PennyAdmin01Baseline/runserver.py (Page 1 of 1)

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

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

import sys
import penny

def main():

    if len(sys.argv) != 2:
        print(f'usage: {sys.argv[0]} port', file=sys.stderr)
        sys.exit(1)

    try:
        port = int(sys.argv[1])
    except Exception:
        print(f'{sys.argv[0]}: Port must be an integer.',
            file=sys.stderr)
        sys.exit(1)

    try:
        penny.app.run(host='0.0.0.0', port=port, debug=True)
    except Exception as ex:
        print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
        sys.exit(1)

if __name__ == '__main__':
    main()
```

PennyAdmin01Baseline/penny.sql (Page 1 of 1)

```
DROP TABLE IF EXISTS books;
CREATE TABLE books (isbn TEXT PRIMARY KEY, author TEXT, title TEXT);
INSERT INTO books (isbn, author, title)
   VALUES ('123', 'Kernighan','The Practice of Programming');
INSERT INTO books (isbn, author, title)
   VALUES ('234', 'Kernighan','The C Programming Language');
INSERT INTO books (isbn, author, title)
   VALUES ('345', 'Sedgewick','Algorithms in C');
```

PennyAdmin01Baseline/database.py (Page 1 of 2)

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

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

import sqlite3
import contextlib

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

_DATABASE_URL = 'file:penny.sqlite'

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

def get_books():

    books = []

    with contextlib.closing(
        sqlite3.connect(_DATABASE_URL + '?mode=ro',
        isolation_level=None, uri=True)) as connection:
        with contextlib.closing(connection.cursor()) as cursor:
            query_str = '''
                SELECT isbn, author, title FROM books
            '''
            cursor.execute(query_str)

            table = cursor.fetchall()
            for row in table:
                book = {'isbn': row[0], 'author': row[1],
                    'title': row[2]}
                books.append(book)

    return books

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

def add_book(isbn, author, title):

    with contextlib.closing(
        sqlite3.connect(_DATABASE_URL + '?mode=rw',
        isolation_level=None, uri=True)) as connection:
        with contextlib.closing(connection.cursor()) as cursor:
            query_str = f'''
                INSERT INTO books (isbn, author, title)
                VALUES ('{isbn}', '{author}', '{title}')
                '''
            try:
                cursor.execute(query_str)
            except sqlite3.IntegrityError:
                return False
            return True

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

def delete_book(isbn):

    with contextlib.closing(
        sqlite3.connect(_DATABASE_URL + '?mode=rw',
        isolation_level=None, uri=True)) as connection:
        with contextlib.closing(connection.cursor()) as cursor:
            query_str = f'''
                DELETE FROM books WHERE isbn = '{isbn}'
```

PennyAdmin01Baseline/database.py (Page 2 of 2)

```
                '''
            cursor.execute(query_str)

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

# For testing:

def _write_books(books):
    for book in books:
        print(f'{book['isbn']} | {book['author']} | {book['title']}')

def _test():
    print('-----------------------------------')
    print('Testing get_books()')
    print('-----------------------------------')
    print()
    books = get_books()
    _write_books(books)
    print()

    print('-----------------------------------')
    print('Testing add_book()')
    print('-----------------------------------')
    print()
    successful = add_book('456', 'Kernighan', 'New Book')
    if successful:
        print('Add was successful')
        print()
        books = get_books()
        _write_books(books)
        print()
    else:
        print('Add was unsuccessful')
        print()
        _write_books(books)
        print()
    successful = add_book('456', 'Kernighan', 'New Book')
    if successful:
        print('Add was successful')
        print()
        books = get_books()
        _write_books(books)
        print()
    else:
        print('Add was unsuccessful')
        print()
        _write_books(books)
        print()

    print('-----------------------------------')
    print('Testing delete_book()')
    print('-----------------------------------')
    print()
    delete_book('456')
    books = get_books()
    _write_books(books)
    print()
    delete_book('456')
    books = get_books()
    _write_books(books)

if __name__ == '__main__':
    _test()
```

PennyAdmin01Baseline/penny.py (Page 1 of 5)

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

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

import flask
import database

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

app = flask.Flask(__name__)

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

def get_header():

    html_code = '''
        <hr>
        Hello, and welcome to <strong>PennyAdmin</strong>
        <hr>
        '''
    return html_code

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

def get_footer():

    html_code = '''
        <hr>
        Created by
        <a href="https://www.cs.princeton.edu/~rdondero">
        Bob Dondero</a>
        <hr>
        '''
    return html_code

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

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

    html_code = f'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                Click to <a href="/menu">begin</a>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

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

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

PennyAdmin01Baseline/penny.py (Page 2 of 5)

```

    html_code = f'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                <a href="/show">Show all books</a><br>
                <a href="/add">Add a book</a><br>
                <a href="/delete">Delete a book by ISBN</a><br>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

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

def convert_to_html(books):

    if len(books) == 0:
        return '(None)'
    html_code = ''
    for book in books:
        html_code += f'''
            {book['isbn']}: <strong>{book['author']}</strong>:
            {book['title']}<br>
            '''
    return html_code

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

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

    books = database.get_books()

    html_code = f'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                <h1>Show All Books</h1>
                {convert_to_html(books)}
                <br>
                <a href="/menu">Return to menu page</a>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

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

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

PennyAdmin01Baseline/penny.py (Page 3 of 5)

```
131:
132:     html_code = fr'''
133:         <!DOCTYPE html>
134:         <html>
135:             <head>
136:                 <title>PennyAdmin</title>
137:             </head>
138:             <body>
139:                 {get_header()}
140:                 <h1>Add a Book</h1>
141:                 <form action="/handleadd" method="post">
142:
143:                     Enter an ISBN:
144:                     <input type="text" name="isbn" autofocus
145:                         required pattern=".*\S.*"
146:                         title="At least one non-white-space char">
147:                     <br>
148:
149:                     Enter an author:
150:                     <input type="text" name="author"
151:                         required pattern=".*\S.*"
152:                         title="At least one non-white-space char">
153:                     <br>
154:
155:                     Enter a title:
156:                     <input type="text" name="title"
157:                         required pattern=".*\S.*"
158:                         title="At least one non-white-space char">
159:                     <br>
160:
161:                     <input type="submit" value="Go">
162:
163:                 </form>
164:
165:                 <br>
166:                 <a href="/menu">Return to menu page</a>
167:                 {get_footer()}
168:             </body>
169:         </html>
170:         '''
171:
172:     response = flask.make_response(html_code)
173:     return response
174:
175: #-----------------------------------------------------------------------
176:
177: @app.route('/handleadd', methods=['POST'])
178: def handle_add():
179:
180:     isbn = flask.request.form.get('isbn')
181:     if (isbn is None) or (isbn.strip() == ''):
182:         message = 'The addition was unsuccessful: missing ISBN'
183:     else:
184:         author = flask.request.form.get('author')
185:         if (author is None) or (author.strip() == ''):
186:             message = 'The addition was unsuccessful: missing author'
187:         else:
188:             title = flask.request.form.get('title')
189:             if (title is None) or (title.strip() == ''):
190:                 message = 'The addition was unsuccessful: missing title'
191:             else:
192:                 isbn = isbn.strip()
193:                 author = author.strip()
194:                 title = title.strip()
195:
```

PennyAdmin01Baseline/penny.py (Page 4 of 5)

```
196:                 successful = database.add_book(isbn, author, title)
197:                 if not successful:
198:                     message = 'The addition was unsuccessful: '
199:                     message += 'duplicate ISBN'
200:                 else:
201:                     message = 'The addition was successful'
202:
203:     html_code = f'''
204:         <!DOCTYPE html>
205:         <html>
206:             <head>
207:                 <title>PennyAdmin</title>
208:             </head>
209:             <body>
210:                 {get_header()}
211:                 <h1>Results</h1>
212:                 {message}
213:                 <br>
214:                 <br>
215:                 <br>
216:                 <a href="/menu">Return to menu page</a>
217:                 {get_footer()}
218:             </body>
219:         </html>
220:         '''
221:
222:     response = flask.make_response(html_code)
223:     return response
224:
225: #-----------------------------------------------------------------------
226:
227: @app.route('/delete', methods=['GET'])
228: def delete():
229:
230:     html_code = fr'''
231:         <!DOCTYPE html>
232:         <html>
233:             <head>
234:                 <title>PennyAdmin</title>
235:             </head>
236:             <body>
237:                 {get_header()}
238:                 <h1>Delete a Book</h1>
239:                 <form action="/handledelete" method="post">
240:                     Enter an ISBN:
241:                     <input type="text" name="isbn" autofocus
242:                         required pattern=".*\S.*"
243:                         title="At least one non-white-space char">
244:                     <br>
245:                     <input type="submit" value="Go">
246:                 </form>
247:                 <br>
248:                 <br>
249:                 <a href="/menu">Return to menu page</a>
250:                 {get_footer()}
251:             </body>
252:         </html>
253:         '''
254:
255:     response = flask.make_response(html_code)
256:     return response
257:
258: #-----------------------------------------------------------------------
259:
260: @app.route('/handledelete', methods=['POST'])
```

PennyAdmin01Baseline/penny.py (Page 5 of 5)

```
261: def handle_delete():
262:
263:     isbn = flask.request.form.get('isbn')
264:     if (isbn is None) or (isbn.strip() == ''):
265:         message = 'The deletion was unsuccessful: missing ISBN'
266:     else:
267:         isbn = isbn.strip()
268:         database.delete_book(isbn)
269:         message = 'The deletion was successful'
270:
271:     html_code = f'''
272:         <!DOCTYPE html>
273:         <html>
274:             <head>
275:                 <title>PennyAdmin</title>
276:             </head>
277:             <body>
278:                 {get_header()}
279:                 <h1>Results</h1>
280:                 {message}
281:                 <br>
282:                 <br>
283:                 <br>
284:                 <a href="/menu">Return to menu page</a>
285:                 {get_footer()}
286:             </body>
287:         </html>
288:         '''
289:     response = flask.make_response(html_code)
290:     return response
```

blank (Page 1 of 1)

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

PennyAdmin02aPrepared/database.py (Page 1 of 2)

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

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

import sqlite3
import contextlib

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

_DATABASE_URL = 'file:penny.sqlite'

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

def get_books():

    books = []

    with contextlib.closing(
        sqlite3.connect(_DATABASE_URL + '?mode=ro',
        isolation_level=None, uri=True)) as connection:
        with contextlib.closing(connection.cursor()) as cursor:
            query_str = '''
                SELECT isbn, author, title FROM books
            '''
            cursor.execute(query_str)

            table = cursor.fetchall()
            for row in table:
                book = {'isbn': row[0], 'author': row[1],
                    'title': row[2]}
                books.append(book)

    return books

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

def add_book(isbn, author, title):

    with contextlib.closing(
        sqlite3.connect(_DATABASE_URL + '?mode=rw',
        isolation_level=None, uri=True)) as connection:
        with contextlib.closing(connection.cursor()) as cursor:
            query_str = '''
                INSERT INTO books (isbn, author, title)
                VALUES (?, ?, ?)
                '''
            try:
                cursor.execute(query_str, [isbn, author, title])
            except sqlite3.IntegrityError:
                return False
            return True

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

def delete_book(isbn):
    with contextlib.closing(
        sqlite3.connect(_DATABASE_URL + '?mode=rw',
        isolation_level=None, uri=True)) as connection:
        with contextlib.closing(connection.cursor()) as cursor:

            query_str = '''
                DELETE FROM books WHERE isbn = ?
```

PennyAdmin02aPrepared/database.py (Page 2 of 2)

```
                '''
            cursor.execute(query_str, [isbn])

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

# For testing:

def _write_books(books):
    for book in books:
        print(f'{book['isbn']} | {book['author']} | {book['title']}')

def _test():
    print('-----------------------------------')
    print('Testing get_books()')
    print('-----------------------------------')
    print()
    books = get_books()
    _write_books(books)
    print()

    print('-----------------------------------')
    print('Testing add_book()')
    print('-----------------------------------')
    print()
    successful = add_book('456', 'Kernighan', 'New Book')
    if successful:
        print('Add was successful')
        print()
        books = get_books()
        _write_books(books)
        print()
    else:
        print('Add was unsuccessful')
        print()
        _write_books(books)
        print()
    successful = add_book('456', 'Kernighan', 'New Book')
    if successful:
        print('Add was successful')
        print()
        books = get_books()
        _write_books(books)
        print()
    else:
        print('Add was unsuccessful')
        print()
        _write_books(books)
        print()

    print('-----------------------------------')
    print('Testing delete_book()')
    print('-----------------------------------')
    print()
    delete_book('456')
    books = get_books()
    _write_books(books)
    print()
    delete_book('456')
    books = get_books()
    _write_books(books)

if __name__ == '__main__':
    _test()
```

PennyAdmin02bAlchemy/database.py (Page 1 of 2)

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

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

import os
import sqlalchemy
import sqlalchemy.orm
import dotenv

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

dotenv.load_dotenv()
_DATABASE_URL = os.getenv('DATABASE_URL', 'sqlite:///penny.sqlite')
_DATABASE_URL = _DATABASE_URL.replace('postgres://', 'postgresql://')
_DATABASE_URL = _DATABASE_URL.replace(
    'postgresql://', 'postgresql+psycopg://')

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

class Base(sqlalchemy.orm.DeclarativeBase):
    pass

class Book (Base):
    __tablename__ = 'books'
    isbn = sqlalchemy.Column(sqlalchemy.String, primary_key=True)
    author = sqlalchemy.Column(sqlalchemy.String)
    title = sqlalchemy.Column(sqlalchemy.String)

_engine = sqlalchemy.create_engine(_DATABASE_URL)

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

def get_books():

    books = []

    with sqlalchemy.orm.Session(_engine) as session:
        query = session.query(Book)
        table = query.all()
        for row in table:
            book = {'isbn': row.isbn, 'author': row.author,
                'title': row.title}
            books.append(book)

    return books

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

def add_book(isbn, author, title):

    with sqlalchemy.orm.Session(_engine) as session:
        row = Book(isbn=isbn, author=author, title=title)
        session.add(row)
        try:
            session.commit()
            return True
        except sqlalchemy.exc.IntegrityError:
            return False

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

def delete_book(isbn):
```

PennyAdmin02bAlchemy/database.py (Page 2 of 2)

```

    with sqlalchemy.orm.Session(_engine) as session:
        session.query(Book).filter(Book.isbn==isbn).delete()
        session.commit()

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

# For testing:

def _write_books(books):
    for book in books:
        print(f'{book['isbn']} | {book['author']} | {book['title']}')

def _test():
    print('-----------------------------------')
    print('Testing get_books()')
    print('-----------------------------------')
    print()
    books = get_books()
    _write_books(books)
    print()

    print('-----------------------------------')
    print('Testing add_book()')
    print('-----------------------------------')
    print()
    successful = add_book('456', 'Kernighan', 'New Book')
    if successful:
        print('Add was successful')
        print()
        books = get_books()
        _write_books(books)
        print()
    else:
        print('Add was unsuccessful')
        print()
        _write_books(books)
        print()
    successful = add_book('456', 'Kernighan', 'New Book')
    if successful:
        print('Add was successful')
        print()
        books = get_books()
        _write_books(books)
        print()
    else:
        print('Add was unsuccessful')
        print()
        _write_books(books)
        print()

    print('-----------------------------------')
    print('Testing delete_book()')
    print('-----------------------------------')
    print()
    delete_book('456')
    books = get_books()
    _write_books(books)
    print()
    delete_book('456')
    books = get_books()
    _write_books(books)

if __name__ == '__main__':
    _test()
```

PennyAdmin03aEscape/penny.py (Page 1 of 5)

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

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

import html
import flask
import database

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

app = flask.Flask(__name__)

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

def get_header():

    html_code = '''
        <hr>
        Hello, and welcome to <strong>PennyAdmin</strong>
        <hr>
        '''
    return html_code

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

def get_footer():

    html_code = '''
        <hr>
        Created by
        <a href="https://www.cs.princeton.edu/~rdondero">
        Bob Dondero</a>
        <hr>
        '''
    return html_code

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

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

    html_code = f'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                Click to <a href="/menu">begin</a>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

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

@app.route('/menu', methods=['GET'])
```

PennyAdmin03aEscape/penny.py (Page 2 of 5)

```
def menu():

    html_code = f'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                <a href="/show">Show all books</a><br>
                <a href="/add">Add a book</a><br>
                <a href="/delete">Delete a book by ISBN</a><br>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

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

def convert_to_html(books):

    if len(books) == 0:
        return '(None)'
    html_code = ''
    for book in books:
        html_code += f'''
            {html.escape(book['isbn'])}:
            <strong>{html.escape(book['author'])}</strong>:
            {html.escape(book['title'])}<br>
            '''
    return html_code

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

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

    books = database.get_books()

    html_code = f'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                <h1>Show All Books</h1>
                {convert_to_html(books)}
                <br>
                <a href="/menu">Return to menu page</a>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

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

```

PennyAdmin03aEscape/penny.py (Page 3 of 5)

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

    html_code = fr'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                <h1>Add a Book</h1>
                <form action="/handleadd" method="post">

                    Enter an ISBN:
                    <input type="text" name="isbn" autofocus
                        required pattern=".*\S.*"
                        title="At least one non-white-space char">
                    <br>

                    Enter an author:
                    <input type="text" name="author"
                        required pattern=".*\S.*"
                        title="At least one non-white-space char">
                    <br>

                    Enter a title:
                    <input type="text" name="title"
                        required pattern=".*\S.*"
                        title="At least one non-white-space char">
                    <br>

                    <input type="submit" value="Go">

                </form>

                <br>
                <a href="/menu">Return to menu page</a>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

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

@app.route('/handleadd', methods=['POST'])
def handle_add():

    isbn = flask.request.form.get('isbn')
    if (isbn is None) or (isbn.strip() == ''):
        message = 'The addition was unsuccessful: missing ISBN'
    else:
        author = flask.request.form.get('author')
        if (author is None) or (author.strip() == ''):
            message = 'The addition was unsuccessful: missing author'
        else:
            title = flask.request.form.get('title')
            if (title is None) or (title.strip() == ''):
                message = 'The addition was unsuccessful: missing title'
            else:
                isbn = isbn.strip()
                author = author.strip()
```

PennyAdmin03aEscape/penny.py (Page 4 of 5)

```
                title = title.strip()

                successful = database.add_book(isbn, author, title)
                if not successful:
                    message = 'The addition was unsuccessful: '
                    message += 'duplicate ISBN'
                else:
                    message = 'The addition was successful'

    html_code = f'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                <h1>Results</h1>
                {message}
                <br>
                <br>
                <br>
                <a href="/menu">Return to menu page</a>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

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

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

    html_code = fr'''
        <!DOCTYPE html>
        <html>
            <head>
                <title>PennyAdmin</title>
            </head>
            <body>
                {get_header()}
                <h1>Delete a Book</h1>
                <form action="/handledelete" method="post">
                    Enter an ISBN:
                    <input type="text" name="isbn" autofocus
                        required pattern=".*\S.*"
                        title="At least one non-white-space char">
                    <br>
                    <input type="submit" value="Go">
                </form>
                <br>
                <br>
                <a href="/menu">Return to menu page</a>
                {get_footer()}
            </body>
        </html>
        '''

    response = flask.make_response(html_code)
    return response

#-----------------------------------------------------------------------
```

PennyAdmin03aEscape/penny.py (Page 5 of 5)

```
261:
262: @app.route('/handledelete', methods=['POST'])
263: def handle_delete():
264:
265:     isbn = flask.request.form.get('isbn')
266:     if (isbn is None) or (isbn.strip() == ''):
267:         message = 'The deletion was unsuccessful: missing ISBN'
268:     else:
269:         isbn = isbn.strip()
270:         database.delete_book(isbn)
271:         message = 'The deletion was successful'
272:
273:     html_code = f'''
274:         <!DOCTYPE html>
275:         <html>
276:             <head>
277:                 <title>PennyAdmin</title>
278:             </head>
279:             <body>
280:                 {get_header()}
281:                 <h1>Results</h1>
282:                 {message}
283:                 <br>
284:                 <br>
285:                 <br>
286:                 <a href="/menu">Return to menu page</a>
287:                 {get_footer()}
288:             </body>
289:         </html>
290:         '''
291:     response = flask.make_response(html_code)
292:     return response
```

blank (Page 1 of 1)

1: This page is intentionally blank.

PennyAdmin03bJinja/header.html (Page 1 of 1)

```
1: <hr>Hello, and welcome to <strong>PennyAdmin</strong><hr>
```

PennyAdmin03bJinja/footer.html (Page 1 of 1)

```
1: <hr>
2: Created by <a href="https://www.cs.princeton.edu/~rdondero">
3: Bob Dondero</a>
4: <hr>
```

PennyAdmin03bJinja/index.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>PennyAdmin</title>
   </head>
   <body>
      <hr>Hello, and welcome to <strong>PennyAdmin</strong><hr>
      Click to <a href="/menu">begin</a>
      <hr>
      Created by <a href="https://www.cs.princeton.edu/~rdondero">
      Bob Dondero</a>
      <hr>
   </body>
</html>
```

PennyAdmin03bJinja/menu.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>PennyAdmin</title>
   </head>
   <body>
      {% include 'header.html' %}
      <br>
      <a href="/show">Show all books</a><br>
      <a href="/add">Add a book</a><br>
      <a href="/delete">Delete a book by ISBN</a><br>
      <br>
      {% include 'footer.html' %}
   </body>
</html>
```

PennyAdmin03bJinja/show.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>PennyAdmin</title>
   </head>
   <body>
      {% include 'header.html' %}
      <h1>Show All Books</h1>
      {% if books|length == 0: %}
         (None)
      {% else %}
         {% for book in books: %}
             {{book['isbn']}}:
             <strong>{{book['author']}}</strong>:
             {{book['title']}}<br>
         {% endfor %}
      {% endif %}
      <br>
      <a href="/menu">Return to menu page</a>
      <br>
      {% include 'footer.html' %}
   </body>
</html>
```

PennyAdmin03bJinja/add.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>PennyAdmin</title>
   </head>
   <body>
      {% include 'header.html' %}
      <h1>Add a Book</h1>
      <form action="/handleadd" method="post">
         Enter an ISBN:
         <input type="text" name="isbn" autofocus
            required pattern=".*\S.*"
            title="At least one non-white-space char">
         <br>

         Enter an author:
         <input type="text" name="author"
            required pattern=".*\S.*"
            title="At least one non-white-space char">
         <br>

         Enter a title:
         <input type="text" name="title"
            required pattern=".*\S.*"
            title="At least one non-white-space char">
         <br>

         <input type="submit" value="Go">
      </form>
      <br>
      <a href="/menu">Return to menu page</a>
      <br>
      {% include 'footer.html' %}
   </body>
</html>
```

PennyAdmin03bJinja/delete.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>PennyAdmin</title>
   </head>
   <body>
      {% include 'header.html' %}
      <h1>Delete a Book</h1>
      <form action="/handledelete" method="post">
         Enter an ISBN:
         <input type="text" name="isbn" autofocus
            required pattern=".*\S.*"
            title="At least one non-white-space char">
         <input type="submit" value="Go">
      </form>
      <br>
      <br>
      <a href="/menu">Return to menu page</a>
      <br>
      {% include 'footer.html' %}
   </body>
</html>
```

PennyAdmin03bJinja/reportresults.html (Page 1 of 1)

```
<!DOCTYPE html>
<html>
   <head>
      <title>PennyAdmin</title>
   </head>
   <body>
      {% include 'header.html' %}
      <h1>Results</h1>
      {{message}}
      <br>
      <br>
      <a href="/menu">Return to menu page</a>
      <br>
      <br>
      {% include 'footer.html' %}
   </body>
</html>
```

PennyAdmin03bJinja/penny.py (Page 1 of 2)

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

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

import flask
import database

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

app = flask.Flask(__name__, template_folder='.')

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

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

    html_code = flask.render_template('index.html')
    response = flask.make_response(html_code)
    return response

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

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

    html_code = flask.render_template('menu.html')
    response = flask.make_response(html_code)
    return response

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

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

    books = database.get_books()
    html_code = flask.render_template('show.html', books=books)
    response = flask.make_response(html_code)
    return response

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

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

    html_code = flask.render_template('add.html')
    response = flask.make_response(html_code)
    return response

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

@app.route('/handleadd', methods=['POST'])
def handle_add():

    isbn = flask.request.form.get('isbn')
    if (isbn is None) or (isbn.strip() == ''):
        message = 'The addition was unsuccessful: missing ISBN'
    else:
        author = flask.request.form.get('author')
        if (author is None) or (author.strip() == ''):
            message = 'The addition was unsuccessful: missing author'
        else:
```

PennyAdmin03bJinja/penny.py (Page 2 of 2)

```
            title = flask.request.form.get('title')
            if (title is None) or (title.strip() == ''):
                message = 'The addition was unsuccessful: missing title'
            else:
                isbn = isbn.strip()
                author = author.strip()
                title = title.strip()

                successful = database.add_book(isbn, author, title)
                if not successful:
                    message = 'The addition was unsuccessful: '
                    message += 'duplicate ISBN'
                else:
                    message = 'The addition was successful'

    html_code = flask.render_template('reportresults.html',
        message=message)
    response = flask.make_response(html_code)
    return response

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

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

    html_code = flask.render_template('delete.html')
    response = flask.make_response(html_code)
    return response

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

@app.route('/handledelete', methods=['POST'])
def handle_delete():

    isbn = flask.request.form.get('isbn')
    if (isbn is None) or (isbn.strip() == ''):
        message = 'The deletion was unsuccessful: missing ISBN'
    else:
        isbn = isbn.strip()
        database.delete_book(isbn)
        message = 'The deletion was successful'

    html_code = flask.render_template('reportresults.html',
        message=message)
    response = flask.make_response(html_code)
    return response
```