

Module Five Assignment

August 3, 2025

```
[1]: # Setup: Ensure the animal_shelter module is in the Python path
import sys
from pathlib import Path

candidate = Path.cwd()
project_root = None
# Search up for the project root to ensure module can be imported
for _ in range(6):
    if (candidate / "animal_shelter").is_dir():
        project_root = candidate
        break
    candidate = candidate.parent

if project_root and str(project_root) not in sys.path:
    sys.path.insert(0, str(project_root))
```

```
[2]: # Import necessary libraries
from dash import Dash, dcc, html, dash_table
from dash.dependencies import Input, Output
import pandas as pd
import qrcode
import base64
from io import BytesIO

# Import the CRUD module
from animal_shelter.animal_shelter import AnimalShelter

# Build App
app = Dash(__name__)

# Professional styling
app.layout = html.Div([
    # Header
    html.Div([
        html.H1("Walter Mobley - CS 340 Dashboard",
            style={
                'textAlign': 'center',
                'color': '#2c3e50',
```

```

        'fontFamily': 'Segoe UI, Arial, sans-serif',
        'marginBottom': '10px',
        'fontWeight': '300'
    }),
    html.P("AAC Database Query Interface",
        style={
            'textAlign': 'center',
            'color': '#7f8c8d',
            'fontSize': '16px'
        })
], style={'padding': '30px 20px', 'borderBottom': '1px solid #ecf0f1'}),

# Login form
html.Div([
    html.Div([
        dcc.Input(
            id="input_user",
            type="text",
            placeholder="Username",
            style={
                'width': '100%',
                'padding': '12px',
                'fontSize': '14px',
                'border': '1px solid #ddd',
                'borderRadius': '4px',
                'marginBottom': '15px'
            }
        ),
        dcc.Input(
            id="input_passwd",
            type="password",
            placeholder="Password",
            style={
                'width': '100%',
                'padding': '12px',
                'fontSize': '14px',
                'border': '1px solid #ddd',
                'borderRadius': '4px',
                'marginBottom': '20px'
            }
        ),
        html.Button(
            'Query Database',
            id='submit-val',
            n_clicks=0,
            style={
                'width': '100%',

```

```

        'padding': '12px',
        'fontSize': '14px',
        'backgroundColor': '#3498db',
        'color': 'white',
        'border': 'none',
        'borderRadius': '4px',
        'cursor': 'pointer'
    }
)
], style={
    'maxWidth': '300px',
    'margin': '0 auto',
    'padding': '20px'
})
]),

# Results
html.Div(id="query-out", style={'padding': '20px'}),

# QR Code (subtle)
html.Div(id="qr-code", style={'textAlign': 'center', 'padding': '20px'})

], style={
    'fontFamily': 'Segoe UI, Arial, sans-serif',
    'backgroundColor': '#fafafa',
    'minHeight': '100vh'
})

def generate_qr_code():
    """Generate QR code with professional message"""
    message = "CS 340 Project - Walter Mobley"
    qr = qrcode.QRCode(version=1, box_size=8, border=4)
    qr.add_data(message)
    qr.make(fit=True)

    img = qr.make_image(fill_color="black", back_color="white")
    buffer = BytesIO()
    img.save(buffer, format='PNG')
    buffer.seek(0)
    img_str = base64.b64encode(buffer.getvalue()).decode()

    return f"data:image/png;base64,{img_str}"

@app.callback(
    Output('query-out', 'children'),
    [Input('input_user', 'value'),
     Input('input_passwd', 'value')],

```

```

        Input('submit-val', 'n_clicks')])
    )
def update_output(username, password, n_clicks):
    if n_clicks > 0:
        if not username or not password:
            return html.Div("Please enter both username and password.",
                             style={'color': '#e74c3c', 'textAlign': 'center',
                                     ↪'padding': '20px'})

        try:
            shelter = AnimalShelter(user=username, password=password)
            query = {"animal_type": "Dog", "name": "Lucy"}
            results = shelter.read(query)

            if not results:
                return html.Div("No records found matching the query.",
                                 style={'textAlign': 'center', 'padding': '20px',
                                         ↪'color': '#7f8c8d'})

            df = pd.DataFrame(results)
            if '_id' in df.columns:
                df = df.drop(columns=['_id'])

            return html.Div([
                html.H3(f"Query Results ({len(results)} record{'s' if
                                     ↪len(results) != 1 else ''})",
                           style={'color': '#2c3e50', 'marginBottom': '20px'}),
                dash_table.DataTable(
                    data=df.to_dict('records'),
                    columns=[{"name": col.replace('_', ' ').title(), "id": col}
                               ↪for col in df.columns],
                    style_cell={'textAlign': 'left', 'padding': '10px',
                               ↪'fontSize': '14px'},
                    style_header={'backgroundColor': '#f8f9fa', 'fontWeight':
                               ↪'bold'},
                    style_data={'backgroundColor': 'white'},
                    style_table={'overflowX': 'auto'}
                )
            ], style={'maxWidth': '1000px', 'margin': '0 auto'})

        except Exception as e:
            return html.Div(f"Database connection failed: {str(e)}",
                             style={'color': '#e74c3c', 'textAlign': 'center',
                                     ↪'padding': '20px'})

    return html.Div("Enter credentials to query the database.",

```

```

        style={'textAlign': 'center', 'padding': '20px', 'color': '
↪'#7f8c8d'})

@app.callback(
    Output('qr-code', 'children'),
    Input('submit-val', 'n_clicks')
)
def show_qr_code(n_clicks):
    if n_clicks > 0:
        qr_img = generate_qr_code()
        return html.Img(src=qr_img, style={'maxWidth': '150px', 'opacity': '0.
↪7'})
    return html.Div()

app.run(jupyter_mode='external')

```

Dash app running on <http://127.0.0.1:8050/>