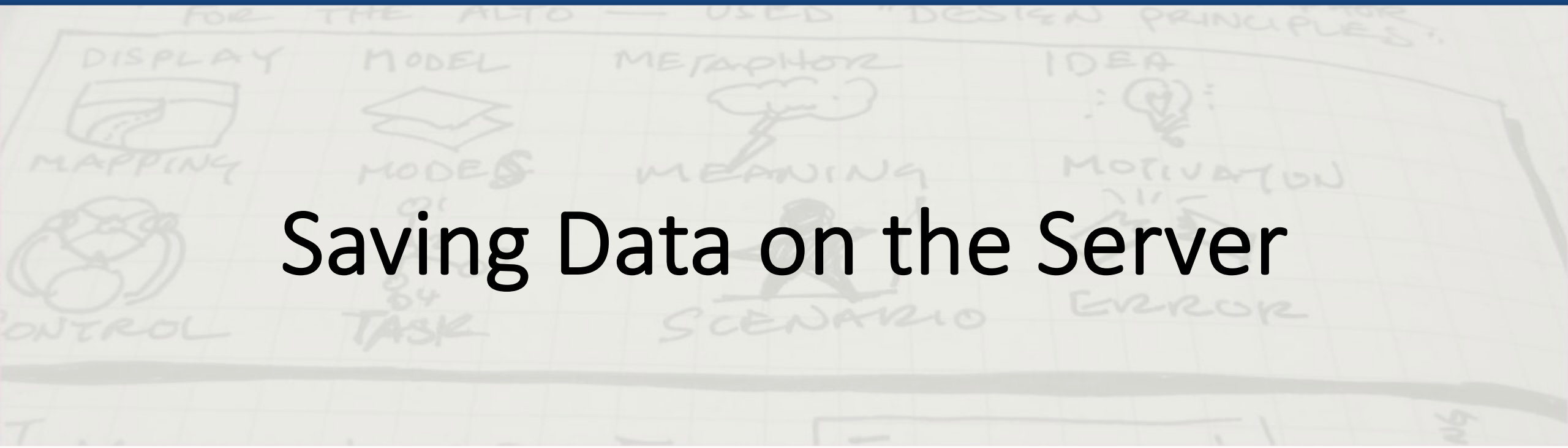


Saving Data on the Server

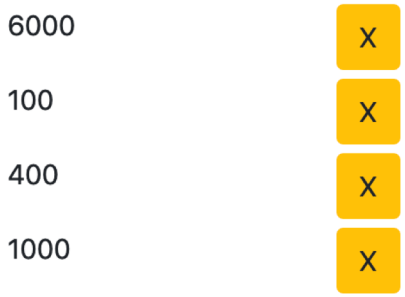
Prof. Lydia Chilton

COMS 4170

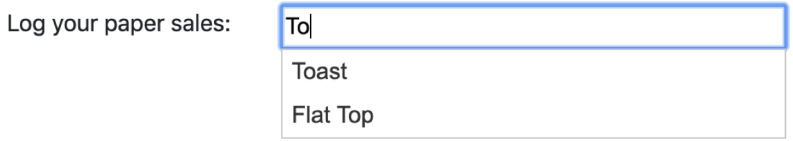
19 February 2025

In HW4, you created widgets dynamically

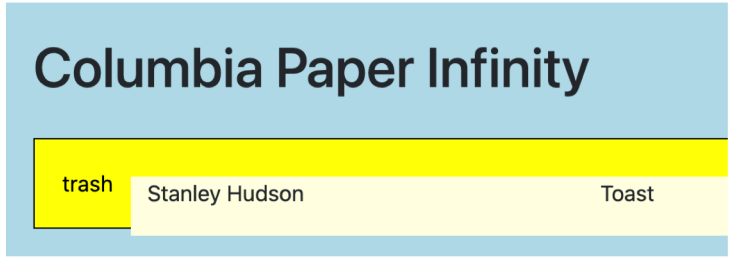
Buttons



Autocomplete

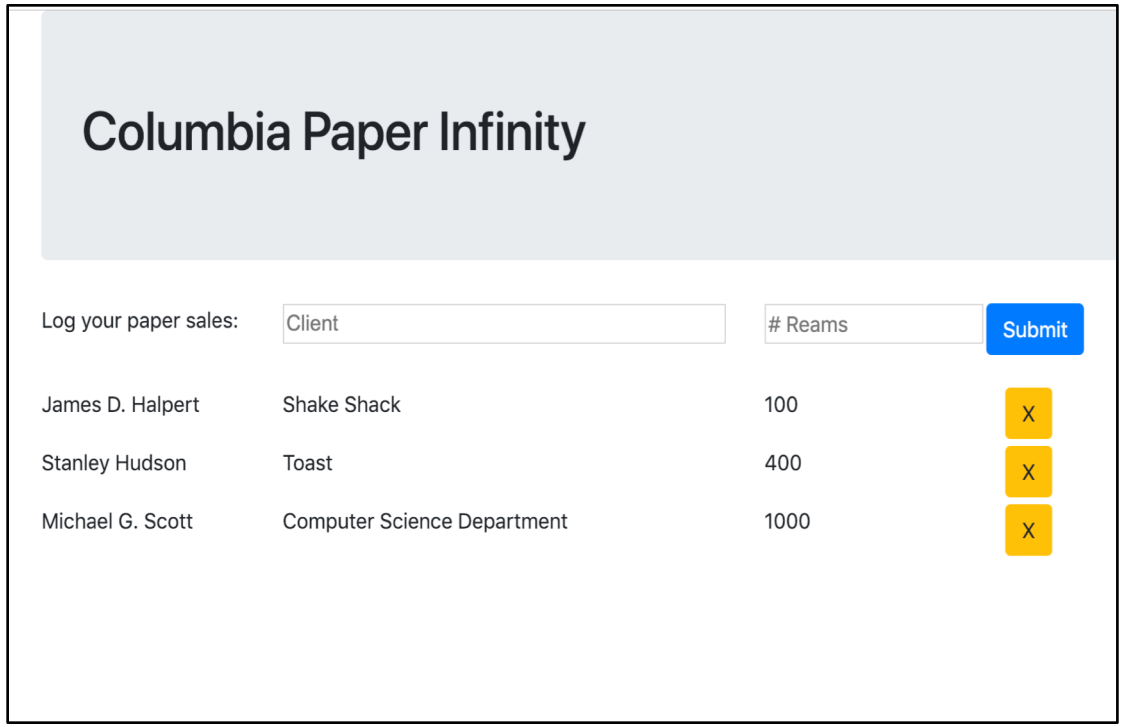


Drag and Drop

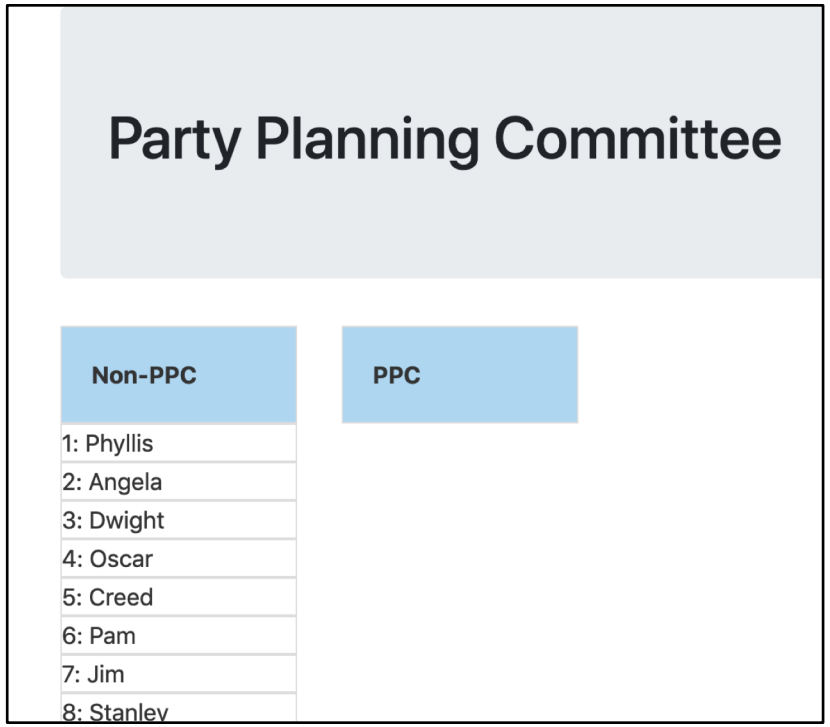


Added customization
(hovering and drop target feedback)

You allowed users to interact with data



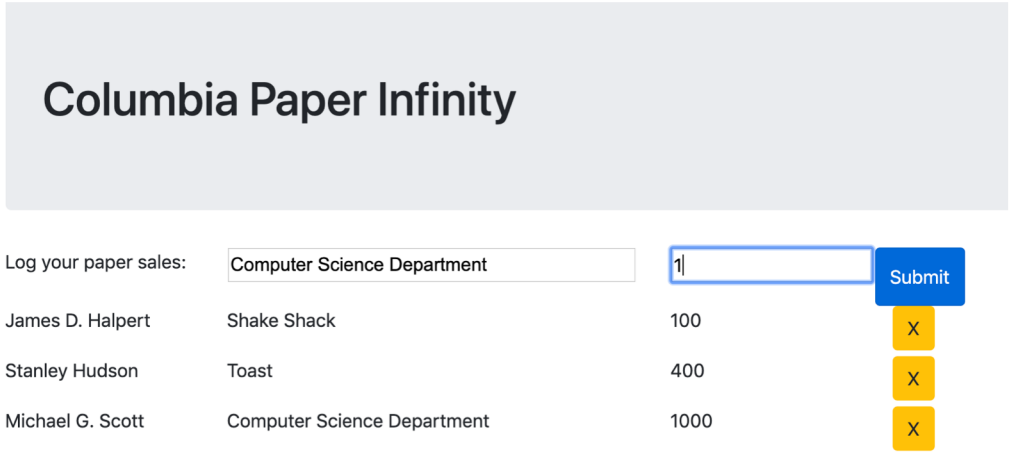
Create / Delete data



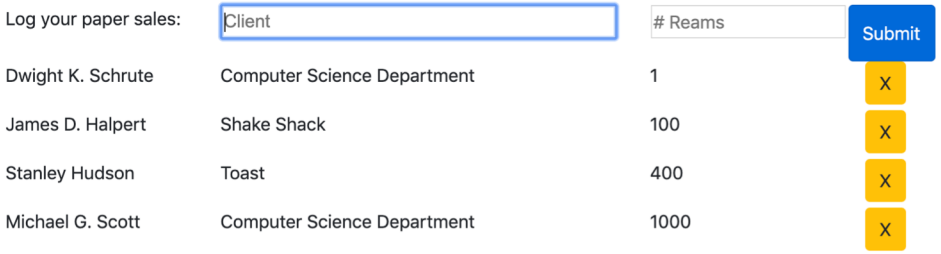
Update data

But there’s a big problem:

Add data

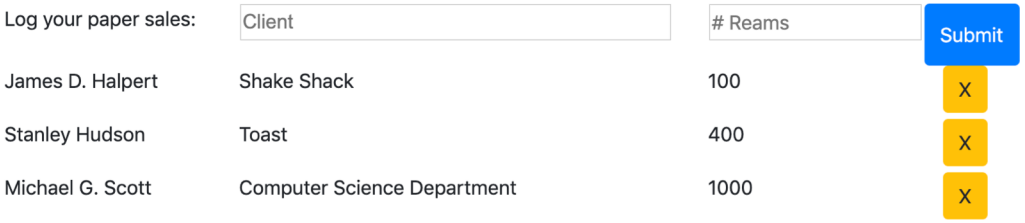


Data appears



REFRESH PAGE

Data is gone!

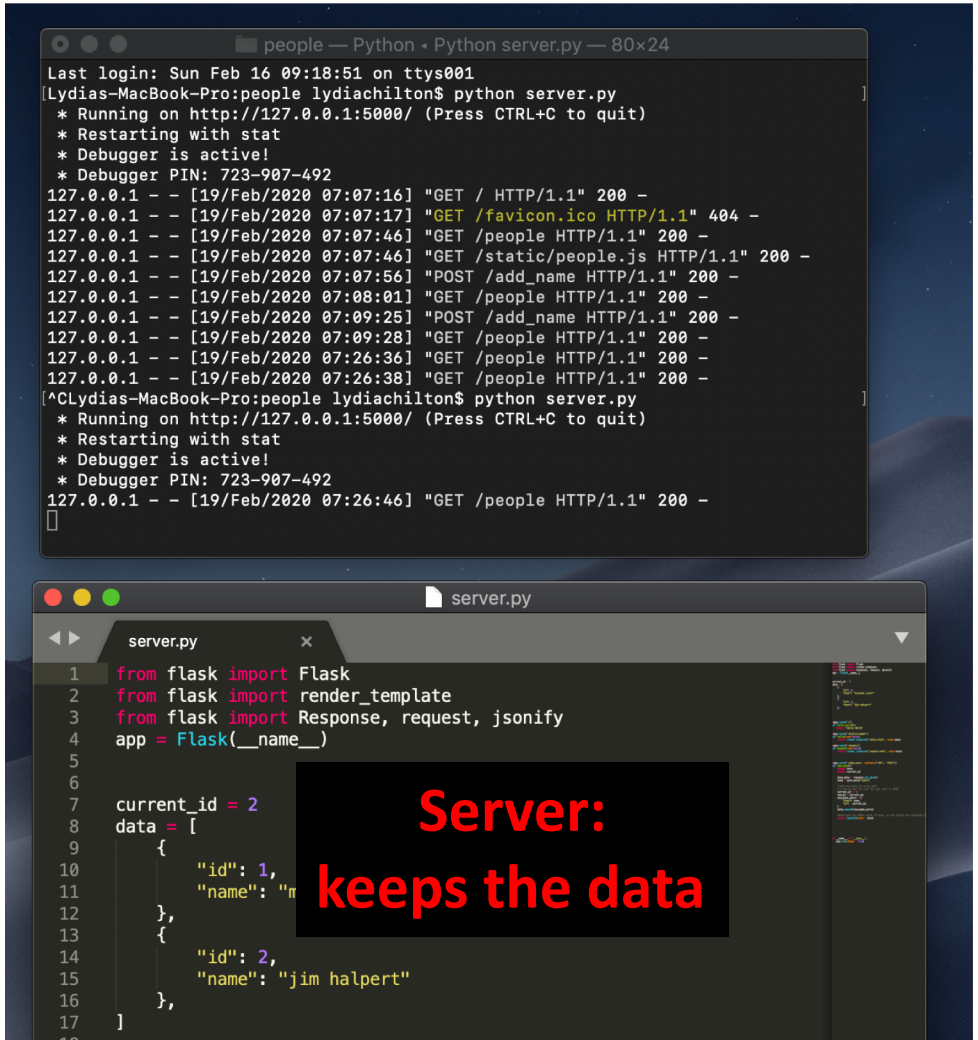


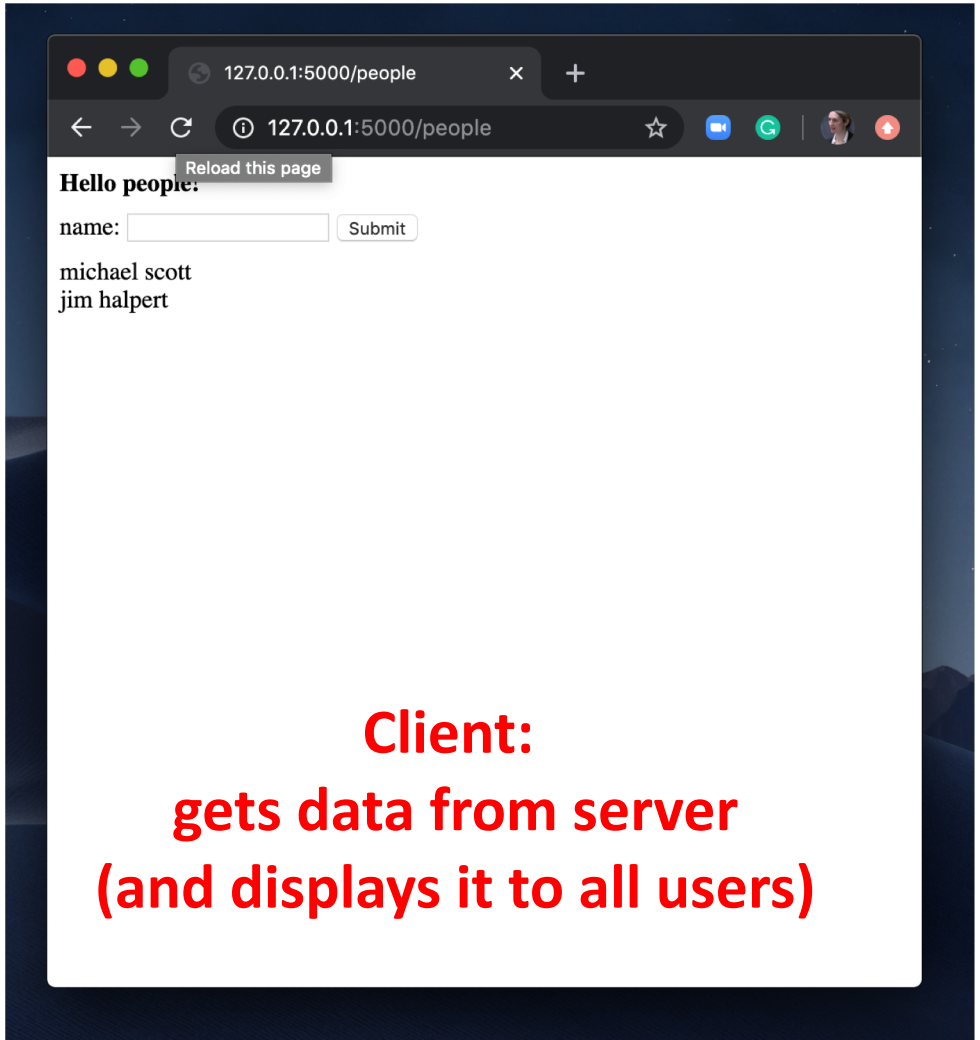
The data doesn’t save

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>

```

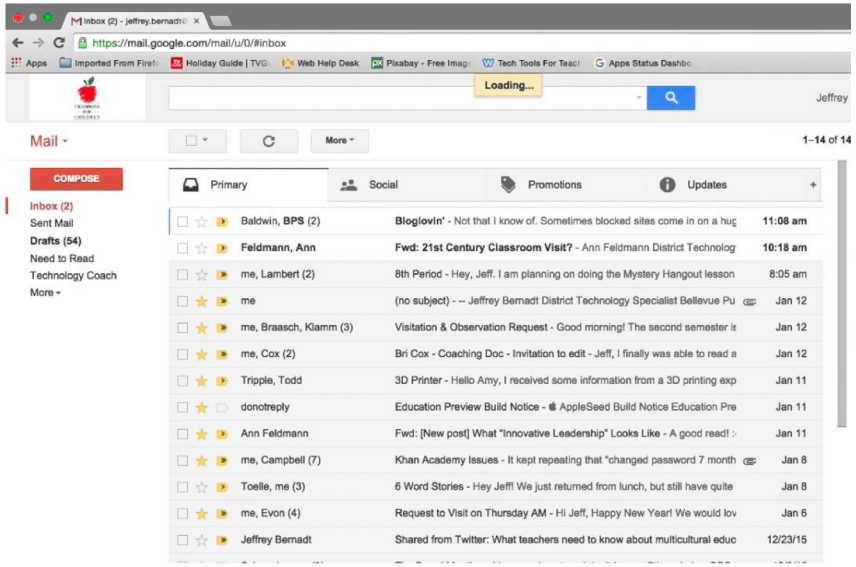
To keep data around, we need to store it somewhere else – another computer that will never get turned off.





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”
    },
]
```



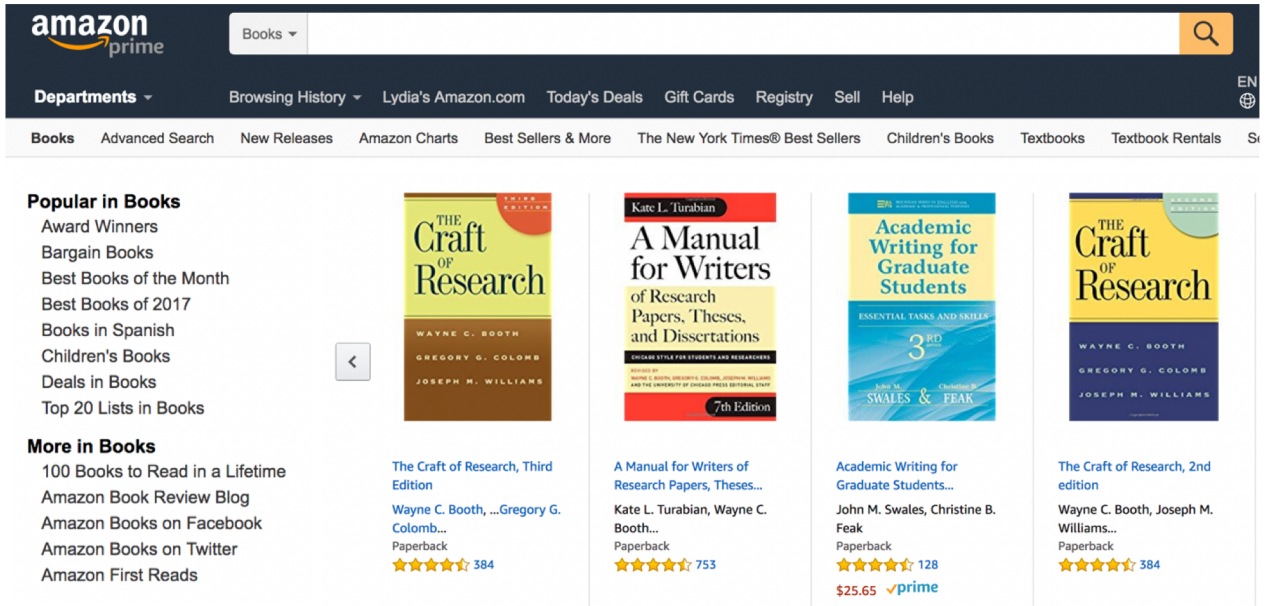
Server:
keeps the data

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”
    },
]
```

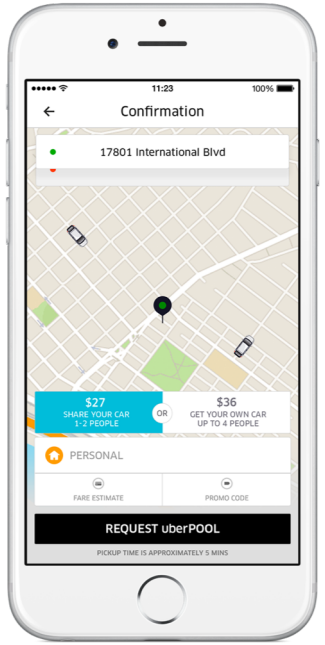
Server:
keeps the data



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”: “griffin newbold”,
    “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”: “kynnedy”,
    “image”: “./kynnedy.png”,
    “likes”: “1000”,
    ”dislikes”: 0,
    },
    {
    “id”: 821212134,
    “name”: “salia”,
    “image”: “./salia.png”,
    “likes”: “1000”,
    “dislikes”: 0,
    },
]
```

Server:

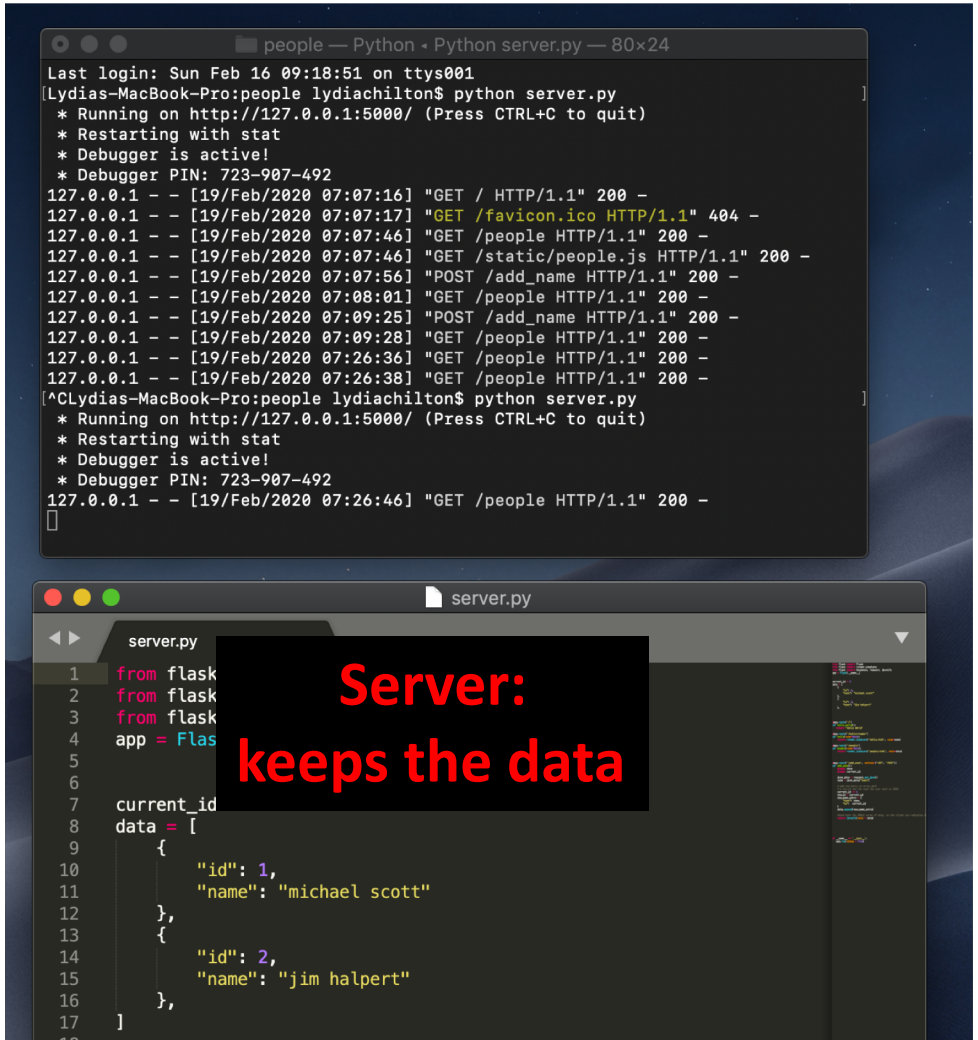
keeps the data

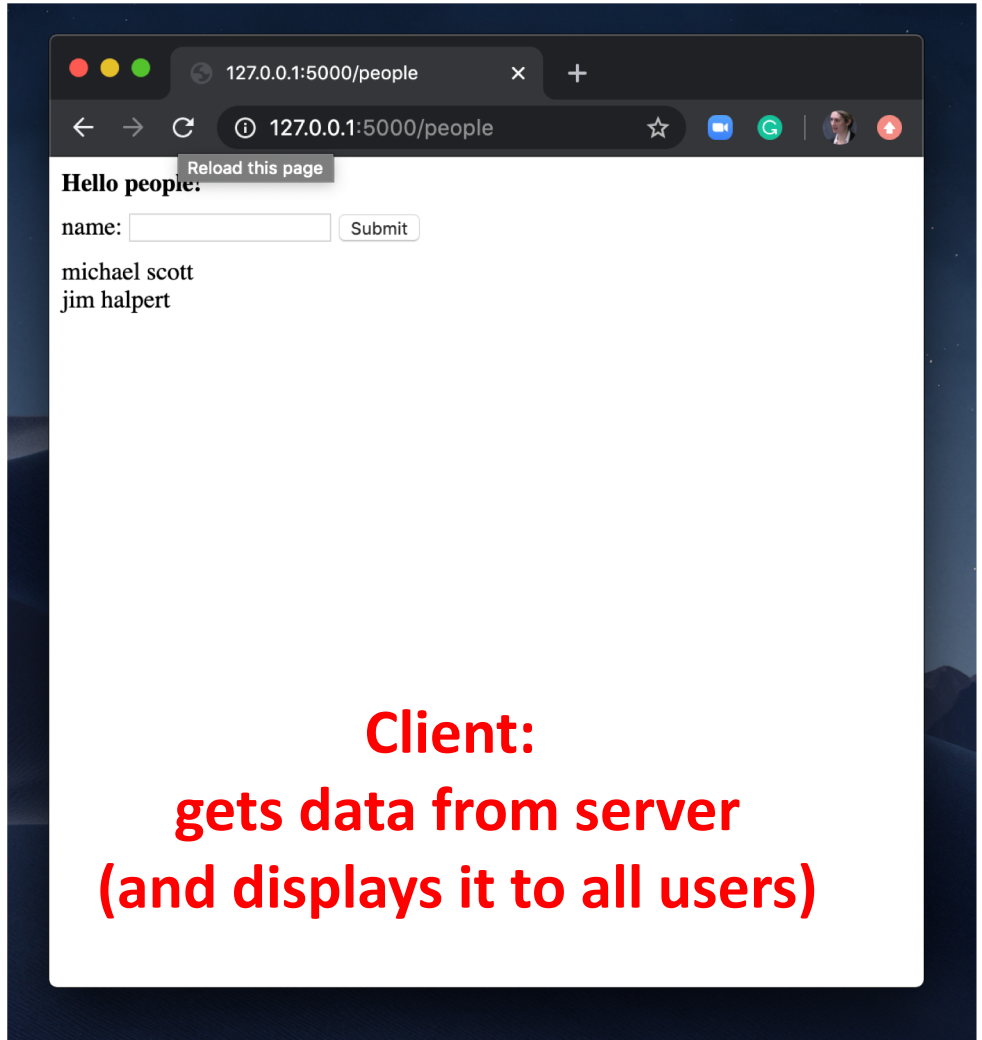
Client:

gets data from server

(and displays it to all users)

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





Example application:

Storing and Serving data in Flask

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

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

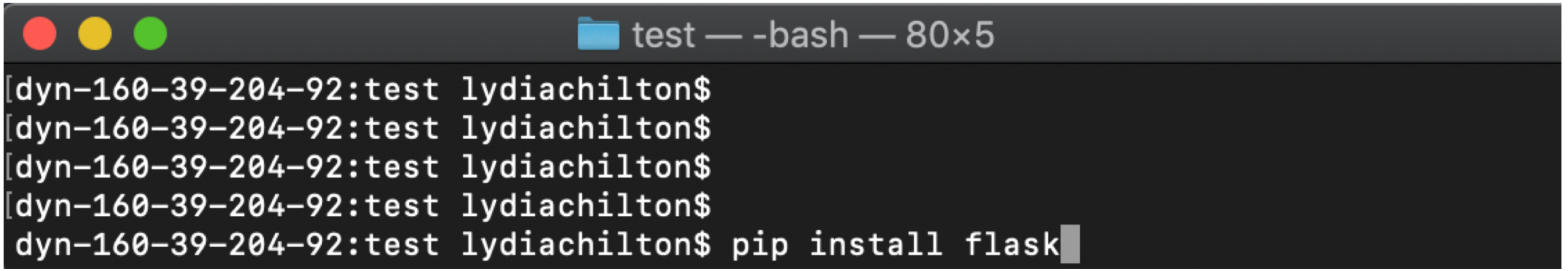
FEBRUARY 14

Homework 5 out

Saving Data on the Server

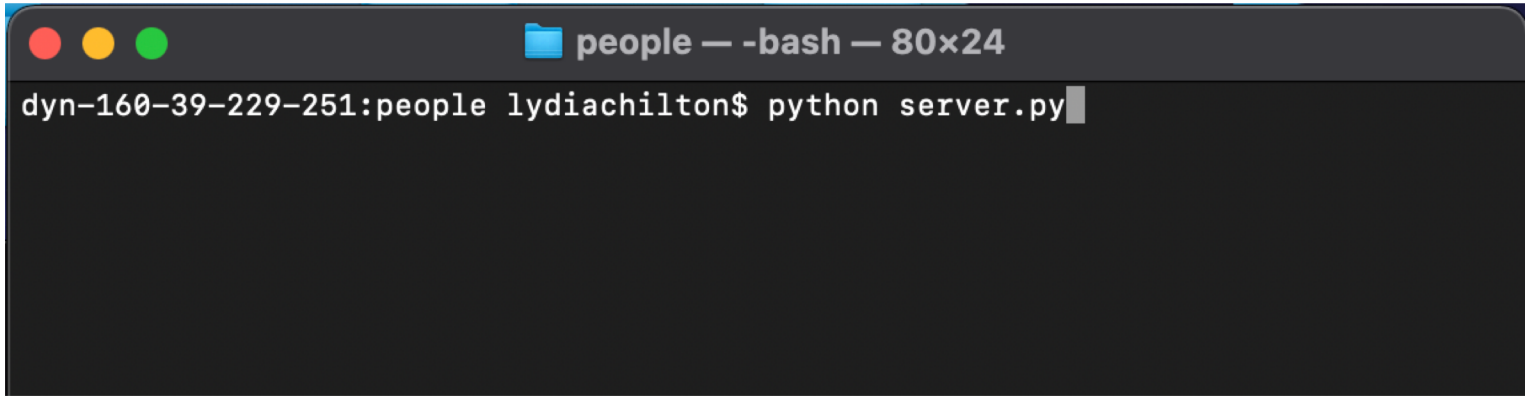
people.zip

You must first install Flask

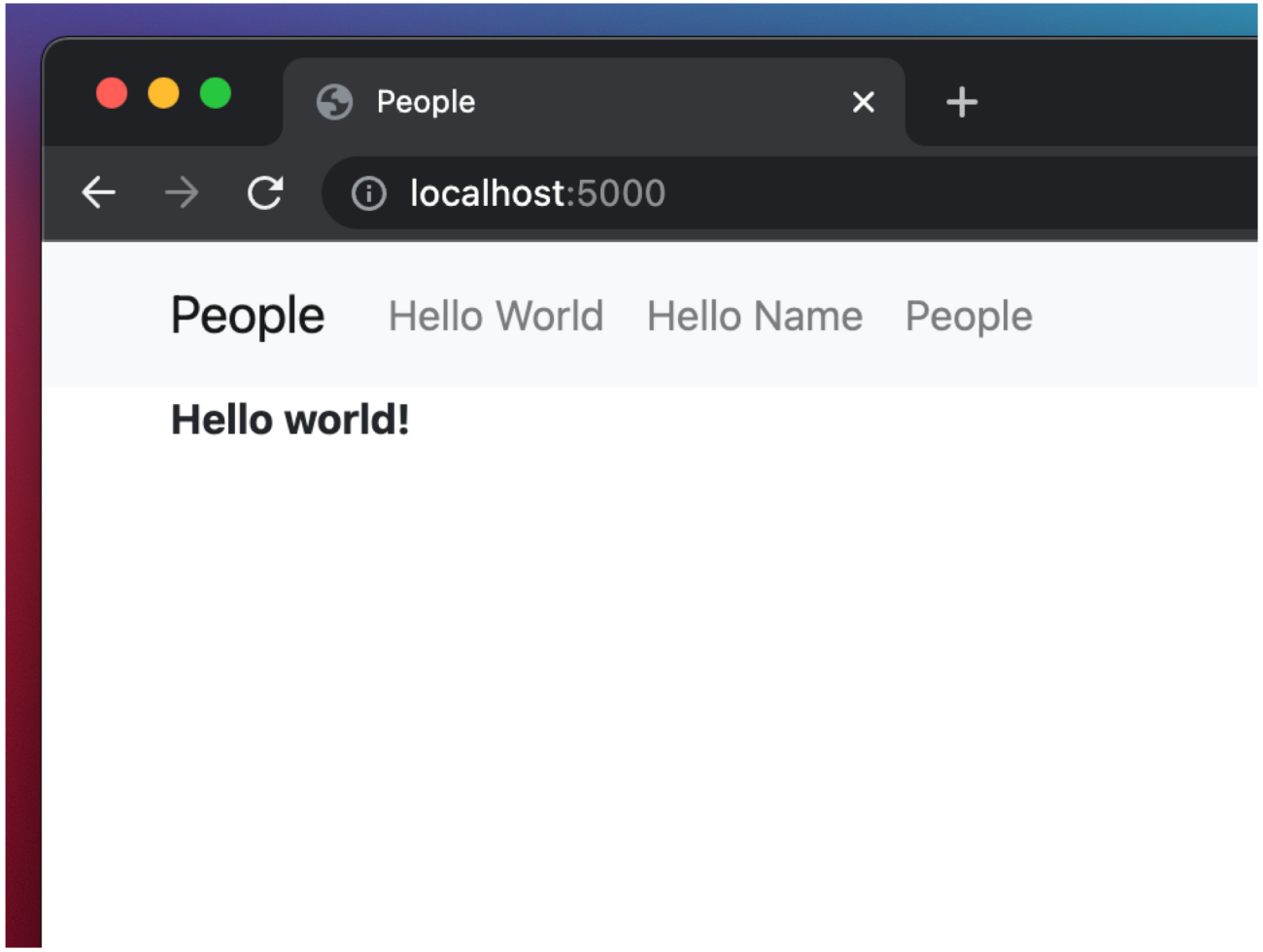


Then run the server.py file.

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



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

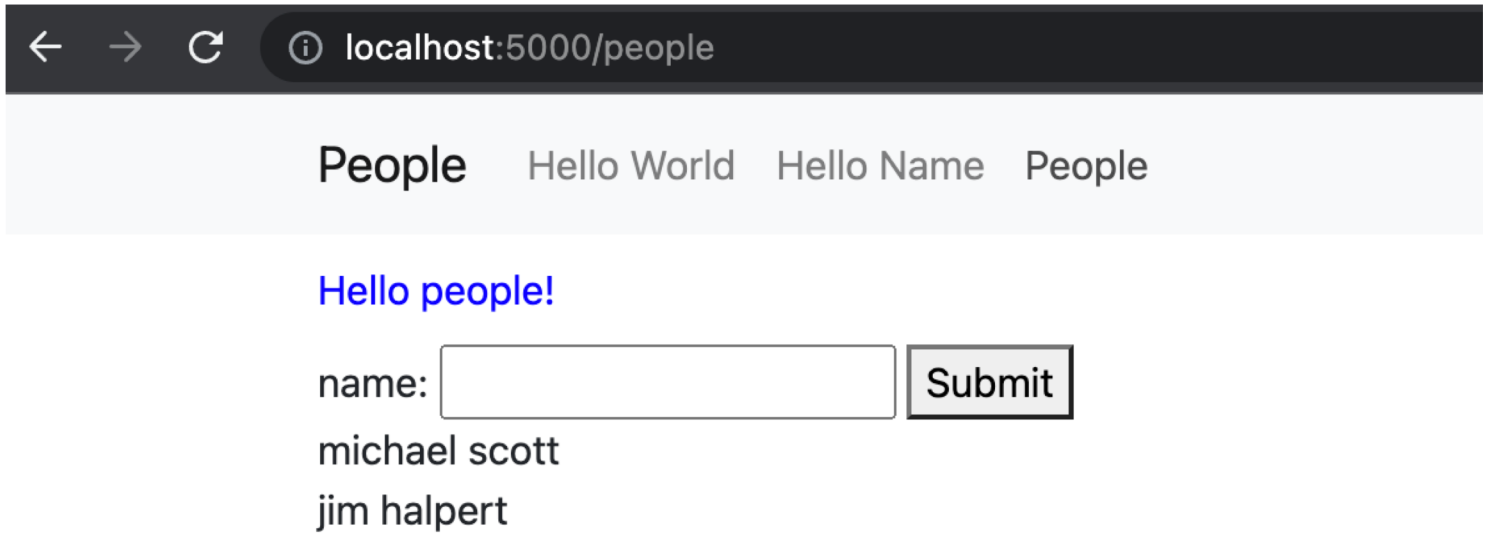


Navbar!

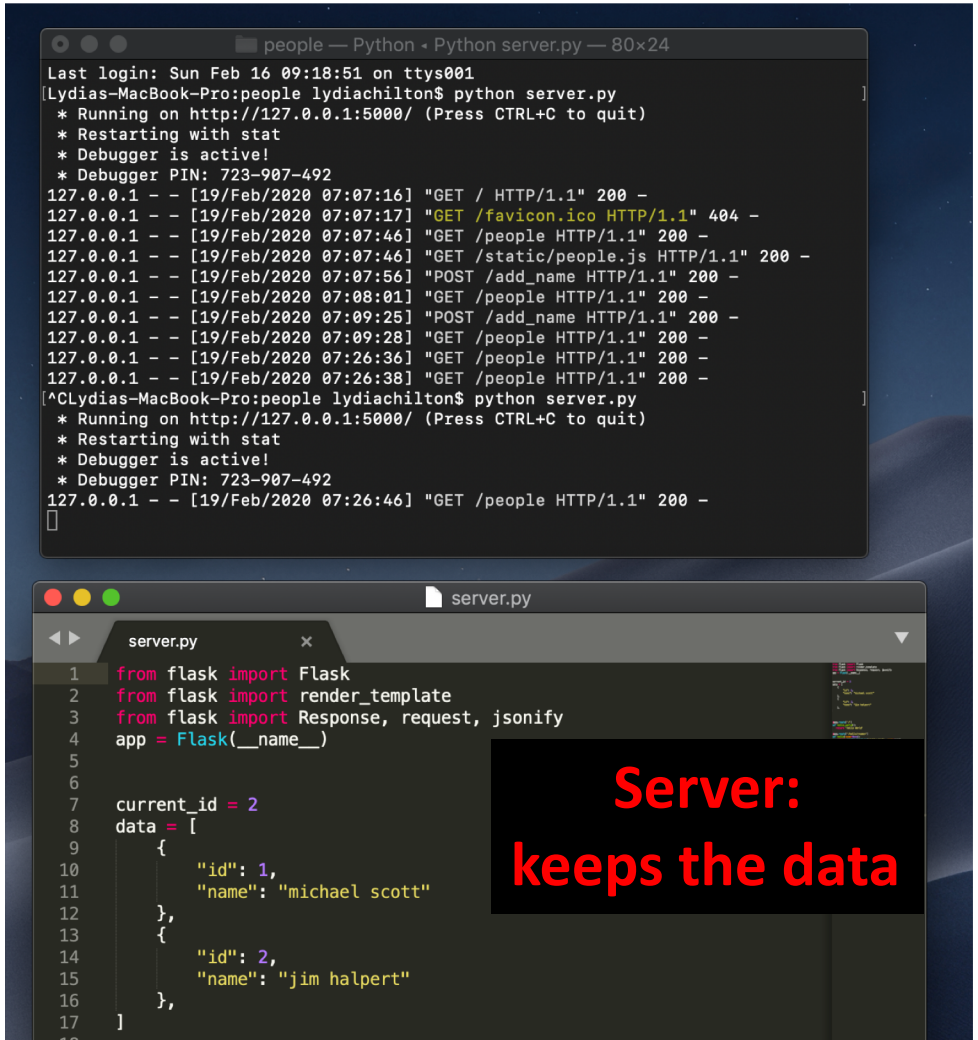
Content block!

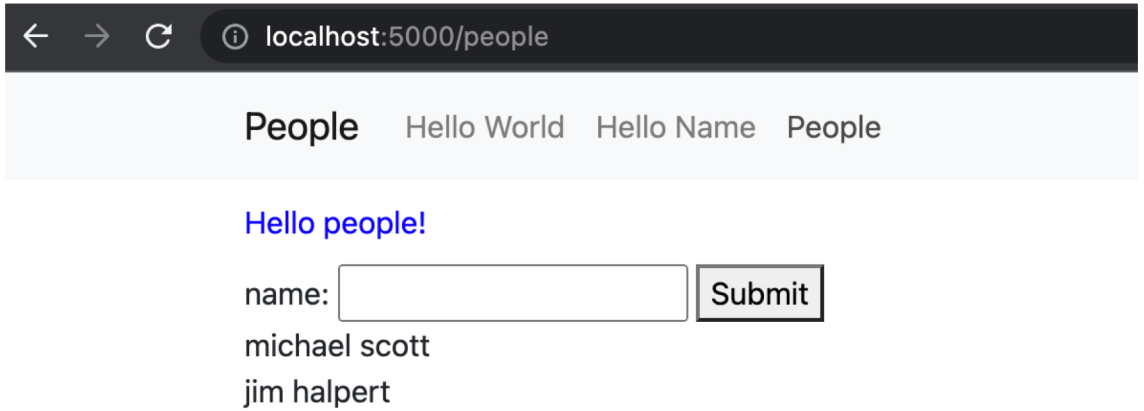
http://localhost:5000/people

lets you create a list of names (look familiar?)



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



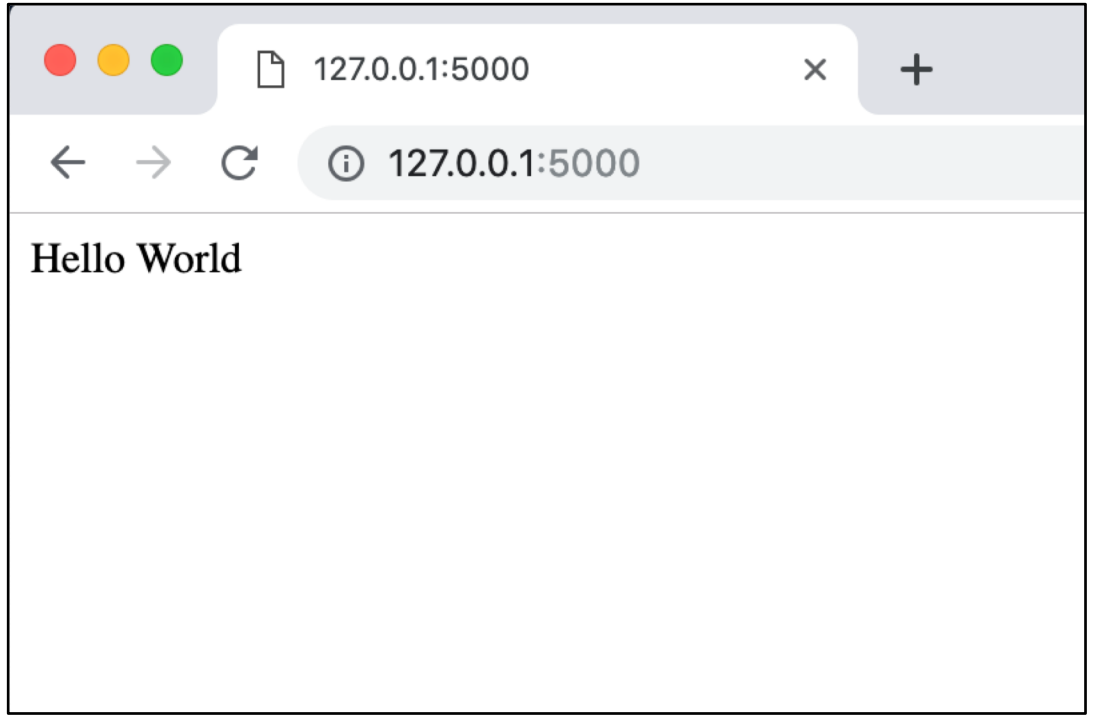


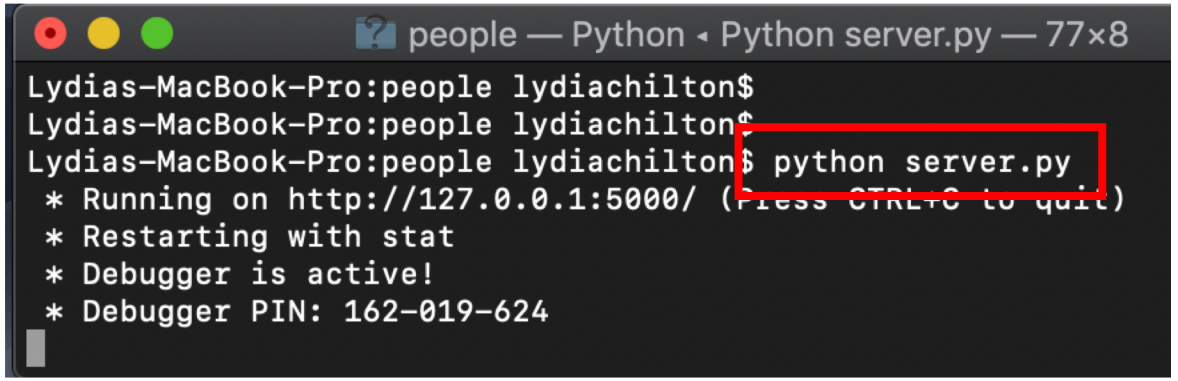
Client:
gets data from server
(and displays it to all users)

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

server.py

```
1  from flask import Flask
2  app = Flask(__name__)
3
4  @app.route('/')
5  def hello_world():
6      return 'Hello World'
7
8  if __name__ == '__main__':
9      app.run()
```

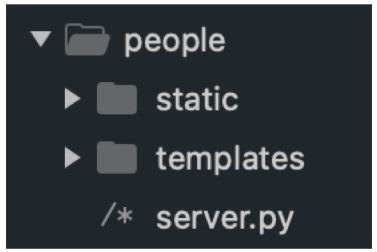


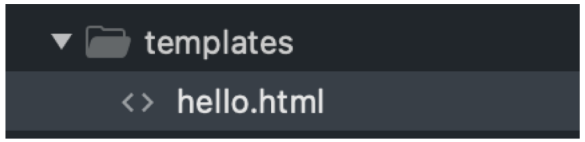


How to render an HTML page with data

server.py

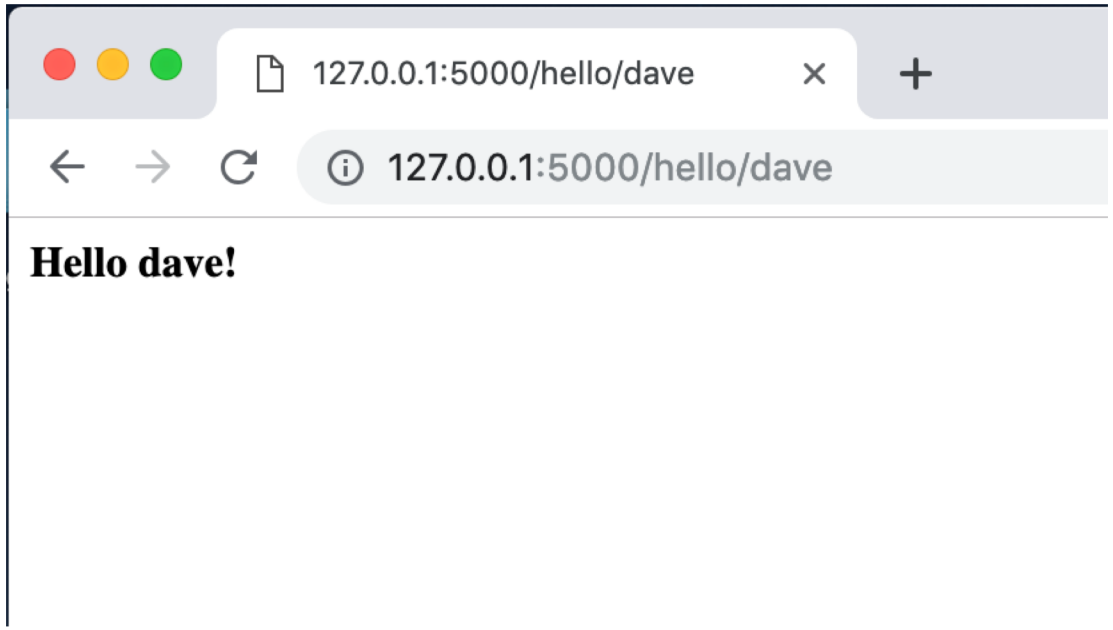
```python
from flask import Flask
from flask import render_template
app = Flask(__name__)

@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()
```





server.py | hello.html

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

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



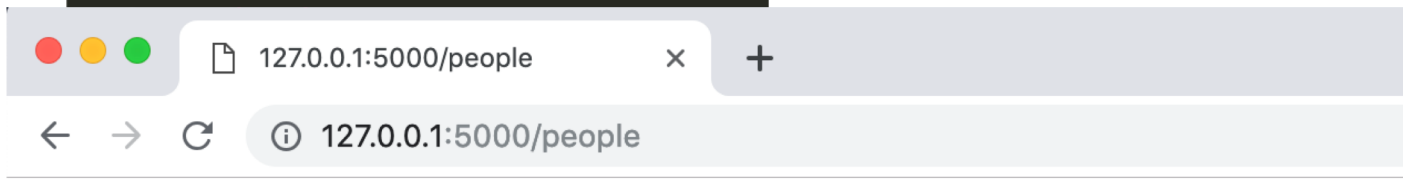
How to send an array of data to JavaScript?

server.py

```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()
```

people.html

```html
<html>
<head>

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



Hello people!

How to send an array of data to JavaScript?

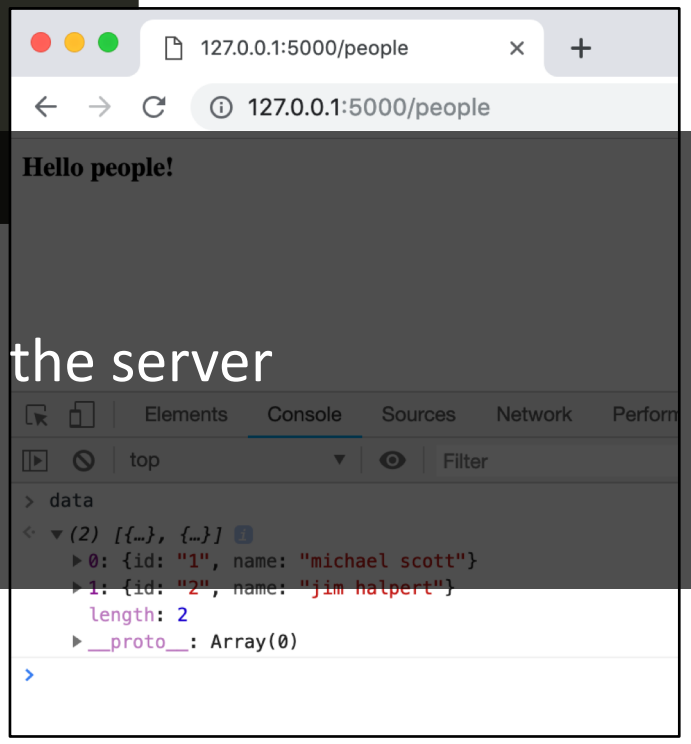
server.py

```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/')
@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()
```

people.html

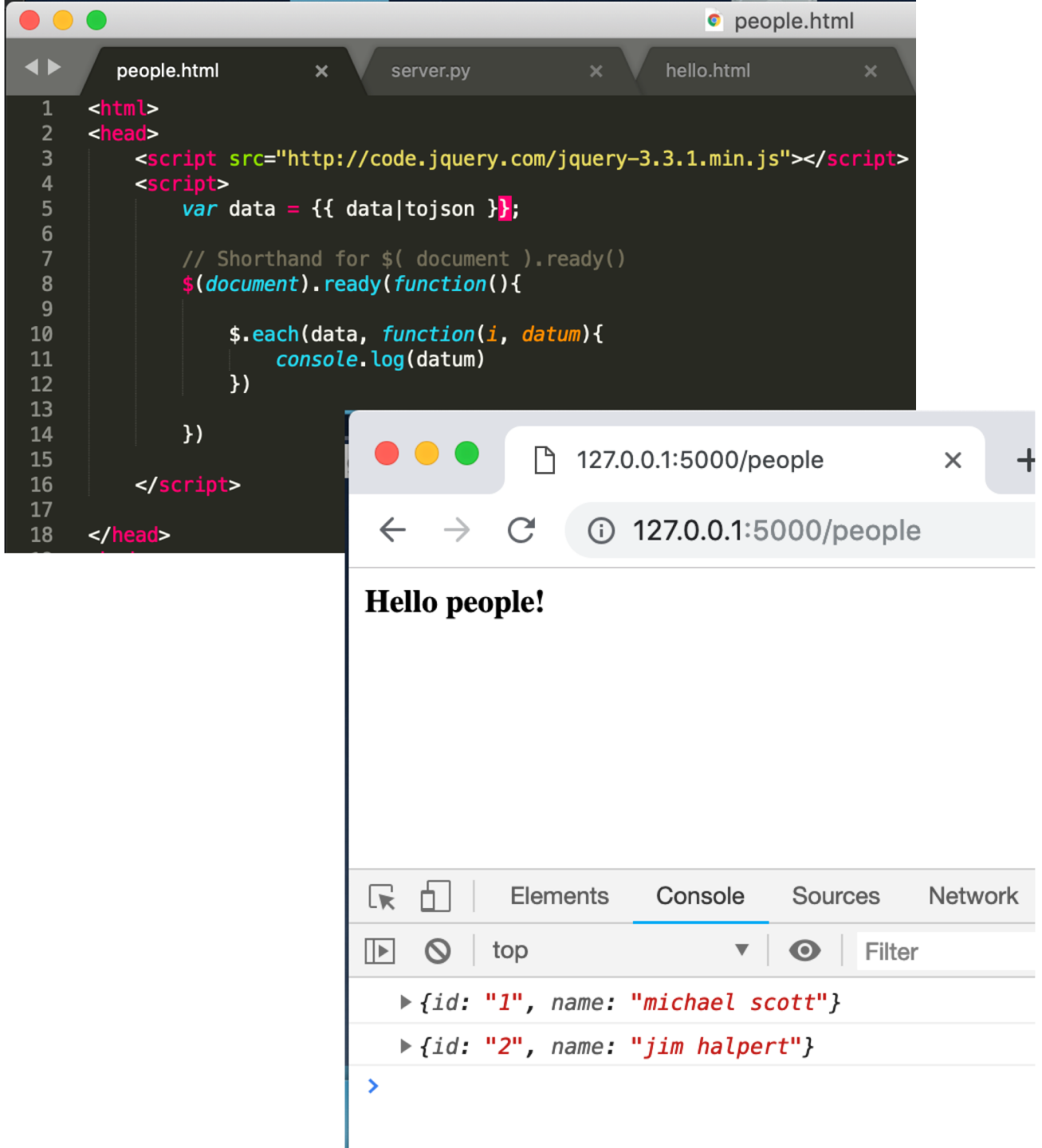
```html
<html>
<head>

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

</head>
<body>

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

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



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



Display all the names

people.html

people.html | server.py | hello.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>
```

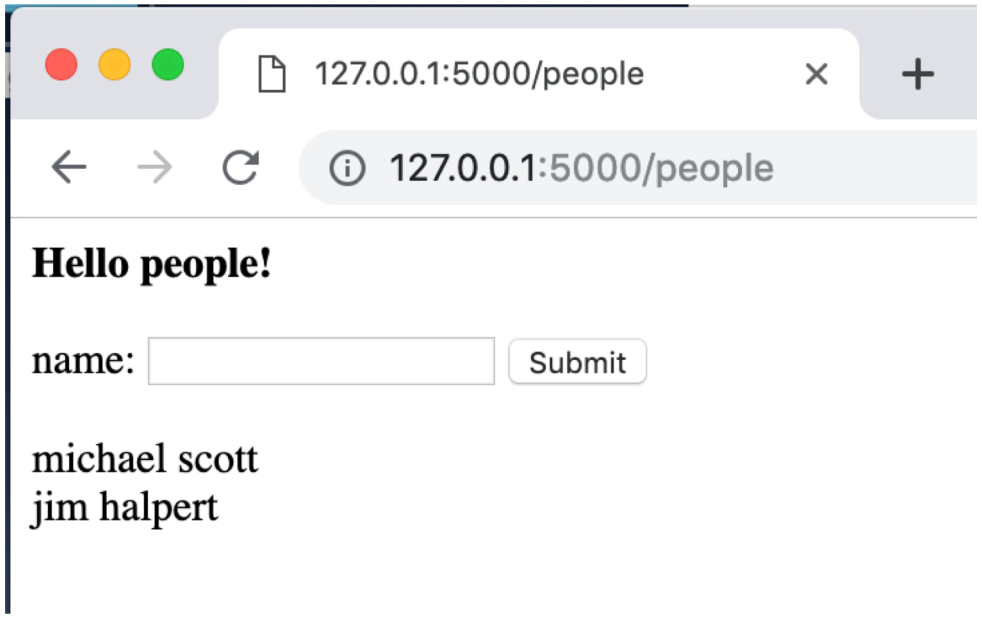
127.0.0.1:5000/people

127.0.0.1:5000/people

Hello people!
michael scott
jim halpert

How do users submit names? (two ways)

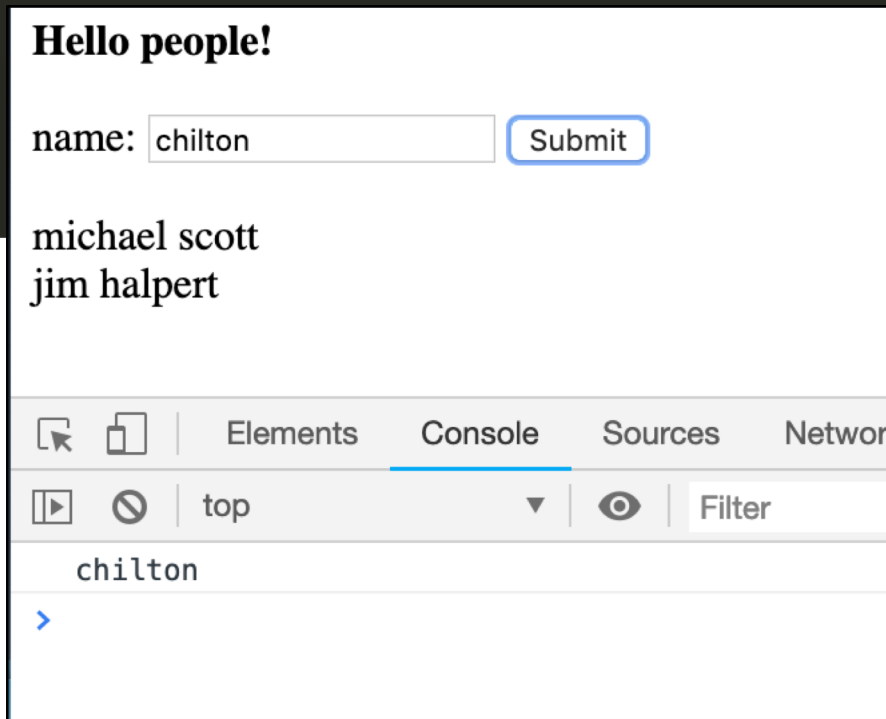
```
<b>Hello people!</b>
<br>
<br>

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



What's the first thing the click handler does?

people.html | server.py | hello.html

```
1  <html>
2  <head>
3      <script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>
4      <script>
5          var data = {{ data|tojson }};
6
7          // Shorthand for $( document ).ready()
8          $(document).ready(function(){
9
10             $.each(data, function(i, datum){
11                 var new_name= $("<div>"+datum["name"]+"</div>")
12                 $("#people_container").append(new_name)
13             })
14
15             $("#submit_name").click(function(){
16
17                 var name = $("#new_name").val()
18                 console.log(name)
19
20             })
21
22         })
23
24     </script>
25
```



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

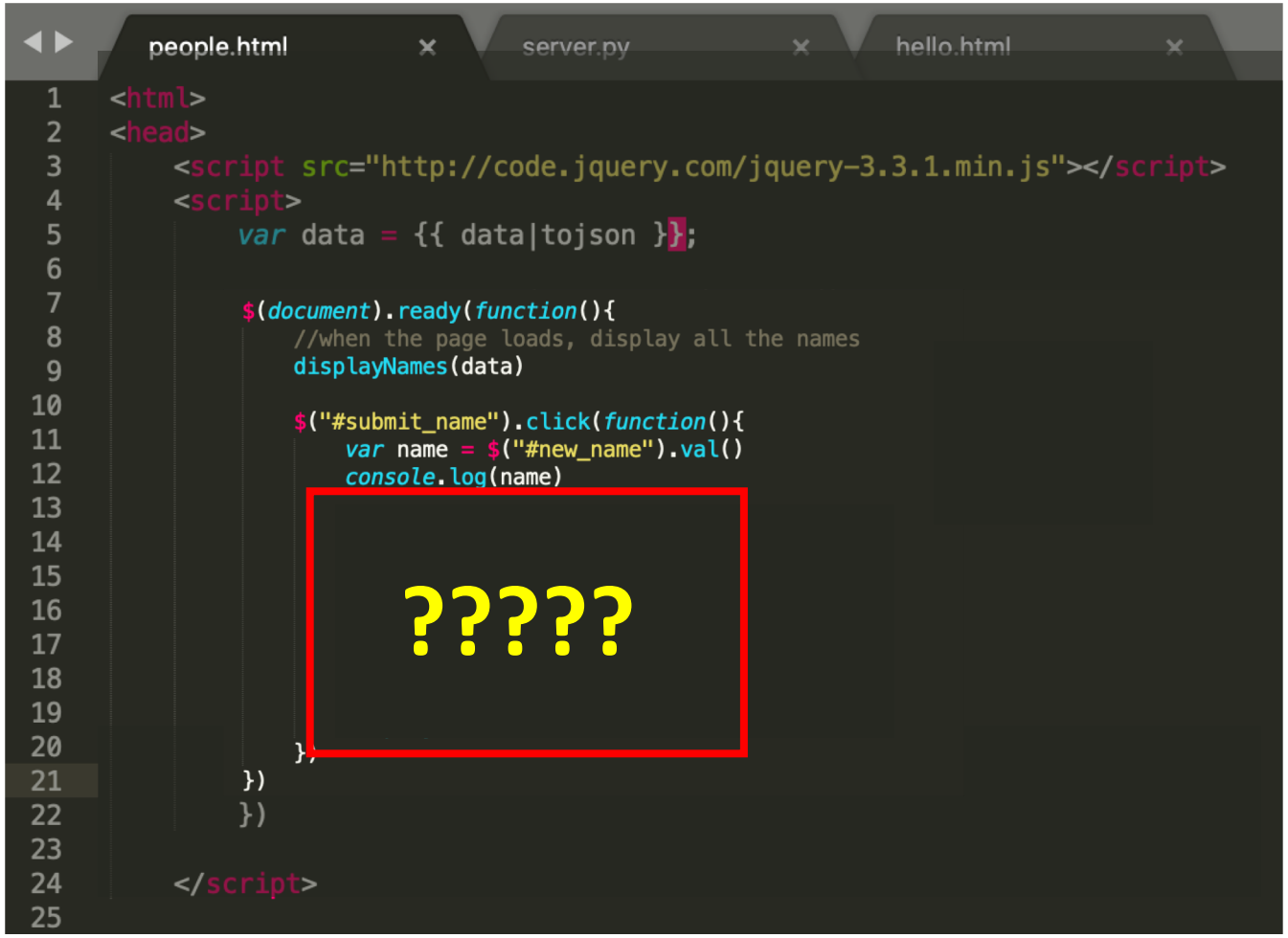
But this won't save data to the server.

What code do we need to write instead?

people.html | server.py | hello.html

```
<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>
```

Save the data to the server



Save the data to the server

people.html | server.py | hello.html

```
<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)

                save_name(name)

            })
        })
        })

    </script>
```

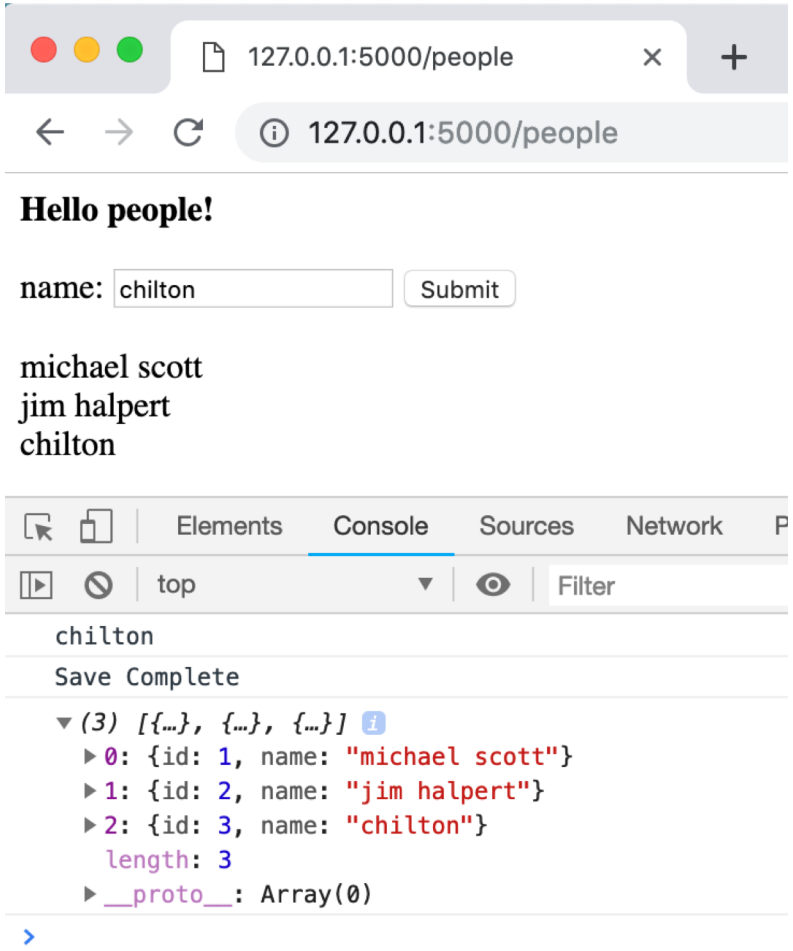
server.py hello.html people

```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)
```

What happens on the server?

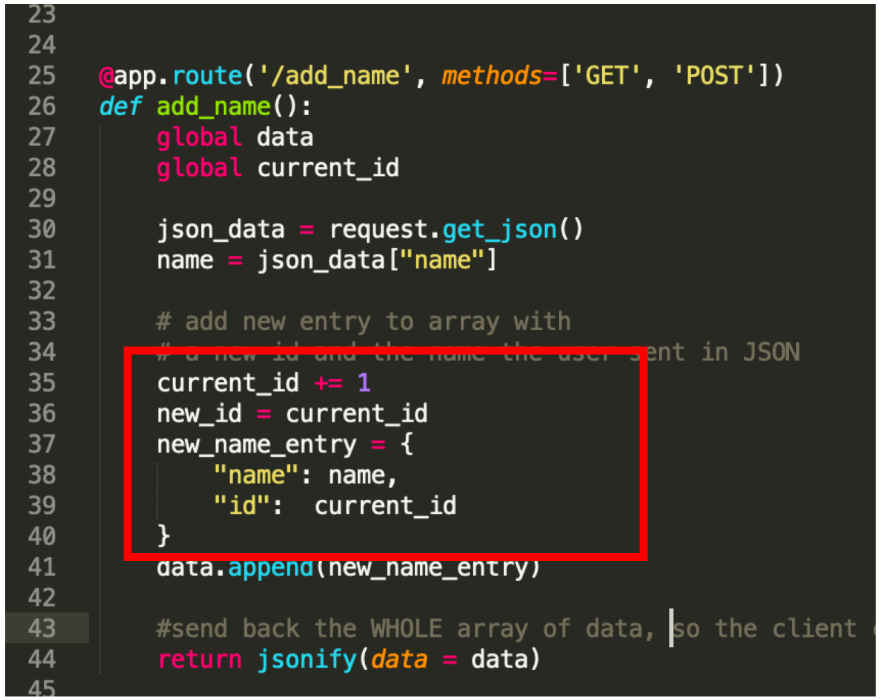
```javascript
var saveName = function(name){
    var data_to_save = {"name": name}
    $.ajax({
        type: "POST",
        url: "add_name",
        dataType : "json",
        contentType: "application/json; charset=utf-8",
        data : JSON.stringify(data_to_save),
        success: function(result){
            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?



Refresh the page to see if the new data stays

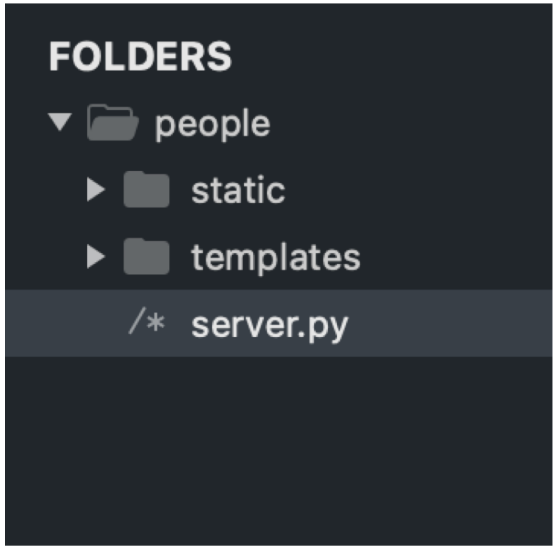
We MUST calculate the id on the server, not the client. Why?



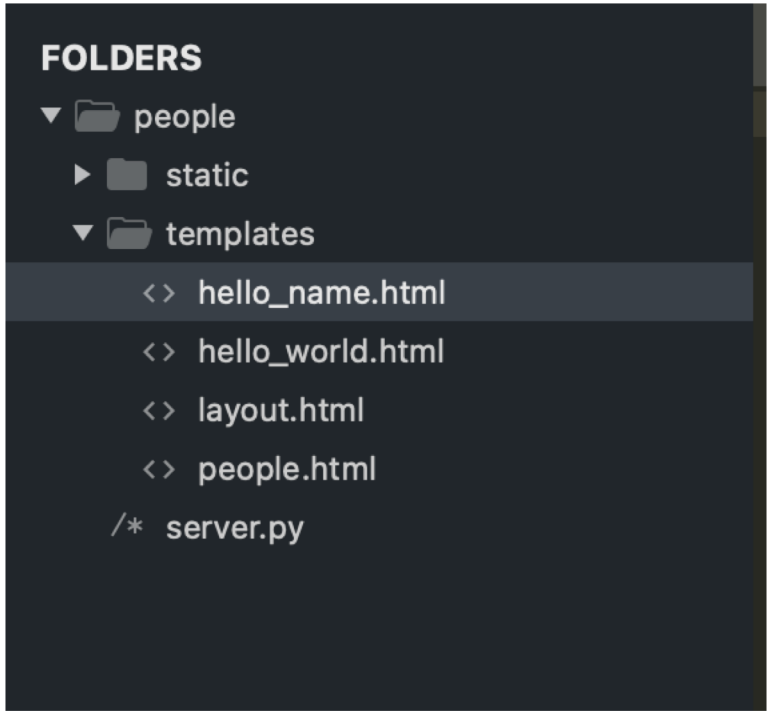
```
23
24
25  @app.route('/add_name', methods=['GET', 'POST'])
26  def add_name():
27      global data
28      global current_id
29
30      json_data = request.get_json()
31      name = json_data["name"]
32
33      # add new entry to array with
34      # a new id and the name the user sent in JSON
35      current_id += 1
36      new_id = current_id
37      new_name_entry = {
38          "name": name,
39          "id":  current_id
40      }
41      data.append(new_name_entry)
42
43      #send back the WHOLE array of data, so the client
44      return jsonify(data = data)
45
```

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

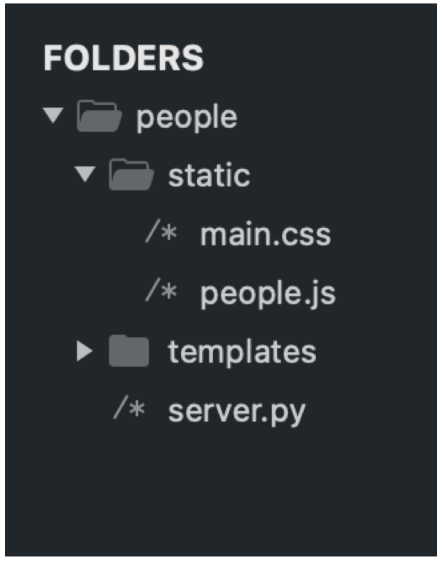
Flask projects have a very specific structure



Server.py
goes directly inside the project folder

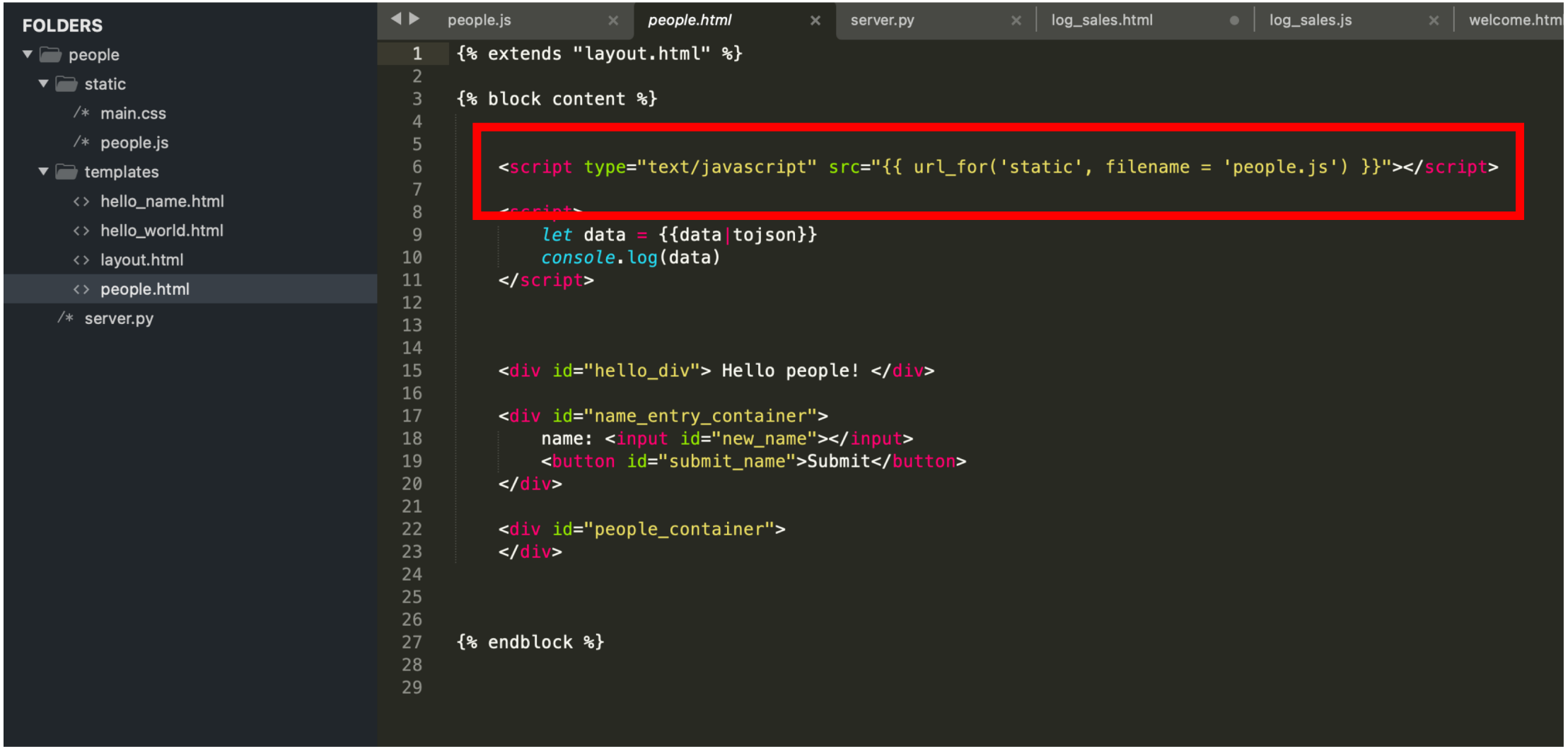


/templates (lower case)
Has HTML files

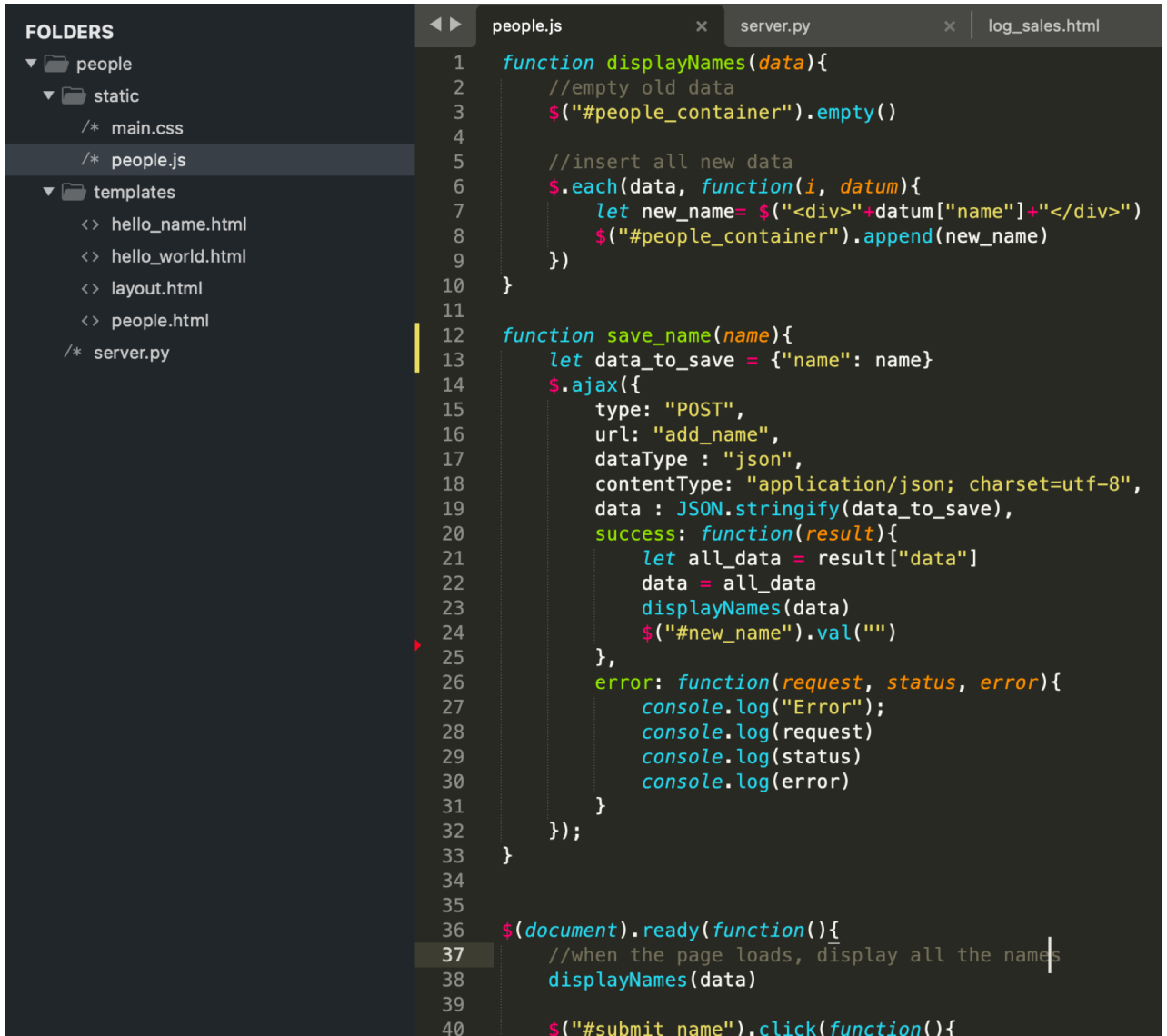


/static (lower case)
.js and .css files
(and image files)

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

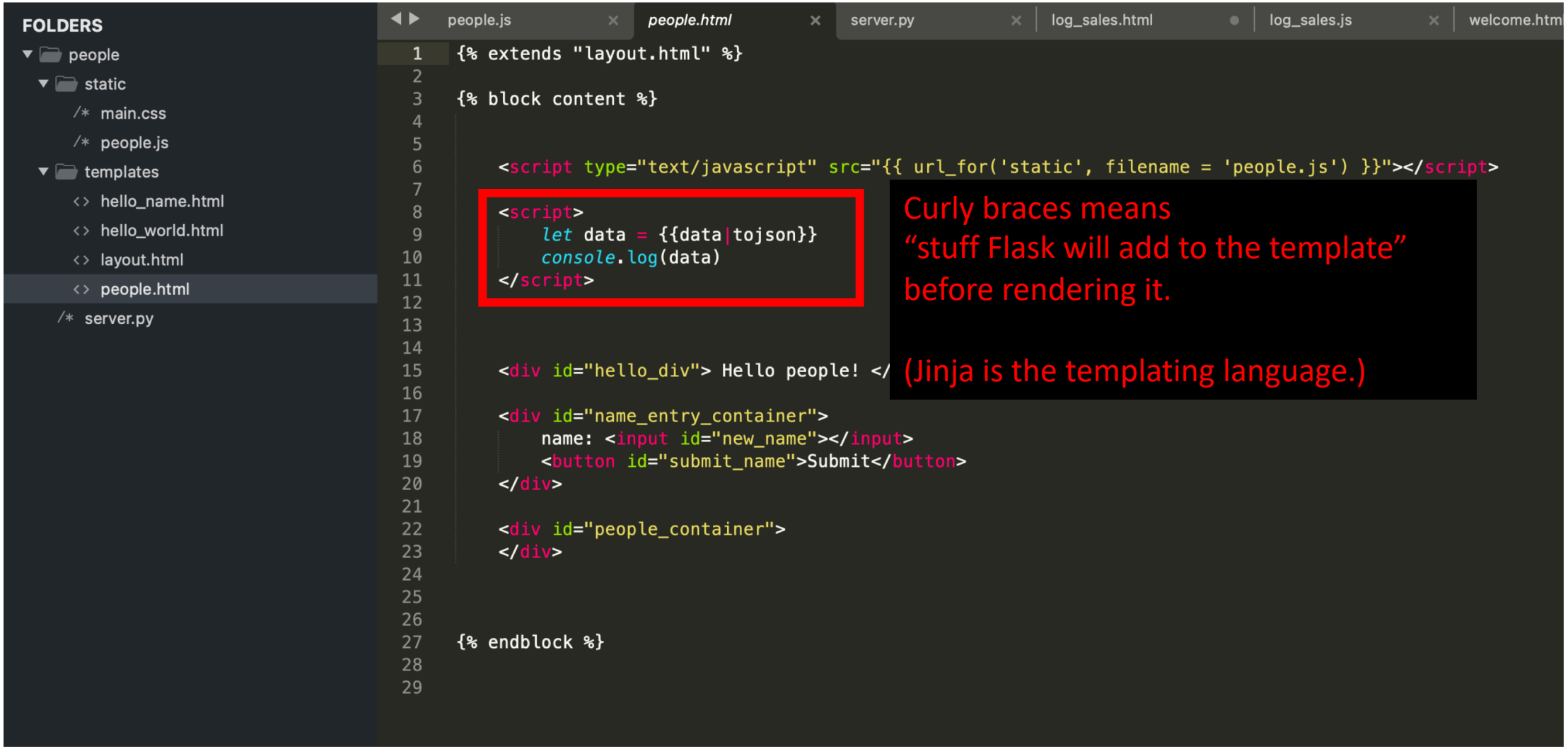


People.js is in the static folder.



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



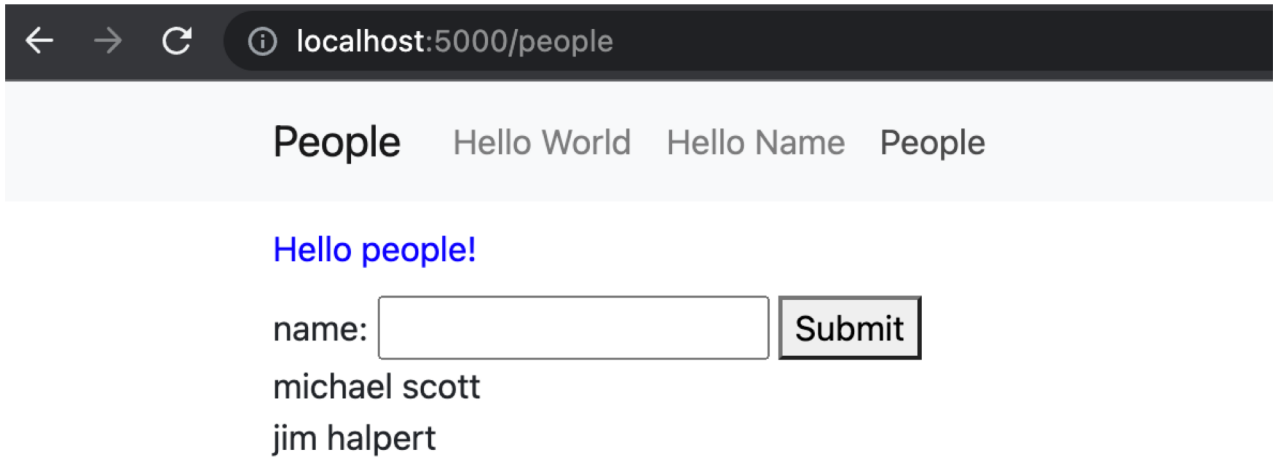
Homework 5

Putting a database behind HW4

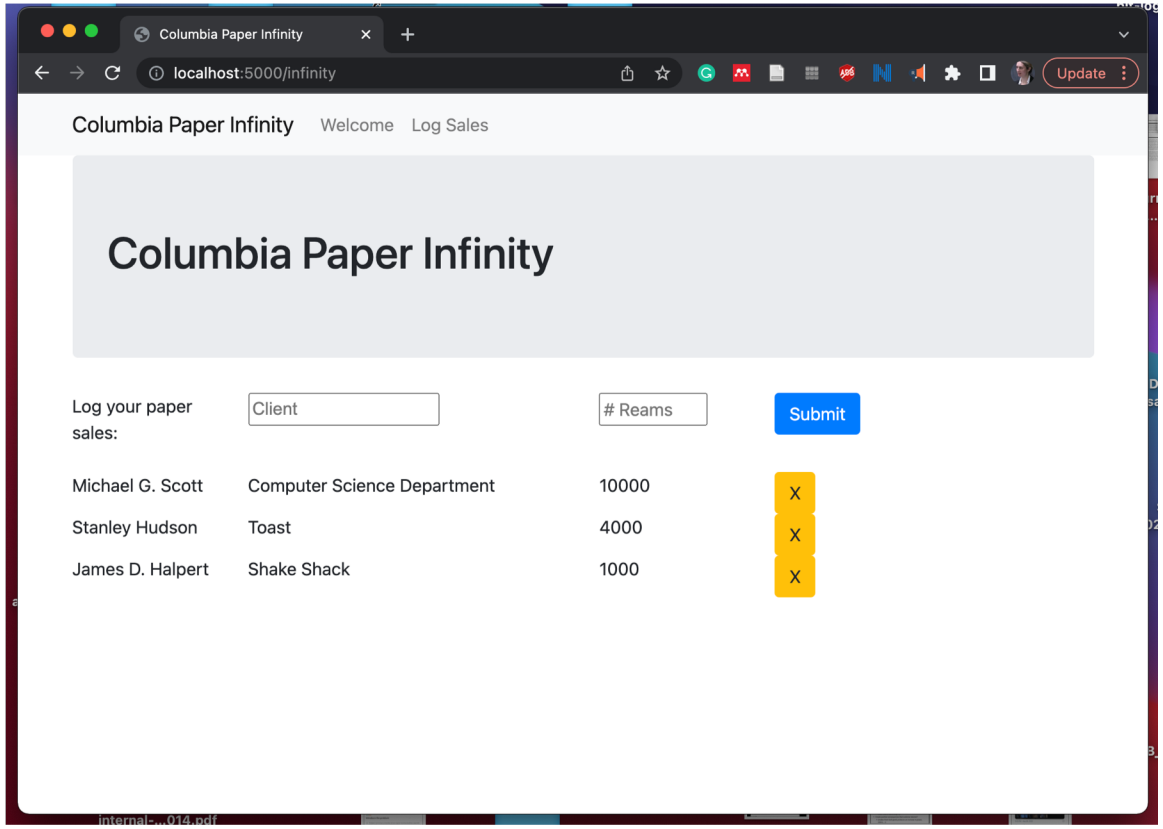
Warm up: Get the Flask sample code to run

server.py ● hello.html × people

```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)
```



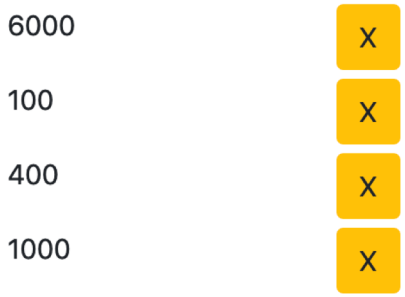
Main. Put a backend behind Log Sales and save the data.



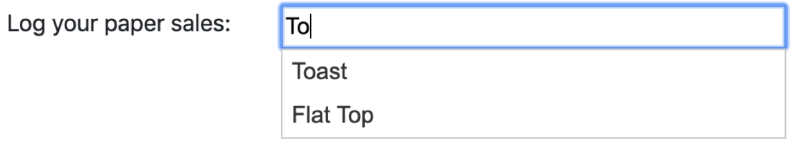
Tip: start by copying the people folder and editing it

In HW4, you dynamically created widgets

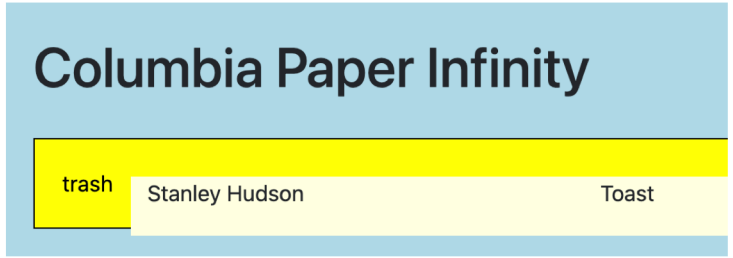
Buttons



Autocomplete

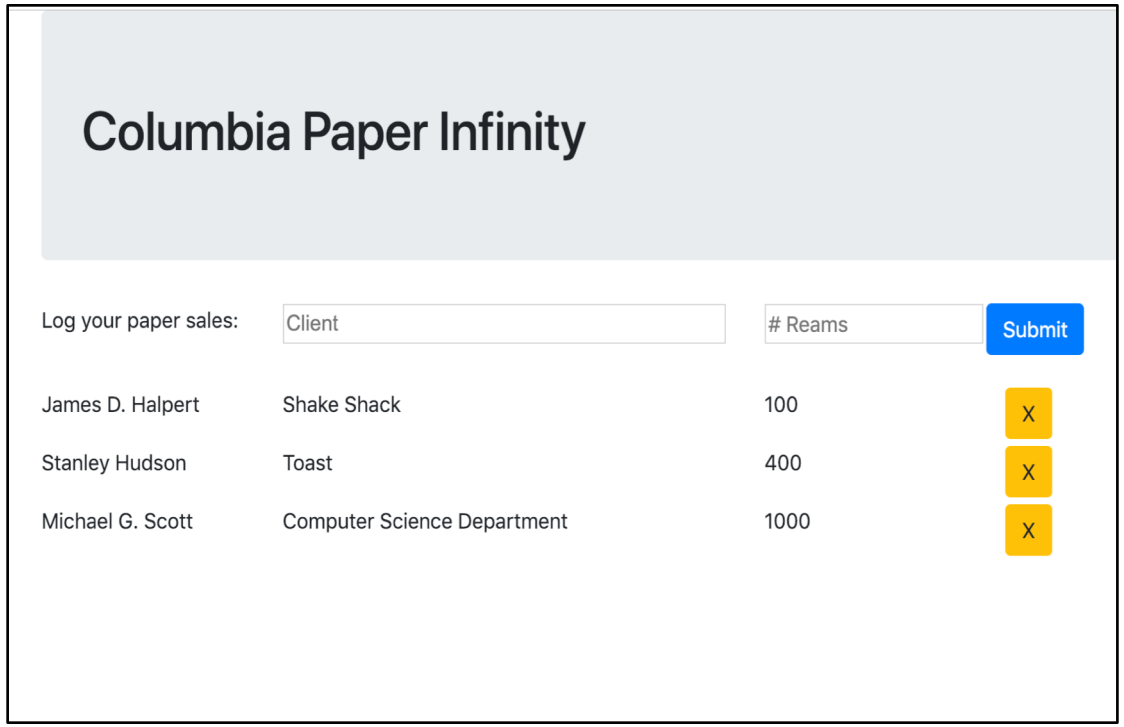


Drag and Drop

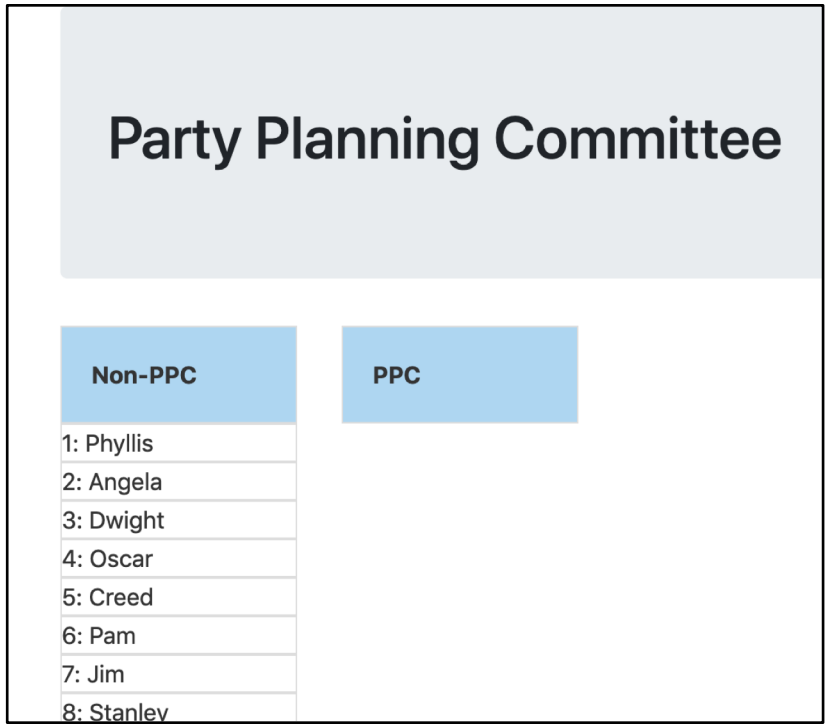


Added customization
(hovering and drop target feedback)

You allowed users to interact with data



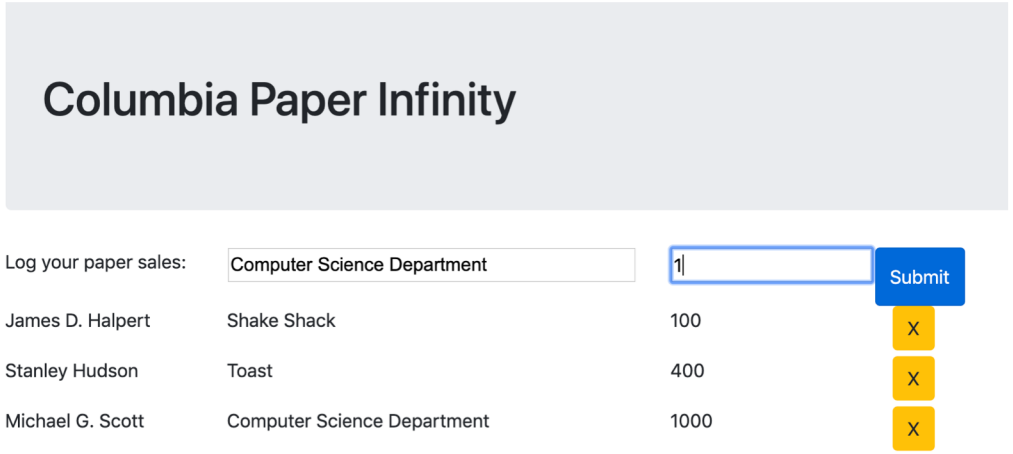
Create / Delete data



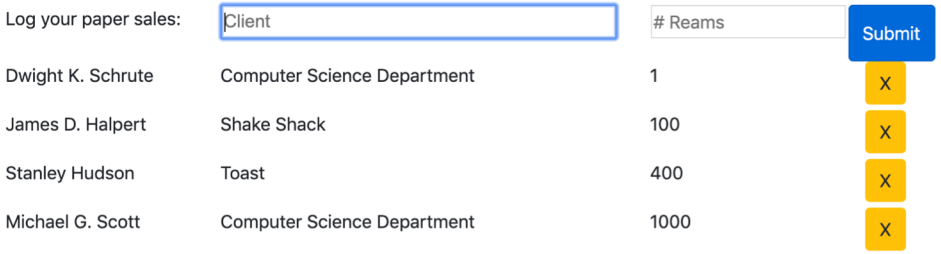
Update data

But there’s a big problem:

Add data

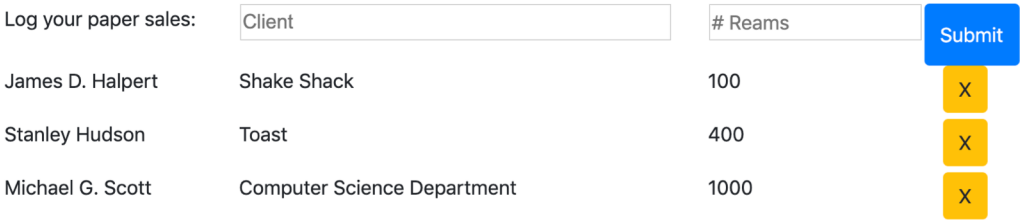


Data appears



REFRESH PAGE

Data is gone!

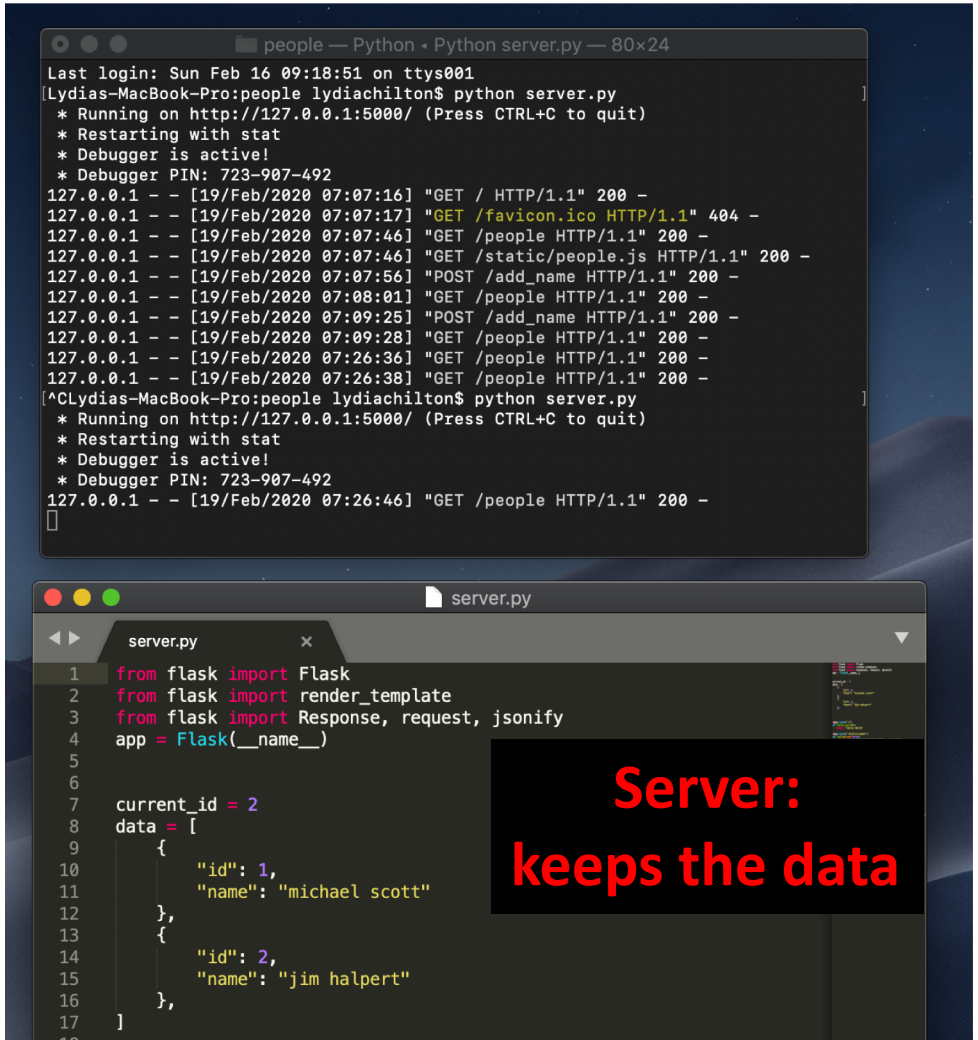


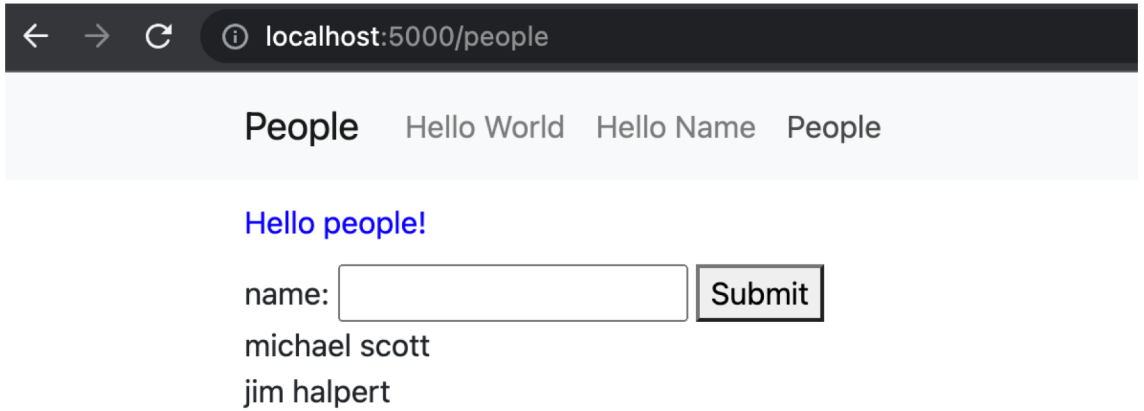
The data doesn’t save

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>

```

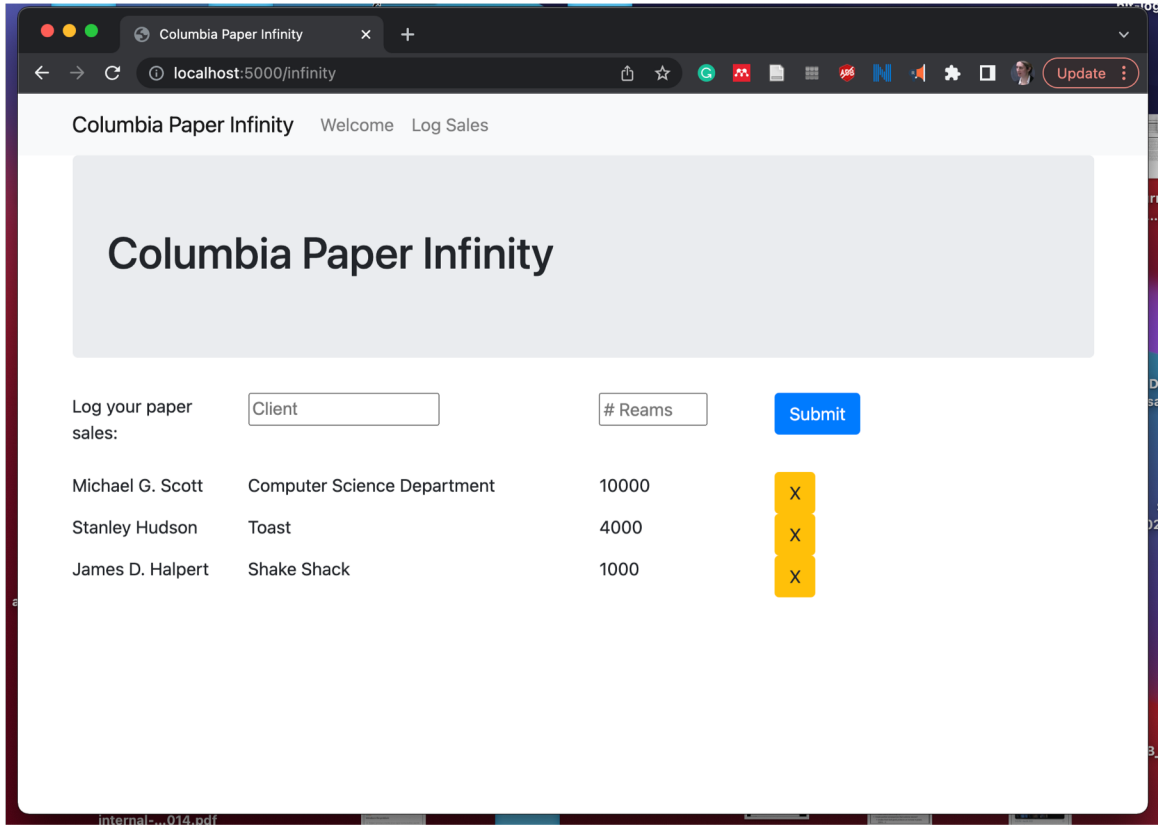
Solution: Store data on the server, display and edit data on the client.





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

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



Tip: start by copying the people folder and editing it