PennyViewport/penny.py (Page 1 of 1)

```
#!/usr/bin/env python

#-----------------------------------------------------------------------
# penny.py
# Author: Bob Dondero
#-----------------------------------------------------------------------

import json
import flask
import database

#-----------------------------------------------------------------------

app = flask.Flask(__name__)

#-----------------------------------------------------------------------

@app.route('/', methods=['GET'])
@app.route('/index', methods=['GET'])
def index():

    return flask.send_file('index.html')

#-----------------------------------------------------------------------

@app.route('/searchresults', methods=['GET'])
def search_results():

    author = flask.request.args.get('author')
    if author is None:
        author = ''
    author = author.strip()

    if author == '':
        books = []
    else:
        books = database.get_books(author) # Exception handling omitted

    json_doc = json.dumps(books)
    response = flask.make_response(json_doc)
    response.headers['Content-Type'] = 'application/json'
    return response
```

blank (Page 1 of 1)

```
This page is intentionally blank.
```

PennyViewport/index.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>

      <meta name="viewport"
         content="width=device-width, initial-scale=1">
   </head>
   <body>
      <hr>
      Good <span id="ampmSpan"></span> and welcome to
      <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      Please enter an author name:
      <input type="text" id="authorInput" autoFocus>
      <hr>

      <div id="resultsDiv"></div>

      <hr>
      Date and time: <span id="datetimeSpan"></span><br>
      Created by <a href="https://www.cs.princeton.edu/~rdondero">
      Bob Dondero</a>
      <hr>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse() {
```

PennyViewport/index.html (Page 2 of 2)

```
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let resultsDiv = document.getElementById('resultsDiv');
            resultsDiv.innerHTML = html;
         }

         function handleError() {
            alert('Error: Failed to fetch data from server');
         }

         let request = null;

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            if (request !== null)
               request.abort();
            request = new XMLHttpRequest();
            request.onload = handleResponse;
            request.onerror = handleError;
            request.open('GET', url);
            request.send();
         }

         let timer = null;

         function debouncedGetResults() {
            clearTimeout(timer);
            timer = setTimeout(getResults, 500);
         }

         function setupHeader() {
            getAmPm();
            let apInterval = window.setInterval(getAmPm, 1000);
            window.addEventListener('beforeunload',
               function(e) {window.clearInterval(apInterval);})
         }

         function setupFooter() {
            getDateTime();
            let dtInterval = window.setInterval(getDateTime, 1000);
            window.addEventListener('beforeunload',
               function(e) {window.clearInterval(dtInterval);})
         }

         function setup() {
            setupHeader();
            setupFooter();
            let authorInput = document.getElementById('authorInput');
            authorInput.addEventListener('input', debouncedGetResults);
         }

         document.addEventListener('DOMContentLoaded', setup);

      </script>
   </body>
</html>
```

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

```
/*--------------------------------------------------------------------*/
/* penny.css                                                          */
/* Author: Bob Dondero                                                */
/*--------------------------------------------------------------------*/

/* Element rules */

body {font-family:Arial, Helvetica, sans-serif; font-size:16px;}

input[type="text"] {font-size:16px;}

a {color:#aaaaff;}

table {border-spacing: 16px;}

/*--------------------------------------------------------------------*/

/* Class rules */

.header {background-color:#295078; text-align:center; color:white;}

.footer {background-color:#295078; text-align:center; color:white;}

.grayborder {border-style:solid; border-color:gray; border-width:1px;}
```

blank (Page 1 of 1)

```
This page is intentionally blank.
```

PennyCss/index.html (Page 1 of 3)

```
<!DOCTYPE html>
<html>

   <head>
      <title>Penny.com</title>
      <meta name="viewport"
         content="width=device-width, initial-scale=1">
      <link rel="stylesheet" href="static/penny.css">
   </head>

   <body>
      <div class="header">
         <h2>Penny.com</h2>
         <h3>Good <span id="ampmSpan"></span> and welcome to
            Penny.com</h3>
      </div>

      <br>

      <table width="100%">
         <tbody>
            <tr>
               <td width="25%"><strong>Author name:</strong></td>
               <td width="75%">
                  <input type="text" id="authorInput" autoFocus>
               </td>
            </tr>
            <tr>
               <td width="25%"><strong>Books:</strong></td>
               <td width="75%" class="grayborder" id="resultsTd"></td>
            </tr>
         </tbody>
      </table>

      <br>

      <div class="footer">
         Date and time: <span id="datetimeSpan"></span><br>
         Created by <a href="https://www.cs.princeton.edu/~rdondero">
         Bob Dondero</a>
      </div>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>
         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
```

PennyCss/index.html (Page 2 of 3)

```
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse() {
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let resultsTd = document.getElementById('resultsTd');
            resultsTd.innerHTML = html;
         }

         function handleError() {
            alert('Error: Failed to fetch data from server');
         }

         let request = null;

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            if (request !== null)
               request.abort();
            request = new XMLHttpRequest();
            request.onload = handleResponse;
            request.onerror = handleError;
            request.open('GET', url);
            request.send();
         }

         let timer = null;

         function debouncedGetResults() {
            clearTimeout(timer);
            timer = setTimeout(getResults, 500);
         }

         function setupHeader() {
            getAmPm();
            let apInterval = window.setInterval(getAmPm, 1000);
            window.addEventListener('beforeunload',
               function(e) {window.clearInterval(apInterval);})
         }

         function setupFooter() {
            getDateTime();
            let dtInterval = window.setInterval(getDateTime, 1000);
            window.addEventListener('beforeunload',
               function(e) {window.clearInterval(dtInterval);})
         }

```

PennyCss/index.html (Page 3 of 3)

```
131:          function setup() {
132:             setupHeader();
133:             setupFooter();
134:             let authorInput = document.getElementById('authorInput');
135:             authorInput.addEventListener('input', debouncedGetResults);
136:          }
137:
138:          document.addEventListener('DOMContentLoaded', setup);
139:       </script>
140:    </body>
141: </html>
```

blank (Page 1 of 1)

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

PennyResponsive/static/penny.css (Page 1 of 2)

```
/*-------------------------------------------------------------------------*/
/* penny.css                                                               */
/* Author: Bob Dondero                                                     */
/*-------------------------------------------------------------------------*/

/* Element rules */

body {font-family:Arial, Helvetica, sans-serif; font-size:16px;}

input[type="text"] {font-size:16px;}

a {color:#aaaaff;}

/*-------------------------------------------------------------------------*/

/* Class rules */

.header {background-color:#295078; text-align:center; color:white;}

.footer {background-color:#295078; text-align:center; color:white;}

.grayborder {border-style:solid; border-color:gray; border-width:1px;}

/*-------------------------------------------------------------------------*/

/* The following sections are derived from:
   https://www.w3schools.com/css/css_rwd_grid.asp */

/*-------------------------------------------------------------------------*/

/* Make sure that the padding and border are included in the total
   width and height of all elements. */

* {box-sizing: border-box;}

/*-------------------------------------------------------------------------*/

/* Divide the window into 12 columns. An element with style "col-1"
   consumes 1 column, an element with style "col-2" consumes 2
   columns, and so forth. Each "col-" element should be placed within
   a "row" element. The "col-" elements within each row element
   should consume 12 columns. Each "col-" element floats to the left
   of its "row" element, and has a padding of 5 pixels. */

.col-1  {width: 8.33%;  float:left; padding:5px;}
.col-2  {width: 16.66%; float:left; padding:5px;}
.col-3  {width: 25%;    float:left; padding:5px;}
.col-4  {width: 33.33%; float:left; padding:5px;}
.col-5  {width: 41.66%; float:left; padding:5px;}
.col-6  {width: 50%;    float:left; padding:5px;}
.col-7  {width: 58.33%; float:left; padding:5px;}
.col-8  {width: 66.66%; float:left; padding:5px;}
.col-9  {width: 75%;    float:left; padding:5px;}
.col-10 {width: 83.33%; float:left; padding:5px;}
.col-11 {width: 91.66%; float:left; padding:5px;}
.col-12 {width: 100%;   float:left; padding:5px;}

/*-------------------------------------------------------------------------*/

/* The "col-" elements inside a "row" element float to the left, and
   so normally would be "out of the flow of the page." Other
   elements would be placed as if the "col-" elements do not exist.
   Clear the flow to prevent that. */

.row::after
```

PennyResponsive/static/penny.css (Page 2 of 2)

```
{
   content: "";
   clear: both;
   display: table;
}

/*-------------------------------------------------------------------------*/

/* On small windows (those whose maximum width is 600 pixels or less)
   suppress the display of "h2" elements. Also change the "col-"
   classes such that each "col-" element consumes all 12 columns
   within its "row" element. So multiple "col-" elements within a
   "row" element will be arranged vertically. */

@media (max-width: 600px)
{
   h2 {display:none;}

   /*
   .col-1  {width: 100%; float:left; padding:5px;}
   .col-2  {width: 100%; float:left; padding:5px;}
   .col-3  {width: 100%; float:left; padding:5px;}
   .col-4  {width: 100%; float:left; padding:5px;}
   .col-5  {width: 100%; float:left; padding:5px;}
   .col-6  {width: 100%; float:left; padding:5px;}
   .col-7  {width: 100%; float:left; padding:5px;}
   .col-8  {width: 100%; float:left; padding:5px;}
   .col-9  {width: 100%; float:left; padding:5px;}
   .col-10 {width: 100%; float:left; padding:5px;}
   .col-11 {width: 100%; float:left; padding:5px;}
   .col-12 {width: 100%; float:left; padding:5px;}
   */

   /* Shortcut: */
   [class*="col-"] {width: 100%; float:left; padding:5px;}
}
```

PennyResponsive/index.html (Page 1 of 3)

```
<!DOCTYPE html>
<html>

   <head>
      <title>Penny.com</title>
      <meta name="viewport"
            content="width=device-width, initial-scale=1">
      <link rel="stylesheet" href="static/penny.css">
   </head>

   <body>

      <div class="header">
         <h2>Penny.com</h2>
         <h3>Good <span id="ampmSpan"></span> and welcome to
            Penny.com</h3>
      </div>

      <br>

      <div class="row">
         <div class="col-3"><strong>Author name:</strong></div>
         <div class="col-9" grayborder>
            <input type="text" id="authorInput" autoFocus>
         </div>
      </div>

      <br>

      <div class="row">
         <div class="col-3"><strong>Books:</strong></div>
         <div class="col-9 grayborder" id="resultsDiv"></div>
      </div>

      <br>

      <div class="footer">
         Date and time: <span id="datetimeSpan"></span><br>
         Created by <a href="https://www.cs.princeton.edu/~rdondero">
         Bob Dondero</a>
      </div>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>
         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
```

PennyResponsive/index.html (Page 2 of 3)

```
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse() {
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let resultsDiv = document.getElementById('resultsDiv');
            resultsDiv.innerHTML = html;
         }

         function handleError() {
            alert('Error: Failed to fetch data from server');
         }

         let request = null;

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            if (request !== null)
               request.abort();
            request = new XMLHttpRequest();
            request.onload = handleResponse;
            request.onerror = handleError;
            request.open('GET', url);
            request.send();
         }

         let timer = null;

         function debouncedGetResults() {
            clearTimeout(timer);
            timer = setTimeout(getResults, 500);
         }

         function setupHeader() {
            getAmPm();
            let apInterval = window.setInterval(getAmPm, 1000);
            window.addEventListener('beforeunload',
               function(e) {window.clearInterval(apInterval);})
         }

         function setupFooter() {
            getDateTime();
            let dtInterval = window.setInterval(getDateTime, 1000);
            window.addEventListener('beforeunload',
               function(e) {window.clearInterval(dtInterval);})
         }

```

PennyResponsive/index.html (Page 3 of 3)

```
131:          function setup() {
132:             setupHeader();
133:             setupFooter();
134:             let authorInput = document.getElementById('authorInput');
135:             authorInput.addEventListener('input', debouncedGetResults);
136:          }
137:
138:          document.addEventListener('DOMContentLoaded', setup);
139:       </script>
140:    </body>
141: </html>
```

blank (Page 1 of 1)

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

PennyBootstrap/index.html (Page 1 of 3)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>

      <meta name="viewport"
         content="width=device-width, initial-scale=1">

      <link rel="stylesheet" href=
         "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstra
p.min.css">

      <style>
         .header, .footer {background-color:#295078; color:white}
      </style>

   </head>

   <body>
      <div class="container-fluid, header">
         <center>
            <h2 class="d-none d-sm-block">Penny.com</h2>
            <h3>Good <span id="ampmSpan"></span> and welcome to
            Penny.com</h3>
         </center>
      </div>

      <br>

      <div class="container-fluid">
         <div class="row">
            <div class="col-sm-3"><h5>Author name:</h5></div>
            <div class="col-sm-9">
               <input type="text" class="form-control"
                  id="authorInput" autoFocus>
            </div>
         </div>
         <br>
         <div class="row">
            <div class="col-sm-3"><h5>Books:</h5></div>
            <div class="col-sm-9">
               <div class="border" id="resultsDiv"></div>
            </div>
         </div>
      </div>

      <br>

      <div class="container-fluid, footer">
         <center>
            Date and time: <span id="datetimeSpan"></span><br>
            Created by <a
            href="http://www.cs.princeton.edu/~rdondero">
            Bob Dondero</a>
         </center>
      </div>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>

         'use strict';

```

PennyBootstrap/index.html (Page 2 of 3)

```
         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse() {
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let resultsDiv = document.getElementById('resultsDiv');
            resultsDiv.innerHTML = html;
         }

         function handleError() {
            alert('Error: Failed to fetch data from server');
         }

         let request = null;

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            if (request !== null)
               request.abort();
            request = new XMLHttpRequest();
            request.onload = handleResponse;
            request.onerror = handleError;
            request.open('GET', url);
            request.send();
         }

         let timer = null;

         function debouncedGetResults() {
            clearTimeout(timer);
            timer = setTimeout(getResults, 500);
```

PennyBootstrap/index.html (Page 3 of 3)

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

blank (Page 1 of 1)

1: This page is intentionally blank.

PennyModal/indexRaw.html (Page 1 of 3)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>

      <meta name="viewport"
         content="width=device-width, initial-scale=1">

      <script src=
         "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap
.bundle.min.js">
      </script>

      <link rel="stylesheet" href=
         "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstra
p.min.css">

      <style>
         .header, .footer {background-color:#295078; color:white}
      </style>

   </head>

   <body>

      <div class="modal" id="booksModal">
         <div class="modal-dialog">
            <div class="modal-content">
               <div class="modal-header">
                  <h4 class="modal-title">Books</h4>
                  <button class="btn-close" data-bs-dismiss="modal">
                  </button>
               </div>
               <div class="modal-body" id="booksModalBody">
                  Modal body; to be replaced.
               </div>
               <div class="modal-footer">
                  <button class="btn btn-danger"
                     data-bs-dismiss="modal">
                     Close
                  </button>
               </div>
            </div>
         </div>
      </div>

      <div class="container-fluid, header">
         <center>
            <h2 class="d-none d-sm-block">Penny.com</h2>
            <h3>Good <span id="ampmSpan"></span> and welcome to
            Penny.com</h3>
         </center>
      </div>

      <br>

      <div class="container-fluid">
         <div class="row">
            <div class="col-sm-2"><h5>Author name:</h5></div>
            <div class="col-sm-8">
               <input type="text" class="form-control"
                  id="authorInput" autoFocus>
            </div>
            <div class="col-sm-2">
               <button id="submitbutton">Submit</button>
```

PennyModal/indexRaw.html (Page 2 of 3)

```
            </div>
         </div>
      </div>

      <br>

      <div class="container-fluid, footer">
         <center>
            Date and time: <span id="datetimeSpan"></span><br>
            Created by <a
            href="http://www.cs.princeton.edu/~rdondero">
            Bob Dondero</a>
         </center>
      </div>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse() {
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let booksModalBodyNode =
               document.getElementById('booksModalBody');
            booksModalBodyNode.innerHTML = html;
            let booksModalNode =
               document.getElementById('booksModal');
            let booksModal = new bootstrap.Modal(booksModalNode);
```

PennyModal/indexRaw.html (Page 3 of 3)

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

blank (Page 1 of 1)

1: This page is intentionally blank.

PennyModal/indexJQuery.html (Page 1 of 3)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>

      <meta name="viewport"
         content="width=device-width, initial-scale=1">

      <script src=
         "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap
.bundle.min.js">
      </script>

      <link rel="stylesheet" href=
         "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstra
p.min.css">

      <style>
         .header, .footer {background-color:#295078; color:white}
      </style>

   </head>

   <body>

      <div class="modal" id="booksModal">
         <div class="modal-dialog">
            <div class="modal-content">
               <div class="modal-header">
                  <h4 class="modal-title">Books</h4>
                  <button class="btn-close" data-bs-dismiss="modal">
                  </button>
               </div>
               <div class="modal-body" id="booksModalBody">
                  Modal body; to be replaced.
               </div>
               <div class="modal-footer">
                  <button class="btn btn-danger"
                     data-bs-dismiss="modal">
                     Close
                  </button>
               </div>
            </div>
         </div>
      </div>

      <div class="container-fluid, header">
         <center>
            <h2 class="d-none d-sm-block">Penny.com</h2>
            <h3>Good <span id="ampmSpan"></span> and welcome to
            Penny.com</h3>
         </center>
      </div>

      <br>

      <div class="container-fluid">
         <div class="row">
            <div class="col-sm-2"><h5>Author name:</h5></div>
            <div class="col-sm-8">
               <input type="text" class="form-control"
                  id="authorInput" autoFocus>
            </div>
            <div class="col-sm-2">
               <button id="submitbutton">Submit</button>
```

PennyModal/indexJQuery.html (Page 2 of 3)

```
            </div>
         </div>
      </div>

      <br>

      <div class="container-fluid, footer">
         <center>
            Date and time: <span id="datetimeSpan"></span><br>
            Created by <a
            href="http://www.cs.princeton.edu/~rdondero">
            Bob Dondero</a>
         </center>
      </div>

      <script src=
         "https://cdn.jsdelivr.net/npm/jquery@3.7.1/dist/jquery.min.js">
      </script>

      <script src=
          "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = 'morning';
            if (hours >= 12)
               amPm = 'afternoon';
            $('#ampmSpan').html(amPm);
         }

         function getDateTime() {
            let dateTime = new Date();
            $('#datetimeSpan').html(dateTime.toLocaleString());
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse(books) {
            let html = convertToHtml(books);
            $('#booksModalBody').html(html);
            $('#booksModal').modal('show');
         }

         function handleError(request) {
            if (request.statusText !== 'abort')
               alert('Error: Failed to fetch data from server');
         }
```

PennyModal/indexJQuery.html (Page 3 of 3)

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

blank (Page 1 of 1)

1: This page is intentionally blank.

PennyModal/indexReact.html (Page 1 of 4)

```
<!DOCTYPE html>
<html>
   <head>
      <meta name="viewport"
         content="width=device-width, initial-scale=1">

      <!-- For development: -->
      <!--
      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.development.js">
      </script>
      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.development.js">
      </script>
      -->

      <!-- For production: -->
      <script src=
         "https://cdn.jsdelivr.net/npm/react@18.3.1/umd/react.production.min.js">
      </script>
      <script src=
         "https://cdn.jsdelivr.net/npm/react-dom@18.3.1/umd/react-dom.production.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/babel-standalone@6.26.0/babel.min.js">
      </script>

      <script src=
         "https://cdn.jsdelivr.net/npm/react-bootstrap@2.10.0/dist/react-bootstrap.min.js">
      </script>

      <link rel="stylesheet" href=
         "https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css">

      <style>
         .header, .footer {background-color:#295078; color:white}
      </style>

      <title>Penny.com</title>
   </head>

   <body>

      <div id="root"></div>

      <script type="text/babel">

         'use strict';

         //-------------------------------------------------------------

         function PennyHeader() {
            const [datetime, setDatetime] = React.useState(new Date());

            function updateHeader() {
               let apInterval = window.setInterval(
                  () => {setDatetime(new Date());},
```

PennyModal/indexReact.html (Page 2 of 4)

```
                  1000
               );
               return () => {window.clearInterval(apInterval);};
            }
            React.useEffect(updateHeader, []);

            let hours = datetime.getHours();
            let ampm = (hours < 12) ? 'morning' : 'afternoon';
            return (
               <ReactBootstrap.Container fluid className="header">
                  <center>
                     <h2>Penny.com</h2>
                     <h3>Good {ampm} and welcome to Penny.com</h3>
                  </center>
               </ReactBootstrap.Container>
            );
         }

         //-------------------------------------------------------------

         function BooksModal(props) {
            let books = props.b;
            let show = props.s;
            let showCallback = props.scb;

            if (books === null) return;

            function handleClose() {
               showCallback(false);
            }

            return (
               <ReactBootstrap.Modal show={show}
                  onHide={handleClose}>
                  <ReactBootstrap.Modal.Header closeButton>
                     <ReactBootstrap.Modal.Title>
                        Books
                     </ReactBootstrap.Modal.Title>
                  </ReactBootstrap.Modal.Header>
                  <ReactBootstrap.Modal.Body>
                     {books.map((book) => (
                        <div key={book.isbn}>
                           {book.isbn}: 
                           <strong>{book.author}</strong>: 
                           {book.title}<br />
                        </div>
                     ))}
                  </ReactBootstrap.Modal.Body>
                  <ReactBootstrap.Modal.Footer>
                     <ReactBootstrap.Button variant="danger"
                        onClick={handleClose}>
                        Close
                     </ReactBootstrap.Button>
                  </ReactBootstrap.Modal.Footer>
               </ReactBootstrap.Modal>
            );
         }

         //-------------------------------------------------------------

         function PennySearch() {
            const [books, setBooks] = React.useState(null);
            const [show, setShow] = React.useState(false);

            const inputRef = React.useRef();
```

PennyModal/indexReact.html (Page 3 of 4)

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

PennyModal/indexReact.html (Page 4 of 4)

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