

PennyFlask/runserver.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # runserver.py
5: # Author: Bob Dondero
6: #-----
7:
8: import sys
9: import penny
10:
11: def main():
12:
13:     if len(sys.argv) != 2:
14:         print('Usage: ' + sys.argv[0] + ' port', file=sys.stderr)
15:         sys.exit(1)
16:
17:     try:
18:         port = int(sys.argv[1])
19:     except Exception:
20:         print('Port must be an integer.', file=sys.stderr)
21:         sys.exit(1)
22:
23:     try:
24:         penny.app.run(host='0.0.0.0', port=port, debug=True)
25:     except Exception as ex:
26:         print(ex, file=sys.stderr)
27:         sys.exit(1)
28:
29: if __name__ == '__main__':
30:     main()

```

PennyFlask/penny.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # penny.py
5: # Author: Bob Dondero
6: #-----
7:
8: import html # html_code.escape() is used to thwart XSS attacks
9: import flask
10: import common
11: import database
12:
13: #-----
14:
15: app = flask.Flask(__name__)
16:
17: #-----
18:
19: @app.route('/', methods=['GET'])
20: @app.route('/index', methods=['GET'])
21: def index():
22:
23:     html_code = ''
24:     html_code += '<!DOCTYPE html>'
25:     html_code += '<html>'
26:     html_code += '<head>'
27:     html_code += '<title>Penny.com</title>'
28:     html_code += '</head>'
29:     html_code += '<body>'
30:     html_code += common.get_header()
31:     html_code += '<br>'
32:     html_code += '<Click here to <a href="/searchform">begin</a>.<br>'
33:     html_code += '<br>'
34:     html_code += common.get_footer()
35:     html_code += '</body>'
36:     html_code += '</html>'
37:
38:     response = flask.make_response(html_code)
39:     return response
40:
41: #-----
42:
43: @app.route('/searchform', methods=['GET'])
44: def search_form():
45:
46:     prev_author = flask.request.cookies.get('prev_author')
47:     if prev_author is None:
48:         prev_author = '(None)'
49:
50:     html_code = ''
51:     html_code += '<!DOCTYPE html>'
52:     html_code += '<html>'
53:     html_code += '<head>'
54:     html_code += '<title>Penny.com</title>'
55:     html_code += '</head>'
56:     html_code += '<body>'
57:     html_code += common.get_header()
58:     html_code += '<h1>Author Search</h1>'
59:     html_code += '<form action="/searchresults" method="get">'
60:     html_code += 'Please enter an author name:'
61:     html_code += '<input type="text" name="author" autofocus>'
62:     html_code += '<input type="submit" value="Go">'
63:     html_code += '</form>'
64:     html_code += '<br>'
65:     html_code += '<br>'

```

PennyFlask/penny.py (Page 2 of 2)

```

66:     html_code += '<strong>Previous author search:</strong> '
67:     html_code += html.escape(prev_author)
68:     html_code += '<br>'
69:     html_code += '<br>'
70:     html_code += common.get_footer()
71:     html_code += '</body>'
72:     html_code += '</html>'
73:
74:     response = flask.make_response(html_code)
75:     return response
76:
77: #-----
78:
79: @app.route('/searchresults', methods=['GET'])
80: def search_results():
81:
82:     author = flask.request.args.get('author')
83:     if author is None:
84:         author = ''
85:     author = author.strip()
86:
87:     if author == '':
88:         prev_author = '(None)'
89:         books = []
90:     else:
91:         prev_author = author
92:         books = database.get_books(author) # Exception handling omitted
93:
94:     html_code = ''
95:     html_code += '<!DOCTYPE html>'
96:     html_code += '<html>'
97:     html_code += '<head>'
98:     html_code += '<title>Penny.com</title>'
99:     html_code += '</head>'
100:    html_code += '<body>'
101:    html_code += common.get_header()
102:    html_code += '<h1>Author Search Results</h1>'
103:    html_code += '<h2>Books by ' + html.escape(prev_author) + ' :</h2>'
104:
105:    if len(books) == 0:
106:        html_code += '(None)<br>'
107:    else:
108:        pattern = '%s: <strong>%s</strong>: %s<br>'
109:        for book in books:
110:            html_code += pattern % (book['isbn'], book['author'],
111:                                   book['title'])
112:
113:    html_code += '<br>'
114:    html_code += '<br>'
115:    html_code += '<Click here to do another '
116:    html_code += '<a href="/searchform">author search</a>.'
117:    html_code += '<br>'
118:    html_code += '<br>'
119:    html_code += '<br>'
120:    html_code += '<br>'
121:    html_code += common.get_footer()
122:    html_code += '</body>'
123:    html_code += '</html>'
124:
125:    response = flask.make_response(html_code)
126:    response.set_cookie('prev_author', prev_author)
127:    return response

```

blank (Page 1 of 1)

1: This page is intentionally blank.

PennyFlaskJinja/header.html (Page 1 of 1)

```
1: <hr>
2: Good {{ampm}} and welcome to <strong>Penny.com</strong>
3: <hr>
```

PennyFlaskJinja/footer.html (Page 1 of 1)

```
1: <hr>
2: Today is {{current_time}}.<br>
3: Created by <a href="https://www.cs.princeton.edu/~rdonero">
4: Bob Dondero</a>
5: <hr>
```

PennyFlaskJinja/index.html (Page 1 of 1)

```
1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>Penny.com</title>
5:   </head>
6:   <body>
7:     {% include 'header.html' %}
8:     <br>
9:     Click here to <a href="/searchform">begin</a>.<br>
10:    <br>
11:    {% include 'footer.html' %}
12:  </body>
13: </html>
```

PennyFlaskJinja/searchform.html (Page 1 of 1)

```
1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>Penny.com</title>
5:   </head>
6:   <body>
7:     {% include 'header.html' %}
8:     <h1>Author Search</h1>
9:     <form action="/searchresults" method="get">
10:       Please enter an author name:
11:       <input type="text" name="author" autofocus>
12:       <input type="submit" value="Go">
13:     </form>
14:     <br>
15:     <br>
16:     <strong>Previous author search: {{prev_author}}</strong>
17:     <br>
18:     <br>
19:     {% include 'footer.html' %}
20:   </body>
21: </html>
```

PennyFlaskJinja/searchresults.html (Page 1 of 1)

```
1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>Penny.com</title>
5:   </head>
6:   <body>
7:     {% include 'header.html' %}
8:     <h1>Author Search Results</h1>
9:     <h2>Books by {{author}}:</h2>
10:    {% if books|length == 0: %}
11:      (None)
12:    {% else %}
13:      {% for book in books: %}
14:        {{book['isbn']}}:
15:        <strong>{{book['author']}}</strong>:
16:        {{book['title']}}<br>
17:      {% endfor %}
18:    {% endif %}
19:    <br>
20:    <br>
21:    Click here to do another
22:    <a href="/searchform">author search</a>.
23:    <br>
24:    <br>
25:    <br>
26:    <br>
27:    {% include 'footer.html' %}
28:  </body>
29: </html>
```

blank (Page 1 of 1)

```
1: This page is intentionally blank.
```

PennyFlaskJinja/penny.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # penny.py
5: # Author: Bob Dondero
6: #-----
7:
8: import time
9: import flask
10: import database
11:
12: #-----
13:
14: app = flask.Flask(__name__, template_folder='.')
15:
16: #-----
17:
18: def get_ampm():
19:     if time.strftime('%p') == "AM":
20:         return 'morning'
21:     return 'afternoon'
22:
23: def get_current_time():
24:     return time.asctime(time.localtime())
25:
26: #-----
27:
28: @app.route('/', methods=['GET'])
29: @app.route('/index', methods=['GET'])
30: def index():
31:
32:     html_code = flask.render_template('index.html',
33:         ampm=get_ampm(),
34:         current_time=get_current_time())
35:     response = flask.make_response(html_code)
36:     return response
37:
38: #-----
39:
40: @app.route('/searchform', methods=['GET'])
41: def search_form():
42:
43:     prev_author = flask.request.cookies.get('prev_author')
44:     if prev_author is None:
45:         prev_author = '(None)'
46:
47:     html_code = flask.render_template('searchform.html',
48:         ampm=get_ampm(),
49:         current_time=get_current_time(),
50:         prev_author=prev_author)
51:     response = flask.make_response(html_code)
52:     return response
53:
54: #-----
55:
56: @app.route('/searchresults', methods=['GET'])
57: def search_results():
58:
59:     author = flask.request.args.get('author')
60:     if author is None:
61:         author = ''
62:     author = author.strip()
63:
64:     if author == '':
65:         prev_author = '(None)'

```

PennyFlaskJinja/penny.py (Page 2 of 2)

```

66:     books = []
67:     else:
68:         prev_author = author
69:         books = database.get_books(author) # Exception handling omitted
70:
71:     html_code = flask.render_template('searchresults.html',
72:         ampm=get_ampm(),
73:         current_time=get_current_time(),
74:         author=prev_author,
75:         books=books)
76:     response = flask.make_response(html_code)
77:     response.set_cookie('prev_author', prev_author)
78:     return response

```