

## PennyAdmin01Baseline/runserver.py (Page 1 of 1)

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

## PennyAdmin01Baseline/penny.sql (Page 1 of 1)

```
1: DROP TABLE IF EXISTS books;
2: CREATE TABLE books (isbn TEXT PRIMARY KEY, author TEXT, title TEXT);
3: INSERT INTO books (isbn, author, title)
4:     VALUES ('123', 'Kernighan', 'The Practice of Programming');
5: INSERT INTO books (isbn, author, title)
6:     VALUES ('234', 'Kernighan', 'The C Programming Language');
7: INSERT INTO books (isbn, author, title)
8:     VALUES ('345', 'Sedgewick', 'Algorithms in C');
```

## PennyAdmin01Baseline/database.py (Page 1 of 2)

```

1:#!/usr/bin/env python
2:
3:-----
4:# database.py
5:# Author: Bob Dondero
6:-----
7:
8:import sqlite3
9:import contextlib
10:
11:-----
12:
13:_DATABASE_URL = 'file:penny.sqlite'
14:
15:-----
16:
17:def get_books():
18:
19:    books = []
20:
21:    with contextlib.closing(
22:        sqlite3.connect(_DATABASE_URL + '?mode=ro',
23:            isolation_level=None, uri=True)) as connection:
24:        with contextlib.closing(connection.cursor()) as cursor:
25:            query_str = '''
26:                SELECT isbn, author, title FROM books
27:            '''
28:            cursor.execute(query_str)
29:
30:            table = cursor.fetchall()
31:            for row in table:
32:                book = {'isbn': row[0], 'author': row[1],
33:                    'title': row[2]}
34:                books.append(book)
35:
36:    return books
37:
38:-----
39:
40:def add_book(isbn, author, title):
41:
42:    with contextlib.closing(
43:        sqlite3.connect(_DATABASE_URL + '?mode=rw',
44:            isolation_level=None, uri=True)) as connection:
45:        with contextlib.closing(connection.cursor()) as cursor:
46:            query_str = f'''
47:                INSERT INTO books (isbn, author, title)
48:                VALUES ('{isbn}', '{author}', '{title}')
49:            '''
50:            try:
51:                cursor.execute(query_str)
52:            except sqlite3.IntegrityError:
53:                return False
54:            return True
55:
56:-----
57:
58:def delete_book(isbn):
59:
60:    with contextlib.closing(
61:        sqlite3.connect(_DATABASE_URL + '?mode=rw',
62:            isolation_level=None, uri=True)) as connection:
63:        with contextlib.closing(connection.cursor()) as cursor:
64:            query_str = f'''
65:                DELETE FROM books WHERE isbn = '{isbn}'

```

## PennyAdmin01Baseline/database.py (Page 2 of 2)

```

66:            '''
67:            cursor.execute(query_str)
68:
69:-----
70:
71:# For testing:
72:
73:def _write_books(books):
74:    for book in books:
75:        print(f'{book["isbn"]} | {book["author"]} | {book["title"]}')
76:
77:def _test():
78:    print('-----')
79:    print('Testing get_books()')
80:    print('-----')
81:    print()
82:    books = get_books()
83:    _write_books(books)
84:    print()
85:
86:    print('-----')
87:    print('Testing add_book()')
88:    print('-----')
89:    print()
90:    successful = add_book('456', 'Kernighan', 'New Book')
91:    if successful:
92:        print('Add was successful')
93:        print()
94:        books = get_books()
95:        _write_books(books)
96:        print()
97:    else:
98:        print('Add was unsuccessful')
99:        print()
100:        _write_books(books)
101:        print()
102:    successful = add_book('456', 'Kernighan', 'New Book')
103:    if successful:
104:        print('Add was successful')
105:        print()
106:        books = get_books()
107:        _write_books(books)
108:        print()
109:    else:
110:        print('Add was unsuccessful')
111:        print()
112:        _write_books(books)
113:        print()
114:
115:    print('-----')
116:    print('Testing delete_book()')
117:    print('-----')
118:    print()
119:    delete_book('456')
120:    books = get_books()
121:    _write_books(books)
122:    print()
123:    delete_book('456')
124:    books = get_books()
125:    _write_books(books)
126:
127:if __name__ == '__main__':
128:    _test()

```

## PennyAdmin01Baseline/penny.py (Page 1 of 5)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # penny.py
5: # Author: Bob Dondero
6: #-----
7:
8: import flask
9: import database
10:
11: #-----
12:
13: app = flask.Flask(__name__)
14:
15: #-----
16:
17: def get_header():
18:
19:     html_code = '''
20:         <hr>
21:         Hello, and welcome to <strong>PennyAdmin</strong>
22:         <hr>
23:         '''
24:     return html_code
25:
26: #-----
27:
28: def get_footer():
29:
30:     html_code = '''
31:         <hr>
32:         Created by
33:         <a href="https://www.cs.princeton.edu/~rdondero">
34:         Bob Dondero</a>
35:         <hr>
36:         '''
37:     return html_code
38:
39: #-----
40:
41: @app.route('/', methods=['GET'])
42: @app.route('/index', methods=['GET'])
43: def index():
44:
45:     html_code = f'''
46:         <!DOCTYPE html>
47:         <html>
48:             <head>
49:                 <title>PennyAdmin</title>
50:             </head>
51:             <body>
52:                 {get_header()}
53:                 Click to <a href="/menu">begin</a>
54:                 {get_footer()}
55:             </body>
56:         </html>
57:         '''
58:
59:     response = flask.make_response(html_code)
60:     return response
61:
62: #-----
63:
64: @app.route('/menu', methods=['GET'])
65: def menu():

```

## PennyAdmin01Baseline/penny.py (Page 2 of 5)

```

66:
67:     html_code = f'''
68:         <!DOCTYPE html>
69:         <html>
70:             <head>
71:                 <title>PennyAdmin</title>
72:             </head>
73:             <body>
74:                 {get_header()}
75:                 <a href="/show">Show all books</a><br>
76:                 <a href="/add">Add a book</a><br>
77:                 <a href="/delete">Delete a book by ISBN</a><br>
78:                 {get_footer()}
79:             </body>
80:         </html>
81:         '''
82:
83:     response = flask.make_response(html_code)
84:     return response
85:
86: #-----
87:
88: def convert_to_html(books):
89:
90:     if len(books) == 0:
91:         return '(None)'
92:     html_code = ''
93:     for book in books:
94:         html_code += f'''
95:             {book['isbn']}: <strong>{book['author']}</strong>:
96:             {book['title']}<br>
97:             '''
98:     return html_code
99:
100: #-----
101:
102: @app.route('/show', methods=['GET'])
103: def show():
104:
105:     books = database.get_books()
106:
107:     html_code = f'''
108:         <!DOCTYPE html>
109:         <html>
110:             <head>
111:                 <title>PennyAdmin</title>
112:             </head>
113:             <body>
114:                 {get_header()}
115:                 <h1>Show All Books</h1>
116:                 {convert_to_html(books)}
117:                 <br>
118:                 <a href="/menu">Return to menu page</a>
119:                 {get_footer()}
120:             </body>
121:         </html>
122:         '''
123:
124:     response = flask.make_response(html_code)
125:     return response
126:
127: #-----
128:
129: @app.route('/add', methods=['GET'])
130: 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)

```

1:#!/usr/bin/env python
2:
3:-----
4:# database.py
5:# Author: Bob Dondero
6:-----
7:
8:import sqlite3
9:import contextlib
10:
11:-----
12:
13:_DATABASE_URL = 'file:penny.sqlite'
14:
15:-----
16:
17:def get_books():
18:
19:    books = []
20:
21:    with contextlib.closing(
22:        sqlite3.connect(_DATABASE_URL + '?mode=ro',
23:            isolation_level=None, uri=True)) as connection:
24:        with contextlib.closing(connection.cursor()) as cursor:
25:            query_str = '''
26:                SELECT isbn, author, title FROM books
27:            '''
28:            cursor.execute(query_str)
29:
30:            table = cursor.fetchall()
31:            for row in table:
32:                book = {'isbn': row[0], 'author': row[1],
33:                    'title': row[2]}
34:                books.append(book)
35:
36:    return books
37:
38:-----
39:
40:def add_book(isbn, author, title):
41:
42:    with contextlib.closing(
43:        sqlite3.connect(_DATABASE_URL + '?mode=rw',
44:            isolation_level=None, uri=True)) as connection:
45:        with contextlib.closing(connection.cursor()) as cursor:
46:            query_str = '''
47:                INSERT INTO books (isbn, author, title)
48:                VALUES (?, ?, ?)
49:            '''
50:            try:
51:                cursor.execute(query_str, [isbn, author, title])
52:            except sqlite3.IntegrityError:
53:                return False
54:            return True
55:
56:-----
57:
58:def delete_book(isbn):
59:    with contextlib.closing(
60:        sqlite3.connect(_DATABASE_URL + '?mode=rw',
61:            isolation_level=None, uri=True)) as connection:
62:        with contextlib.closing(connection.cursor()) as cursor:
63:
64:            query_str = '''
65:                DELETE FROM books WHERE isbn = ?

```

## PennyAdmin02aPrepared/database.py (Page 2 of 2)

```

66:        '''
67:        cursor.execute(query_str, [isbn])
68:
69:-----
70:
71:# For testing:
72:
73:def _write_books(books):
74:    for book in books:
75:        print(f'{book["isbn"]} | {book["author"]} | {book["title"]}')
76:
77:def _test():
78:    print('-----')
79:    print('Testing get_books()')
80:    print('-----')
81:    print()
82:    books = get_books()
83:    _write_books(books)
84:    print()
85:
86:    print('-----')
87:    print('Testing add_book()')
88:    print('-----')
89:    print()
90:    successful = add_book('456', 'Kernighan', 'New Book')
91:    if successful:
92:        print('Add was successful')
93:        print()
94:        books = get_books()
95:        _write_books(books)
96:        print()
97:    else:
98:        print('Add was unsuccessful')
99:        print()
100:        _write_books(books)
101:        print()
102:    successful = add_book('456', 'Kernighan', 'New Book')
103:    if successful:
104:        print('Add was successful')
105:        print()
106:        books = get_books()
107:        _write_books(books)
108:        print()
109:    else:
110:        print('Add was unsuccessful')
111:        print()
112:        _write_books(books)
113:        print()
114:
115:    print('-----')
116:    print('Testing delete_book()')
117:    print('-----')
118:    print()
119:    delete_book('456')
120:    books = get_books()
121:    _write_books(books)
122:    print()
123:    delete_book('456')
124:    books = get_books()
125:    _write_books(books)
126:
127:if __name__ == '__main__':
128:    _test()

```

## PennyAdmin02bAlchemy/database.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # database.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9: import sqlalchemy
10: import sqlalchemy.orm
11: import dotenv
12:
13: #-----
14:
15: dotenv.load_dotenv()
16: _DATABASE_URL = os.getenv('DATABASE_URL', 'sqlite:///penny.sqlite')
17: _DATABASE_URL = _DATABASE_URL.replace('postgres://', 'postgresql://')
18: _DATABASE_URL = _DATABASE_URL.replace(
19:     'postgresql://', 'postgresql+psycopg://')
20:
21: #-----
22:
23: class Base(sqlalchemy.orm.DeclarativeBase):
24:     pass
25:
26: class Book(Base):
27:     __tablename__ = 'books'
28:     isbn = sqlalchemy.Column(sqlalchemy.String, primary_key=True)
29:     author = sqlalchemy.Column(sqlalchemy.String)
30:     title = sqlalchemy.Column(sqlalchemy.String)
31:
32: _engine = sqlalchemy.create_engine(_DATABASE_URL)
33:
34: #-----
35:
36: def get_books():
37:
38:     books = []
39:
40:     with sqlalchemy.orm.Session(_engine) as session:
41:         query = session.query(Book)
42:         table = query.all()
43:         for row in table:
44:             book = {'isbn': row.isbn, 'author': row.author,
45:                   'title': row.title}
46:             books.append(book)
47:
48:     return books
49:
50: #-----
51:
52: def add_book(isbn, author, title):
53:
54:     with sqlalchemy.orm.Session(_engine) as session:
55:         row = Book(isbn=isbn, author=author, title=title)
56:         session.add(row)
57:         try:
58:             session.commit()
59:             return True
60:         except sqlalchemy.exc.IntegrityError:
61:             return False
62:
63: #-----
64:
65: def delete_book(isbn):

```

## PennyAdmin02bAlchemy/database.py (Page 2 of 2)

```

66:
67:     with sqlalchemy.orm.Session(_engine) as session:
68:         session.query(Book).filter(Book.isbn==isbn).delete()
69:         session.commit()
70:
71: #-----
72:
73: # For testing:
74:
75: def _write_books(books):
76:     for book in books:
77:         print(f'{book["isbn"]} | {book["author"]} | {book["title']}')
78:
79: def _test():
80:     print('-----')
81:     print('Testing get_books()')
82:     print('-----')
83:     print()
84:     books = get_books()
85:     _write_books(books)
86:     print()
87:
88:     print('-----')
89:     print('Testing add_book()')
90:     print('-----')
91:     print()
92:     successful = add_book('456', 'Kernighan', 'New Book')
93:     if successful:
94:         print('Add was successful')
95:         print()
96:         books = get_books()
97:         _write_books(books)
98:         print()
99:     else:
100:         print('Add was unsuccessful')
101:         print()
102:         _write_books(books)
103:         print()
104:     successful = add_book('456', 'Kernighan', 'New Book')
105:     if successful:
106:         print('Add was successful')
107:         print()
108:         books = get_books()
109:         _write_books(books)
110:         print()
111:     else:
112:         print('Add was unsuccessful')
113:         print()
114:         _write_books(books)
115:         print()
116:
117:     print('-----')
118:     print('Testing delete_book()')
119:     print('-----')
120:     print()
121:     delete_book('456')
122:     books = get_books()
123:     _write_books(books)
124:     print()
125:     delete_book('456')
126:     books = get_books()
127:     _write_books(books)
128:
129: if __name__ == '__main__':
130:     _test()

```

## PennyAdmin03aEscape/penny.py (Page 1 of 5)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # penny.py
5: # Author: Bob Dondero
6: #-----
7:
8: import html
9: import flask
10: import database
11:
12: #-----
13:
14: app = flask.Flask(__name__)
15:
16: #-----
17:
18: def get_header():
19:
20:     html_code = '''
21:         <hr>
22:         Hello, and welcome to <strong>PennyAdmin</strong>
23:         <hr>
24:         '''
25:     return html_code
26:
27: #-----
28:
29: def get_footer():
30:
31:     html_code = '''
32:         <hr>
33:         Created by
34:         <a href="https://www.cs.princeton.edu/~rdondero">
35:         Bob Dondero</a>
36:         <hr>
37:         '''
38:     return html_code
39:
40: #-----
41:
42: @app.route('/', methods=['GET'])
43: @app.route('/index', methods=['GET'])
44: def index():
45:
46:     html_code = f'''
47:     <!DOCTYPE html>
48:     <html>
49:         <head>
50:             <title>PennyAdmin</title>
51:         </head>
52:         <body>
53:             {get_header()}
54:             Click to <a href="/menu">begin</a>
55:             {get_footer()}
56:         </body>
57:     </html>
58:     '''
59:
60:     response = flask.make_response(html_code)
61:     return response
62:
63: #-----
64:
65: @app.route('/menu', methods=['GET'])

```

## PennyAdmin03aEscape/penny.py (Page 2 of 5)

```

66: def menu():
67:
68:     html_code = f'''
69:     <!DOCTYPE html>
70:     <html>
71:         <head>
72:             <title>PennyAdmin</title>
73:         </head>
74:         <body>
75:             {get_header()}
76:             <a href="/show">Show all books</a><br>
77:             <a href="/add">Add a book</a><br>
78:             <a href="/delete">Delete a book by ISBN</a><br>
79:             {get_footer()}
80:         </body>
81:     </html>
82:     '''
83:
84:     response = flask.make_response(html_code)
85:     return response
86:
87: #-----
88:
89: def convert_to_html(books):
90:
91:     if len(books) == 0:
92:         return '(None)'
93:     html_code = ''
94:     for book in books:
95:         html_code += f'''
96:             {html.escape(book['isbn'])}:
97:             <strong>{html.escape(book['author'])}</strong>:
98:             {html.escape(book['title'])}<br>
99:             '''
100:     return html_code
101:
102: #-----
103:
104: @app.route('/show', methods=['GET'])
105: def show():
106:
107:     books = database.get_books()
108:
109:     html_code = f'''
110:     <!DOCTYPE html>
111:     <html>
112:         <head>
113:             <title>PennyAdmin</title>
114:         </head>
115:         <body>
116:             {get_header()}
117:             <h1>Show All Books</h1>
118:             {convert_to_html(books)}
119:             <br>
120:             <a href="/menu">Return to menu page</a>
121:             {get_footer()}
122:         </body>
123:     </html>
124:     '''
125:
126:     response = flask.make_response(html_code)
127:     return response
128:
129: #-----
130:

```

## PennyAdmin03aEscape/penny.py (Page 3 of 5)

```

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

```

## PennyAdmin03aEscape/penny.py (Page 4 of 5)

```

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

```

## 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 = '''
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)

```
1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>PennyAdmin</title>
5:   </head>
6:   <body>
7:     <hr>Hello, and welcome to <strong>PennyAdmin</strong><hr>
8:     Click to <a href="/menu">begin</a>
9:     <hr>
10:    Created by <a href="https://www.cs.princeton.edu/~rdondero">
11:      Bob Dondero</a>
12:    <hr>
13:  </body>
14: </html>
```

## PennyAdmin03bJinja/menu.html (Page 1 of 1)

```
1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>PennyAdmin</title>
5:   </head>
6:   <body>
7:     {% include 'header.html' %}
8:     <br>
9:     <a href="/show">Show all books</a><br>
10:    <a href="/add">Add a book</a><br>
11:    <a href="/delete">Delete a book by ISBN</a><br>
12:    <br>
13:    {% include 'footer.html' %}
14:  </body>
15: </html>
```

## PennyAdmin03bJinja/show.html (Page 1 of 1)

```

1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>PennyAdmin</title>
5:   </head>
6:   <body>
7:     {% include 'header.html' %}
8:     <h1>Show All Books</h1>
9:     {% if books|length == 0: %}
10:      (None)
11:     {% else %}
12:       {% for book in books: %}
13:         {{book['isbn']}}:
14:         <strong>{{book['author']}}</strong>:
15:         {{book['title']}}<br>
16:       {% endfor %}
17:     {% endif %}
18:     <br>
19:     <a href="/menu">Return to menu page</a>
20:     <br>
21:     {% include 'footer.html' %}
22:   </body>
23: </html>

```

## PennyAdmin03bJinja/add.html (Page 1 of 1)

```

1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>PennyAdmin</title>
5:   </head>
6:   <body>
7:     {% include 'header.html' %}
8:     <h1>Add a Book</h1>
9:     <form action="/handleadd" method="post">
10:      Enter an ISBN:
11:      <input type="text" name="isbn" autofocus
12:        required pattern=".*\S.*"
13:        title="At least one non-white-space char">
14:      <br>
15:      Enter an author:
16:      <input type="text" name="author"
17:        required pattern=".*\S.*"
18:        title="At least one non-white-space char">
19:      <br>
20:      Enter a title:
21:      <input type="text" name="title"
22:        required pattern=".*\S.*"
23:        title="At least one non-white-space char">
24:      <br>
25:      <input type="submit" value="Go">
26:    </form>
27:    <br>
28:    <a href="/menu">Return to menu page</a>
29:    <br>
30:    {% include 'footer.html' %}
31:  </body>
32: </html>

```

## PennyAdmin03bJinja/delete.html (Page 1 of 1)

```
1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>PennyAdmin</title>
5:   </head>
6:   <body>
7:     {% include 'header.html' %}
8:     <h1>Delete a Book</h1>
9:     <form action="/handledelete" method="post">
10:      Enter an ISBN:
11:      <input type="text" name="isbn" autofocus
12:        required pattern=".*\S.*"
13:        title="At least one non-white-space char">
14:      <input type="submit" value="Go">
15:    </form>
16:    <br>
17:    <br>
18:    <a href="/menu">Return to menu page</a>
19:    <br>
20:    {% include 'footer.html' %}
21:  </body>
22: </html>
```

## PennyAdmin03bJinja/reportresults.html (Page 1 of 1)

```
1: <!DOCTYPE html>
2: <html>
3:   <head>
4:     <title>PennyAdmin</title>
5:   </head>
6:   <body>
7:     {% include 'header.html' %}
8:     <h1>Results</h1>
9:     {{message}}
10:    <br>
11:    <br>
12:    <a href="/menu">Return to menu page</a>
13:    <br>
14:    <br>
15:    {% include 'footer.html' %}
16:  </body>
17: </html>
```

## PennyAdmin03bJinja/penny.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # penny.py
5: # Author: Bob Dondero
6: #-----
7:
8: import flask
9: import database
10:
11: #-----
12:
13: app = flask.Flask(__name__, template_folder='.')
14:
15: #-----
16:
17: @app.route('/', methods=['GET'])
18: @app.route('/index', methods=['GET'])
19: def index():
20:
21:     html_code = flask.render_template('index.html')
22:     response = flask.make_response(html_code)
23:     return response
24:
25: #-----
26:
27: @app.route('/menu', methods=['GET'])
28: def menu():
29:
30:     html_code = flask.render_template('menu.html')
31:     response = flask.make_response(html_code)
32:     return response
33:
34: #-----
35:
36: @app.route('/show', methods=['GET'])
37: def show():
38:
39:     books = database.get_books()
40:     html_code = flask.render_template('show.html', books=books)
41:     response = flask.make_response(html_code)
42:     return response
43:
44: #-----
45:
46: @app.route('/add', methods=['GET'])
47: def add():
48:
49:     html_code = flask.render_template('add.html')
50:     response = flask.make_response(html_code)
51:     return response
52:
53: #-----
54:
55: @app.route('/handleadd', methods=['POST'])
56: def handle_add():
57:
58:     isbn = flask.request.form.get('isbn')
59:     if (isbn is None) or (isbn.strip() == ''):
60:         message = 'The addition was unsuccessful: missing ISBN'
61:     else:
62:         author = flask.request.form.get('author')
63:         if (author is None) or (author.strip() == ''):
64:             message = 'The addition was unsuccessful: missing author'
65:         else:

```

## PennyAdmin03bJinja/penny.py (Page 2 of 2)

```

66:         title = flask.request.form.get('title')
67:         if (title is None) or (title.strip() == ''):
68:             message = 'The addition was unsuccessful: missing title'
69:         else:
70:             isbn = isbn.strip()
71:             author = author.strip()
72:             title = title.strip()
73:
74:             successful = database.add_book(isbn, author, title)
75:             if not successful:
76:                 message = 'The addition was unsuccessful: '
77:                 message += 'duplicate ISBN'
78:             else:
79:                 message = 'The addition was successful'
80:
81:             html_code = flask.render_template('reportresults.html',
82:                 message=message)
83:             response = flask.make_response(html_code)
84:             return response
85:
86: #-----
87:
88: @app.route('/delete', methods=['GET'])
89: def delete():
90:
91:     html_code = flask.render_template('delete.html')
92:     response = flask.make_response(html_code)
93:     return response
94:
95: #-----
96:
97: @app.route('/handledelete', methods=['POST'])
98: def handle_delete():
99:
100:     isbn = flask.request.form.get('isbn')
101:     if (isbn is None) or (isbn.strip() == ''):
102:         message = 'The deletion was unsuccessful: missing ISBN'
103:     else:
104:         isbn = isbn.strip()
105:         database.delete_book(isbn)
106:         message = 'The deletion was successful'
107:
108:     html_code = flask.render_template('reportresults.html',
109:         message=message)
110:     response = flask.make_response(html_code)
111:     return response

```