

Saving Data on the Backend

No screens



Prof. Lydia Chilton
COMS 4170
27 February 2019

Say your name



Review

Homework 5

Dynamic JavaScript + UI Widgets

Logging Paper Sales

Columbia Paper Infinity

Log your paper sales:

Client	# Reams	
James D. Halpert	Shake Shack	100 <input type="button" value="X"/>
Stanley Hudson	Toast	400 <input type="button" value="X"/>
Michael G. Scott	Computer Science Department	1000 <input type="button" value="X"/>

```
36 <body>
37 <div class="container">
38   <div class="jumbotron">
39     <h1>Columbia Paper Infinity</h1>
40   </div>
41   <div id="logsales" >
42     <div class="row">
43       <div class="col-md-2">
44         Log your paper sales:
45       </div>
46       <div class="col-md-4">
47         <div class="ui-widget">
48           <input type="text" id="enter_client" placeholder="Client" >
49         </div>
50       </div>
51       <div class="col-md-2">
52         <input type="text" id="enter_reams" placeholder="# Reams" >
53       </div>
54       <button class="btn btn-primary" id="submit_sale">Submit</button>
55     </div>
56     <br>
57     <div id="sales">
58     </div>
59   </div>
60 </div>
61 </body>
62
63
64
65
```

Autocomplete Client names

Log your paper sales:

Toast
Flat Top

```
36 <body>
37 <div class="container">
38   <div class="jumbotron">
39     <h1>Columbia Paper Infinity</h1>
40   </div>
41   <div id="logsales" >
42     <div class="row">
43       <div class="col-md-2">
44         Log your paper sales:
45       </div>
46       <div class="col-md-4">
47         <div class="ui-widget">
48           <input type="text" id="enter_client" placeholder="Client" >
49         </div>
50       </div>
51       <div class="col-md-2">
52         <input type="text" id="enter_reams" placeholder="# Reams" >
53       </div>
54       <button class="btn btn-primary" id="submit_sale">Submit</button>
55     </div>
56   </div>
57   <br>
58   <div id="sales">
59   </div>
60 </div>
61 </div>
62 </div>
63 </body>
64 </body>
65
```

1_log_sales.html

```
20
21 var clients = [
22   "Shake Shack",
23   "Toast",
24   "Computer Science Department",
25   "Teacher's College",
26   "Starbucks",
27   "Subsconscious",
28   "Flat Top",
29   "Joe's Coffee",
30   "Max Caffe",
31   "Nussbaum & Wu",
32   "Taco Bell",
33 ];
34
35
36
37
38 $(document).ready(function(){
39   $( "#enter_client" ).autocomplete({
40     source: clients
41   });
42
43
```

1_log_sales.js

Load sales from data

Columbia Paper Infinity

Log your paper sales:

James D. Halpert

Shake Shack

100

Stanley Hudson

Toast

400

Michael G. Scott

Computer Science Department

1000

```
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
  }
]
```

```
66 var display_sales_list = function(sales){
67   $("#sales").empty()
68
69   if(sales.length == 0){
70     var row = $("<div class='row'>")
71     var col_client = $("<div class='col-md-4'>")
72     $(col_client).append("No Sales")
73     $("#sales").append(row)
74   }else {
75     $.each(sales, function(i, sale){
76
77       var row = $("<div class='row bottom_row_padding'>")
78       var col_salesperson = $("<div class='col-md-2'>")
79       $(col_salesperson).append(sale["salesperson"])
80       $(row).append(col_salesperson)
81
82       var col_client = $("<div class='col-md-4'>")
83       $(col_client).append(sale["client"])
84       $(row).append(col_client)
85
86       var col_reams = $("<div class='col-md-2'>")
87       $(col_reams).append(sale["reams"])
88       $(row).append(col_reams)
89
90       var col_salesperson = $("<div class='col-md-3'>")
91       var delete_button = $("<button class='btn btn-warning'>X</button>")
92       $(delete_button).click(function(){
93         //alert(i)
94         sales.splice(i, 1)
95         display_sales_list(sales)
96       })
97
98       $(col_salesperson).append(delete_button)
99       $(row).append(col_salesperson)
100
101       $("#sales").append(row)
102     })
103   }
104 }
105
106
107
108 }
```

Enter sale part 1 – Validate new sale data

Columbia Paper Infinity

Log your paper sales:

<input type="text" value="Toast"/>	<input type="text" value="6000"/>	<input type="button" value="Submit"/>
James D. Halpert	Shake Shack	100 <input type="button" value="X"/>
Stanley Hudson	Toast	400 <input type="button" value="X"/>
Michael G. Scott	Computer Science Department	1000 <input type="button" value="X"/>

```
38 $(document).ready(function(){
39
40     $("#submit_sale").click(function(){
41         submitSale()
42     })
43
44     $("#enter_reams").keypress(function(e){
45         if(e.which == 13) {
46             submitSale()
47         }
48     })
49
```

```
var submitSale = function(){
    var client = $("#enter_client").val()
    var reams = $.trim( $("#enter_reams").val() )

    if($.trim(client) == ""){
        alert("Hey! The client can't be empty!")
        $("#enter_client").val("")
        $("#enter_client").focus()
    }else if (reams == ""){
        alert("Hey! The # reams can't be empty!")
        $("#enter_reams").val("")
        $("#enter_reams").focus()
    }else if (!$.isNumeric(reams)){ //$.isNumeric( "-10" )
        alert("Hey! The # reams had to be a number!")
        //$("#enter_reams").val("")
        $("#enter_reams").focus()
    }else{

        var new_sale = {
            "salesperson": salesperson,
            "client": client,
            "reams": reams
        }

        addSale(new_sale)
    }
}
```

Enter sale part 2 – update data (Model + ViewController Style)

Columbia Paper Infinity

Log your paper sales:

Dwight K. Schrute	Toast	6000	<input type="button" value="X"/>
James D. Halpert	Shake Shack	100	<input type="button" value="X"/>
Stanley Hudson	Toast	400	<input type="button" value="X"/>
Michael G. Scott	Computer Science Department	1000	<input type="button" value="X"/>

```
2
3 var sales = [
4   {
5     "salesperson": "James D. Halpert",
6     "client": "Shake Shack",
7     "reams": 1
8   },
9   {
10    "salespers
11    "client":
12    "reams": 4
13  },
14  {
15    "salespers
16    "client":
17    "reams": 1
18  },
19 ]
20
21 var clients = [
22   "Shake Shack",
23   "Toast",
24   "Computer Scie
25   "Teacher's Col
26   "Starbucks",
27   "Subconscious"
28   "Flat Top",
29   "Joe's Coffee"
30   "Max Caffe",
31   "Nussbaum & Wu
32   "Taco Bell",
33 ];
34
35
154
155 //reset focus and values to
156 // "entering data state"
157 $("#enter_client").val("")
158 $("#enter_reams").val("")
159 $("#enter_client").focus()
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

Deleting Records (Model + ViewController Style)

```
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  
  },  
]
```

Columbia Paper Infinity

Log your paper sales:

James D. Halpert

Shake Shack

100

Stanley Hudson

Toast

400

```
108  
109 var display_sales_list = function(sales){  
110     $("#sales").empty()  
111  
112     $.each(sales, function(i, sale){  
113         //var row = $("<div class='row bottom_row_padding'>")  
114         //CREATE UI HERE  
115  
116         var delete_button = $("<button class='btn btn-warning'>X</button>")  
117         $(delete_button).click(function(){  
118             sales.splice(i, 1)  
119             display_sales_list(sales)  
120         })  
121     })  
122  
123     //ATTACH UI STUFF HERE  
124     //$("#sales").append(row)  
125  
126     })  
127  
128  
129  
130  
131 }
```

Model + ViewController

update the data, Re-render the display

Columbia Paper Infinity

Log your paper sales: # Reams

James D. Halpert	Shake Shack	100	<input type="button" value="X"/>
Stanley Hudson	Toast	400	<input type="button" value="X"/>
Michael G. Scott	Computer Science Department	1000	<input type="button" value="X"/>

```
139
140 var addSale = function(new_sale){
141     //Update the data
142     // prepend new sale to the array of sales
143     sales.unshift(new_sale)
144
145     //redisplay ALL DATA
146     display_sales_list(sales)
147
148     //if it's a new client, add to clients
149     client = new_sale["client"]
150     if( $.inArray(client, clients) < 0){
151         clients.push(client)
152     }
153
154     //reset focus and values to
155     //"entering data state"
156     $("#enter_client").val("")
157     $("#enter_reams").val("")
158     $("#enter_client").focus()
159
160 }
161
```

Add sale

```
108
109 var display_sales_list = function(sales){
110     $("#sales").empty()
111
112     $.each(sales, function(i, sale){
113         //var row = $("<div class='row bottom_row_padding'>")
114         //CREATE UI HERE
115
116         var delete_button = $("<button class='btn btn-warning'>X</button>")
117         $(delete_button).click(function(){
118
119             sales.splice(i, 1)
120
121             display_sales_list(sales)
122
123         })
124     })
125
126     //ATTACH UI STUFF HERE
127     $("#sales").append(row)
128
129
130
131 }
```

Delete sale

Drag and Drop the PPC members

Party Planning Committee

Non-PPC	PPC
1: Phyllis	
2: Angela	
3: Dwight	
4: Oscar	
5: Creed	
6: Pam	
7: Jim	
8: Stanley	
9: Michael	
10: Kevin	
11: Kelly	

```
53
54 <body>
55 <div class="container">
56   <div class="jumbotron">
57     <h1>Party Planning Committee</h1>
58   </div>
59
60
61   <div class="row">
62     <div class="col-md-12">
63
64       <div class="row" >
65         <div class="col-md-2" >
66           <div id="non_ppc_label" class="list_label ui-widget-header" >Non-PPC</div>
67           <div id="non_ppc">
68             </div>
69         </div>
70
71         <div class="col-md-2" >
72           <div id="ppc_label" class="list_label ui-widget-header" >PPC</div>
73           <div id="ppc" >
74             </div>
75         </div>
76       </div>
77     </div>
78   </div>
79 </div>
80 </div>
81
82
83 </div>
84 </body>
85
```

Make the names appear

Party Planning Committee	
Non-PPC	PPC
1: Phyllis	
2: Angela	
3: Dwight	
4: Oscar	
5: Creed	
6: Pam	
7: Jim	
8: Stanley	
9: Michael	
10: Kevin	
11: Kelly	

```
var non_ppc = [  
  "Phyllis",  
  "Angela",  
  "Dwight",  
  "Oscar",  
  "Creed",  
  "Pam",  
  "Jim",  
  "Stanley",  
  "Michael",  
  "Kevin",  
  "Kelly"  
]  
  
var ppc = [  
  "Phyllis",  
  "Angela",  
  "Dwight",  
  "Oscar",  
  "Creed",  
  "Pam",  
  "Jim",  
  "Stanley",  
  "Michael",  
  "Kevin",  
  "Kelly"  
]  
  
19  
20 function makeListItemWrapper(text, value, list){  
21   return "<div class='list_text ui-widget-content "+list+"' data-value="+value  
22 }  
23  
24 function make_non_ppc(non_ppc){  
25   $("#non_ppc").empty()  
26  
27   $.each(non_ppc, function( index, value ) {  
28     var text = (index+1) + ": " + value  
29     var text_wrapper = makeListItemWrapper(text, value, "non_ppc_list")  
30     $("#non_ppc").append(text_wrapper)  
31  
32     $(".non_ppc_list").draggable({  
33       revert: function (droppableObj) {  
34         //if false then no object drop occurred.  
35         if (droppableObj === false) {  
36           return true;  
37         } else {  
38           return false;  
39         }  
40       }  
41     });  
42   });  
43  
44 });  
45 }  
46
```

Make the list headers drop targets

Party Planning Committee

Non-PPC	PPC
1: Phyllis	
2: Angela	
3: Dwight	
4: Oscar	
5: Creed	
6: Pam	
7: Jim	
8: Stanley	
9: Michael	
10: Kevin	
11: Kelly	

```
<style>
  .highlightDRAGGING.highlightHOVER{
    background-color: darkblue;
  }

  .highlightDRAGGING{
    background-color: blue ;
  }
</style>
```

```
76
77 $(document).ready(function(){
78   make_ppc(ppc)
79
80   $("#ppc_label").droppable({
81     accept: ".non_ppc_list",
82     //highlightClass: "ui-state-active",
83     classes: {
84       "ui-droppable-active": "highlightDRAGGING",
85       "ui-droppable-hover": "highlightHOVER"
86     },
87     drop: function( event, ui ) {
88       var name_dropped = $(ui.draggable[0]).data("value")
89
90       ppc.push(name_dropped)
91       non_ppc.splice( $.inArray(name_dropped, non_ppc), 1 );
92
93       make_non_ppc(non_ppc)
94       make_ppc(ppc)
95
96     }
97   });
98
99
```

Drop a name (Model + ViewController Style) update the data, Re-render the display

Party Planning Committee

Non-PPC	PPC
1: Phyllis	
2: Angela	
3: Dwight	
4: Oscar	
5: Creed	
6: Pam	
7: Jim	
8: Stanley	
9: Michael	
10: Kevin	
11: Kelly	

```
76
77 $(document).ready(function(){
78     make_ppc(ppc)
79
80     $("#ppc_label").droppable({
81         accept: ".non_ppc_list",
82         //hoverClass: "ui-state-active",
83         classes: {
84             "ui-droppable-active": "highlightDRAGGING",
85             "ui-droppable-hover": "highlightHOVER"
86         },
87         drop: function( event, ui ) {
88             var name_dropped = $(ui.draggable[0]).data("value")
89
90             ppc.push(name_dropped)
91             non_ppc.splice( $.inArray(name_dropped, non_ppc), 1 );
92
93
94             make_non_ppc(non_ppc)
95             make_ppc(ppc)
96
97         }
98     });
99
```

Summary

Dynamically create widgets

Buttons



Autocomplete

Log your paper sales:

- Toast
- Flat Top

Drag and Drop

Non-PPC	PPC
1: Phyllis	
2: Angela	
3: Dwight	
4: Oscar	
5: Creed	
6: Pam	
7: Jim	
8: Stanley	
9: Michael	
10: Kevin	
11: Kelly	

Added minor customization
(hovering and drop target feedback)

Interacting with Data

Columbia Paper Infinity

Log your paper sales:

<input type="text" value="Client"/>	<input type="text" value="# Reams"/>	<input type="button" value="Submit"/>
James D. Halpert	Shake Shack	100
Stanley Hudson	Toast	400
Michael G. Scott	Computer Science Department	1000

Each row in the table has a yellow button with an 'X' icon to its right, indicating a delete action.

Create / Delete data

Party Planning Committee

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

Update data

Model + ViewController

update the data, Re-render the display

```
var addSale = function(new_sale){
  //Update the data
  // prepend new sale to the array of sales
  sales.unshift(new_sale)
  //redisplay ALL DATA
  display_sales_list(sales)
  //if it's a new client, add to clients
  client = new_sale["client"]
  if( $.inArray(client, clients) < 0){
    clients.push(client)
  }
  //reset focus and values to
  //"entering data state"
  $("#enter_client").val("")
  $("#enter_reams").val("")
  $("#enter_client").focus()
}
```

Add data

```
var display_sales_list = function(sales){
  $("#sales").empty()
  $.each(sales, function(i, sale){
    //var row = $("<div class='row bottom_r
    //CREATE UI HERE

    var delete_button = $("<button class='b
    $(delete_button).click(function(){
      sales.splice(i, 1)
      display_sales_list(sales)
    })
  })
  //ATTACH UI STUFF HERE
  //$("#sales").append(row)
}
```

Delete data

```
$( "#ppc_label" ).droppable({
  accept: ".non_ppc_list",
  //hoverClass: "ui-state-active",
  classes: {
    "ui-droppable-active": "highlightDRAGGING",
    "ui-droppable-hover": "highlightHOVER"
  },
  drop: function( event, ui ) {
    var name_dropped = $(ui.draggable[0]).d
    ppc.push(name_dropped)
    non_ppc.splice( $.inArray(name_dropped,
    make_non_ppc(non_ppc)
    make_ppc(ppc)
  }
});
```

Update data

Problem:

The data doesn't save

Saving Data on the Backend

No screens

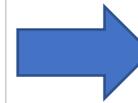
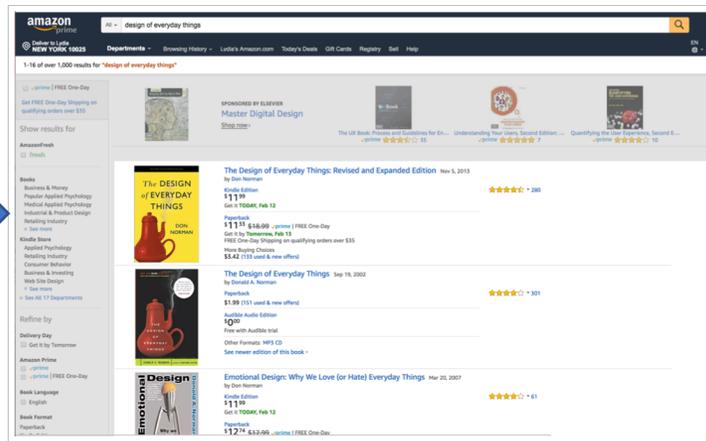
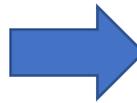
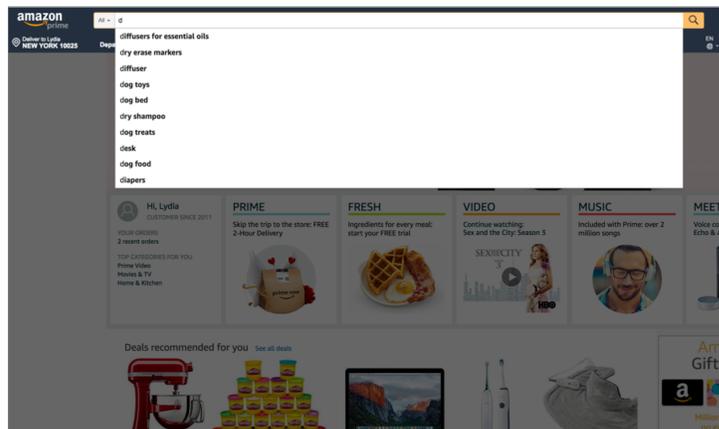


Prof. Lydia Chilton
COMS 4170
27 February 2019

Say your name

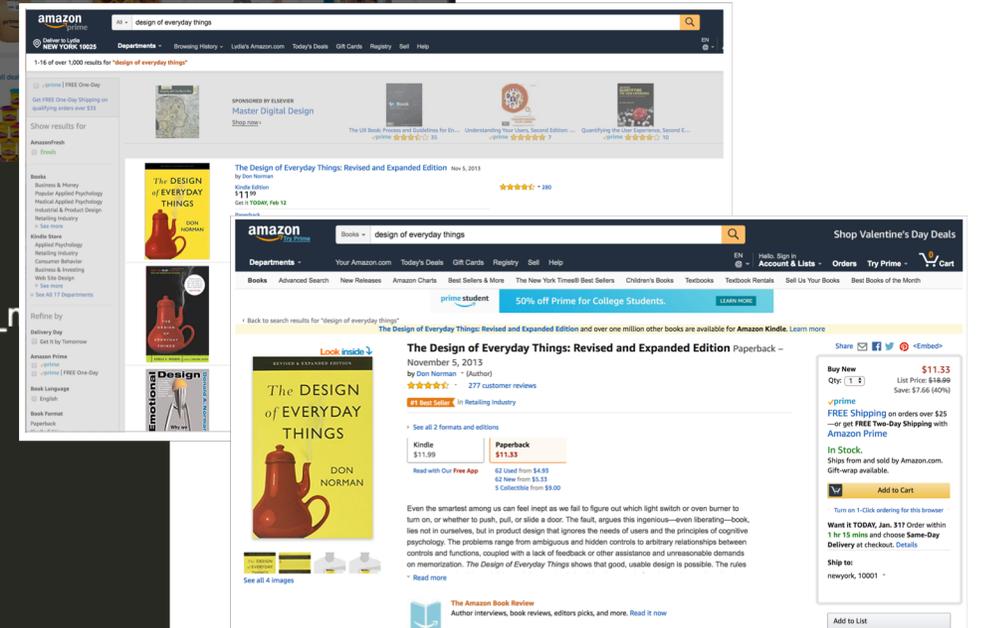
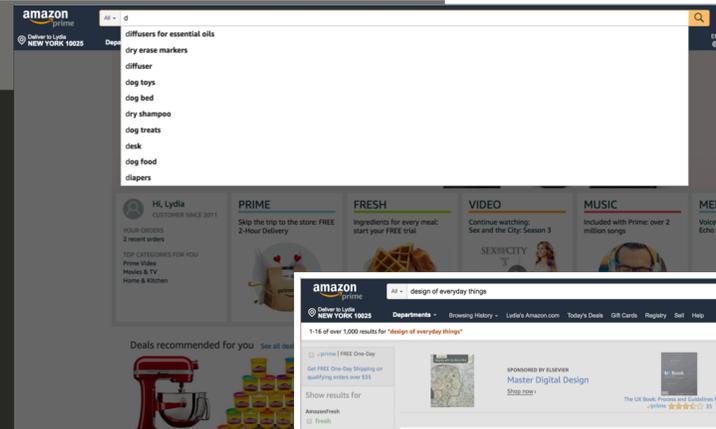


Site use multiple pages. One page per state / task.



To create a website with multiple pages we need a server on the backend.

```
amazon_server.py  
1 from flask import Flask  
2 from flask import render_template  
3 app = Flask(__name__)  
4  
5  
6  
7 @app.route('/')  
8 def home():  
9     return render_template('home.html')  
10  
11 @app.route('/product_results/<product_name>')  
12 def product_search_results(product_name=None):  
13     return render_template('product_search_results.html', data = product_name)  
14  
15 @app.route('/checkout/<product_id>')  
16 def checkout(product_id=None):  
17     return render_template('checkout.html', data = product_id)  
18  
19  
20  
21 if __name__ == '__main__':  
22     app.run()  
23  
24
```



The server also needs to store user data across pages like “what’s in my shopping cart”



```
amazon_server.py
1 from flask import Flask
2 from flask import render_template
3 app = Flask(__name__)
4
5 my_cart = [{id:1, name:"Design of Things"}]
6
7 @app.route('/')
8 def home():
9     return render_template('home.html')
10
11 @app.route('/product_results/<product_name>')
12 def product_search_results(product_name=None):
13     return render_template('product_search_results.html', data = product_name)
14
15 @app.route('/checkout/<product_id>')
16 def checkout(product_id=None):
17     return render_template('checkout.html', data = product_id)
18
19
20
21 if __name__ == '__main__':
22     app.run()
23
24
```

If we store data on the server,
how will we send it to the page?

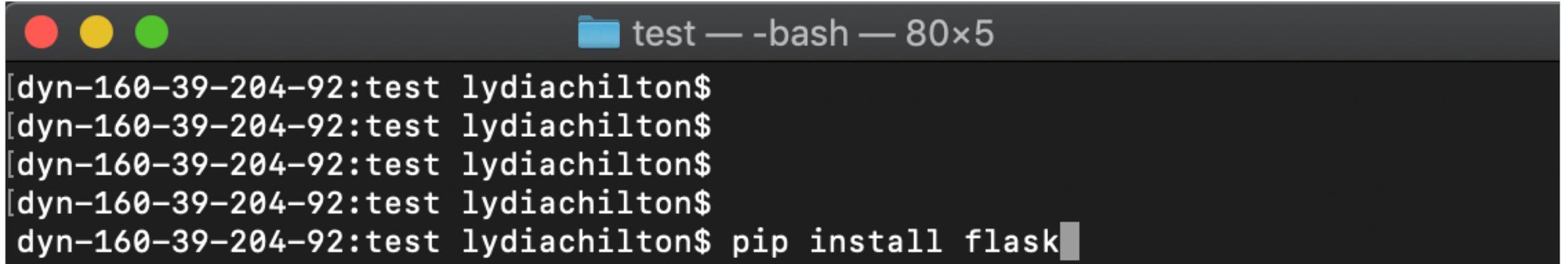


```
amazon_server.py
1 from flask import Flask
2 from flask import render_template
3 app = Flask(__name__)
4
5 my_cart = [{id:1, name:"Design of Things"}]
6
7 @app.route('/')
8 def home():
9     return render_template('home.html')
10
11 @app.route('/product_results/<product_name>')
12 def product_search_results(product_name=None):
13     return render_template('product_search_results.html', data = product_name, cart = my_cart)
14
15 @app.route('/checkout/<product_id>')
16 def checkout(product_id=None):
17     return render_template('checkout.html', data = product_id)
18
19
20
21 if __name__ == '__main__':
22     app.run()
23
24
```

Example application:

Saving data in Flask

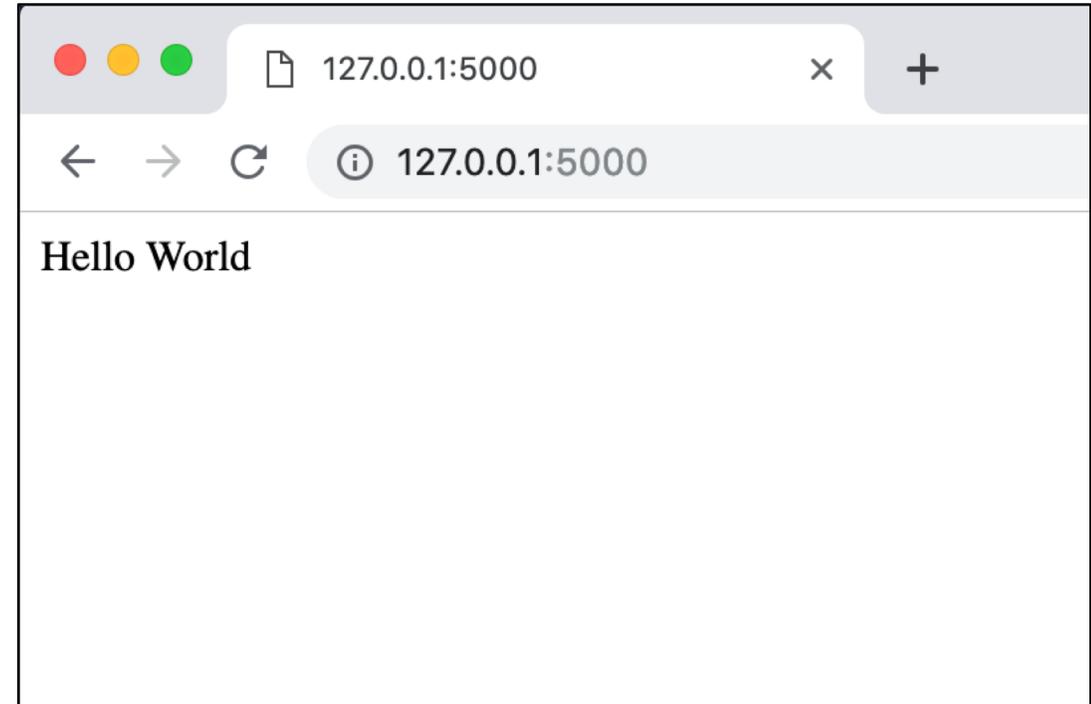
What do you do first?



```
test — -bash — 80x5  
[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
```

You have written the world's smallest Flask app. Now what?

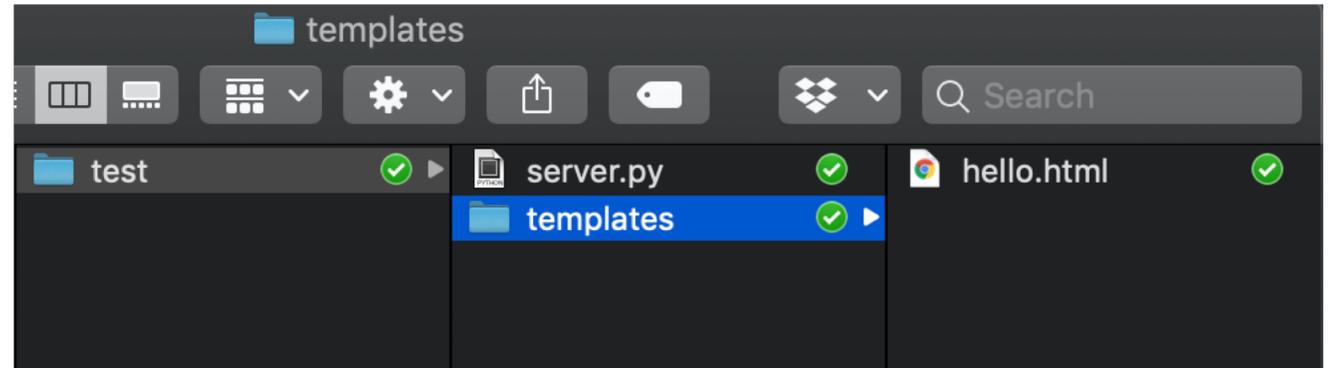
```
server.py
1 from flask import Flask
2 app = Flask(__name__)
3
4 @app.route('/')
5 def hello_world():
6     return 'Hello World'
7
8 if __name__ == '__main__':
9     app.run()
```



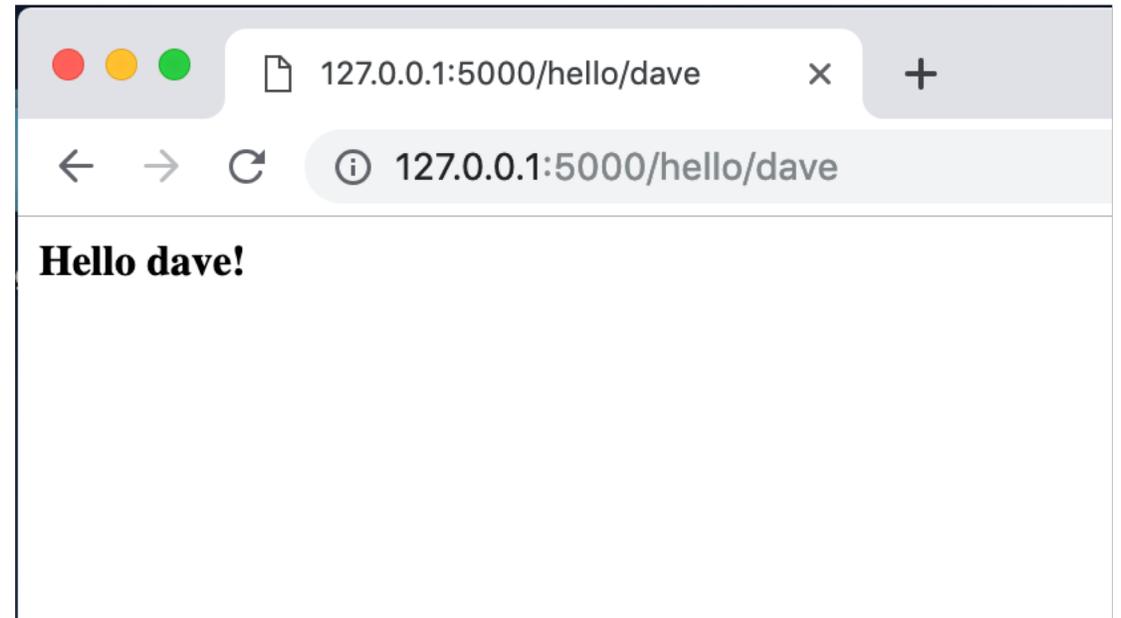
```
people — Python — Python server.py — 77x8
Lydias-MacBook-Pro:people lydiachilton$
Lydias-MacBook-Pro:people lydiachilton$
Lydias-MacBook-Pro:people lydiachilton$ python server.py
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 162-019-624
```

How to render an HTML page with data

```
server.py x hello.html x
1 from flask import Flask
2 from flask import render_template
3 app = Flask(__name__)
4
5 @app.route('/')
6 def hello_world():
7     return 'Hello World'
8
9 @app.route('/hello/<name>')
10 def hello(name=None):
11     return render_template('hello.html', name=name)
12
13 if __name__ == '__main__':
14     app.run()
15
16
```



```
server.py x hello.html x
1 <html>
2 <head></head>
3 <body>
4
5 <b>Hello {{name}}!</b>
6 </body>
7 </html>
8
9
```



How to send an array of data to JavaScript?

```
server.py x hello.html x people.  
1 from flask import Flask  
2 from flask import render_template  
3 app = Flask(__name__)  
4  
5  
6 data = [  
7 {  
8 "id": 1,  
9 "name": "michael scott"  
10 },  
11 {  
12 "id": 2,  
13 "name": "jim halpert"  
14 },  
15 ]  
16  
17  
18  
19 @app.route('/')  
20 def hello_world():  
21 return 'Hello World'  
22  
23 @app.route('/hello/<name>')  
24 def hello(name=None):  
25 return render_template('hello.html', name=name)  
26  
27 @app.route('/people')  
28 def people(name=None):  
29 return render_template('people.html', data=data)  
30  
31 if __name__ == '__main__':  
32 app.run()  
33  
34  
35
```

```
people.html x server  
1 <html>  
2 <head>  
3  
4 <script>  
5 var data = '{{ data }}';  
6 </script>  
7
```

127.0.0.1:5000/people

127.0.0.1:5000/people

Hello people!

Elements Console Sources Network Performance Memory Application Security

top Filter Default levels

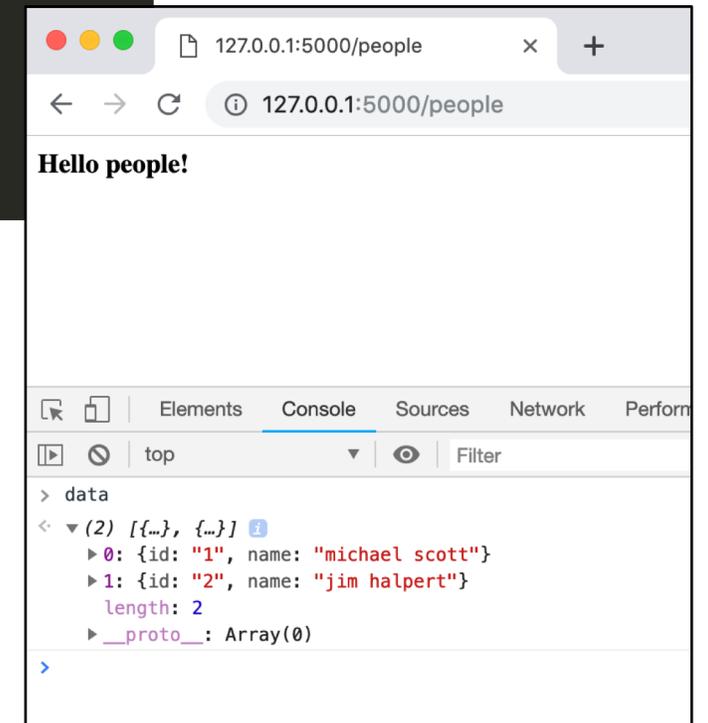
```
> data  
< [{"id": 1, "name": "michael scott"}, {"id": 2, "name": "jim halpert"}]  
> |
```

How to send an array of data to JavaScript?

```
server.py x hello.html x people.  
1 from flask import Flask  
2 from flask import render_template  
3 app = Flask(__name__)  
4  
5  
6 data = [  
7 {  
8     "id": 1,  
9     "name": "michael scott"  
10 },  
11 {  
12     "id": 2,  
13     "name": "jim halpert"  
14 },  
15 ]  
16  
17  
18  
19 @app.route('/')  
20 def hello_world():  
21     return 'Hello World'  
22  
23 @app.route('/hello/<name>')  
24 def hello(name=None):  
25     return render_template('hello.html', name=name)  
26  
27 @app.route('/people')  
28 def people(name=None):  
29     return render_template('people.html', data=data)  
30  
31 if __name__ == '__main__':  
32     app.run()  
33  
34  
35
```

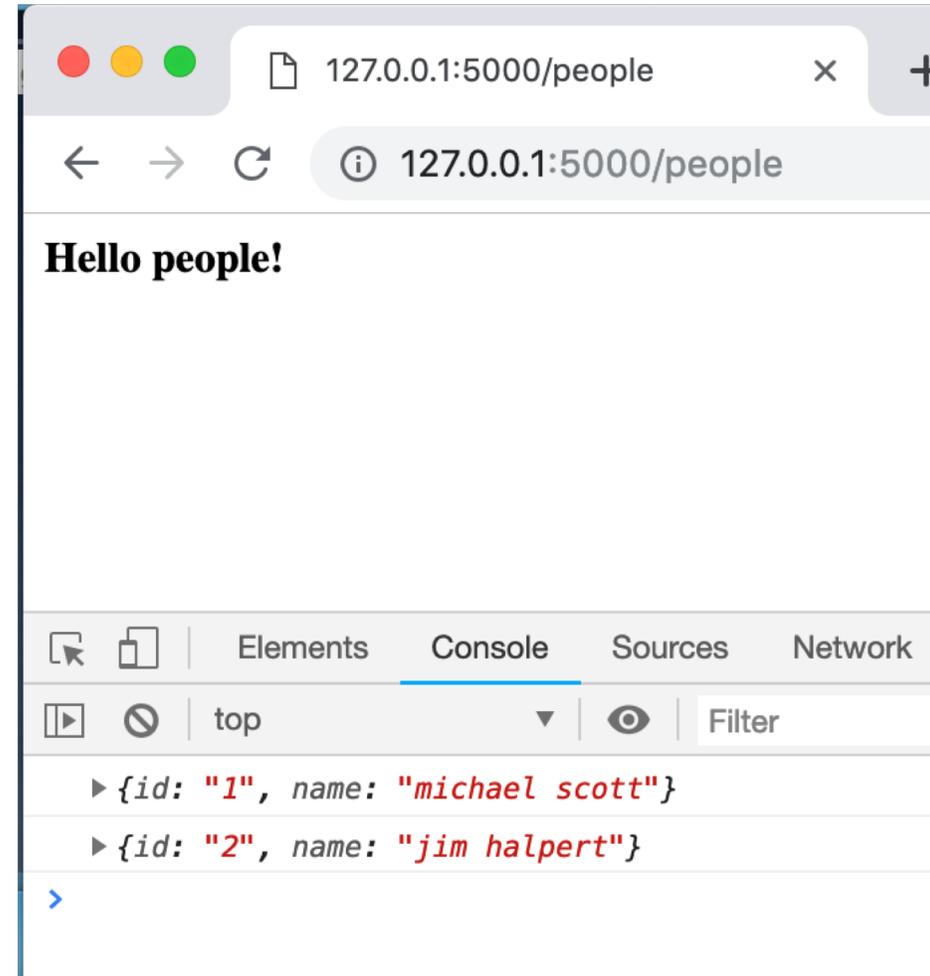
```
people.html x server  
1 <html>  
2 <head>  
3  
4 <script>  
5     var data = '{{ data }}';  
6 </script>  
7  
8 </head>  
9 <body>  
10  
11 <b>Hello people!</b>  
12 </body>  
13 </html>  
14
```

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



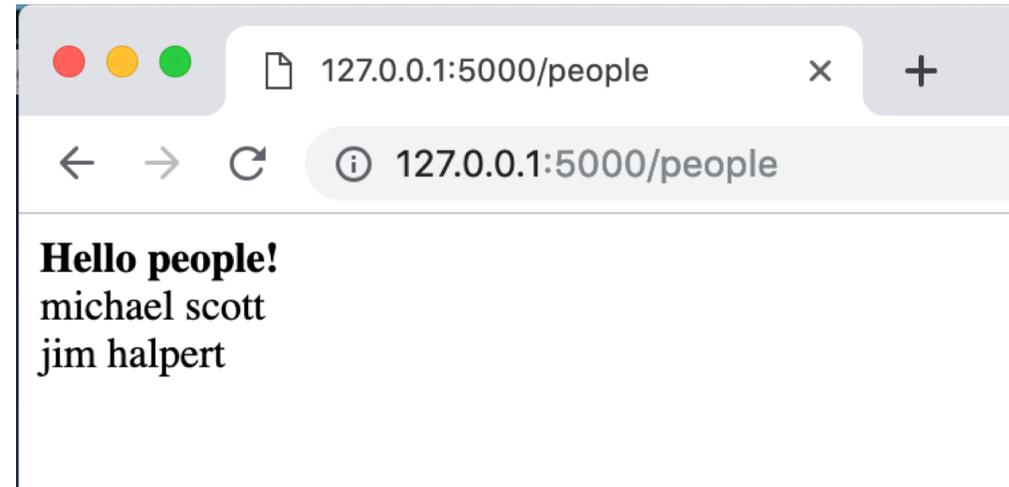
Iterate over the data

```
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        console.log(datum)
12      })
13
14    })
15  </script>
16
17 </head>
18 <body>
19
20
21 <b>Hello people!</b>
22 <div id="people_container">
23 </div>
24
25 </body>
26 </html>
```



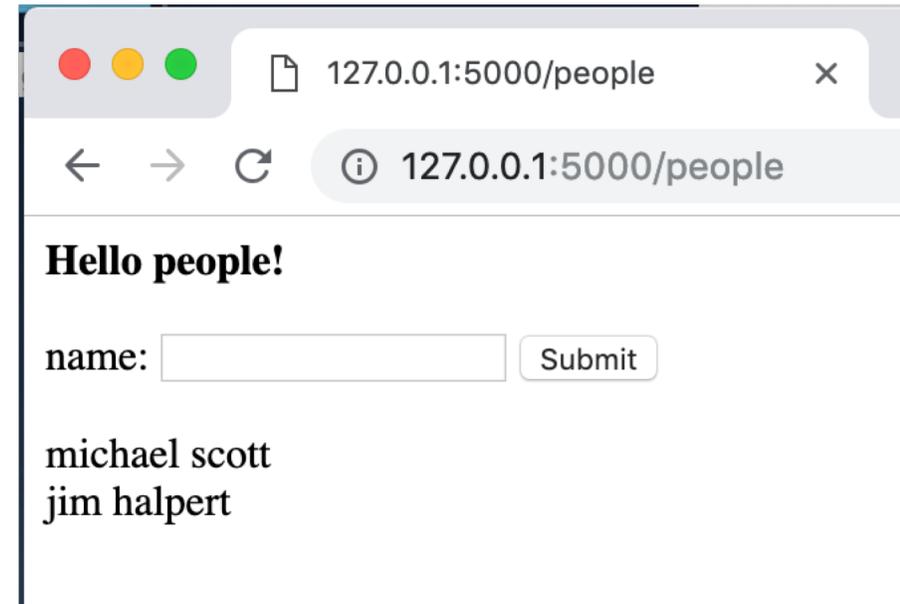
Display all the names

```
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    })
16
17  </script>
18
19 </head>
20 <body>
21
22 <b>Hello people!</b>
23 <div id="people_container">
24 </div>
25
26 </body>
27 </html>
28
29
```



How do we allow people to add names?

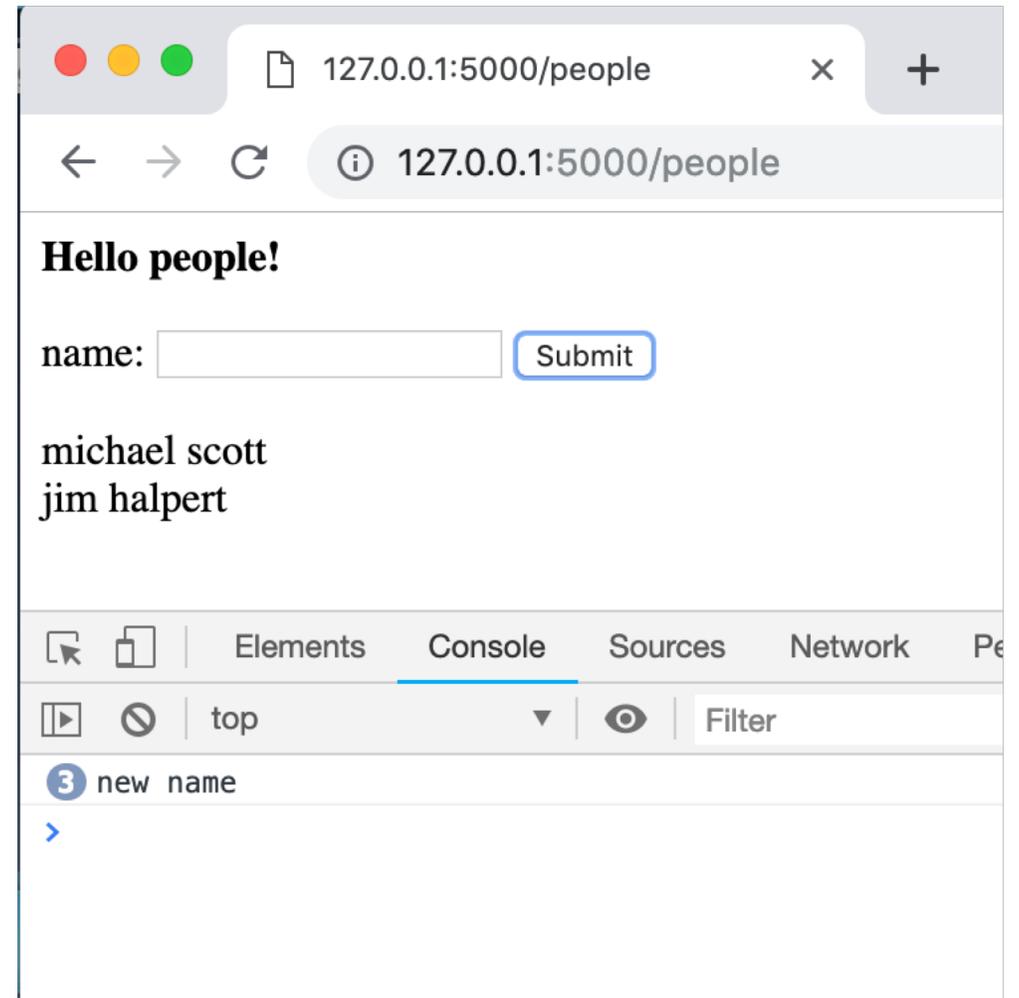
```
<b>Hello people!</b>
<br>
<br>
name: <input id="new_name"></input> <button id="submit_name">Submit</button>
<br>
<br>
<div id="people_container">
</div>
```



What does the click handler do?

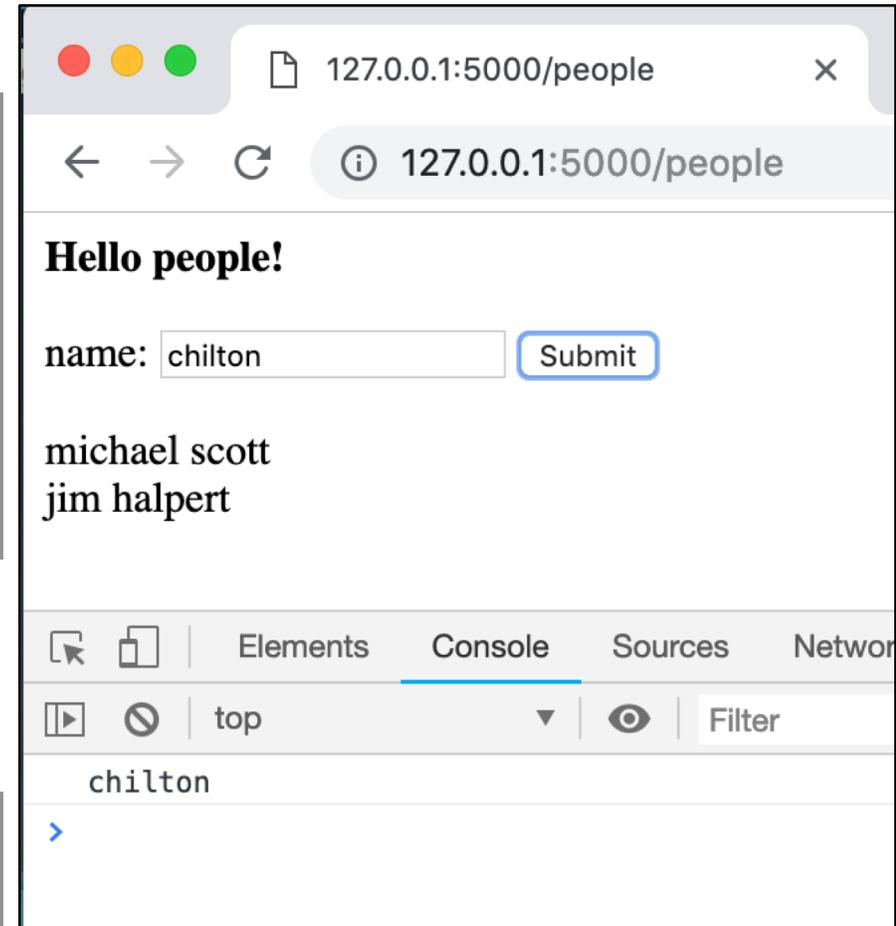
```
people.html
server.py
hello.html

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        console.log("new name")
17      })
18    })
19  }
20
21 </script>
22
23 </head>
24 <body>
25
26 <b>Hello people!</b>
27 <br>
28 <br>
29
30 name: <input id="new_name"></input> <button id="submit_name">Submit</button>
31 <br>
32 </body>
33 </html>
```



Now what does the click handler do?

```
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  </script>
23
24
25
```



This is MVC updating. It will add names to the list.
Will it save? **NO. What line needs to change?**

```
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       //when the page loads, display all the names
10      displayNames(data)
11
12      $("#submit_name").click(function(){
13        var name = $("#new_name").val()
14        console.log(name)
15
16        var new_id = data.length + 1
17        var new_name = name
18        var new_data = {
19          "id": new_id,
20          "name": new_name
21        }
22        data.push(new_data)
23        displayNames(data)
24      })
25    }
26  </script>
```

127.0.0.1:5000/people

127.0.0.1:5000/people

Hello people!

name:

michael scott
jim halpert
chilton

Elements Console Sources Network

top Filter

chilton

> data

< (3) [{...}, {...}, {...}]

- ▶ 0: {id: 1, name: "michael scott"}
- ▶ 1: {id: 2, name: "jim halpert"}
- ▶ 2: {id: 3, name: "chilton"}

length: 3

▶ __proto__: Array(0)

Save the data to the server with AJAX

```
people.html x server.py x hello.html x
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     //when the page loads, display all the names
10    displayNames(data)
11
12    $("#submit_name").click(function(){
13      var name = $("#new_name").val()
14      console.log(name)
15
16      ??????
17
18
19
20
21
22
23
24
25
```

Save the data to the server with AJAX

```
people.html x server.py x hello.html x
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     //when the page loads, display all the names
10    displayNames(data)
11
12    $("#submit_name").click(function(){
13      var name = $("#new_name").val()
14      console.log(name)
15
16      saveName(name)
17
18
19
20
21
22
23
24
25
```

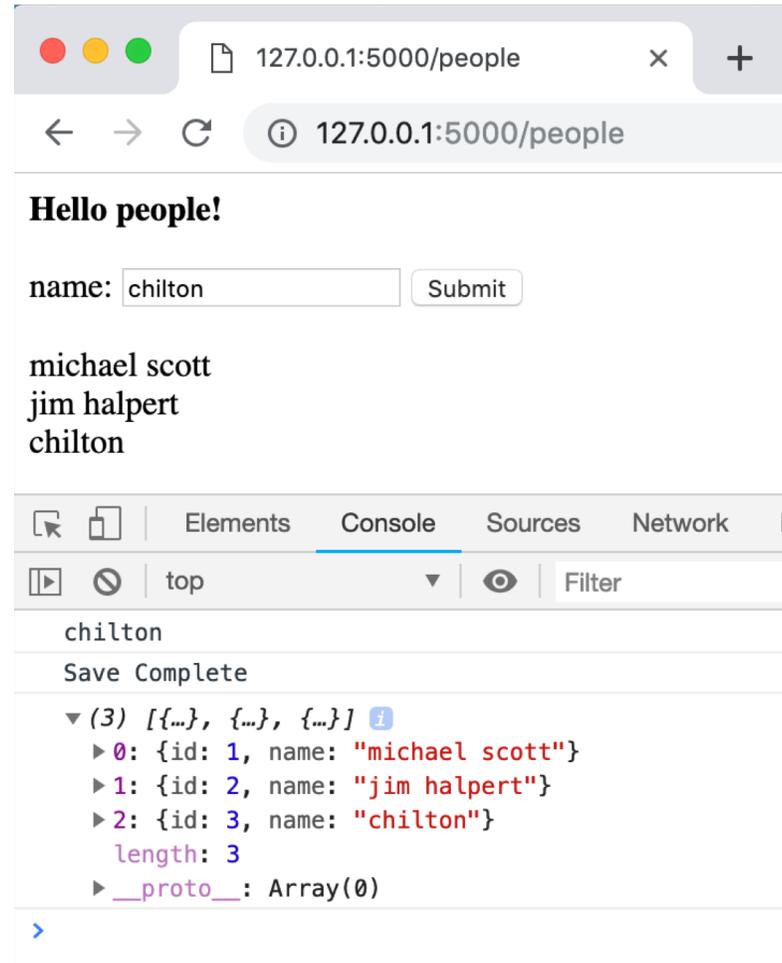
```
var saveName = function(name){
  var data_to_save = {"name": name}
  $.ajax({
    type: "POST",
    url: "add_name",
    dataType: "json",
    contentType: "application/json; charset=utf-8",
    data: JSON.stringify(data_to_save),
    success: function(result){
      var all_data = result["data"]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

```
server.py hello.html people
1 from flask import Flask
2 from flask import render_template
3 from flask import Response, request, jsonify
4 app = Flask(__name__)
5
6
7 current_id = 2
8 data = [
9     {
10         "id": 1,
11         "name": "michael scott"
12     },
13     {
14         "id": 2,
15         "name": "jim halpert"
16     },
17 ]
18
19
20 @app.route('/people')
21 def people(name=None):
22     return render_template('people.html', data=data)
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
```

the server?

```
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

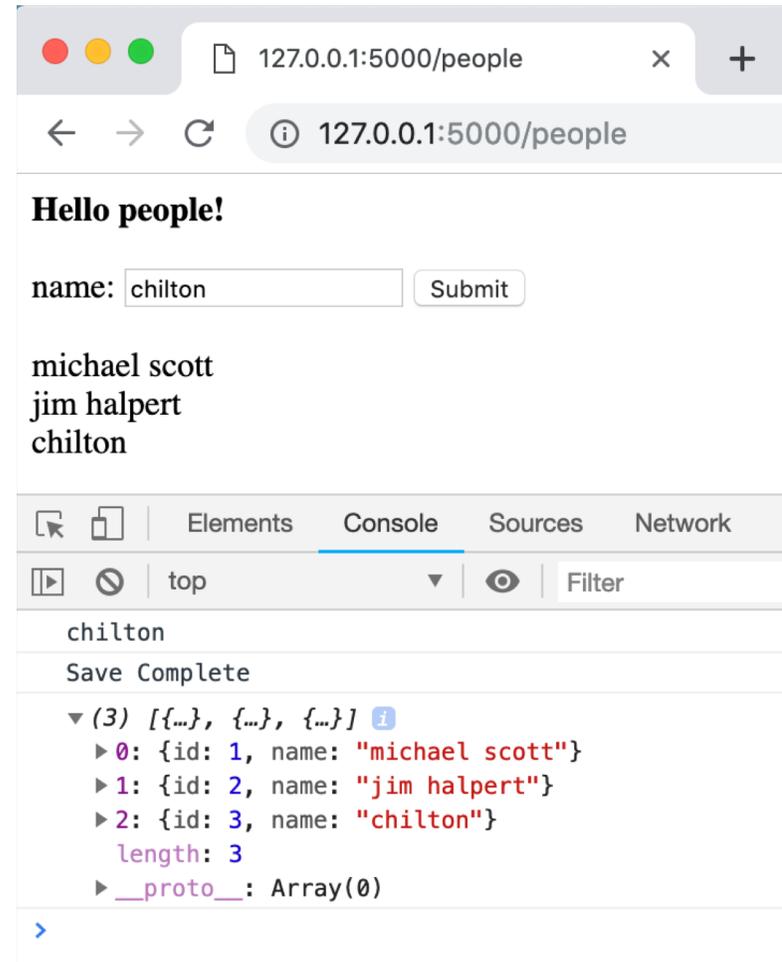
Next Homework

Released today!

Due Next Friday (March 8, 2019) at 4pm

Problem 1. Get the Flask sample code to run

```
server.py
1 from flask import Flask
2 from flask import render_template
3 from flask import Response, request, jsonify
4 app = Flask(__name__)
5
6
7 current_id = 2
8 data = [
9     {
10         "id": 1,
11         "name": "michael scott"
12     },
13     {
14         "id": 2,
15         "name": "jim halpert"
16     },
17 ]
18
19
20 @app.route('/people')
21 def people(name=None):
22     return render_template('people.html', data=data)
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
```



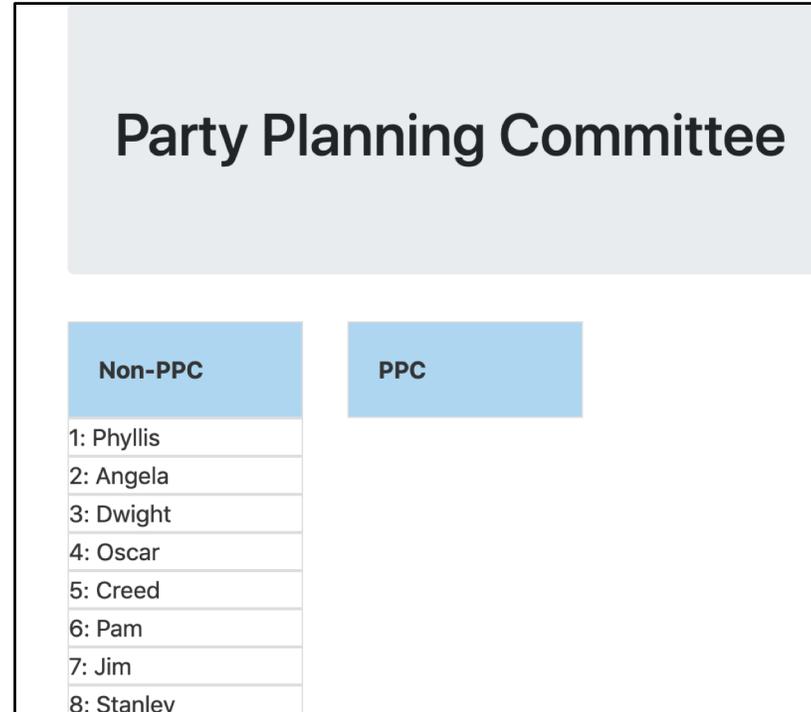
Problem 2. Put a backend behind Log Sales

Columbia Paper Infinity

Log your paper sales:

<input type="text" value="Client"/>	<input type="text" value="# Reams"/>	<input type="submit" value="Submit"/>
James D. Halpert	Shake Shack	100 <input type="button" value="X"/>
Stanley Hudson	Toast	400 <input type="button" value="X"/>
Michael G. Scott	Computer Science Department	1000 <input type="button" value="X"/>

Problem 3. Put a backend behind PPC



Due Friday March 8th:

Problem 1.
Get the Flask sample code to run

127.0.0.1:5000/people

127.0.0.1:5000/people

Hello people!

name:

michael scott
jim halpert
chilton

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

Problem 2.
Put a backend behind Log Sales

Columbia Paper Infinity

Log your paper sales: # Reams

James D. Halpert	Shake Shack	100	<input type="button" value="X"/>
Stanley Hudson	Toast	400	<input type="button" value="X"/>
Michael G. Scott	Computer Science Department	1000	<input type="button" value="X"/>

Problem 3.
Put a backend behind PPC

Party Planning Committee

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

Please get this done by **Friday March 1st**.