

PennyViewport/penny.py (Page 1 of 1)

```
1: #!/usr/bin/env python
2:
3: #-----
4: # penny.py
5: # Author: Bob Dondero
6: #-----
7:
8: import json
9: import flask
10: import database
11:
12: #-----
13:
14: app = flask.Flask(__name__)
15:
16: #-----
17:
18: @app.route('/', methods=['GET'])
19: @app.route('/index', methods=['GET'])
20: def index():
21:
22:     return flask.send_file('index.html')
23:
24: #-----
25:
26: @app.route('/searchresults', methods=['GET'])
27: def search_results():
28:
29:     author = flask.request.args.get('author')
30:     if author is None:
31:         author = ''
32:     author = author.strip()
33:
34:     if author == '':
35:         books = []
36:     else:
37:         books = database.get_books(author) # Exception handling omitted
38:
39:     json_doc = json.dumps(books)
40:     response = flask.make_response(json_doc)
41:     response.headers['Content-Type'] = 'application/json'
42:     return response
```

blank (Page 1 of 1)

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

PennyViewport/index.html (Page 1 of 2)

```

1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>Penny.com</title>
5:
6:     <meta name="viewport"
7:       content="width=device-width, initial-scale=1">
8:   </head>
9:   <body>
10:    <hr>
11:    Good <span id="ampmSpan"></span> and welcome to
12:    <strong>Penny.com</strong>
13:    <hr>
14:
15:    <h1>Author Search</h1>
16:    Please enter an author name:
17:    <input type="text" id="authorInput" autoFocus>
18:    <hr>
19:
20:    <div id="resultsDiv"></div>
21:
22:    <hr>
23:    Date and time: <span id="datetimeSpan"></span><br>
24:    Created by <a href="https://www.cs.princeton.edu/~rdondero">
25:    Bob Dondero</a>
26:    <hr>
27:
28:    <script src=
29:      "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
30:    </script>
31:
32:    <script>
33:      'use strict';
34:
35:      function getAmPm() {
36:        let dateTime = new Date();
37:        let hours = dateTime.getHours();
38:        let amPm = (hours < 12) ? 'morning' : 'afternoon';
39:        let amPmSpan = document.getElementById('ampmSpan');
40:        amPmSpan.innerHTML = amPm;
41:      }
42:
43:      function getDateTime() {
44:        let dateTime = new Date();
45:        let datetimeSpan =
46:          document.getElementById('datetimeSpan');
47:        datetimeSpan.innerHTML = dateTime.toLocaleString();
48:      }
49:
50:      function convertToHtml(books) {
51:        let template = `
52:          {{#books}}
53:            {{isbn}}:
54:            <strong>{{author}}</strong>;
55:            {{title}}
56:            <br>
57:          {{/books}}
58:        `;
59:        let map = {books: books};
60:        let html = Mustache.render(template, map);
61:        return html;
62:      }
63:
64:
65:      function handleResponse() {

```

PennyViewport/index.html (Page 2 of 2)

```

66:        if (this.status !== 200) {
67:          alert('Error: Failed to fetch data from server');
68:          return;
69:        }
70:        let books = JSON.parse(this.response);
71:        let html = convertToHtml(books);
72:        let resultsDiv = document.getElementById('resultsDiv');
73:        resultsDiv.innerHTML = html;
74:      }
75:
76:      function handleError() {
77:        alert('Error: Failed to fetch data from server');
78:      }
79:
80:      let request = null;
81:
82:      function getResults() {
83:        let authorInput = document.getElementById('authorInput');
84:        let author = authorInput.value;
85:        let encodedAuthor = encodeURIComponent(author);
86:        let url = '/searchresults?author=' + encodedAuthor;
87:        if (request !== null)
88:          request.abort();
89:        request = new XMLHttpRequest();
90:        request.onload = handleResponse;
91:        request.onerror = handleError;
92:        request.open('GET', url);
93:        request.send();
94:      }
95:
96:      let timer = null;
97:
98:      function debouncedGetResults() {
99:        clearTimeout(timer);
100:        timer = setTimeout(getResults, 500);
101:      }
102:
103:      function setupHeader() {
104:        getAmPm();
105:        let apInterval = window.setInterval(getAmPm, 1000);
106:        window.addEventListener('beforeunload',
107:          function(e) {window.clearInterval(apInterval);})
108:      }
109:
110:      function setupFooter() {
111:        getDateTime();
112:        let dtInterval = window.setInterval(getDateTime, 1000);
113:        window.addEventListener('beforeunload',
114:          function(e) {window.clearInterval(dtInterval);})
115:      }
116:
117:      function setup() {
118:        setupHeader();
119:        setupFooter();
120:        let authorInput = document.getElementById('authorInput');
121:        authorInput.addEventListener('input', debouncedGetResults);
122:      }
123:
124:      document.addEventListener('DOMContentLoaded', setup);
125:
126:    </script>
127:  </body>
128: </html>

```

PennyCss/static/penny.css (Page 1 of 1)

```

1: /*-----*/
2: /* penny.css */
3: /* Author: Bob Dondero */
4: /*-----*/
5:
6: /* Element rules */
7:
8: body {font-family: Arial, Helvetica, sans-serif; font-size: 16px;}
9:
10: input[type="text"] {font-size: 16px;}
11:
12: /*-----*/
13:
14: /* Class rules */
15:
16: .header, .footer {
17:     background-color: #295078; /* Venice blue */
18:     text-align: center;
19:     color: white;
20: }
21:
22: .headerlink, .footerlink {color: white;}
23:
24: .grayborder {
25:     border-style: solid;
26:     border-color: gray;
27:     border-width: 1px;
28: }
29:
30: .grid-container {
31:     display: grid;
32:     grid-template-columns: 1fr 3fr;
33:     gap: 16px;
34: }

```

PennyCss/index.html (Page 1 of 3)

```

1: <!DOCTYPE html>
2: <html>
3:
4:   <head>
5:     <title>Penny.com</title>
6:     <meta name="viewport"
7:       content="width=device-width, initial-scale=1">
8:     <link rel="stylesheet" href="static/penny.css">
9:   </head>
10:
11:   <body>
12:     <div class="header">
13:       <h2>Penny.com</h2>
14:       <h3>Good <span id="ampmSpan"></span> and welcome to
15:         Penny.com</h3>
16:     </div>
17:
18:     <br>
19:
20:     <div class="grid-container">
21:       <div><strong>Author name:</strong></div>
22:       <div><input type="text" id="authorInput" autoFocus</div>
23:       <div><strong>Books:</strong></div>
24:       <div class="grayborder" id="resultsDiv"></div>
25:     </div>
26:
27:     <br>
28:
29:     <div class="footer">
30:       Date and time: <span id="datetimeSpan"></span><br>
31:       Created by <a class="footerlink"
32:         href="https://www.cs.princeton.edu/~rdondero">
33:         Bob Dondero</a>
34:     </div>
35:
36:     <script src=
37:       "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
38:     </script>
39:
40:     <script>
41:       'use strict';
42:
43:       function getAmPm() {
44:         let dateTime = new Date();
45:         let hours = dateTime.getHours();
46:         let amPm = (hours < 12) ? 'morning' : 'afternoon';
47:         let ampMspan = document.getElementById('ampmSpan');
48:         ampMspan.innerHTML = amPm;
49:       }
50:
51:       function getDateTime() {
52:         let dateTime = new Date();
53:         let datetimeSpan =
54:           document.getElementById('datetimeSpan');
55:         datetimeSpan.innerHTML = dateTime.toLocaleString();
56:       }
57:
58:       function convertToHtml(books) {
59:         let template = `
60:           {{#books}}
61:             {{isbn}}:
62:             <strong>{{author}}</strong>:
63:             {{title}}
64:             <br>
65:           {{/books}}

```

PennyCss/index.html (Page 2 of 3)

```

66:         `;
67:         let map = {books: books};
68:         let html = Mustache.render(template, map);
69:         return html;
70:     }
71:
72:     function handleResponse() {
73:         if (this.status !== 200) {
74:             alert('Error: Failed to fetch data from server');
75:             return;
76:         }
77:         let books = JSON.parse(this.response);
78:         let html = convertToHtml(books);
79:         let resultsDiv = document.getElementById('resultsDiv');
80:         resultsDiv.innerHTML = html;
81:     }
82:
83:     function handleError() {
84:         alert('Error: Failed to fetch data from server');
85:     }
86:
87:     let request = null;
88:
89:     function getResults() {
90:         let authorInput = document.getElementById('authorInput');
91:         let author = authorInput.value;
92:         let encodedAuthor = encodeURIComponent(author);
93:         let url = '/searchresults?author=' + encodedAuthor;
94:         if (request !== null)
95:             request.abort();
96:         request = new XMLHttpRequest();
97:         request.onload = handleResponse;
98:         request.onerror = handleError;
99:         request.open('GET', url);
100:        request.send();
101:    }
102:
103:    let timer = null;
104:
105:    function debouncedGetResults() {
106:        clearTimeout(timer);
107:        timer = setTimeout(getResults, 500);
108:    }
109:
110:    function setupHeader() {
111:        getAmPm();
112:        let apInterval = window.setInterval(getAmPm, 1000);
113:        window.addEventListener('beforeunload',
114:            function(e) {window.clearInterval(apInterval);})
115:    }
116:
117:    function setupFooter() {
118:        getDateTime();
119:        let dtInterval = window.setInterval(getDateTime, 1000);
120:        window.addEventListener('beforeunload',
121:            function(e) {window.clearInterval(dtInterval);})
122:    }
123:
124:    function setup() {
125:        setupHeader();
126:        setupFooter();
127:        let authorInput = document.getElementById('authorInput');
128:        authorInput.addEventListener('input', debouncedGetResults);
129:    }
130:

```

PennyCss/index.html (Page 3 of 3)

```

131:         document.addEventListener('DOMContentLoaded', setup);
132:     </script>
133: </body>
134: </html>

```

PennyResponsive/static/penny.css (Page 1 of 1)

```
1: /*-----*/
2: /* penny.css */
3: /* Author: Bob Dondero */
4: /*-----*/
5:
6: /* Element rules */
7:
8: body {font-family:Arial, Helvetica, sans-serif; font-size:16px;}
9:
10: input[type="text"] {font-size:16px;}
11:
12: /*-----*/
13:
14: /* Class rules */
15:
16: .header, .footer {
17:     background-color: #295078;
18:     text-align: center;
19:     color: white;}
20:
21: .headerlink, .footerlink {color: white;}
22:
23: .grayborder {
24:     border-style: solid;
25:     border-color: gray;
26:     border-width: 1px;
27: }
28:
29: .grid-container {
30:     display: grid;
31:     grid-template-columns: 1fr 3fr;
32:     gap: 16px;
33:     color: blue;
34: }
35:
36: /*-----*/
37:
38: /* Media query */
39:
40: @media (max-width: 600px) {
41:     h2 {display: none;}
42:
43:     .grid-container {
44:         display: grid;
45:         grid-template-columns: 1fr;
46:         gap: 16px;
47:     }
48: }
```

blank (Page 1 of 1)

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

PennyBootstrap/index.html (Page 1 of 3)

```

1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>Penny.com</title>
5:
6:     <meta name="viewport"
7:       content="width=device-width, initial-scale=1">
8:
9:     <link rel="stylesheet" href=
10:       "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstra
p.min.css">
11:
12:     <style>
13:       .header, .footer {background-color:#295078; color:white}
14:       .headerlink, .footerlink {color:white}
15:     </style>
16:
17:   </head>
18:
19:   <body>
20:     <div class="container-fluid sticky-top header">
21:       <center>
22:         <h2 class="d-none d-sm-block">Penny.com</h2>
23:         <h3>Good <span id="ampmSpan"></span> and welcome to
24:         Penny.com</h3>
25:       </center>
26:     </div>
27:
28:     <br>
29:
30:     <div class="container-fluid">
31:       <div class="row">
32:         <div class="col-sm-3"><h5>Author name:</h5></div>
33:         <div class="col-sm-9">
34:           <input type="text" class="form-control"
35:             id="authorInput" autoFocus>
36:         </div>
37:       </div>
38:       <br>
39:       <div class="row">
40:         <div class="col-sm-3"><h5>Books:</h5></div>
41:         <div class="col-sm-9">
42:           <div class="border" id="resultsDiv"></div>
43:         </div>
44:       </div>
45:     </div>
46:
47:     <br>
48:
49:     <div class="container-fluid sticky-bottom footer">
50:       <center>
51:         Date and time: <span id="datetimeSpan"></span><br>
52:         Created by <a class="footerlink"
53:           href="http://www.cs.princeton.edu/~rdondero">
54:           Bob Dondero</a>
55:       </center>
56:     </div>
57:
58:     <script src=
59:       "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
60:     </script>
61:
62:     <script>
63:
64:       'use strict';

```

PennyBootstrap/index.html (Page 2 of 3)

```

65:
66:     function getAmPm() {
67:       let dateTime = new Date();
68:       let hours = dateTime.getHours();
69:       let amPm = (hours < 12) ? 'morning' : 'afternoon';
70:       let ampmspan = document.getElementById('ampmSpan');
71:       ampmspan.innerHTML = amPm;
72:     }
73:
74:     function getDateTime() {
75:       let dateTime = new Date();
76:       let datetimeSpan =
77:         document.getElementById('datetimeSpan');
78:       datetimeSpan.innerHTML = dateTime.toLocaleString();
79:     }
80:
81:     function convertToHtml(books) {
82:       let template = `
83:         {{#books}}
84:         {{isbn}}:
85:         <strong>{{author}}</strong>:
86:         {{title}}
87:         <br>
88:         {{/books}}
89:       `;
90:       let map = {books: books};
91:       let html = Mustache.render(template, map);
92:       return html;
93:     }
94:
95:     function handleResponse() {
96:       if (this.status !== 200) {
97:         alert('Error: Failed to fetch data from server');
98:         return;
99:       }
100:       let books = JSON.parse(this.response);
101:       let html = convertToHtml(books);
102:       let resultsDiv = document.getElementById('resultsDiv');
103:       resultsDiv.innerHTML = html;
104:     }
105:
106:     function handleError() {
107:       alert('Error: Failed to fetch data from server');
108:     }
109:
110:     let request = null;
111:
112:     function getResults() {
113:       let authorInput = document.getElementById('authorInput');
114:       let author = authorInput.value;
115:       let encodedAuthor = encodeURIComponent(author);
116:       let url = '/searchresults?author=' + encodedAuthor;
117:       if (request !== null)
118:         request.abort();
119:       request = new XMLHttpRequest();
120:       request.onload = handleResponse;
121:       request.onerror = handleError;
122:       request.open('GET', url);
123:       request.send();
124:     }
125:
126:     let timer = null;
127:
128:     function debouncedGetResults() {
129:       clearTimeout(timer);

```

PennyBootstrap/index.html (Page 3 of 3)

```
130:         timer = setTimeout(getResults, 500);
131:     }
132:
133:     function setupHeader() {
134:         getAmPm();
135:         let apInterval = window.setInterval(getAmPm, 1000);
136:         window.addEventListener('beforeunload',
137:             function(e) {window.clearInterval(apInterval);})
138:     }
139:
140:     function setupFooter() {
141:         getDateTime();
142:         let dtInterval = window.setInterval(getDateTime, 1000);
143:         window.addEventListener('beforeunload',
144:             function(e) {window.clearInterval(dtInterval);})
145:     }
146:
147:     function setup() {
148:         setupHeader();
149:         setupFooter();
150:         let authorInput = document.getElementById('authorInput');
151:         authorInput.addEventListener('input', debouncedGetResults);
152:     }
153:
154:     document.addEventListener('DOMContentLoaded', setup);
155:
156: </script>
157: </body>
158: </html>
```

blank (Page 1 of 1)

1: This page is intentionally blank.

PennyModal/indexRaw.html (Page 1 of 3)

```

1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>Penny.com</title>
5:
6:     <meta name="viewport"
7:       content="width=device-width, initial-scale=1">
8:
9:     <script src=
10:       "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap
.bundle.min.js">
11:     </script>
12:
13:     <link rel="stylesheet" href=
14:       "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstra
p.min.css">
15:
16:     <style>
17:       .header, .footer {background-color:#295078; color:white}
18:       .headerlink, .footerlink {color:white}
19:     </style>
20:
21:   </head>
22:
23:   <body>
24:
25:     <div class="modal" id="booksModal">
26:       <div class="modal-dialog">
27:         <div class="modal-content">
28:           <div class="modal-header">
29:             <h4 class="modal-title">Books</h4>
30:             <button class="btn-close" data-bs-dismiss="modal">
31:               </button>
32:           </div>
33:           <div class="modal-body" id="booksModalBody">
34:             Modal body; to be replaced.
35:           </div>
36:           <div class="modal-footer">
37:             <button class="btn btn-danger"
38:               data-bs-dismiss="modal">
39:               Close
40:             </button>
41:           </div>
42:         </div>
43:       </div>
44:     </div>
45:
46:     <div class="container-fluid header">
47:       <center>
48:         <h2 class="d-none d-sm-block">Penny.com</h2>
49:         <h3>Good <span id="ampmSpan"></span> and welcome to
50:         Penny.com</h3>
51:       </center>
52:     </div>
53:
54:     <br>
55:
56:     <div class="container-fluid">
57:       <div class="row">
58:         <div class="col-sm-2"><h5>Author name:</h5></div>
59:         <div class="col-sm-8">
60:           <input type="text" class="form-control"
61:             id="authorInput" autoFocus>
62:         </div>
63:       <div class="col-sm-2">

```

PennyModal/indexRaw.html (Page 2 of 3)

```

64:         <button id="submitbutton">Submit</button>
65:       </div>
66:     </div>
67:   </div>
68:
69:   <br>
70:
71:   <div class="container-fluid footer">
72:     <center>
73:       Date and time: <span id="datetimeSpan"></span><br>
74:       Created by <a class="footerlink"
75:         href="http://www.cs.princeton.edu/~rdondero">
76:         Bob Dondero</a>
77:     </center>
78:   </div>
79:
80:   <script src=
81:     "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
82:   </script>
83:
84:   <script>
85:
86:     'use strict';
87:
88:     function getAmPm() {
89:       let dateTime = new Date();
90:       let hours = dateTime.getHours();
91:       let amPm = (hours < 12) ? 'morning' : 'afternoon';
92:       let ampmspan = document.getElementById('ampmSpan');
93:       ampmspan.innerHTML = amPm;
94:     }
95:
96:     function getDateTime() {
97:       let dateTime = new Date();
98:       let datetimeSpan =
99:         document.getElementById('datetimeSpan');
100:       datetimeSpan.innerHTML = dateTime.toLocaleString();
101:     }
102:
103:     function convertToHtml(books) {
104:       let template = `
105:         {{#books}}
106:         {{isbn}}:
107:         <strong>{{author}}</strong>:
108:         {{title}}
109:         <br>
110:         {{/books}}
111:       `;
112:       let map = {books: books};
113:       let html = Mustache.render(template, map);
114:       return html;
115:     }
116:
117:     function handleResponse() {
118:       if (this.status !== 200) {
119:         alert('Error: Failed to fetch data from server');
120:         return;
121:       }
122:       let books = JSON.parse(this.response);
123:       let html = convertToHtml(books);
124:       let booksModalBodyNode =
125:         document.getElementById('booksModalBody');
126:       booksModalBodyNode.innerHTML = html;
127:       let booksModalNode =
128:         document.getElementById('booksModal');

```

PennyModal/indexRaw.html (Page 3 of 3)

```
129:         let booksModal = new bootstrap.Modal(booksModalNode);
130:         booksModal.show();
131:     }
132:
133:     function handleError() {
134:         alert('Error: Failed to fetch data from server');
135:     }
136:
137:     let request = null;
138:
139:     function getResults() {
140:         let authorInput = document.getElementById('authorInput');
141:         let author = authorInput.value;
142:         let encodedAuthor = encodeURIComponent(author);
143:         let url = '/searchresults?author=' + encodedAuthor;
144:         if (request !== null)
145:             request.abort();
146:         request = new XMLHttpRequest();
147:         request.onload = handleResponse;
148:         request.onerror = handleError;
149:         request.open('GET', url);
150:         request.send();
151:     }
152:
153:     function setupHeader() {
154:         getAmPm();
155:         let apInterval = window.setInterval(getAmPm, 1000);
156:         window.addEventListener('beforeunload',
157:             function(e) {window.clearInterval(apInterval);})
158:     }
159:
160:     function setupFooter() {
161:         getDateTime();
162:         let dtInterval = window.setInterval(getDateTime, 1000);
163:         window.addEventListener('beforeunload',
164:             function(e) {window.clearInterval(dtInterval);})
165:     }
166:
167:     function setup() {
168:         setupHeader();
169:         setupFooter();
170:         let submitButton = document.getElementById('submitbutton');
171:         submitButton.addEventListener('click', getResults);
172:     }
173:
174:     document.addEventListener('DOMContentLoaded', setup);
175:
176: </script>
177: </body>
178: </html>
```

blank (Page 1 of 1)

1: This page is intentionally blank.

PennyModal/indexjQuery.html (Page 1 of 3)

```

1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>Penny.com</title>
5:
6:     <meta name="viewport"
7:       content="width=device-width, initial-scale=1">
8:
9:     <script src=
10:       "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap
.bundle.min.js">
11:     </script>
12:
13:     <link rel="stylesheet" href=
14:       "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstra
p.min.css">
15:
16:     <style>
17:       .header, .footer {background-color:#295078; color:white}
18:       .headerlink, .footerlink {color:white}
19:     </style>
20:
21:   </head>
22:
23:   <body>
24:
25:     <div class="modal" id="booksModal">
26:       <div class="modal-dialog">
27:         <div class="modal-content">
28:           <div class="modal-header">
29:             <h4 class="modal-title">Books</h4>
30:             <button class="btn-close" data-bs-dismiss="modal">
31:               </button>
32:           </div>
33:           <div class="modal-body" id="booksModalBody">
34:             Modal body; to be replaced.
35:           </div>
36:           <div class="modal-footer">
37:             <button class="btn btn-danger"
38:               data-bs-dismiss="modal">
39:               Close
40:             </button>
41:           </div>
42:         </div>
43:       </div>
44:     </div>
45:
46:     <div class="container-fluid, header">
47:       <center>
48:         <h2 class="d-none d-sm-block">Penny.com</h2>
49:         <h3>Good <span id="ampmSpan"></span> and welcome to
50:         Penny.com</h3>
51:       </center>
52:     </div>
53:
54:     <br>
55:
56:     <div class="container-fluid">
57:       <div class="row">
58:         <div class="col-sm-2"><h5>Author name:</h5></div>
59:         <div class="col-sm-8">
60:           <input type="text" class="form-control"
61:             id="authorInput" autoFocus>
62:         </div>
63:       <div class="col-sm-2">

```

PennyModal/indexjQuery.html (Page 2 of 3)

```

64:         <button id="submitbutton">Submit</button>
65:       </div>
66:     </div>
67:   </div>
68:
69:   <br>
70:
71:   <div class="container-fluid, footer">
72:     <center>
73:       Date and time: <span id="datetimeSpan"></span><br>
74:       Created by <a class="footerlink"
75:         href="http://www.cs.princeton.edu/~rdontero">
76:         Bob Dondero</a>
77:     </center>
78:   </div>
79:
80:   <script src=
81:     "https://cdn.jsdelivr.net/npm/jquery@3.7.1/dist/jquery.min.js">
82:   </script>
83:
84:   <script src=
85:     "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
86:   </script>
87:
88:   <script>
89:
90:     'use strict';
91:
92:     function getAmPm() {
93:       let dateTime = new Date();
94:       let hours = dateTime.getHours();
95:       let amPm = 'morning';
96:       if (hours >= 12)
97:         amPm = 'afternoon';
98:       $('#ampmSpan').html(amPm);
99:     }
100:
101:     function getDateTime() {
102:       let dateTime = new Date();
103:       $('#datetimeSpan').html(dateTime.toLocaleString());
104:     }
105:
106:     function convertToHtml(books) {
107:       let template = `
108:         {{#books}}
109:           {{isbn}}:
110:           <strong>{{author}}</strong>:
111:           {{title}}
112:           <br>
113:           {{/books}}
114:       `;
115:       let map = {books: books};
116:       let html = Mustache.render(template, map);
117:       return html;
118:     }
119:
120:     function handleResponse(books) {
121:       let html = convertToHtml(books);
122:       $('#booksModalBody').html(html);
123:       $('#booksModal').modal('show');
124:     }
125:
126:     function handleError(request) {
127:       if (request.statusText !== 'abort')
128:         alert('Error: Failed to fetch data from server');

```

PennyModal/indexjQuery.html (Page 3 of 3)

```
129:     }
130:
131:     let request = null;
132:
133:     function getResults() {
134:         let author = $('#authorInput').val();
135:         let encodedAuthor = encodeURIComponent(author);
136:         let url = '/searchresults?author=' + encodedAuthor;
137:         if (request != null)
138:             request.abort();
139:         let requestData = {
140:             type: 'GET',
141:             url: url,
142:             success: handleResponse,
143:             error: handleError
144:         };
145:         request = $.ajax(requestData);
146:     }
147:
148:     function setupHeader() {
149:         getAmPm();
150:         let apInterval = window.setInterval(getAmPm, 1000);
151:         $(window).on('beforeunload',
152:             function(e) {window.clearInterval(apInterval);})
153:     }
154:
155:     function setupFooter() {
156:         getDateTime();
157:         let dtInterval = window.setInterval(getDateTime, 1000);
158:         $(window).on('beforeunload',
159:             function(e) {window.clearInterval(dtInterval);})
160:     }
161:
162:     function setup() {
163:         setupHeader();
164:         setupFooter();
165:         $('#submitbutton').on('click', getResults);
166:     }
167:
168:     $('document').ready(setup);
169:
170:     </script>
171: </body>
172: </html>
```

blank (Page 1 of 1)

1: This page is intentionally blank.

PennyModal/indexReact.html (Page 1 of 4)

```

1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <meta name="viewport"
5:       content="width=device-width, initial-scale=1">
6:
7:     <!-- For development: -->
8:     <!--
9:     <script src=
10:       "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.developmen
t.js">
11:     </script>
12:     <script src=
13:       "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.de
velopment.js">
14:     </script>
15:     -->
16:
17:     <!-- For production: -->
18:     <script src=
19:       "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production
.min.js">
20:     </script>
21:     <script src=
22:       "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.pr
oduction.min.js">
23:     </script>
24:
25:     <script src=
26:       "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min
.js">
27:     </script>
28:
29:     <script src=
30:       "https://cdn.jsdelivr.net/npm/react-bootstrap@2.10.0/dist/react
-bootstrap.min.js">
31:     </script>
32:
33:     <link rel="stylesheet" href=
34:       "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstra
p.min.css">
35:
36:     <style>
37:       .header, .footer {background-color:#295078; color:white}
38:       .headerlink, .footerlink {color:white}
39:     </style>
40:
41:     <title>Penny.com</title>
42:   </head>
43:
44:   <body>
45:
46:     <div id="root"></div>
47:
48:     <script type="text/babel">
49:
50:       'use strict';
51:
52:       //-----
53:
54:       function PennyHeader() {
55:         const [datetime, setDatetime] = React.useState(new Date());
56:
57:         function updateHeader() {
58:           let apInterval = window.setInterval(

```

PennyModal/indexReact.html (Page 2 of 4)

```

59:           () => {setDatetime(new Date());},
60:           1000
61:         );
62:         return () => {window.clearInterval(apInterval)};
63:       }
64:       React.useEffect(updateHeader, []);
65:
66:       let hours = datetime.getHours();
67:       let ampm = (hours < 12) ? 'morning' : 'afternoon';
68:       return (
69:         <ReactBootstrap.Container fluid className="header">
70:           <center>
71:             <h2>Penny.com</h2>
72:             <h3>Good {ampm} and welcome to Penny.com</h3>
73:           </center>
74:         </ReactBootstrap.Container>
75:       );
76:     }
77:
78:     //-----
79:
80:     function BooksModal(props) {
81:       let books = props.b;
82:       let show = props.s;
83:       let showCallback = props.scb;
84:
85:       if (books === null) return;
86:
87:       function handleClose() {
88:         showCallback(false);
89:       }
90:
91:       return (
92:         <ReactBootstrap.Modal show={show}
93:           onHide={handleClose}>
94:           <ReactBootstrap.Modal.Header closeButton>
95:             <ReactBootstrap.Modal.Title>
96:               Books
97:             </ReactBootstrap.Modal.Title>
98:           </ReactBootstrap.Modal.Header>
99:           <ReactBootstrap.Modal.Body>
100:            {books.map(book => (
101:              <div key={book.isbn}>
102:                {book.isbn}:&nbsp;&nbsp;&nbsp;
103:                <strong>{book.author}</strong>:&nbsp;&nbsp;&nbsp;
104:                {book.title}<br />
105:              </div>
106:            ))}
107:           </ReactBootstrap.Modal.Body>
108:           <ReactBootstrap.Modal.Footer>
109:             <ReactBootstrap.Button variant="danger"
110:               onClick={handleClose}>
111:               Close
112:             </ReactBootstrap.Button>
113:           </ReactBootstrap.Modal.Footer>
114:         </ReactBootstrap.Modal>
115:       );
116:     }
117:
118:     //-----
119:
120:     function PennySearch() {
121:       const [books, setBooks] = React.useState(null);
122:       const [show, setShow] = React.useState(false);
123:

```

PennyModal/indexReact.html (Page 3 of 4)

```

124:     const inputRef = React.useRef();
125:
126:     let request = null;
127:
128:     function fetchBooks(author) {
129:         function handleResponse() {
130:             if (this.status !== 200) {
131:                 alert('Error: Failed to fetch data from server');
132:                 return;
133:             }
134:             let books = JSON.parse(this.response);
135:             setBooks(books);
136:             setShow(true);
137:         }
138:
139:         function handleError() {
140:             alert('Error: Failed to fetch data from server');
141:         }
142:
143:         let encodedAuthor = encodeURIComponent(author);
144:         let url = '/searchresults?author=' + encodedAuthor;
145:         if (request !== null)
146:             request.abort();
147:         request = new XMLHttpRequest();
148:         request.onload = handleResponse;
149:         request.onerror = handleError;
150:         request.open('GET', url);
151:         request.send();
152:     }
153:
154:     return (
155:         <div>
156:             <BooksModal s={show} scb={setShow} b={books} />
157:             <ReactBootstrap.Container fluid>
158:                 <ReactBootstrap.Row>
159:                     <ReactBootstrap.Col sm={2}>
160:                         <h5>Author name:</h5>
161:                     </ReactBootstrap.Col>
162:                     <ReactBootstrap.Col sm={8}>
163:                         <ReactBootstrap.Form.Control
164:                             type="text" ref={inputRef} autoFocus
165:                         />
166:                     </ReactBootstrap.Col>
167:                     <ReactBootstrap.Col sm={2}>
168:                         <button onClick={ (event) => {
169:                             fetchBooks(inputRef.current.value);
170:                         } } >
171:                             Submit
172:                         </button>
173:                     </ReactBootstrap.Col>
174:                 </ReactBootstrap.Row>
175:             </ReactBootstrap.Container>
176:         </div>
177:     );
178: }
179:
180: //-----
181:
182: function PennyFooter() {
183:     const [datetime, setDatetime] = React.useState(new Date());
184:
185:     function updateFooter() {
186:         let dtInterval = window.setInterval(
187:             () => {setDatetime(new Date());},
188:             1000

```

PennyModal/indexReact.html (Page 4 of 4)

```

189:         );
190:         return () => {window.clearInterval(dtInterval)};
191:     }
192:     React.useEffect(updateFooter, []);
193:
194:     return (
195:         <ReactBootstrap.Container fluid className="footer">
196:             <center>
197:                 Date and time: {datetime.toLocaleString()}
198:                 <br />
199:                 Created by&nbsp;
200:                 <a class="footerlink"
201:                     href="https://www.cs.princeton.edu/~rdondero">
202:                     Bob Dondero</a>
203:             </center>
204:         </ReactBootstrap.Container>
205:     );
206: }
207:
208: //-----
209:
210: function App() {
211:     return (
212:         <div>
213:             <PennyHeader />
214:             <br />
215:             <PennySearch />
216:             <br />
217:             <PennyFooter />
218:         </div>
219:     );
220: }
221:
222: //-----
223:
224: let domRoot = document.getElementById('root');
225: let reactRoot = ReactDOM.createRoot(domRoot);
226: reactRoot.render(
227:     <React.StrictMode>
228:         <App />
229:     </React.StrictMode>
230: );
231:
232: </script>
233: </body>
234: </html>
235:

```