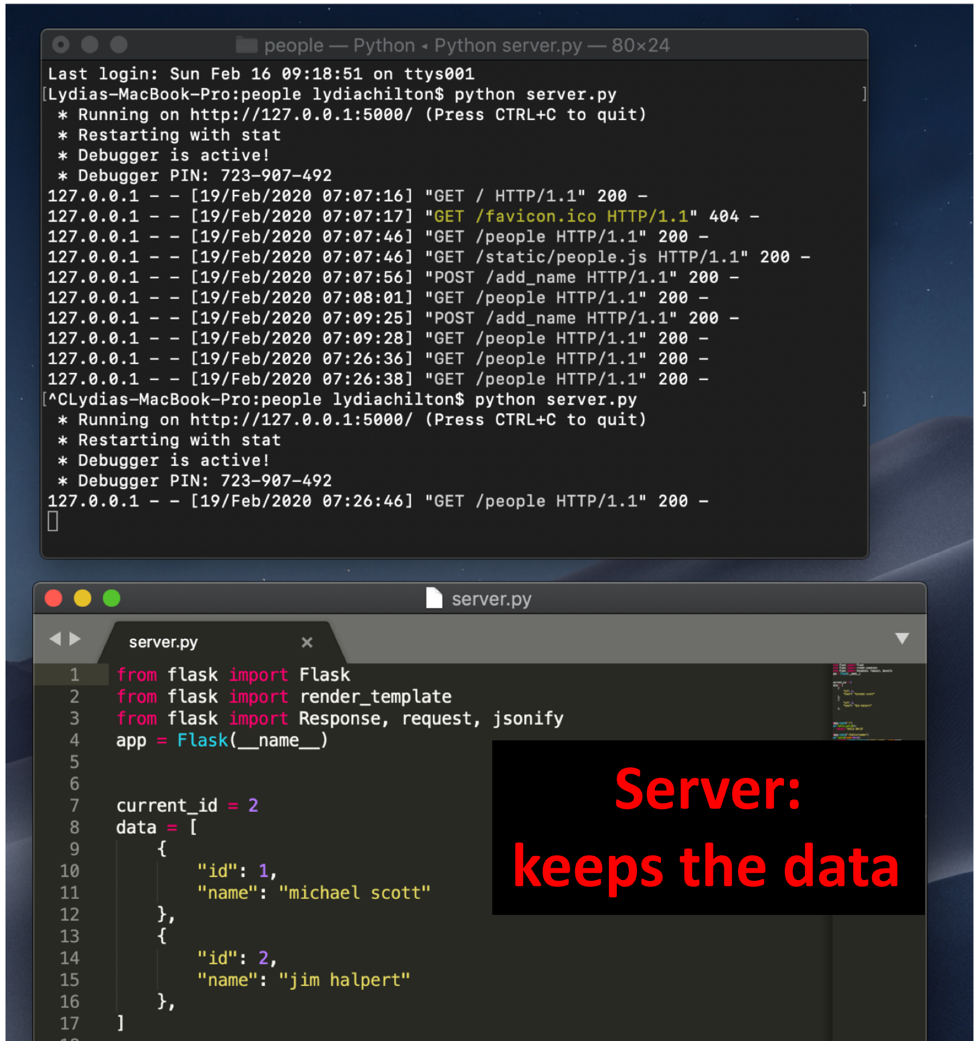
Hello people!

name: Submit

michael scott

jim halpert

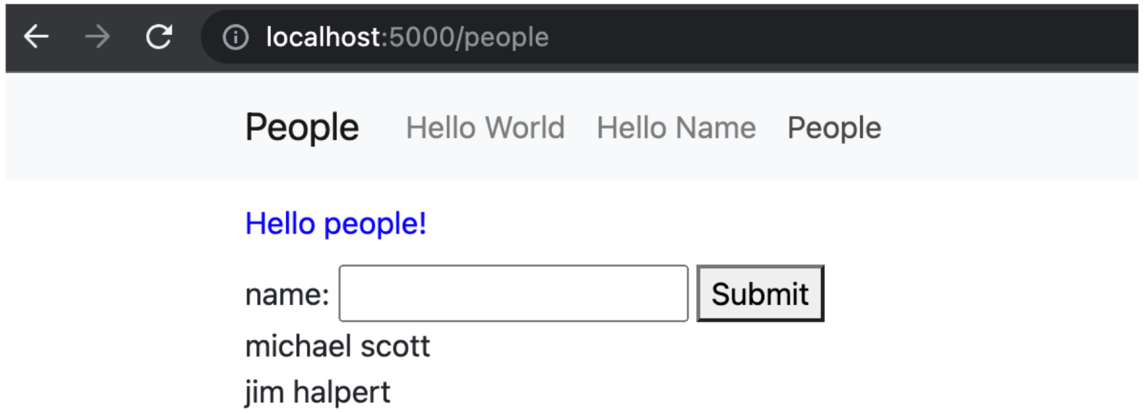
Now the data is stored on the server, not the client



```
Last login: Sun Feb 16 09:18:51 on ttys001
Lydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
127.0.0.1 - - [19/Feb/2020 07:07:16] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:17] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [19/Feb/2020 07:07:46] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:46] "GET /static/people.js HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:56] "POST /add_name HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:08:01] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:09:25] "POST /add_name HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:09:28] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:26:36] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:26:38] "GET /people HTTP/1.1" 200 -
^CLydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
127.0.0.1 - - [19/Feb/2020 07:26:46] "GET /people HTTP/1.1" 200 -
```

```python
from flask import Flask
from flask import render_template
from flask import Response, request, jsonify
app = Flask(__name__)


current_id = 2
data = [
    {
        "id": 1,
        "name": "michael scott"
    },
    {
        "id": 2,
        "name": "jim halpert"
    },
]
```

Server:
keeps the data



Hello people!

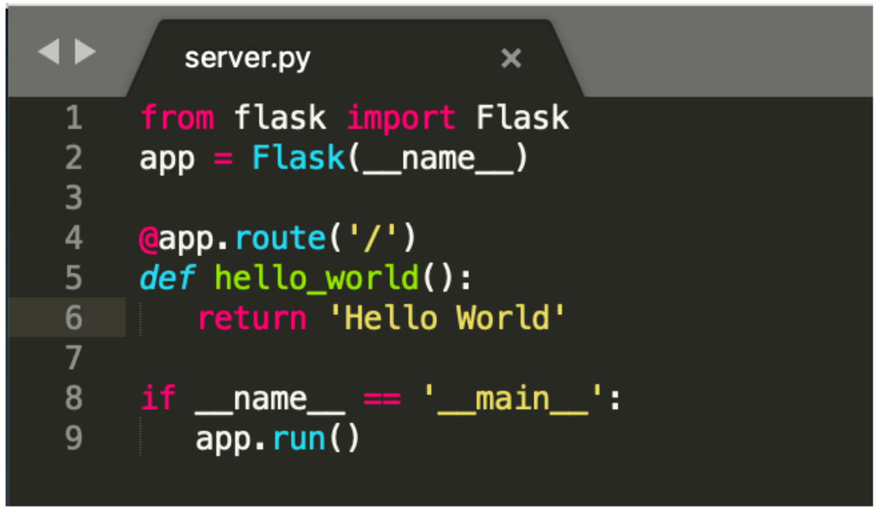
name:

michael scott

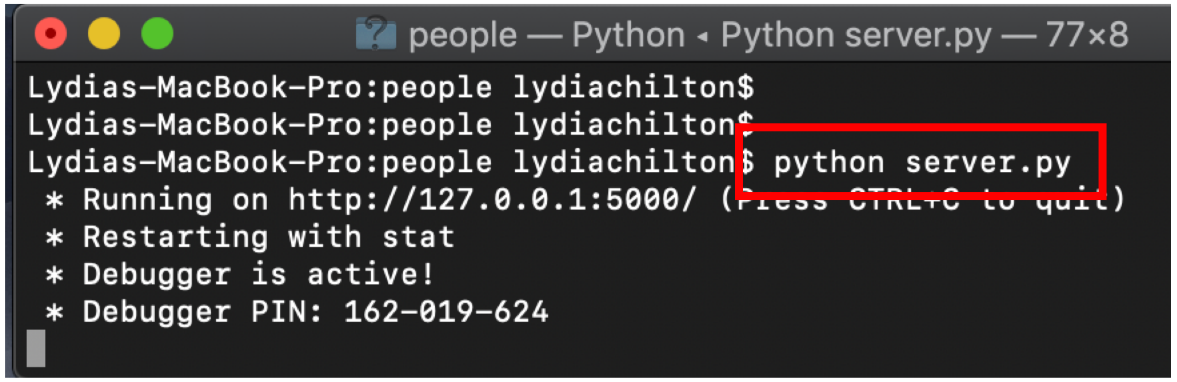
jim halpert

Client:
gets data from server
(and displays it to all users)

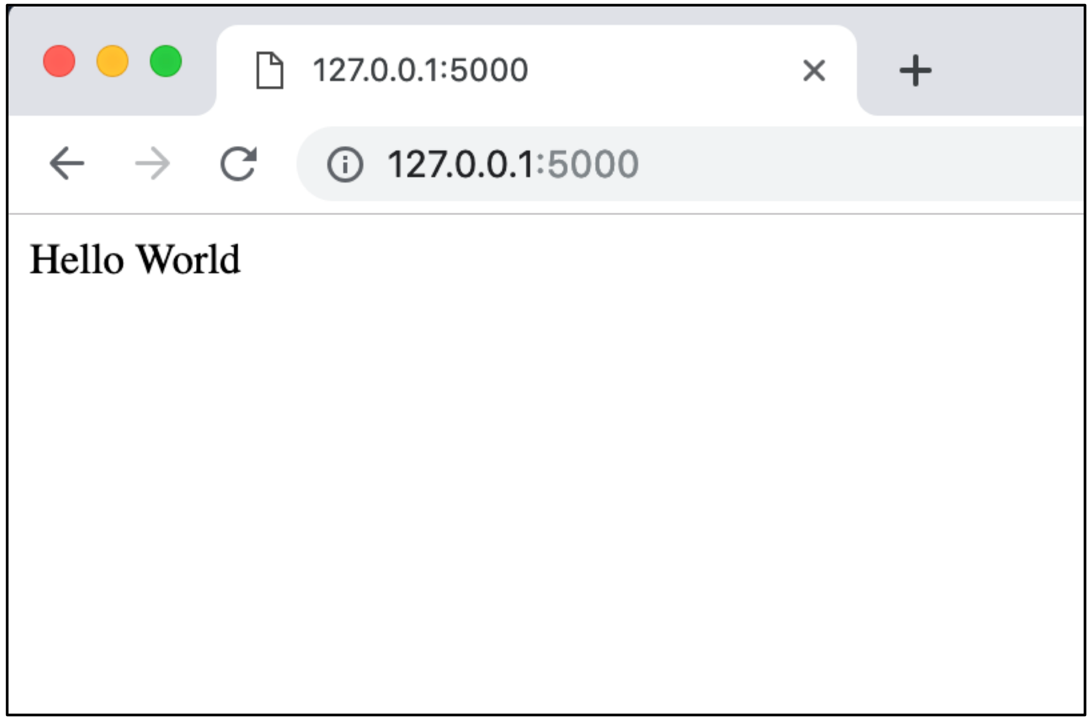
Let’s see the world’s smallest Flask app. Now what?



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

@app.route('/')
def hello_world():
    return 'Hello World'

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

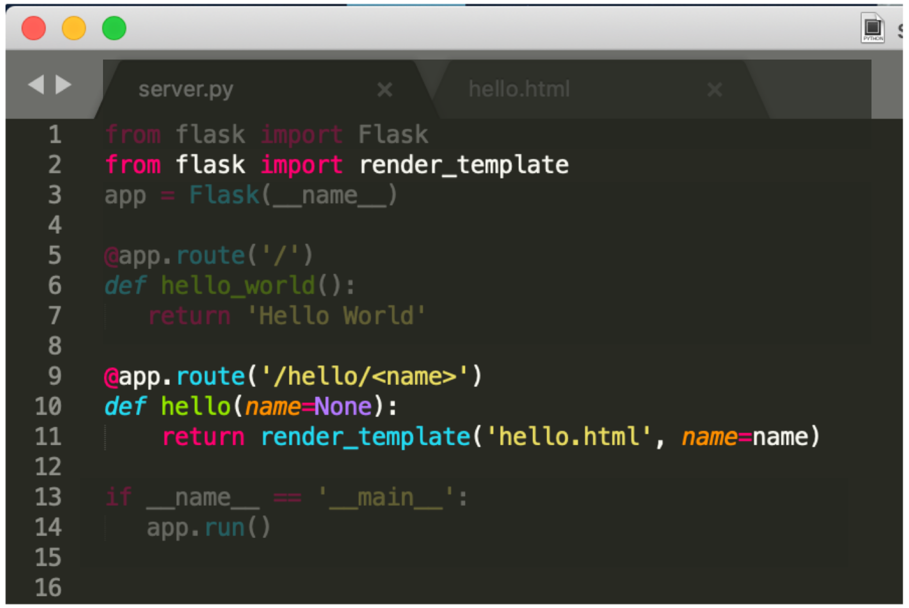


```
Lydias-MacBook-Pro:people lydiachilton$
Lydias-MacBook-Pro:people lydiachilton$
Lydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 162-019-624
```

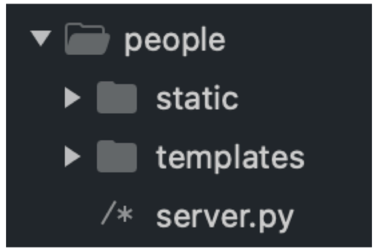


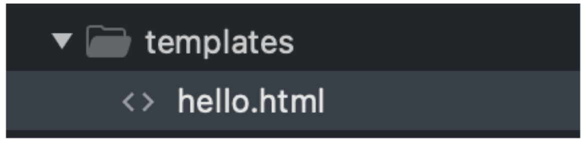
Hello World

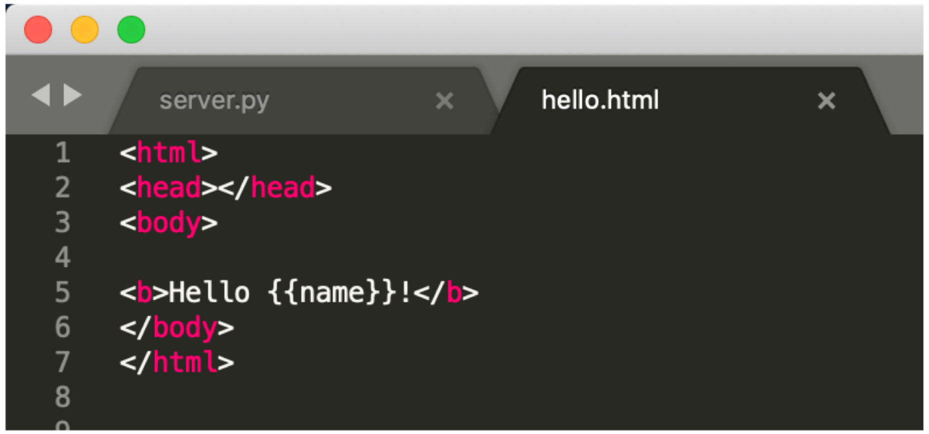
How to render an HTML page with data



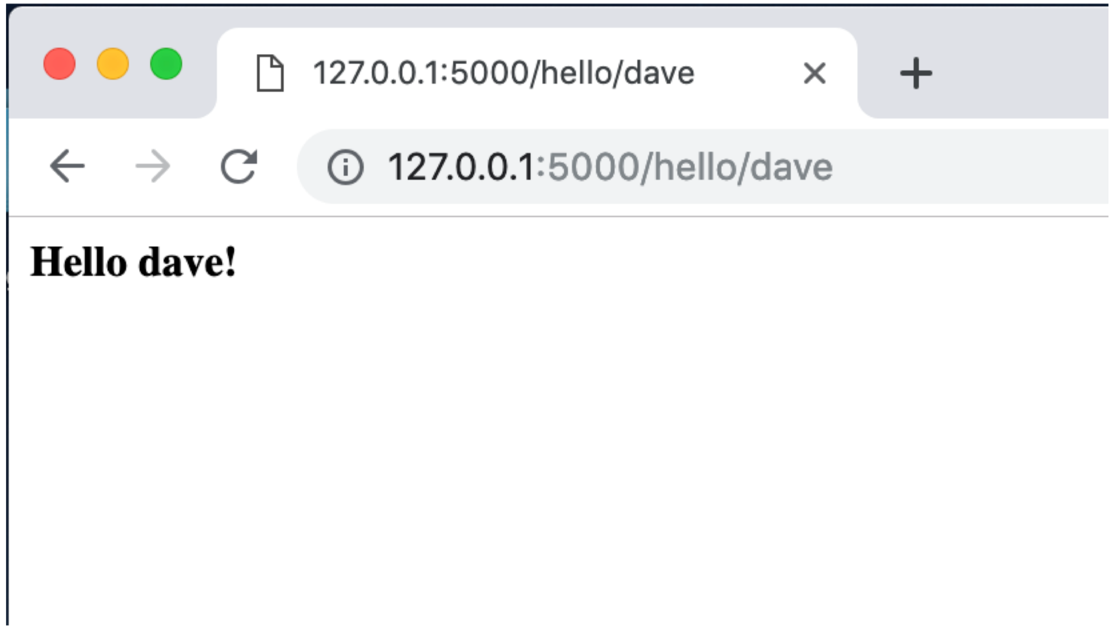
```python
from flask import Flask
from flask import render_template
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello World'

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

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





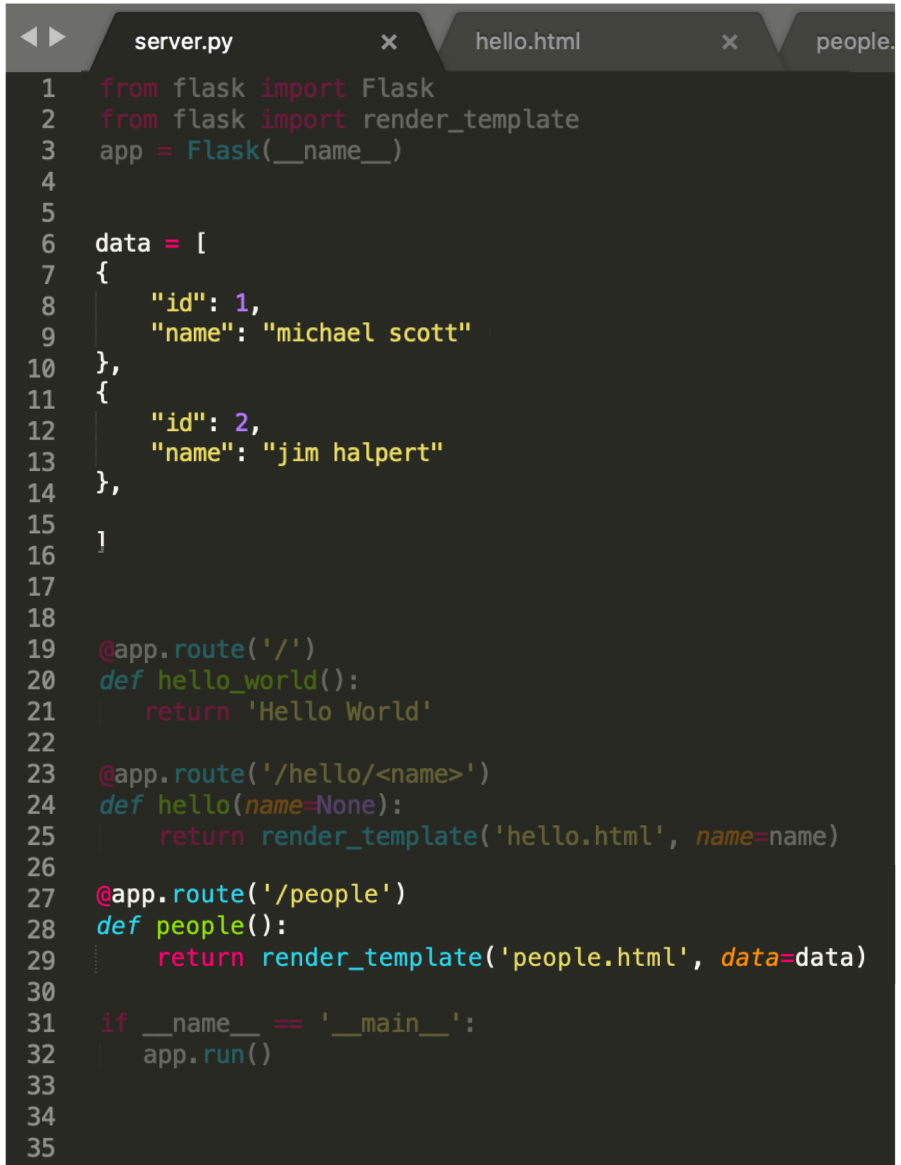


```html
<html>
<head></head>
<body>

<b>Hello {{name}}!</b>
</body>
</html>
```

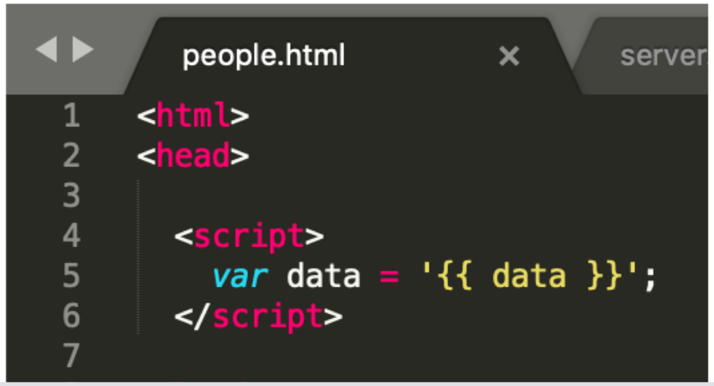


Hello dave!

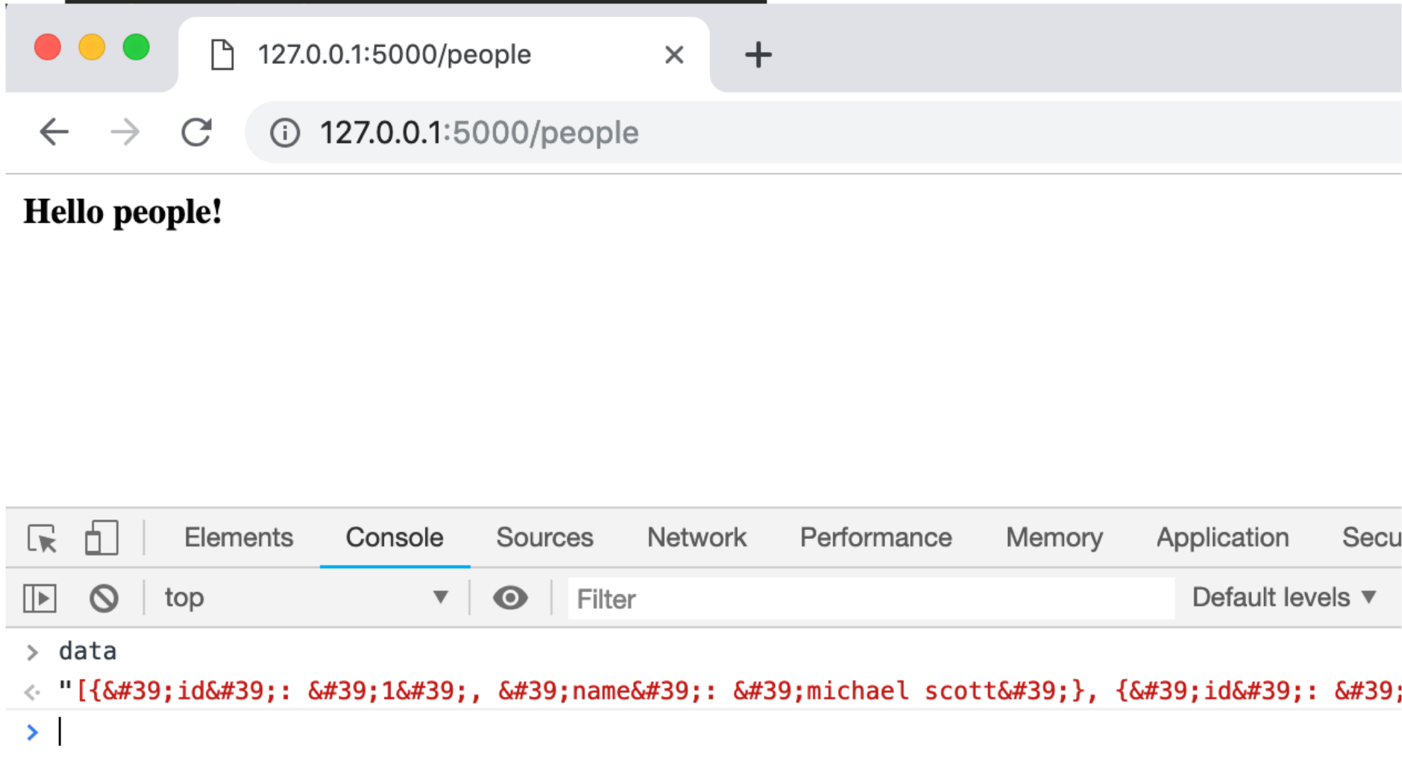
How to send an array of data to JavaScript?



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


data = [
{
    "id": 1,
    "name": "michael scott"
},
{
    "id": 2,
    "name": "jim halpert"
},

]



@app.route('/')
def hello_world():
    return 'Hello World'

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

@app.route('/people')
def people():
    return render_template('people.html', data=data)

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



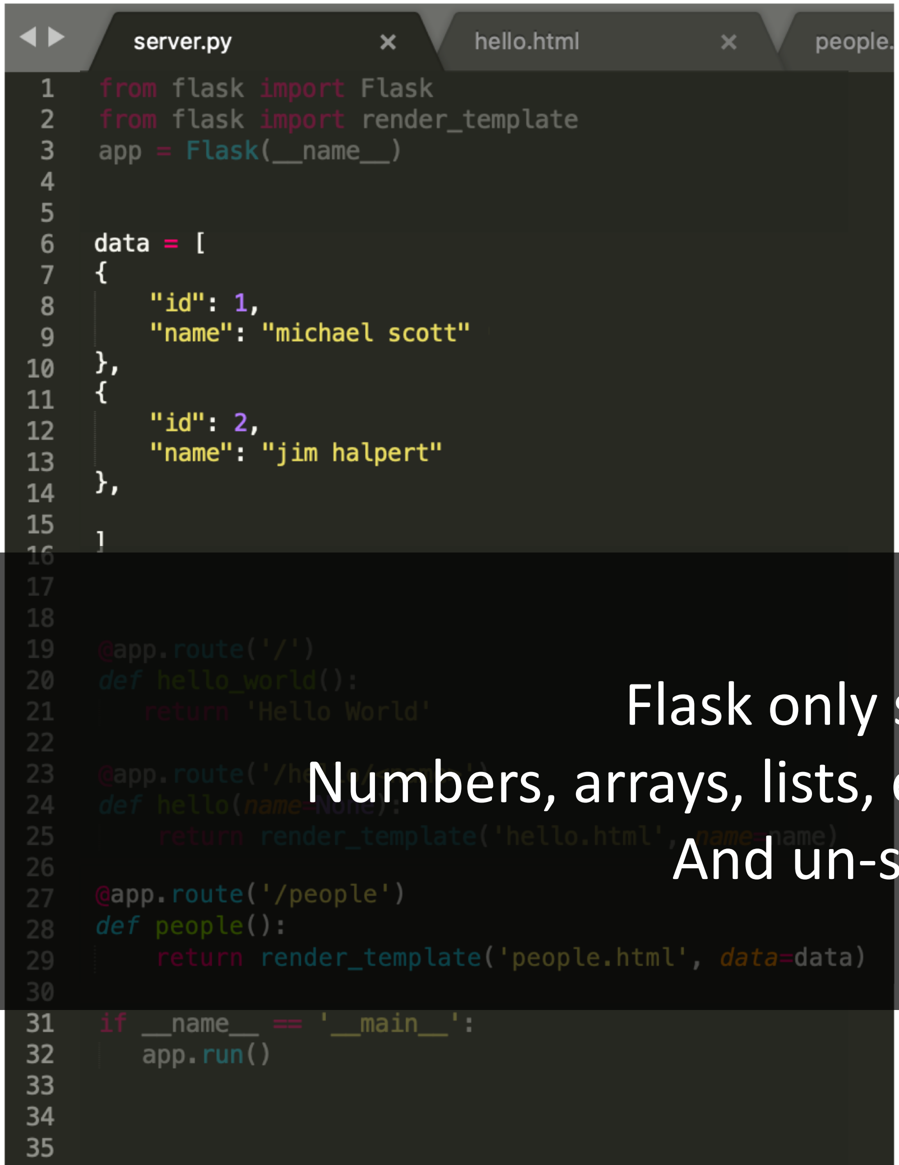
```html
<html>
<head>

  <script>
    var data = '{{ data }}';
  </script>

```



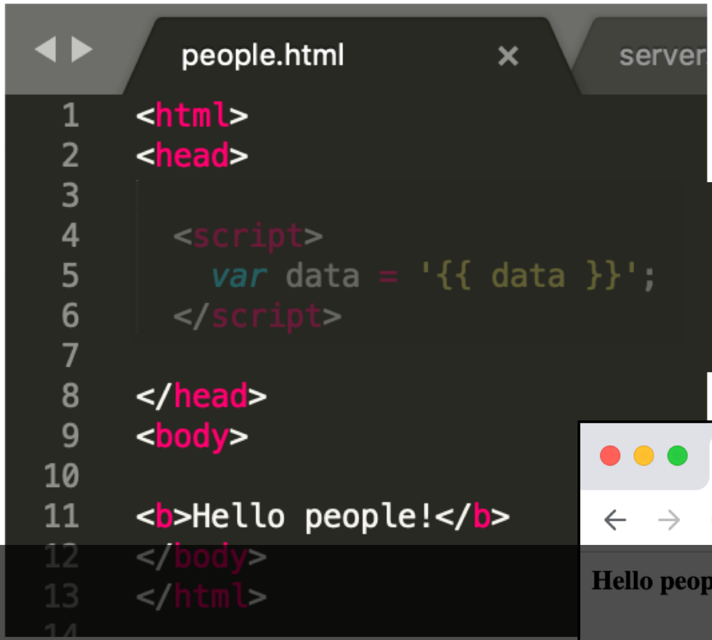
Hello people!

```
> data
<· "[{&#39;id&#39;: &#39;1&#39;, &#39;name&#39;: &#39;michael scott&#39;}, {&#39;id&#39;: &#39;
> |
```

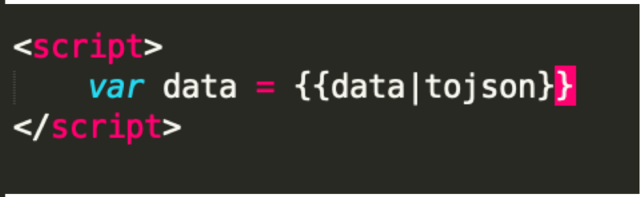
How to send an array of data to JavaScript?



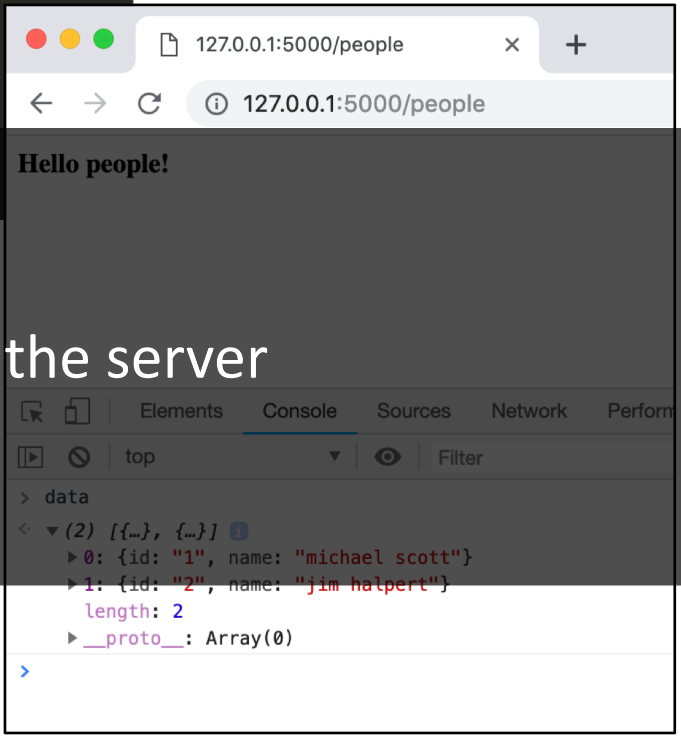
```python
from flask import Flask
from flask import render_template
app = Flask(__name__)


data = [
{
    "id": 1,
    "name": "michael scott"
},
{
    "id": 2,
    "name": "jim halpert"
},

]



@app.route('/')
def hello_world():
    return 'Hello World'

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

@app.route('/people')
def people():
    return render_template('people.html', data=data)

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



```html
<html>
<head>

  <script>
    var data = '{{ data }}';
  </script>

</head>
<body>

<b>Hello people!</b>
</body>
</html>
```



```html
<script>
    var data = {{data|tojson}}
</script>
```

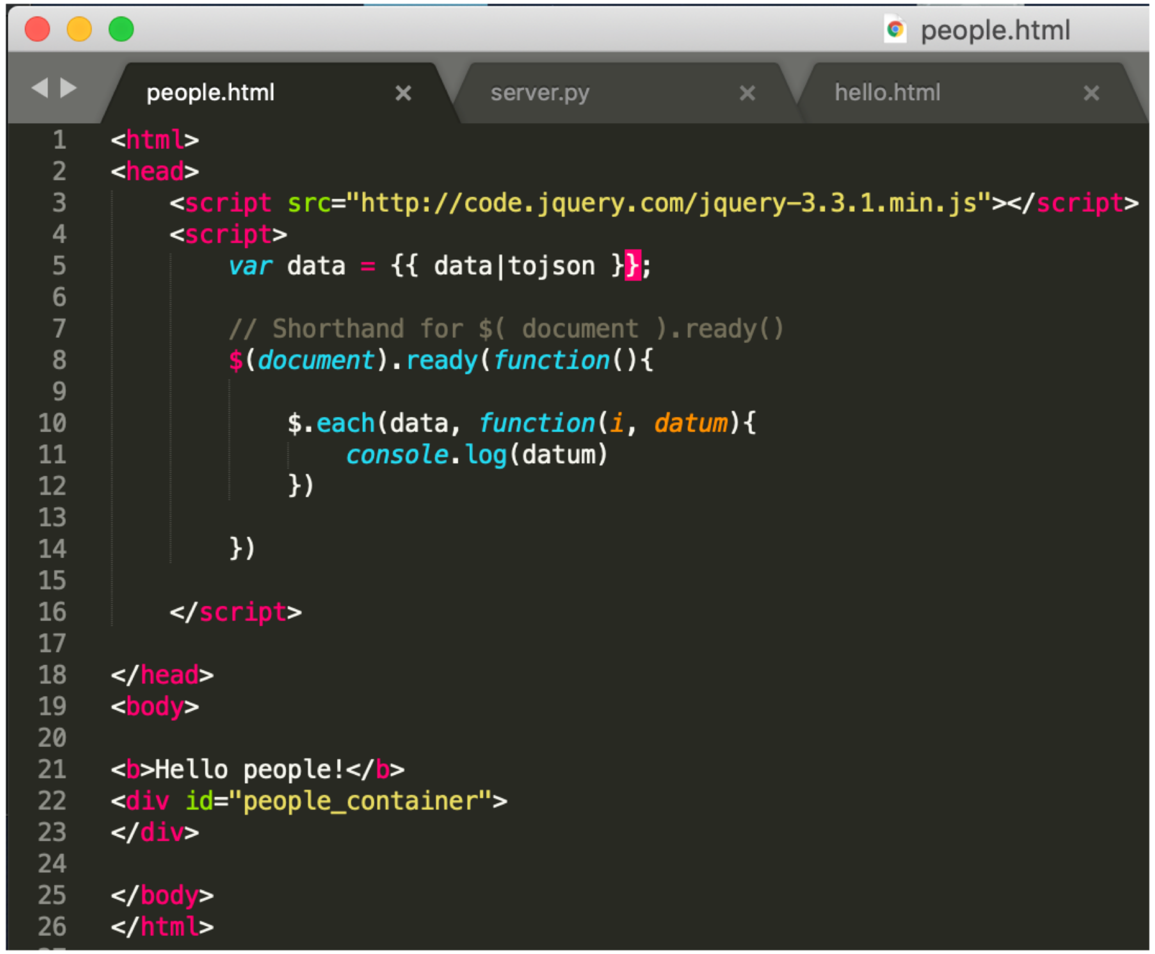


Hello people!

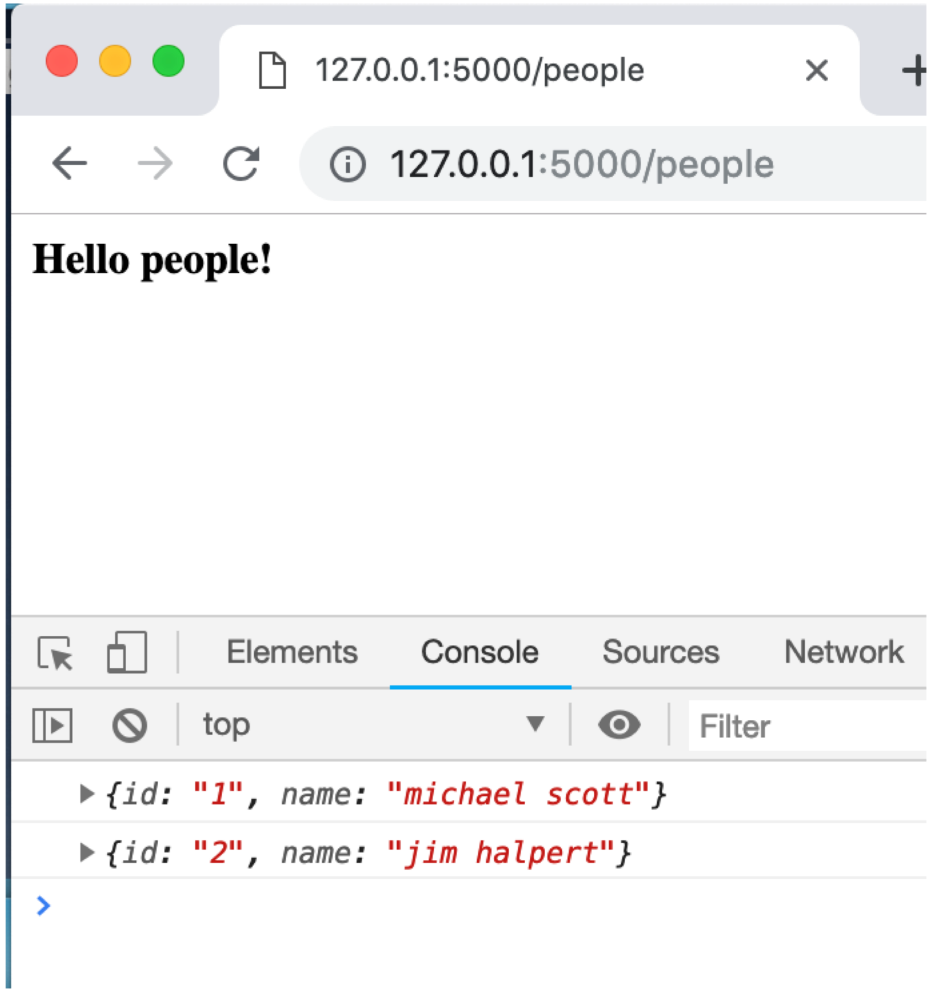
```
> data
< ▼(2) [{…}, {…}]
   ▶0: {id: "1", name: "michael scott"}
   ▶1: {id: "2", name: "jim halpert"}
    length: 2
   ▶__proto__: Array(0)
>
```

Flask only send strings to the client.
Numbers, arrays, lists, etc, must be string-ified on the server
And un-string-ified on the client

Iterate over the data



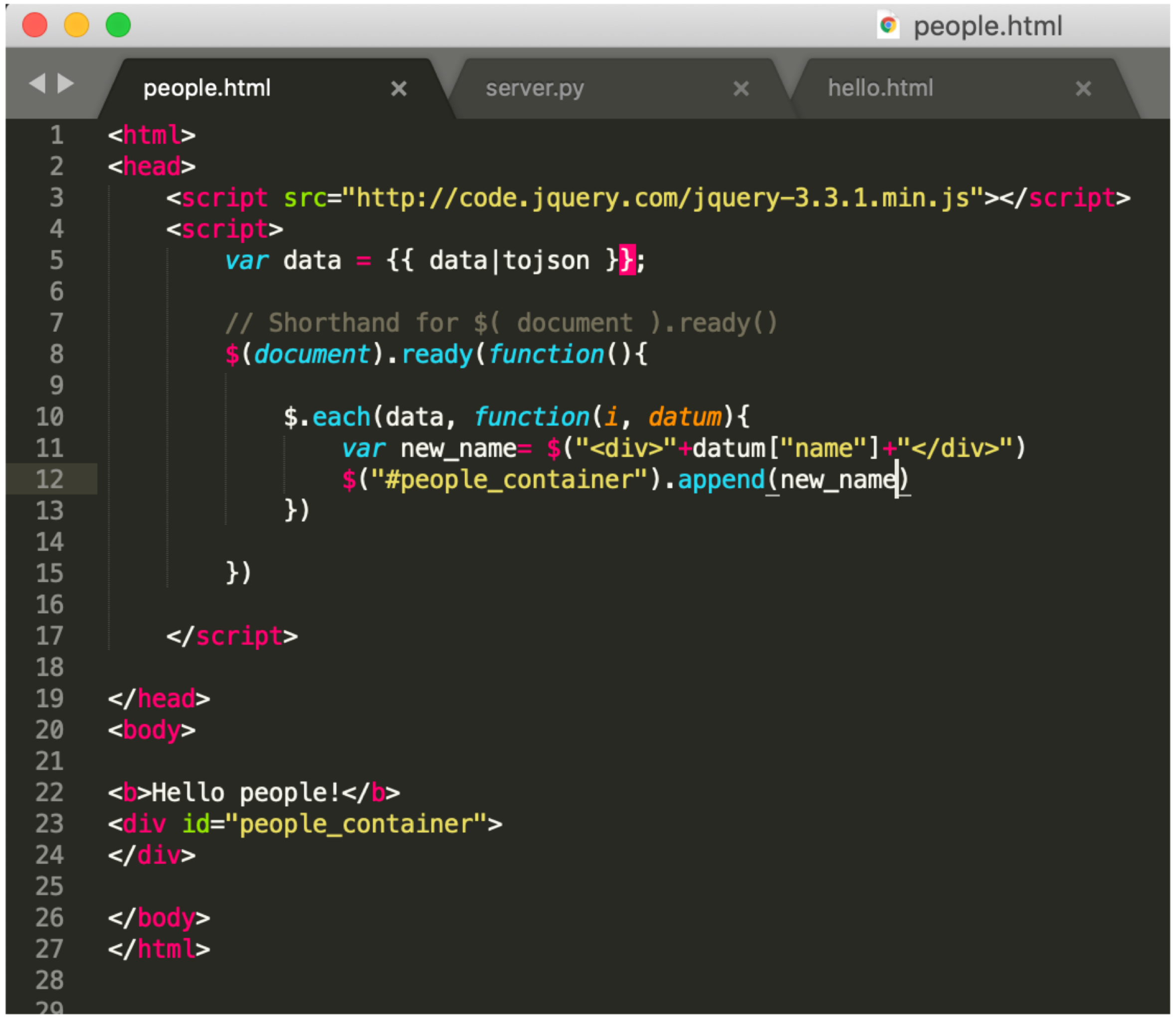
```
<html>
<head>
    <script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        var data = {{ data|tojson }};

        // Shorthand for $( document ).ready()
        $(document).ready(function(){

            $.each(data, function(i, datum){
                console.log(datum)
            })

        })

    </script>

</head>
<body>

<b>Hello people!</b>
<div id="people_container">
</div>

</body>
</html>
```



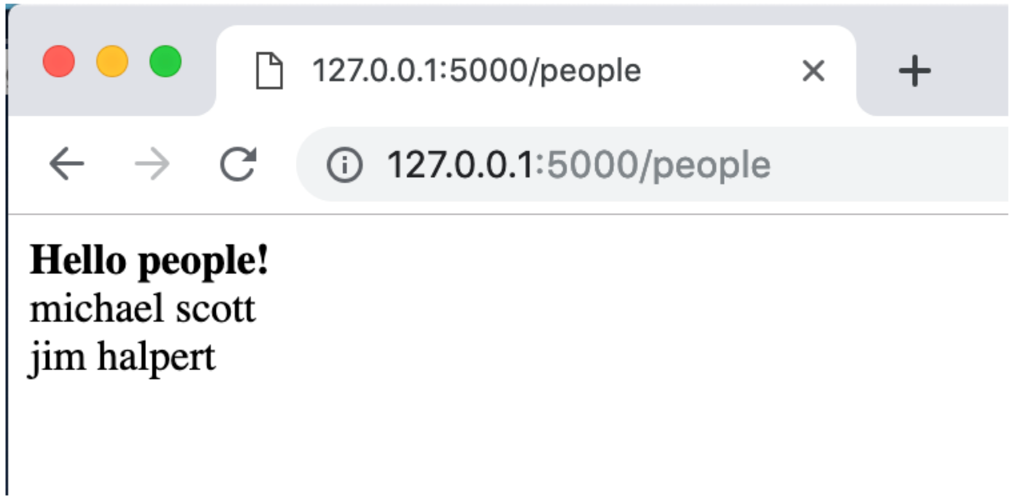
Hello people!

```
▶{id: "1", name: "michael scott"}
▶{id: "2", name: "jim halpert"}
>
```

Display all the names

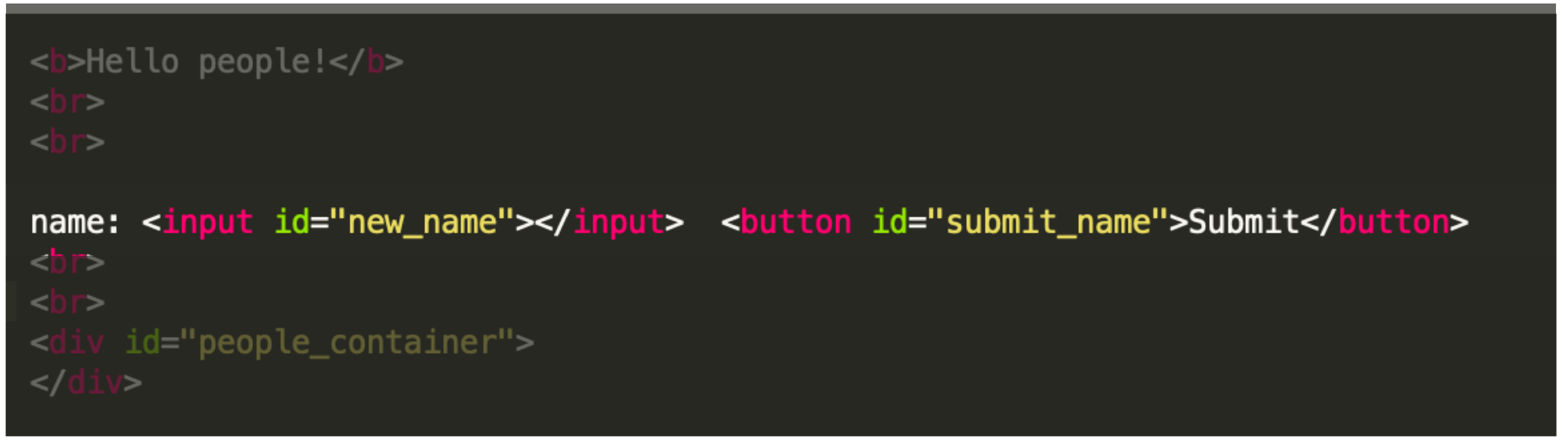


```html
<html>
<head>
    <script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        var data = {{ data|tojson }};

        // Shorthand for $( document ).ready()
        $(document).ready(function(){

            $.each(data, function(i, datum){
                var new_name= $("<div>"+datum["name"]+"</div>")
                $("#people_container").append(new_name)
            })

        })

    </script>

</head>
<body>

<b>Hello people!</b>
<div id="people_container">
</div>

</body>
</html>
```

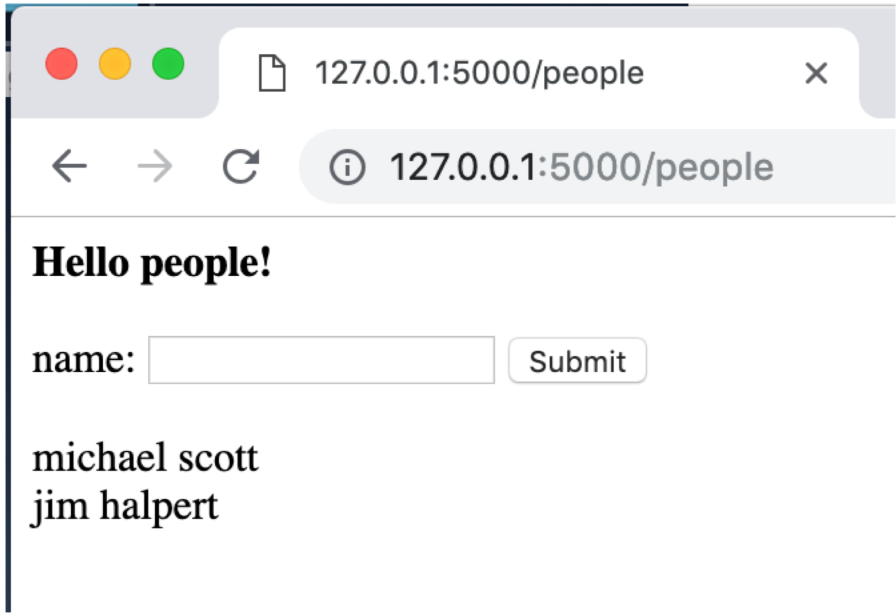


Hello people!
michael scott
jim halpert

How do users submit names? (two ways)



```
<b>Hello people!</b>
<br>
<br>

name: <input id="new_name"></input>  <button id="submit_name">Submit</button>
<br>
<br>
<div id="people_container">
</div>
```



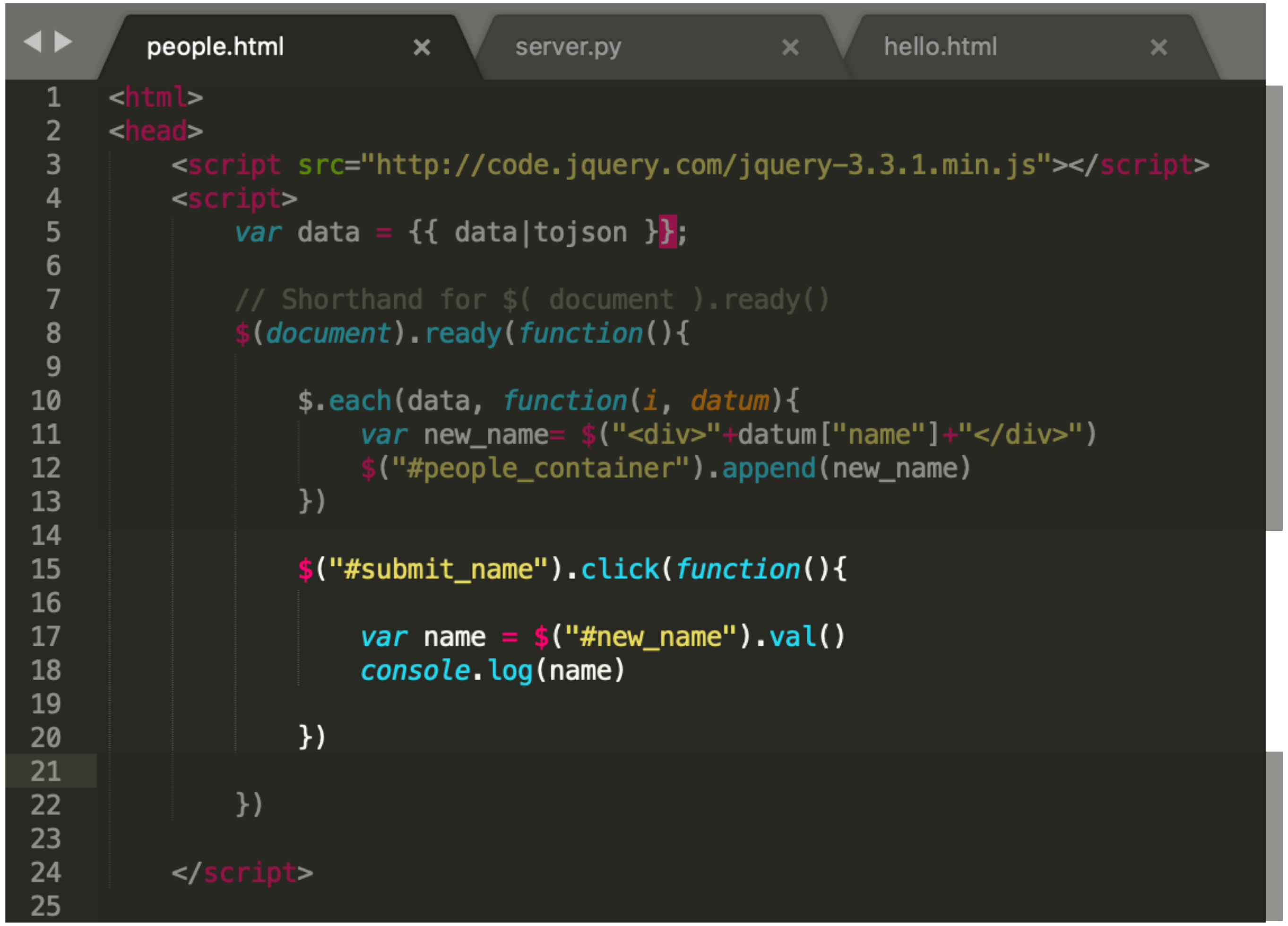
127.0.0.1:5000/people

Hello people!

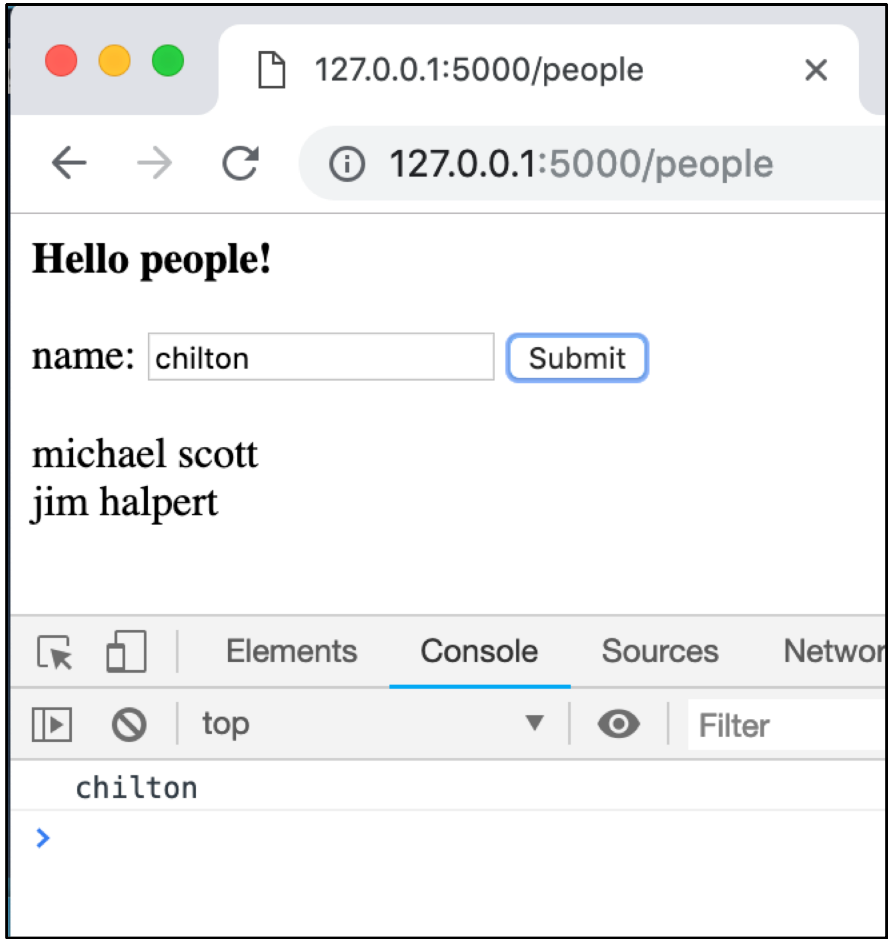
name: Submit

michael scott
jim halpert

What’s the first thing the click handler does?



```
<html>
<head>
    <script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        var data = {{ data|tojson }};

        // Shorthand for $( document ).ready()
        $(document).ready(function(){

            $.each(data, function(i, datum){
                var new_name= $("<div>"+datum["name"]+"</div>")
                $("#people_container").append(new_name)
            })

            $("#submit_name").click(function(){

                var name = $("#new_name").val()
                console.log(name)

            })

        })

    </script>

```



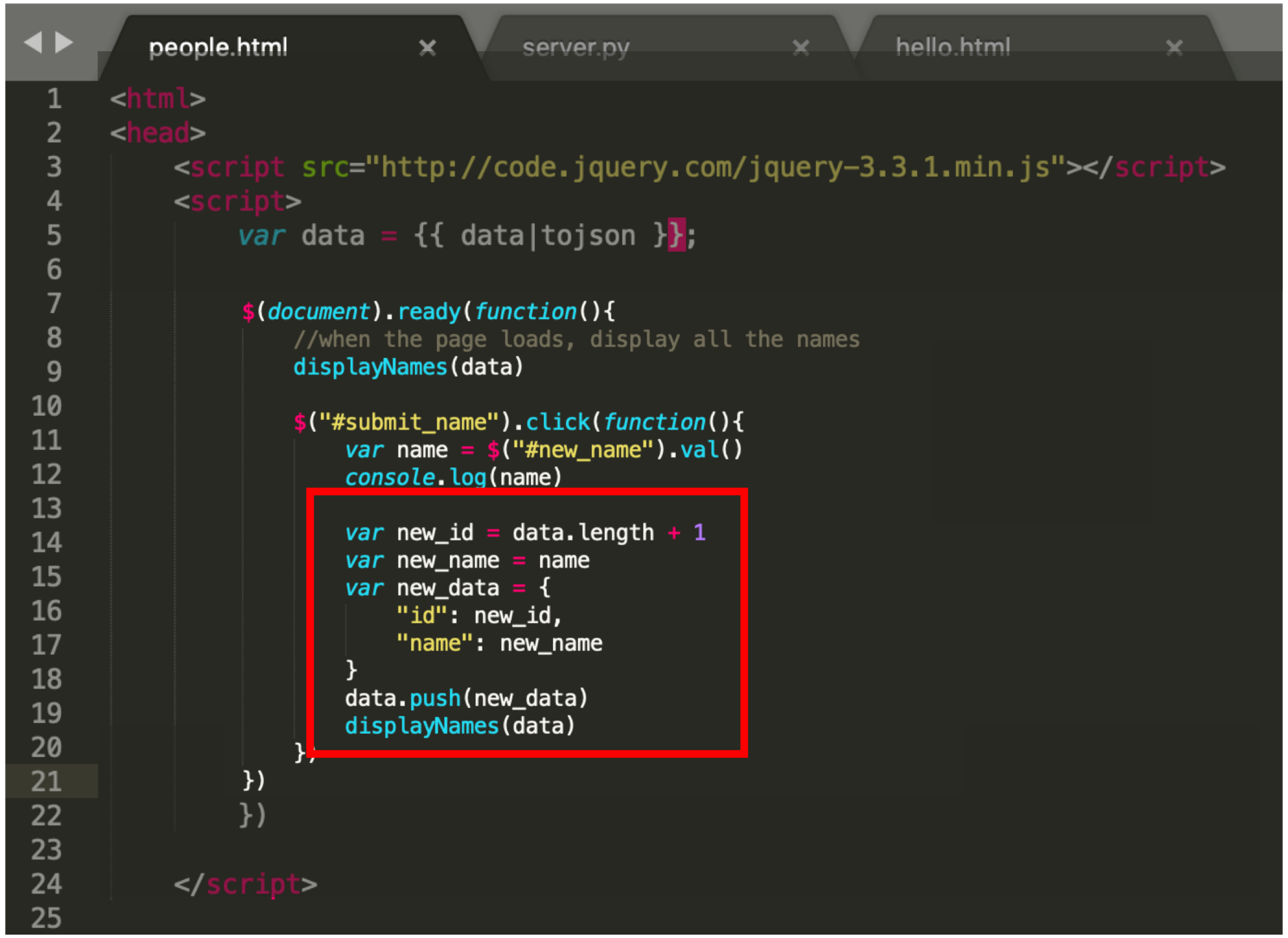
Hello people!

name: chilton Submit

michael scott
jim halpert

```
chilton
>
```

In HW4, we used MVC to update the data on the client, then regenerate the list.

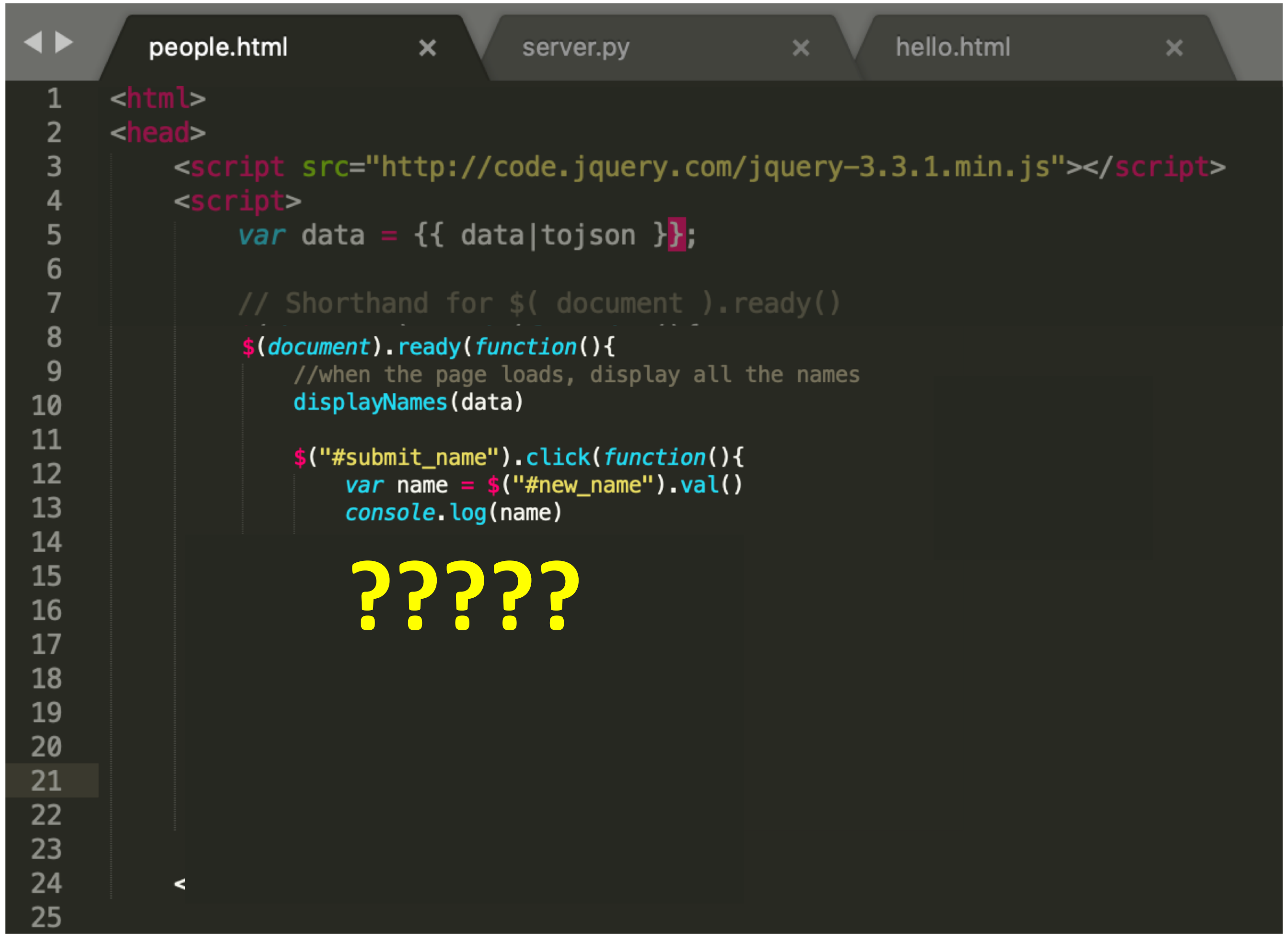


```
<html>
<head>
    <script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        var data = {{ data|tojson }};

        $(document).ready(function(){
            //when the page loads, display all the names
            displayNames(data)

            $("#submit_name").click(function(){
                var name = $("#new_name").val()
                console.log(name)

                var new_id = data.length + 1
                var new_name = name
                var new_data = {
                    "id": new_id,
                    "name": new_name
                }
                data.push(new_data)
                displayNames(data)
            })
        })
        })

    </script>
```

But this won’t save data to the server.

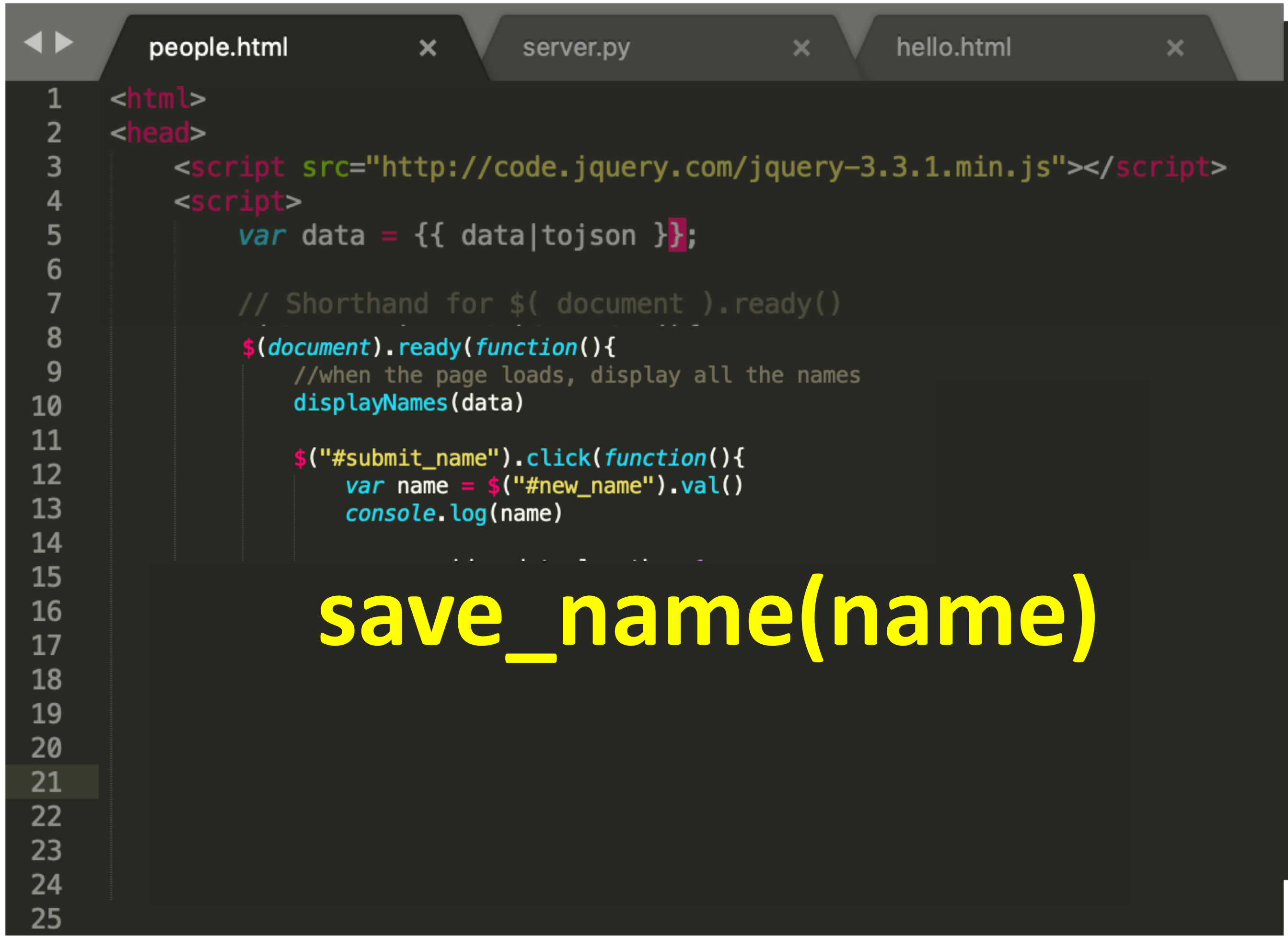
What code do we need to write instead?

Save the data to the server



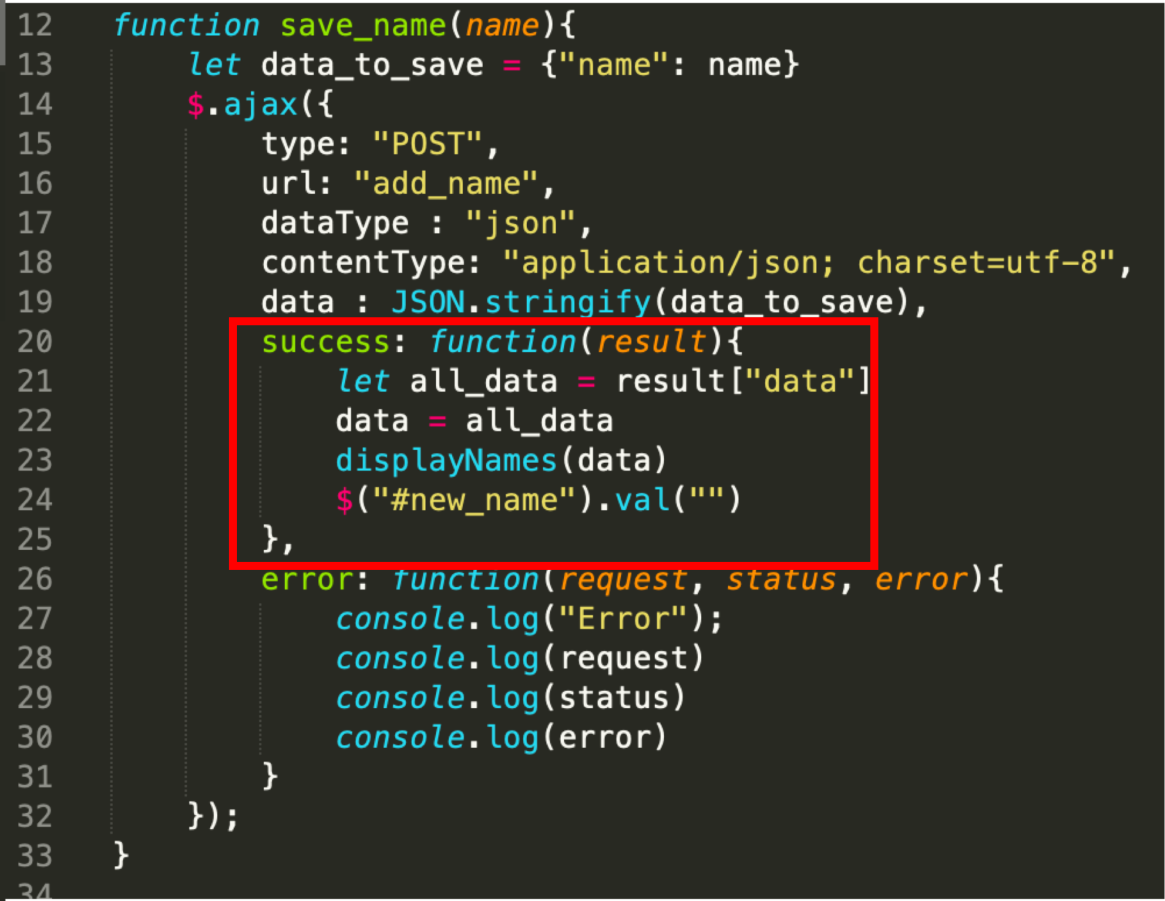
```
<html>
<head>
    <script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        var data = {{ data|tojson }};

        // Shorthand for $( document ).ready()
        $(document).ready(function(){
            //when the page loads, display all the names
            displayNames(data)

            $("#submit_name").click(function(){
                var name = $("#new_name").val()
                console.log(name)

              ?????









    <
```

Save the data to the server

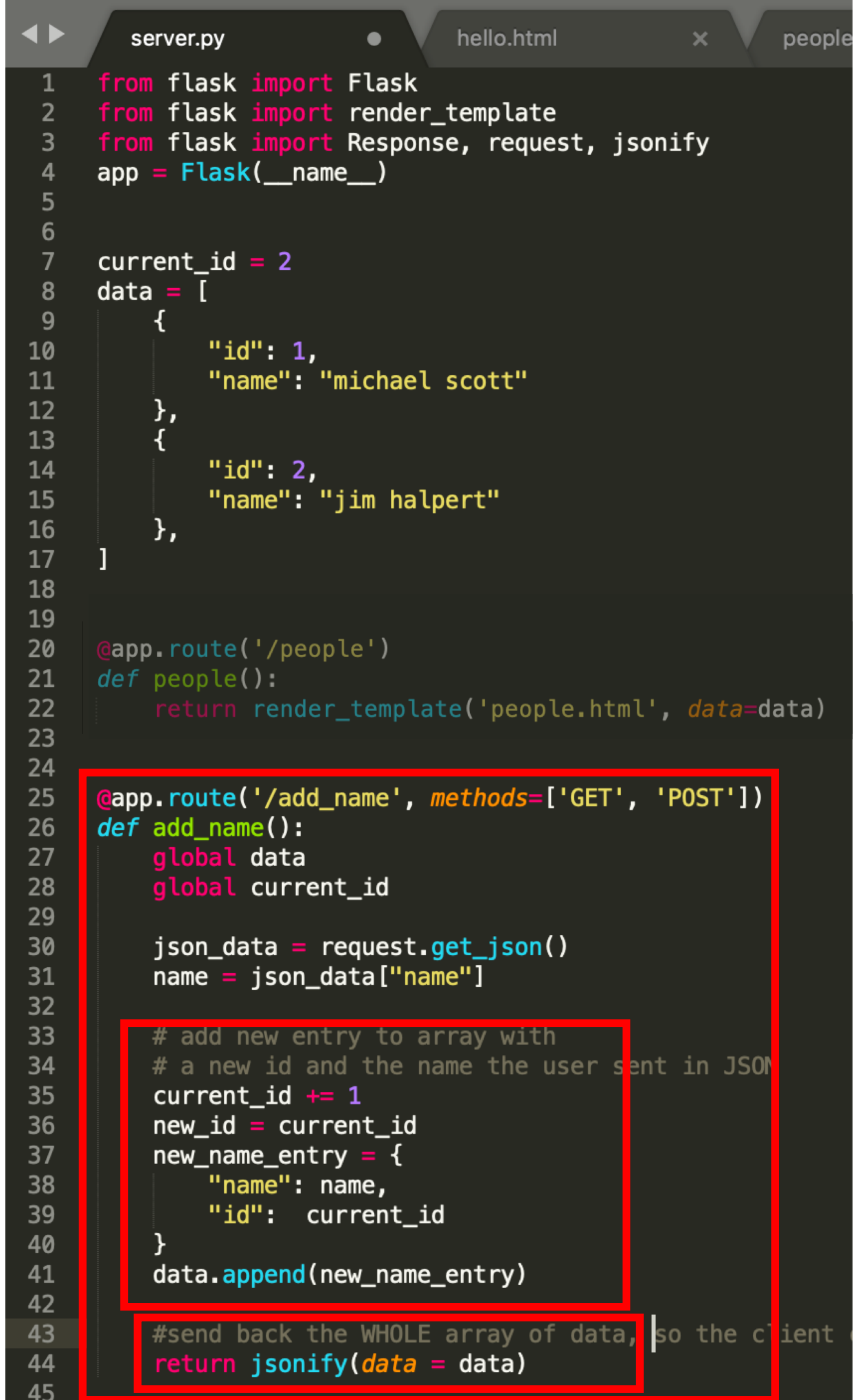


```
<html>
<head>
    <script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        var data = {{ data|tojson }};

        // Shorthand for $( document ).ready()
        $(document).ready(function(){
            //when the page loads, display all the names
            displayNames(data)

            $("#submit_name").click(function(){
                var name = $("#new_name").val()
                console.log(name)
```

save_name(name)

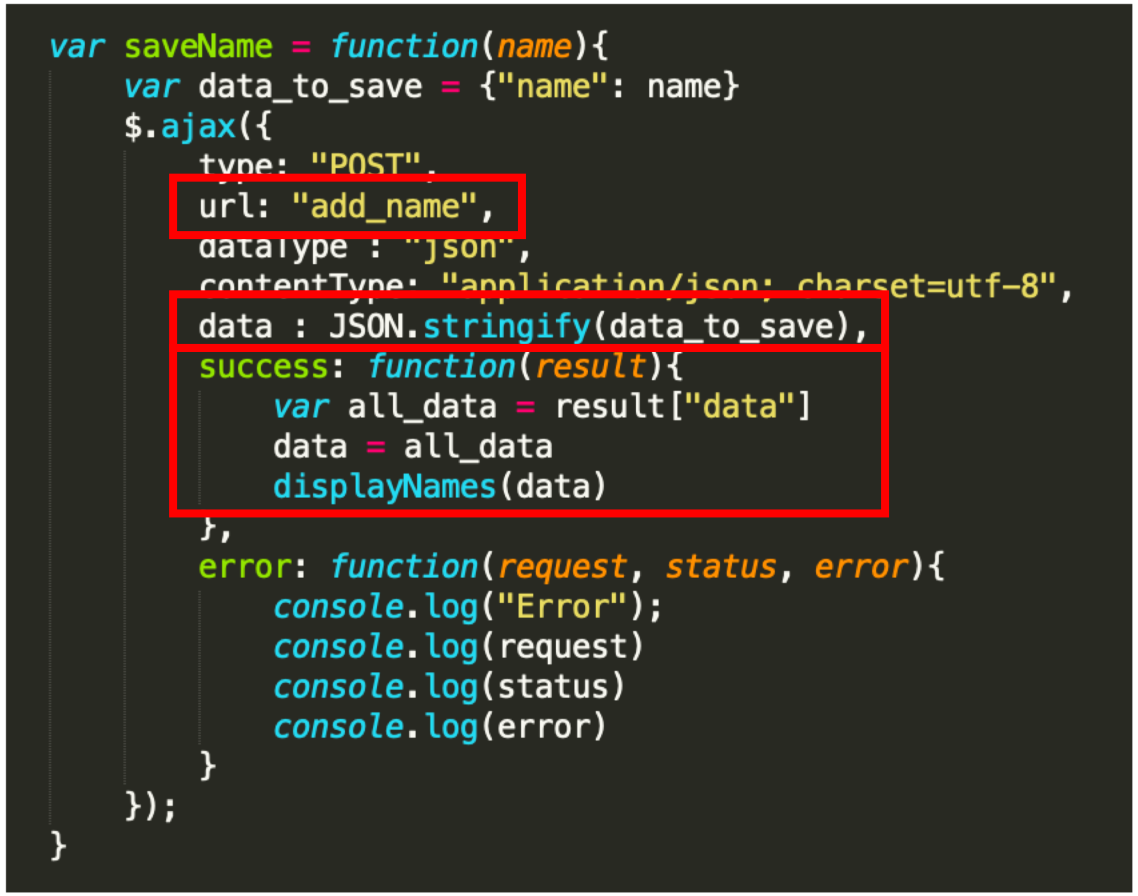


```
function save_name(name){
    let data_to_save = {"name": name}
    $.ajax({
        type: "POST",
        url: "add_name",
        dataType : "json",
        contentType: "application/json; charset=utf-8",
        data : JSON.stringify(data_to_save),
        success: function(result){
            let all_data = result["data"]
            data = all_data
            displayNames(data)
            $("#new_name").val("")
        },
        error: function(request, status, error){
            console.log("Error");
            console.log(request)
            console.log(status)
            console.log(error)
        }
    });
}
```



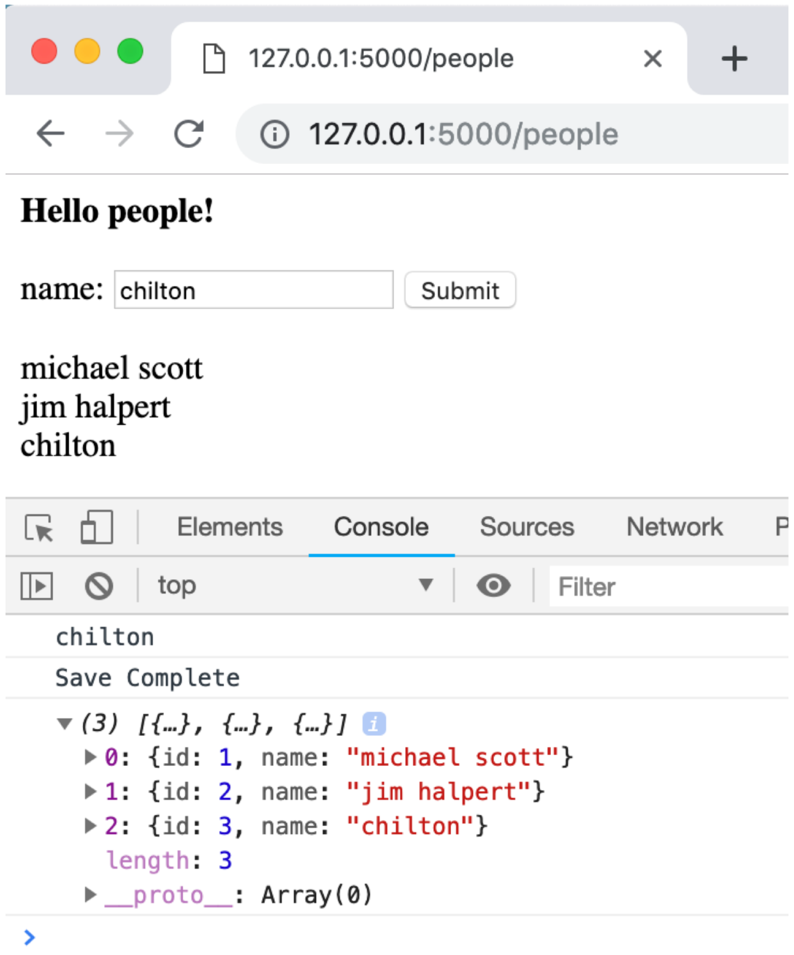
```python
from flask import Flask
from flask import render_template
from flask import Response, request, jsonify
app = Flask(__name__)


current_id = 2
data = [
    {
        "id": 1,
        "name": "michael scott"
    },
    {
        "id": 2,
        "name": "jim halpert"
    },
]


@app.route('/people')
def people():
    return render_template('people.html', data=data)


@app.route('/add_name', methods=['GET', 'POST'])
def add_name():
    global data
    global current_id

    json_data = request.get_json()
    name = json_data["name"]

    # add new entry to array with
    # a new id and the name the user sent in JSON
    current_id += 1
    new_id = current_id
    new_name_entry = {
        "name": name,
        "id":  current_id
    }
    data.append(new_name_entry)

    #send back the WHOLE array of data, so the client
    return jsonify(data = data)
```

the server?



```javascript
var saveName = function(name){
    var data_to_save = {"name": name}
    $.ajax({
        type: "POST",
        url: "add_name",
        dataType : "json",
        contentType: "application/json; charset=utf-8",
        data : JSON.stringify(data_to_save),
        success: function(result){
            var all_data = result["data"]
            data = all_data
            displayNames(data)
        },
        error: function(request, status, error){
            console.log("Error");
            console.log(request)
            console.log(status)
            console.log(error)
        }
    });
}
```

How do we test if the data saves to the server?



Hello people!

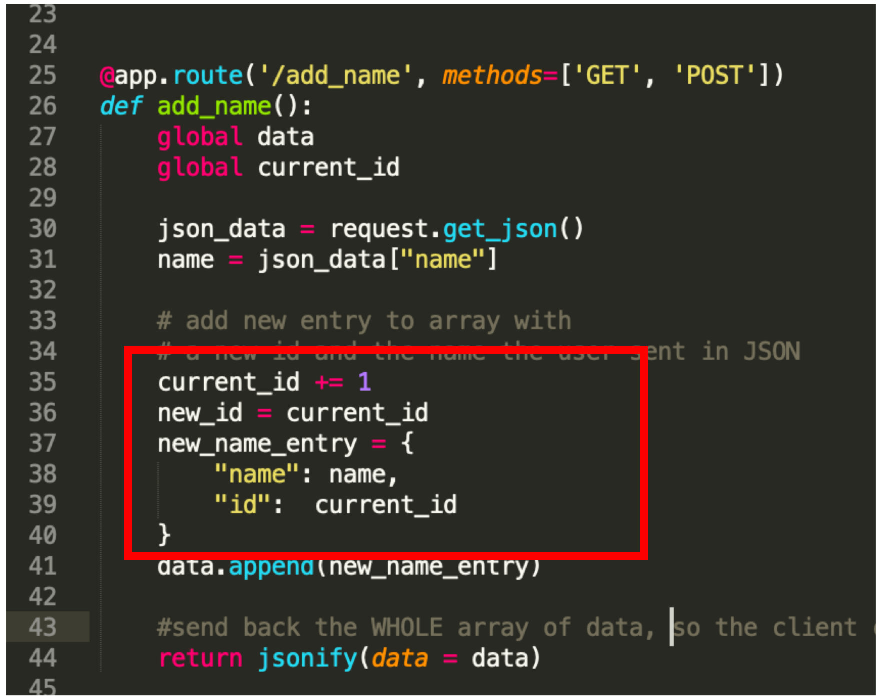
name: chilton Submit

michael scott
jim halpert
chilton

```
chilton
Save Complete
▼(3) [{…}, {…}, {…}]
  ▶0: {id: 1, name: "michael scott"}
  ▶1: {id: 2, name: "jim halpert"}
  ▶2: {id: 3, name: "chilton"}
   length: 3
  ▶__proto__: Array(0)
>
```

Refresh the page to see if the new data stays

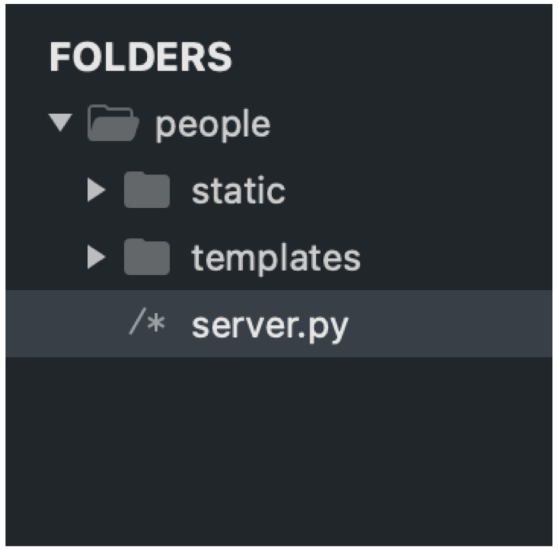
We MUST calculate the id on the server, not the client. Why?



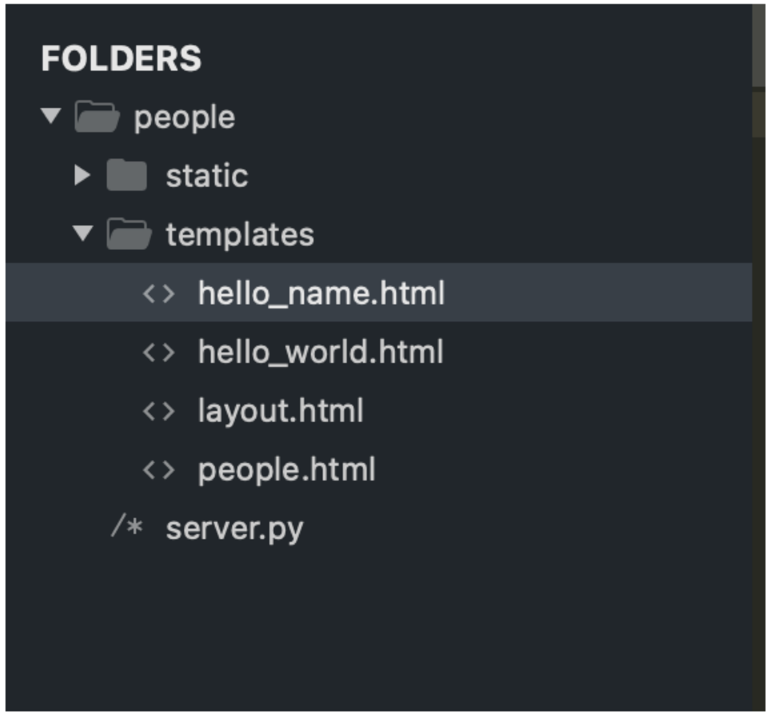
```python
@app.route('/add_name', methods=['GET', 'POST'])
def add_name():
    global data
    global current_id

    json_data = request.get_json()
    name = json_data["name"]

    # add new entry to array with
    # a new id and the name the user sent in JSON
    current_id += 1
    new_id = current_id
    new_name_entry = {
        "name": name,
        "id":  current_id
    }
    data.append(new_name_entry)

    #send back the WHOLE array of data, so the client
    return jsonify(data = data)
```

Multiple people will be able to add name, and we don’t want them to use the same ids.

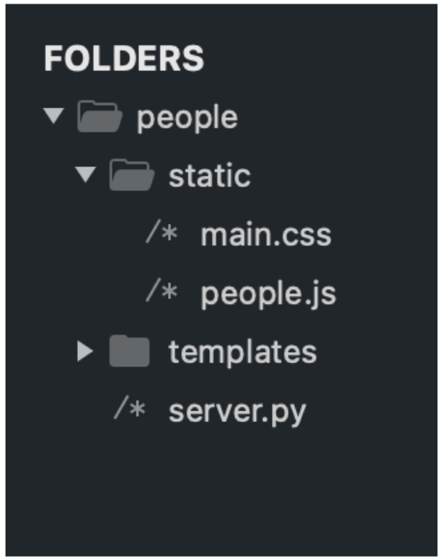
Flask projects have a very specific structure



```
FOLDERS
▼ people
  ▶ static
  ▶ templates
    server.py
```



```
FOLDERS
▼ people
  ▶ static
  ▼ templates
      hello_name.html
      hello_world.html
      layout.html
      people.html
    server.py
```



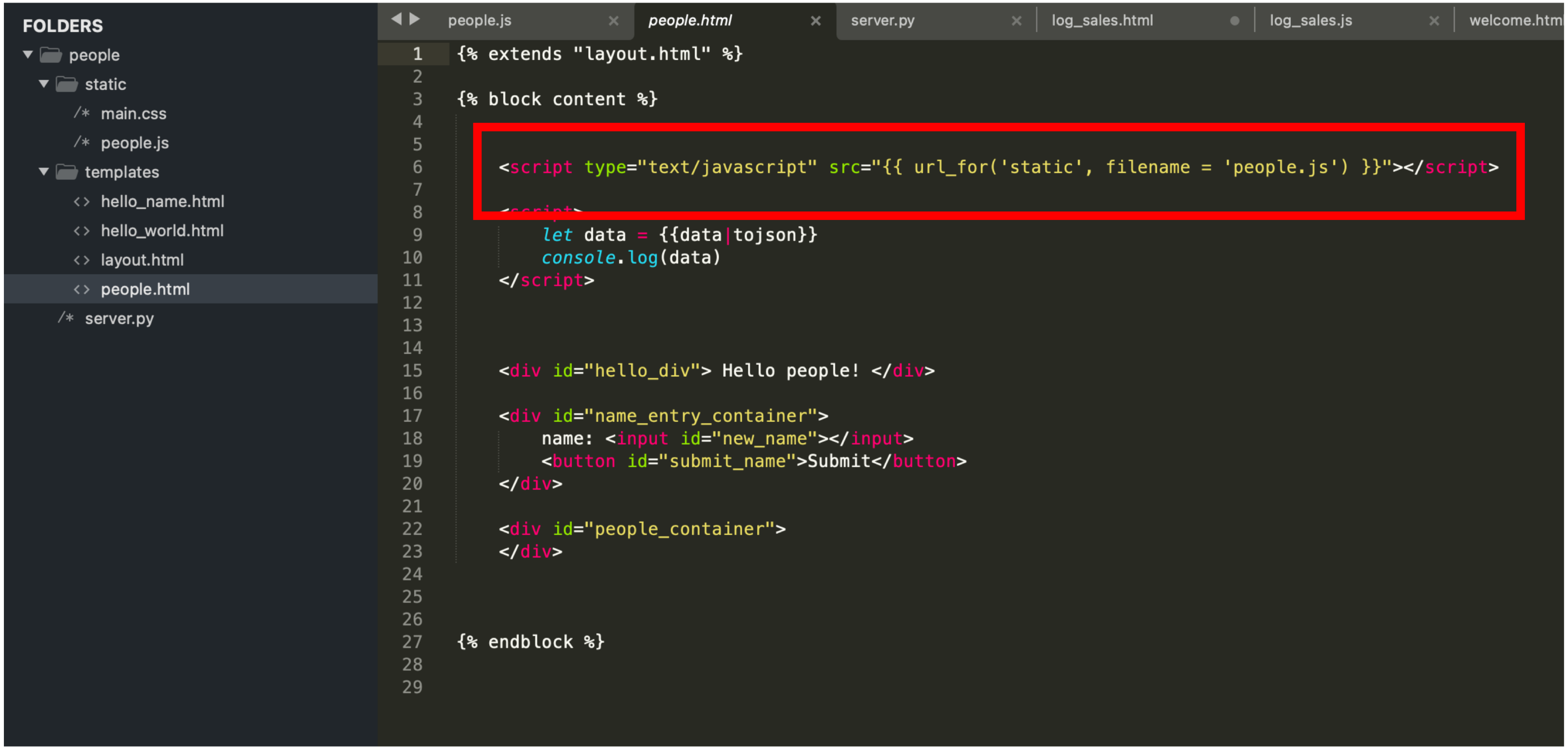
```
FOLDERS
▼ people
  ▼ static
      main.css
      people.js
  ▶ templates
    server.py
```

Server.py
goes directly inside the project folder

/templates (lower case)
Has HTML files

/static (lower case)
.js and .css files
(and image files)

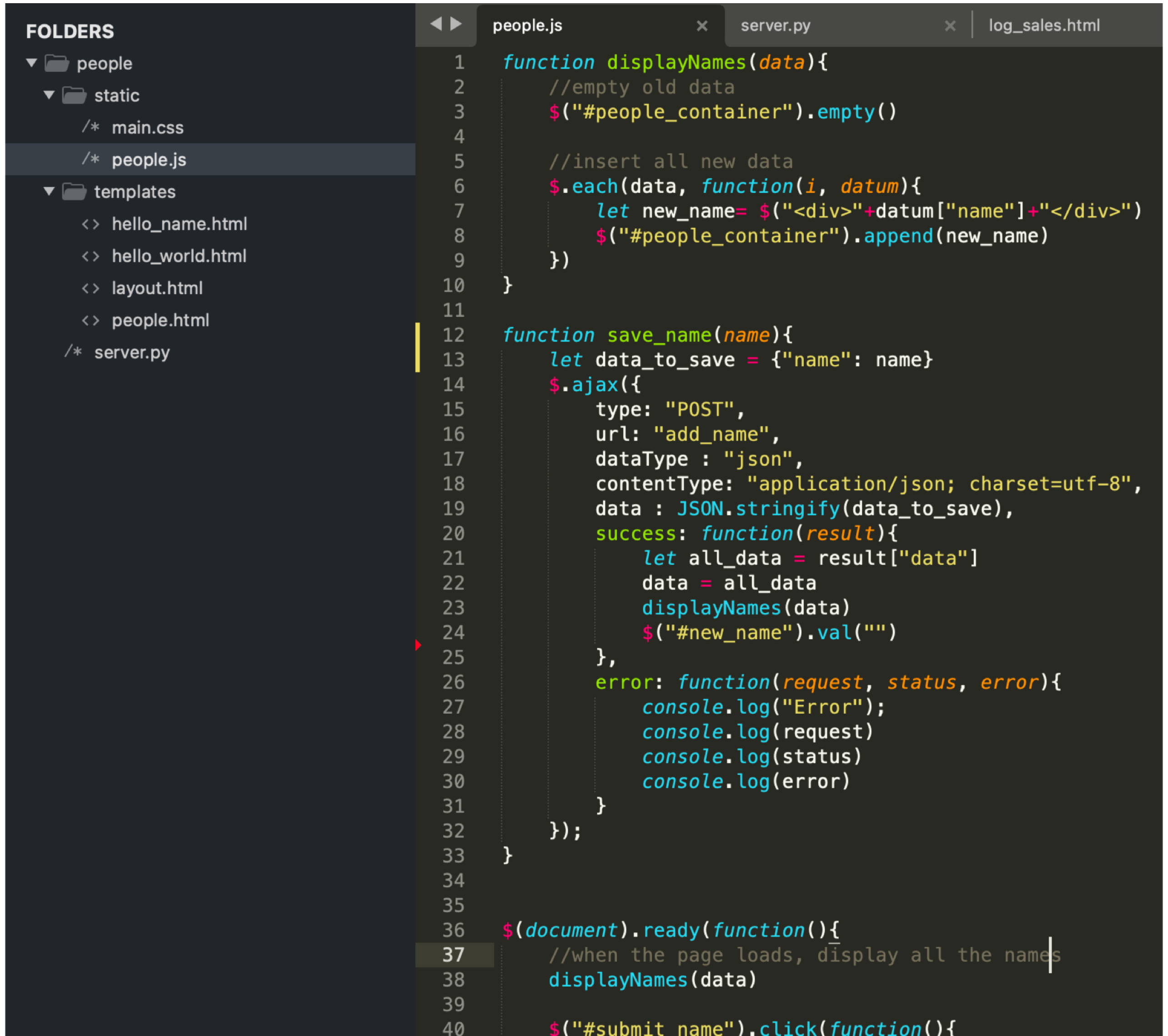
People.html is in templates.
But where's people.js?



```
FOLDERS
▼ people
  ▼ static
      main.css
      people.js
  ▼ templates
      hello_name.html
      hello_world.html
      layout.html
      people.html
    server.py
```

```
{% extends "layout.html" %}

{% block content %}


    <script type="text/javascript" src="{{ url_for('static', filename = 'people.js') }}"></script>

    <script>
        let data = {{data|tojson}}
        console.log(data)
    </script>



    <div id="hello_div"> Hello people! </div>

    <div id="name_entry_container">
        name: <input id="new_name"></input>
        <button id="submit_name">Submit</button>
    </div>

    <div id="people_container">
    </div>



{% endblock %}
```

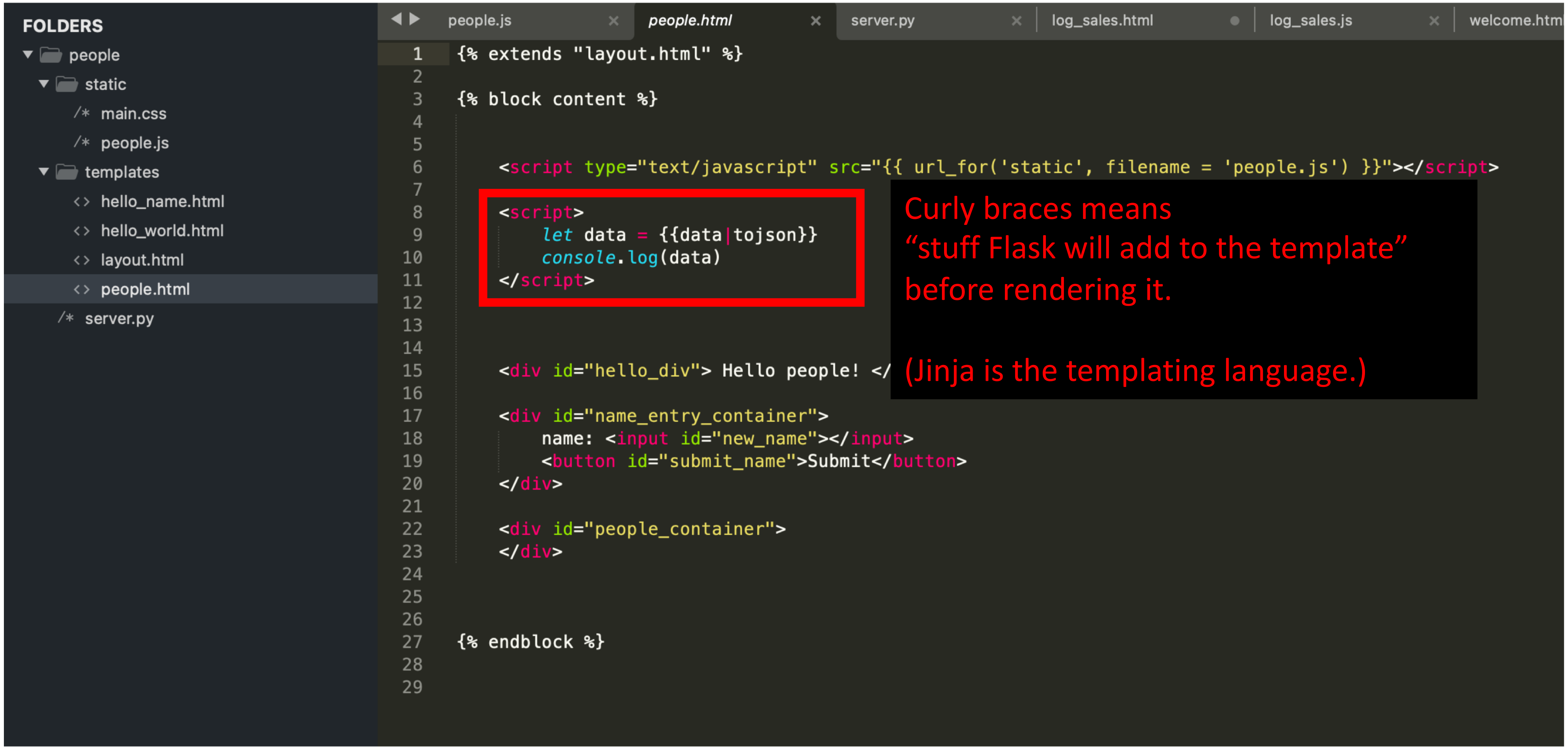
People.js is in the static folder.



```javascript
function displayNames(data){
    //empty old data
    $("#people_container").empty()

    //insert all new data
    $.each(data, function(i, datum){
        let new_name= $("<div>"+datum["name"]+"</div>")
        $("#people_container").append(new_name)
    })
}

function save_name(name){
    let data_to_save = {"name": name}
    $.ajax({
        type: "POST",
        url: "add_name",
        dataType : "json",
        contentType: "application/json; charset=utf-8",
        data : JSON.stringify(data_to_save),
        success: function(result){
            let all_data = result["data"]
            data = all_data
            displayNames(data)
            $("#new_name").val("")
        },
        error: function(request, status, error){
            console.log("Error");
            console.log(request)
            console.log(status)
            console.log(error)
        }
    });
}


$(document).ready(function(){
    //when the page loads, display all the names
    displayNames(data)

    $("#submit_name").click(function(){
```

We already forced you to separate your JS from your HTML, so this isn’t a big deal.

There is a tiny amount of JS in people.html



```
{% extends "layout.html" %}

{% block content %}


    <script type="text/javascript" src="{{ url_for('static', filename = 'people.js') }}"></script>

    <script>
        let data = {{data|tojson}}
        console.log(data)
    </script>



    <div id="hello_div"> Hello people! </

    <div id="name_entry_container">
        name: <input id="new_name"></input>
        <button id="submit_name">Submit</button>
    </div>

    <div id="people_container">
    </div>



{% endblock %}
```

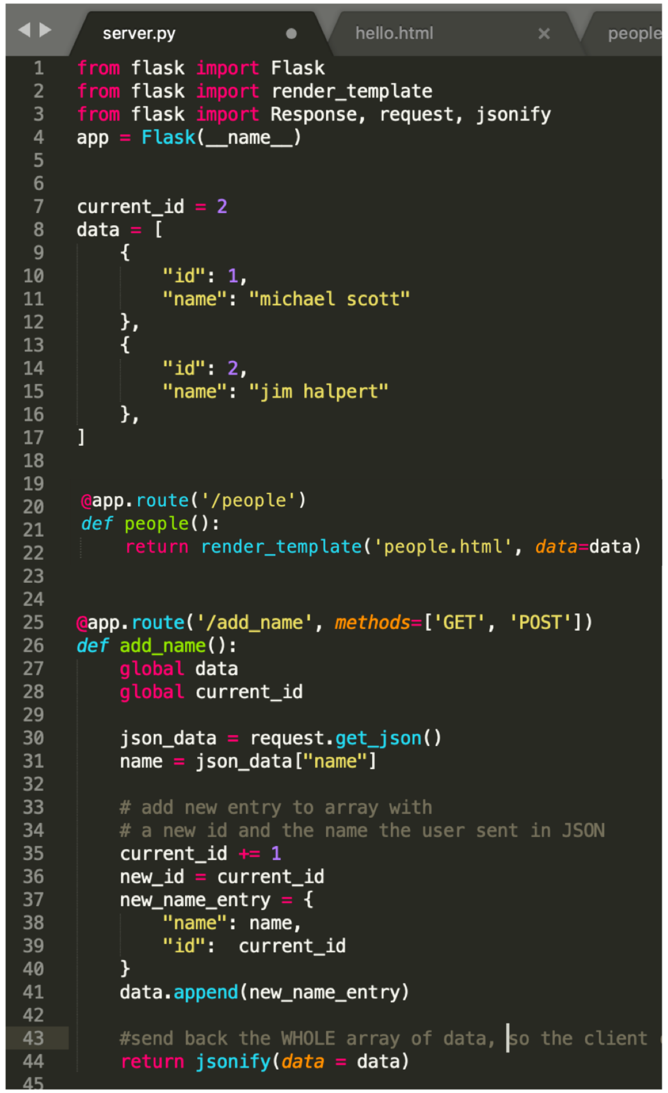
Curly braces means
“stuff Flask will add to the template”
before rendering it.

(Jinja is the templating language.)

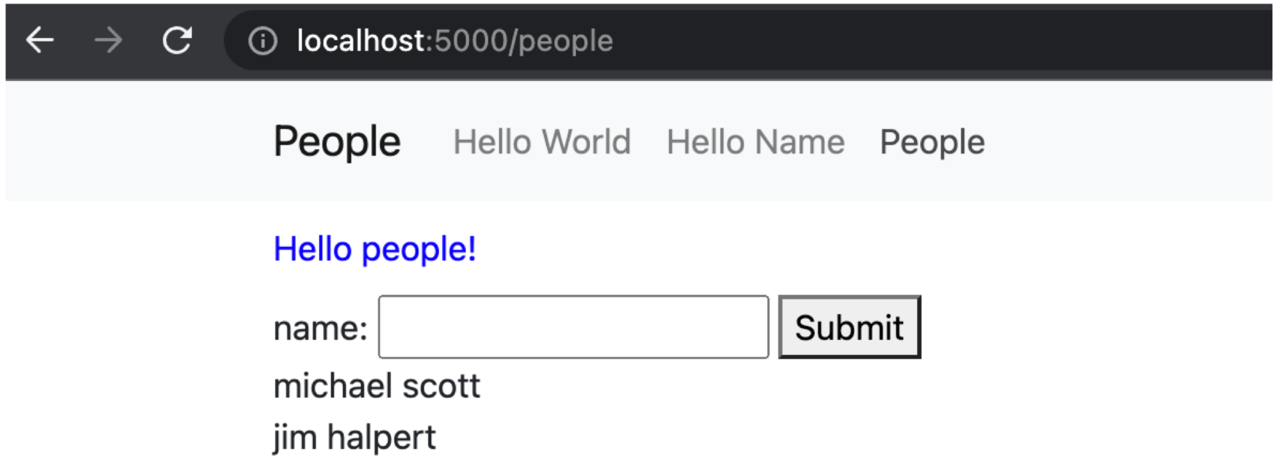
Homework 5

Putting a database behind HW4

Warm up: Get the Flask sample code to run



```python
from flask import Flask
from flask import render_template
from flask import Response, request, jsonify
app = Flask(__name__)


current_id = 2
data = [
    {
        "id": 1,
        "name": "michael scott"
    },
    {
        "id": 2,
        "name": "jim halpert"
    },
]


@app.route('/people')
def people():
    return render_template('people.html', data=data)


@app.route('/add_name', methods=['GET', 'POST'])
def add_name():
    global data
    global current_id

    json_data = request.get_json()
    name = json_data["name"]

    # add new entry to array with
    # a new id and the name the user sent in JSON
    current_id += 1
    new_id = current_id
    new_name_entry = {
        "name": name,
        "id":  current_id
    }
    data.append(new_name_entry)

    #send back the WHOLE array of data, so the client
    return jsonify(data = data)
```

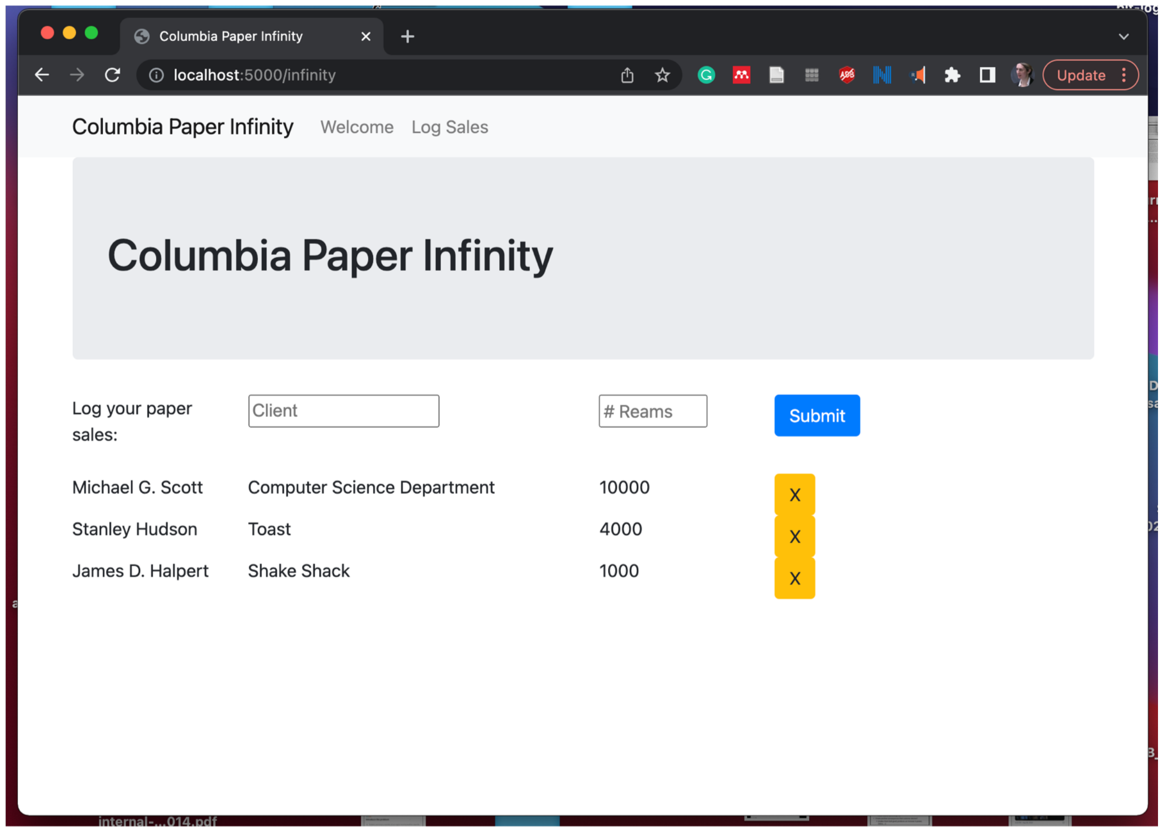


Hello people!

michael scott

jim halpert

Main. Put a backend behind Log Sales and save the data.



Columbia Paper Infinity

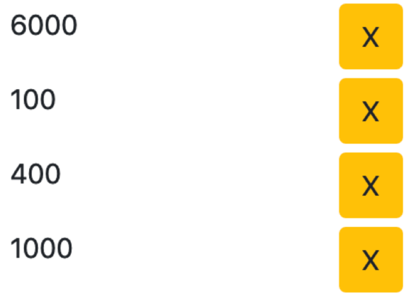
Log your paper sales:

Log your paper sales:	Client	# Reams	Submit
Michael G. Scott	Computer Science Department	10000	X
Stanley Hudson	Toast	4000	X
James D. Halpert	Shake Shack	1000	X

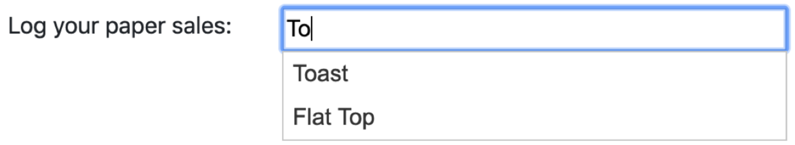
Tip: start by copying the people folder and editing it

In HW4, you dynamically created widgets

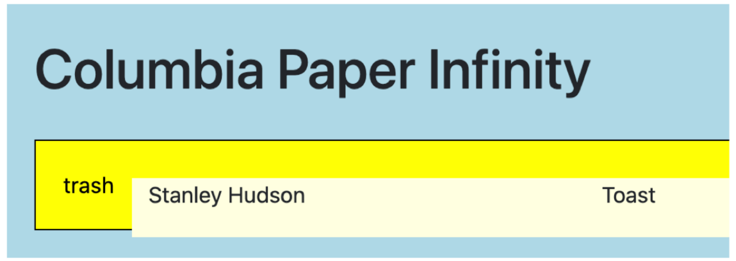
Buttons



Autocomplete

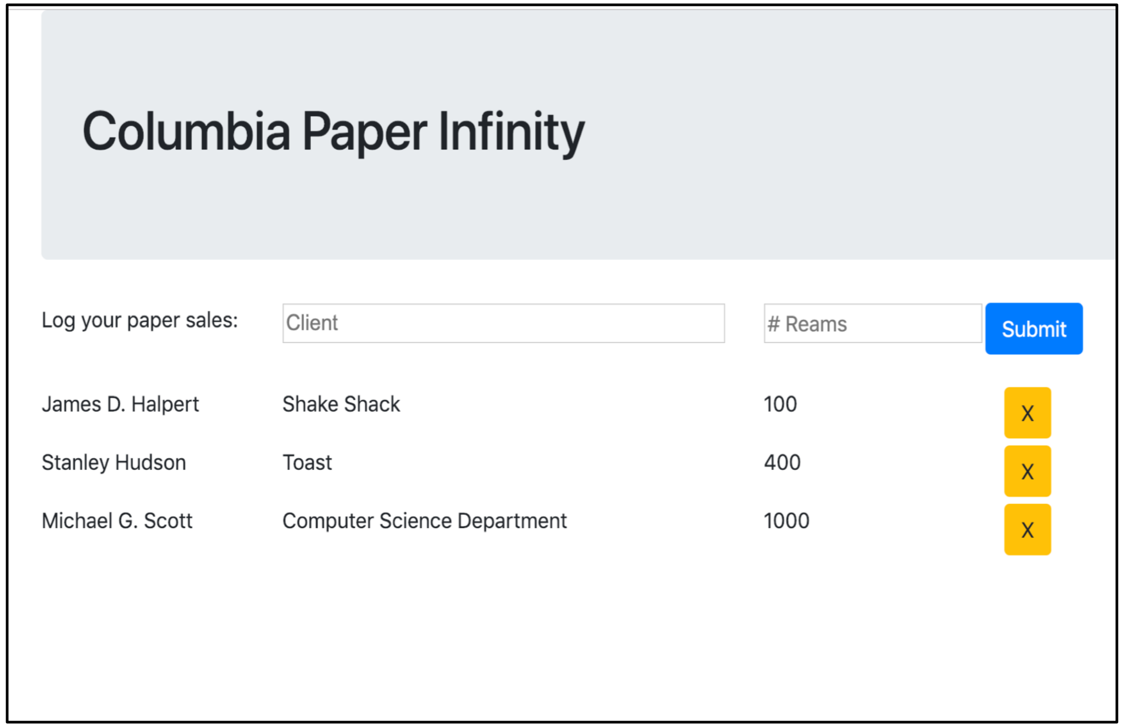


Drag and Drop



Added customization
(hovering and drop target feedback)

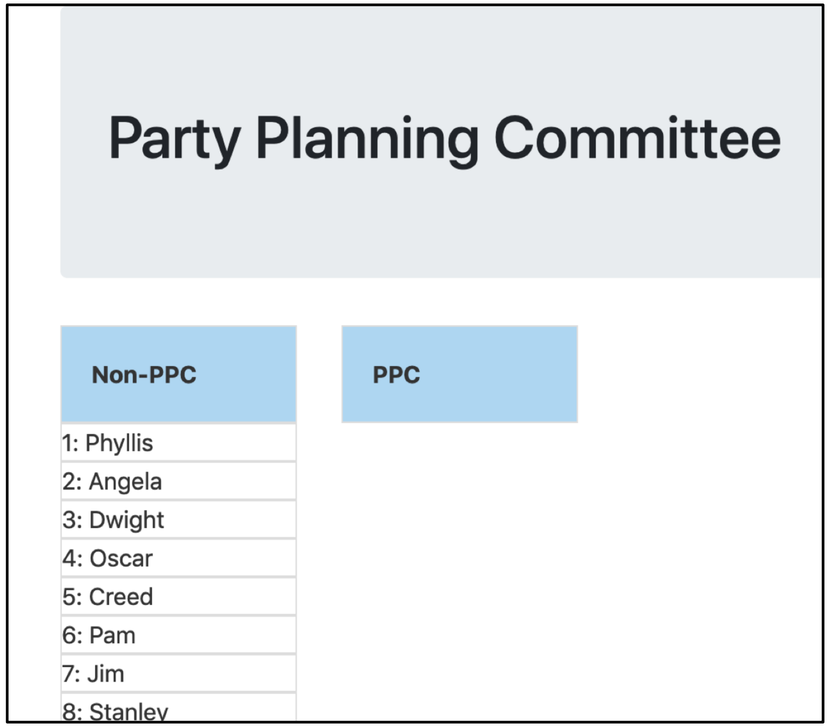
You allowed users to interact with data



Columbia Paper Infinity

Log your paper sales: Client # Reams Submit

James D. Halpert	Shake Shack	100	X
Stanley Hudson	Toast	400	X
Michael G. Scott	Computer Science Department	1000	X



Party Planning Committee

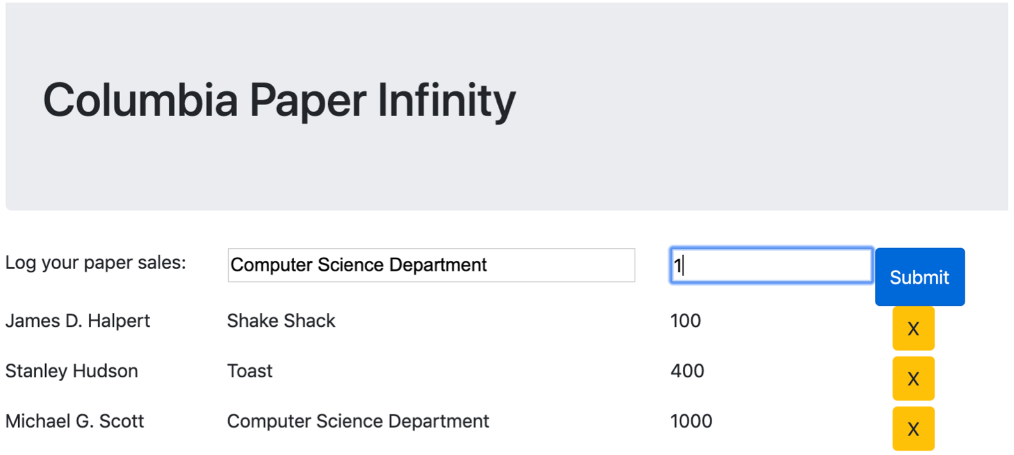
Non-PPC	PPC
1: Phyllis	
2: Angela	
3: Dwight	
4: Oscar	
5: Creed	
6: Pam	
7: Jim	
8: Stanley	

Create / Delete data

Update data

But there's a big problem:

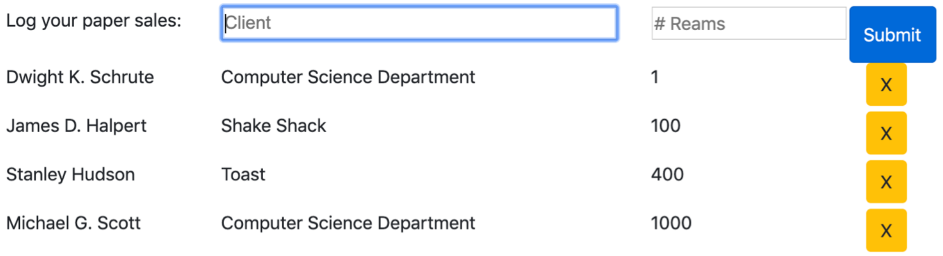
Add data



Columbia Paper Infinity

Log your paper sales:	Computer Science Department	1	Submit
James D. Halpert	Shake Shack	100	X
Stanley Hudson	Toast	400	X
Michael G. Scott	Computer Science Department	1000	X

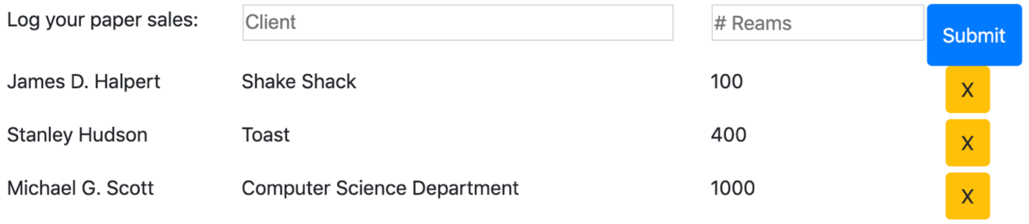
Data appears



Log your paper sales:	Client	# Reams	Submit
Dwight K. Schrute	Computer Science Department	1	X
James D. Halpert	Shake Shack	100	X
Stanley Hudson	Toast	400	X
Michael G. Scott	Computer Science Department	1000	X

REFRESH PAGE

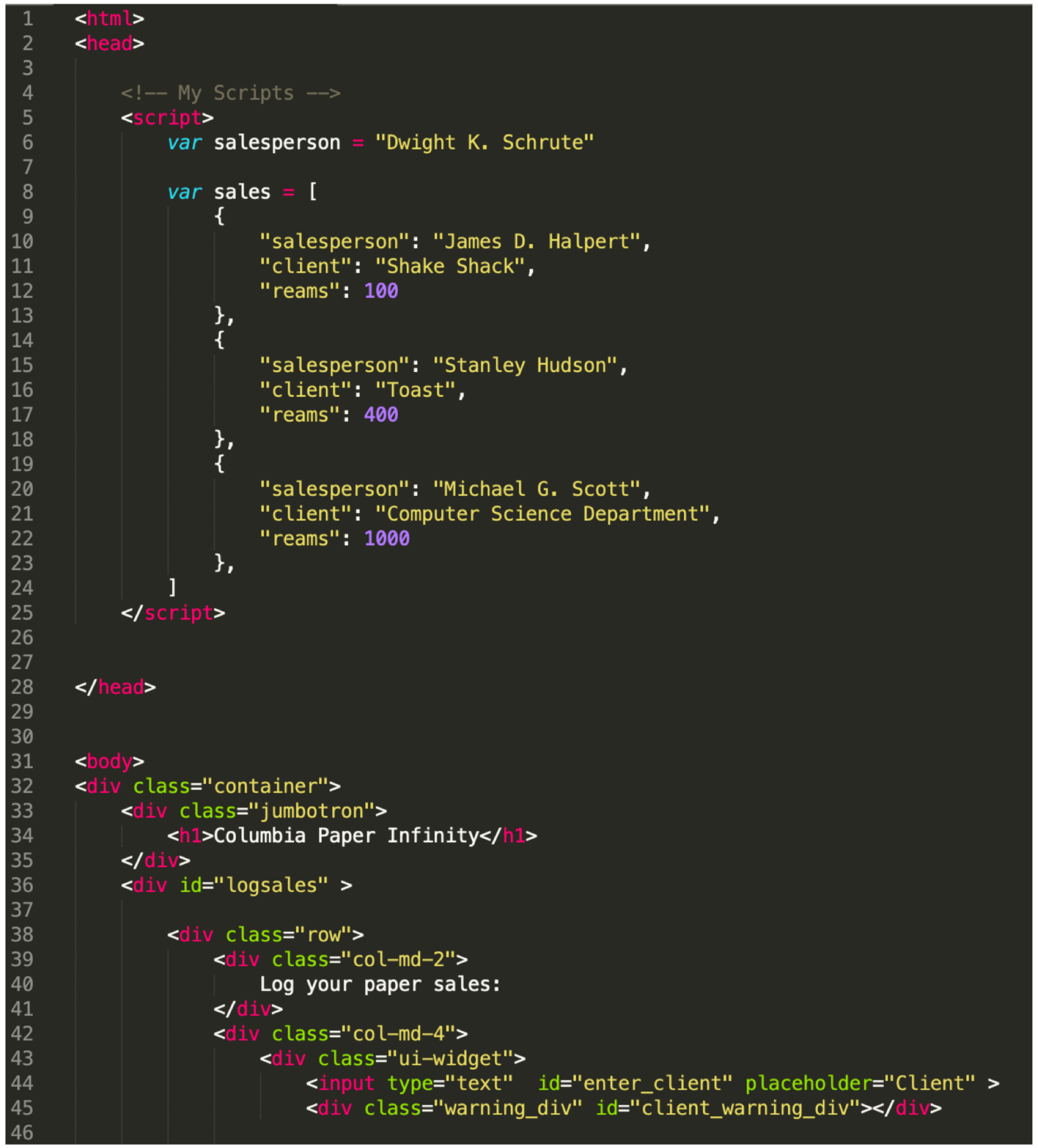
Data is gone!



Log your paper sales:	Client	# Reams	Submit
James D. Halpert	Shake Shack	100	X
Stanley Hudson	Toast	400	X
Michael G. Scott	Computer Science Department	1000	X

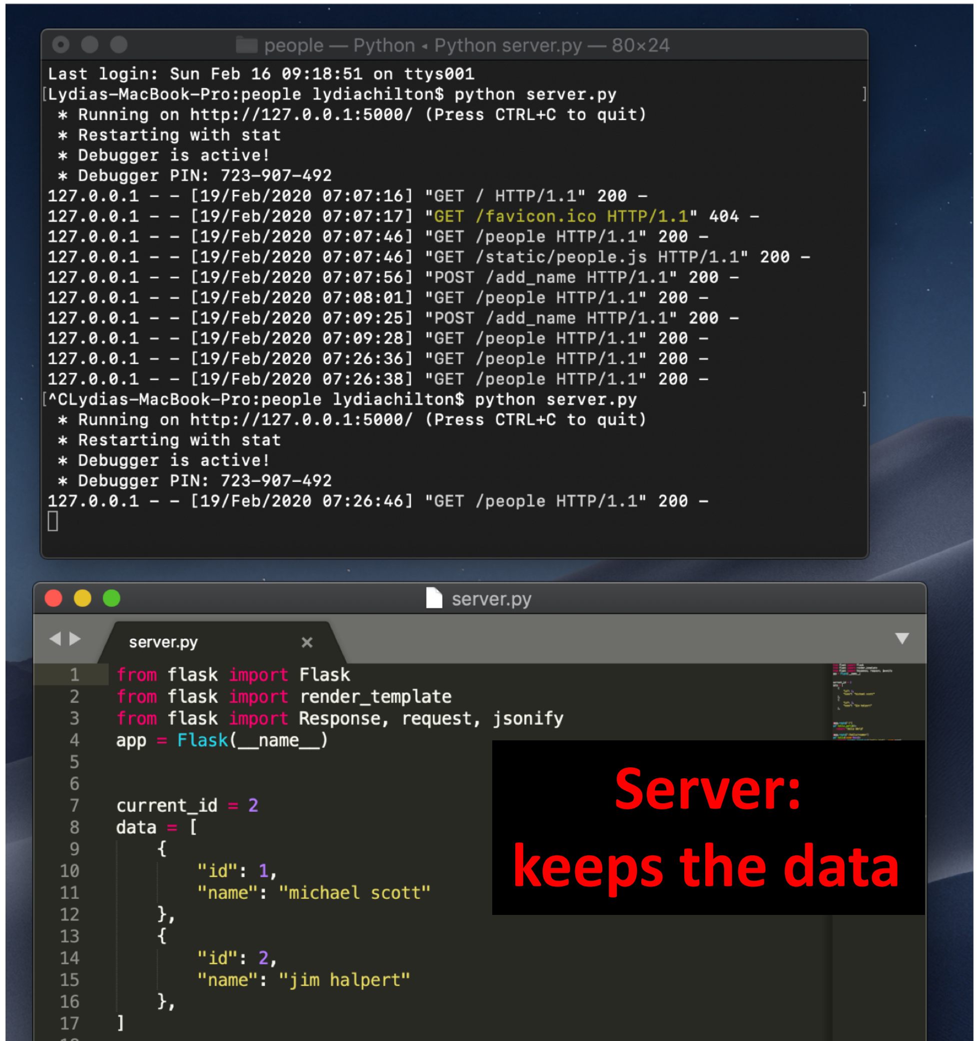
The data doesn't save

In HW4, the data is only stored in the browser



```html
<html>
<head>

    <!-- My Scripts -->
    <script>
        var salesperson = "Dwight K. Schrute"

        var sales = [
            {
                "salesperson": "James D. Halpert",
                "client": "Shake Shack",
                "reams": 100
            },
            {
                "salesperson": "Stanley Hudson",
                "client": "Toast",
                "reams": 400
            },
            {
                "salesperson": "Michael G. Scott",
                "client": "Computer Science Department",
                "reams": 1000
            },
        ]
    </script>


</head>


<body>
<div class="container">
    <div class="jumbotron">
        <h1>Columbia Paper Infinity</h1>
    </div>
    <div id="logsales" >

        <div class="row">
            <div class="col-md-2">
                Log your paper sales:
            </div>
            <div class="col-md-4">
                <div class="ui-widget">
                    <input type="text"  id="enter_client" placeholder="Client" >
                    <div class="warning_div" id="client_warning_div"></div>

```

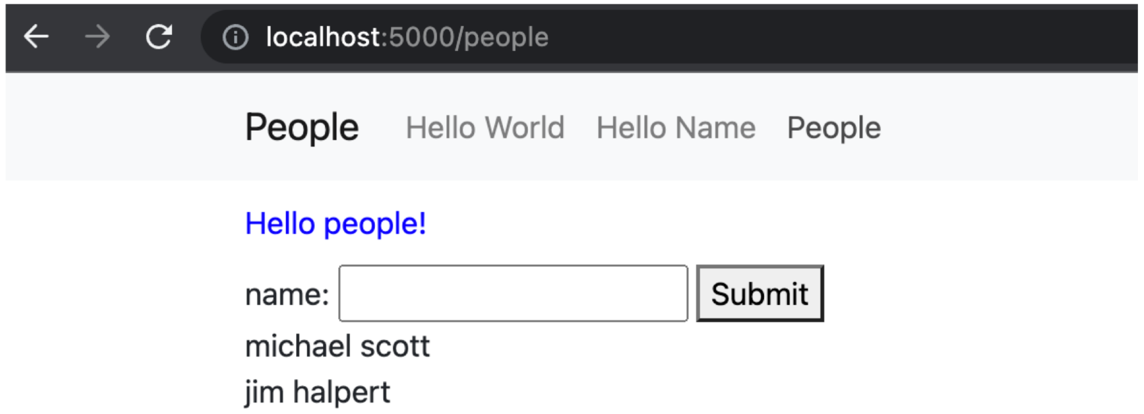
Solution: Store data on the server, display and edit data on the client.



```
Last login: Sun Feb 16 09:18:51 on ttys001
Lydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
127.0.0.1 - - [19/Feb/2020 07:07:16] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:17] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [19/Feb/2020 07:07:46] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:46] "GET /static/people.js HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:07:56] "POST /add_name HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:08:01] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:09:25] "POST /add_name HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:09:28] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:26:36] "GET /people HTTP/1.1" 200 -
127.0.0.1 - - [19/Feb/2020 07:26:38] "GET /people HTTP/1.1" 200 -
^CLydias-MacBook-Pro:people lydiachilton$ python server.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 723-907-492
127.0.0.1 - - [19/Feb/2020 07:26:46] "GET /people HTTP/1.1" 200 -
```

```python
from flask import Flask
from flask import render_template
from flask import Response, request, jsonify
app = Flask(__name__)


current_id = 2
data = [
    {
        "id": 1,
        "name": "michael scott"
    },
    {
        "id": 2,
        "name": "jim halpert"
    },
]
```

Server:
keeps the data



Hello people!

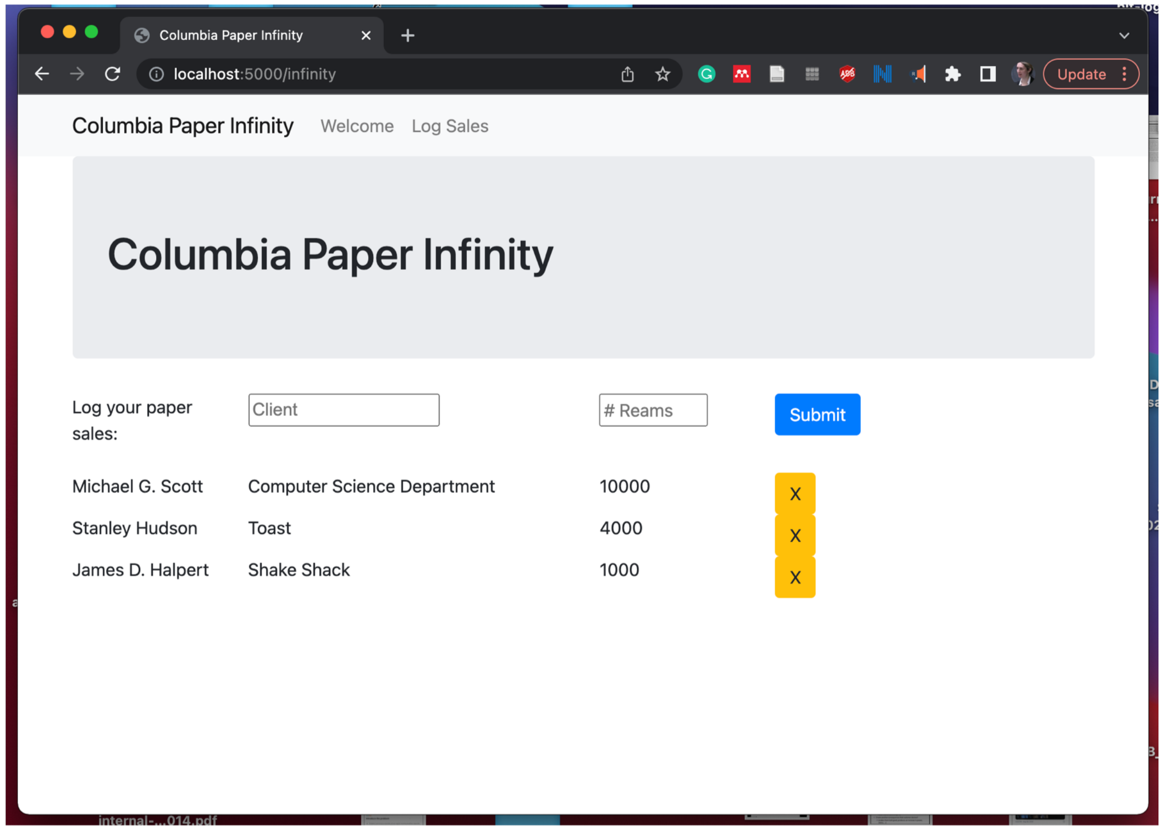
name:

michael scott

jim halpert

Client:
gets data from server
(and displays it to all users)

Main. Put a backend behind Log Sales and save the data.



Columbia Paper Infinity

localhost:5000/infinity

Columbia Paper Infinity Welcome Log Sales

Columbia Paper Infinity

Log your paper sales: Client # Reams Submit

Michael G. Scott	Computer Science Department	10000	X
Stanley Hudson	Toast	4000	X
James D. Halpert	Shake Shack	1000	X

Tip: start by copying the people folder and editing it