APCSP Create Task:

Github: https://github.com/ikoide/MagicLogger
Live Site: https://project.iankoide.dev

This code was pasted on 5/4/20. For the most updated version of the code check out the Github repository.

I did not include any database files as they are not readable although the Magic the Gathering card database can be found here https://mtgjson.com/downloads/all-files/. This site's user database as well as card collection database are stored locally on the server.

Code:

routes.py

```
import secrets
import os
from PIL import Image
from flask import render_template, url_for, flash, redirect, request,
abort, Markup, send_from_directory
from mtglogger import app, db, bcrypt
from flask_login import login_user, current_user, logout_user,
login_required
from mtglogger.forms import SearchForm, LoginForm, RegistrationForm,
AddForm
from mtglogger.models import User, Card
from mtglogger.scripts.card import by_name, by_multiverseId, return_card

@app.errorhandler(404)
def page_not_found(e):
   return render_template('errors/404.html', title="404 Error"), 404

@app.route("/", methods=['GET', 'POST'])
def home():
  return render_template("index.html", title="Home")

@app.route("/search/<search_query>", methods=['GET', 'POST'])
```

```
def results(search_query):
 cards = by_name(search_query)
  return render_template("results.html", title="Search", cards=cards)

# Auth

@app.route("/login", methods=['GET', 'POST'])
def login():
 if current_user.is_authenticated:
   flash('Your already logged in.', 'info')
   return redirect(url_for('home'))

 loginForm = LoginForm()

 if loginForm.validate_on_submit() and loginForm.submit.data:
   user = User.query.filter_by(username=loginForm.username.data).first()
   if user and bcrypt.check_password_hash(user.password,
loginForm.password.data):
     login_user(user)
     next_page = request.args.get('next')
     message = Markup('Login successful! Welcome back <b>' +
current_user.username + '</b>.')
     flash(message, 'success')
     return redirect(next_page) if next_page else
redirect(url_for('home'))
   else:
     flash('Login unsuccessful, please check your username and password.',
'danger')


 return render_template("auth/login.html", title="Login",
loginForm=loginForm)

@app.route("/register", methods=['GET', 'POST'])
def register():
 if current_user.is_authenticated:
   flash('Your already logged in.', 'info')
   return redirect(url_for('home'))

 registrationForm = RegistrationForm()

 if registrationForm.validate_on_submit():
   hashed_password = bcrypt.generate_password_hash(registrationForm.password.data).decode('utf-8')
   user = User(username=registrationForm.username.data, email=registrationForm.email.data, password=hashed_password)
   db.session.add(user)
   db.session.commit()
   flash('Your account has been created! You are now able to login.', 'success')
   return redirect(url_for('login'))

 return render_template("auth/register.html", title="Register", registrationForm=registrationForm)

@app.route("/collection", methods=['GET', 'POST'])
def collection():
 act_cards = []
 for i in current_user.cards:
   card = return_card(i.multiverse_id)
   card["quantity"] = i.quantity
   act_cards.append(card)
 return render_template("collection.html", title="Collection", act_cards=act_cards)

@app.route("/card/<multiverse_id>", methods=['POST', 'GET'])
def card(multiverse_id):
 card = return_card(multiverse_id)
 if card:
   page_title = card["name"]
 else:
   page_title = "Not Valid Card"
 form = AddForm()
 if form.validate_on_submit():
   if not current_user.is_authenticated:
     flash('You must login to add cards to your collection.', 'warning')
     return redirect(url_for('login'))
   else:
     card = Card(multiverse_id=multiverse_id, quantity=form.quantity.data,
owner=current_user)
     db.session.add(card)
     db.session.commit()
     return redirect(url_for('collection'))

 return render_template("card.html", title=page_title, form=form,
card=card)
```

cards.py

```
import sqlite3

def by_name(card_name):
 conn =
sqlite3.connect('/home/lan/Documents/MagicLogger/mtglogger/cards.sqlite')
 c = conn.cursor()
 c.execute("SELECT name, setCode, multiverseId FROM cards WHERE name=?",
(card_name,))
 rows = c.fetchall()
 conn.close()
 if len(rows) > 0:
   cards = rows
 else:
   cards = [['Card Not Found', '']]
 new_cards = []
 for i in cards:
   if i[2]:
     image_url =
str('https://gatherer.wizards.com/Handlers/Image.ashx?multiverseid=' +
str(i[2]) + '&type=card')
     i = i + (image_url, )
     new_cards.append(i)
 return new_cards

def by_multiverseId(id):
 conn =
sqlite3.connect('/home/lan/Documents/MagicLogger/mtglogger/cards.sqlite')
 c = conn.cursor()
 c.execute("SELECT name, setCode, multiverseId FROM cards WHERE
multiverseId=?", (id,))
 rows = c.fetchall()
 if len(rows) > 0:
   card = rows[0]
 image_url =
str('https://gatherer.wizards.com/Handlers/Image.ashx?multiverseid=' +
str(card[2]) + '&type=card')
 card = card + (image_url, )

 return card

def return_card(id):
 conn =
sqlite3.connect('/home/lan/Documents/MagicLogger/mtglogger/cards.sqlite')
 c = conn.cursor()
 c.execute("SELECT name, setCode, multiverseId, rarity, originalType,
power, toughness, text, flavorText, type, manaCost FROM cards WHERE
multiverseId=?", (id,))
 rows = c.fetchall()
 if len(rows) > 0:
   notDictCard = rows[0]

   card = {}
   card["name"] = notDictCard[0]
   card["set"] = notDictCard[1]
   card["id"] = notDictCard[2]
   card["rarity"] = notDictCard[3]
   card["originalType"] = notDictCard[4]
   card["type"] = notDictCard[9]
   card["power"] = notDictCard[5]
   card["toughness"] = notDictCard[6]
   card["text"] = notDictCard[7]
   card["flavor"] = notDictCard[8]
   card["image_url"] =
str('https://gatherer.wizards.com/Handlers/Image.ashx?multiverseid=' +
str(card["id"]) + '&type=card')
   card["manaCost"] = notDictCard[10]
 else:
   card = None
 return card
```

models.py

```
from datetime import datetime
from mtglogger import db, login_manager
from flask_login import UserMixin

@login_manager.user_loader
def load_user(user_id):
 return User.query.get(int(user_id))

class User(db.Model, UserMixin):
 id = db.Column(db.Integer, primary_key=True)
 username = db.Column(db.String(16), unique=True, nullable=False)
 email = db.Column(db.String(32), unique=True, nullable=False)
 password = db.Column(db.String(128), nullable=False)
 join_date = db.Column(db.DateTime, nullable=False,
default=datetime.utcnow())

 cards = db.relationship('Card', backref='owner', lazy=True)

class Card(db.Model, UserMixin):
 id = db.Column(db.Integer, primary_key=True)
 multiverse_id = db.Column(db.String, nullable=False)
 quantity = db.Column(db.Integer, nullable=False)

 user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
```

forms.py

```
from flask_wtf import FlaskForm
from flask_wtf.file import FileField, FileAllowed
from flask_login import current_user
from wtforms import StringField, PasswordField, SubmitField, BooleanField,
TextAreaField, SelectField, FileField
from wtforms.validators import DataRequired, Length, EqualTo,
ValidationError, email
from mtglogger.models import User

class SearchForm(FlaskForm):
 query = StringField('Search', validators=[DataRequired()])
 search = SubmitField('Search')

class LoginForm(FlaskForm):
 username = StringField('Username', validators=[DataRequired()])
 password = PasswordField('Password', validators=[DataRequired()])
 submit = SubmitField('Login')

class RegistrationForm(FlaskForm):
 username = StringField('Username', validators=[DataRequired(),
Length(min=4, max=16)])
 email = StringField('Username', validators=[DataRequired()])
 password = PasswordField('Password',
validators=[DataRequired(Length(min=8, max=128))])
 confirm_password = PasswordField('Confirm', validators=[DataRequired(),
EqualTo('password')])
 submit = SubmitField('Sign Up')
  def validate_username(self, username):
   user = User.query.filter_by(username=username.data).first()
   if user:
     raise ValidationError('That username is taken. Please choose a
different one.')
```

```
class AddForm(FlaskForm):
 condition = SelectField('Condition', choices=[('Near Mint', 'Near Mint (NM)'), ('Lightly Played', 'Lightly Played (LP)'), ('Moderately Played', 'Moderatly Played (MP)'), ('Heavily Played', 'Heavily Played (HP)'), ('Damaged', 'Damaged (D)')])
 quantity = StringField('Quantity', validators=[DataRequired()])
 add = SubmitField('Add to Collection')
```

__init__.py

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt
from flask_login import LoginManager

app = Flask(__name__)
app.config['SECRET_KEY'] = 'my_secret_key'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///site.db'
db = SQLAlchemy(app)
bcrypt = Bcrypt(app)
login_manager = LoginManager(app)
login_manager.login_view = 'login'
login_manager.login_message_category = 'info'

from mtglogger import routes
```

run.py

```
from mtglogger import app

if __name__ == '__main__':
   app.run(debug=True)
```

card.py

```
import sqlite3

def by_name(card_name):
 conn =
sqlite3.connect('/home/lan/Documents/MagicLogger/mtglogger/cards.sqlite')
 c = conn.cursor()
 c.execute("SELECT name, setCode, multiverseId FROM cards WHERE name=?",
(card_name,))
 rows = c.fetchall()
 conn.close()
 if len(rows) > 0:
   cards = rows
 else:
   cards = [['Card Not Found', '']]
 new_cards = []
 for i in cards:
   if i[2]:
     image_url =
str('https://gatherer.wizards.com/Handlers/Image.ashx?multiverseid=' +
str(i[2]) + '&type=card')
     i = i + (image_url, )
     new_cards.append(i)
 return new_cards

def by_multiverseId(id):
 conn =
sqlite3.connect('/home/lan/Documents/MagicLogger/mtglogger/cards.sqlite')
 c = conn.cursor()
 c.execute("SELECT name, setCode, multiverseId FROM cards WHERE
multiverseId=?", (id,))
 rows = c.fetchall()
 if len(rows) > 0:
   card = rows[0]
 image_url =
str('https://gatherer.wizards.com/Handlers/Image.ashx?multiverseid=' +
str(card[2]) + '&type=card')
 card = card + (image_url, )

 return card
```

```
def return_card(id):
 conn = sqlite3.connect('/home/lan/Documents/MagicLogger/mtglogger/cards.sqlite')
 c = conn.cursor()
 c.execute("SELECT name, setCode, multiverseId, rarity, originalType, power, toughness, text, flavorText, type, manaCost FROM cards WHERE multiverseId=?", (id,))
 rows = c.fetchall()
 if len(rows) > 0:
   notDictCard = rows[0]

   card = {}
   card["name"] = notDictCard[0]
   card["set"] = notDictCard[1]
   card["id"] = notDictCard[2]
   card["rarity"] = notDictCard[3]
   card["originalType"] = notDictCard[4]
   card["type"] = notDictCard[9]
   card["power"] = notDictCard[5]
   card["toughness"] = notDictCard[6]
   card["text"] = notDictCard[7]
   card["flavor"] = notDictCard[8]
   card["image_url"] = str('https://gatherer.wizards.com/Handlers/Image.ashx?multiverseid=' + str(card["id"]) + '&type=card')
   card["manaCost"] = notDictCard[10]
 else:
   card = None
 return card
```

auth/login.html

```
{% extends "layout.html" %}
{% block content %}
<div class="auth">
```

```
 <form method="POST" action="">
   {{ loginForm.hidden_tag() }}
   <h4>Login</h4>
   <hr>
   <div class="form-group">
     <label>Username</label>
     {% if loginForm.username.errors %}
         {{ loginForm.username(class="form-control is-invalid") }}
         <div class="invalid-feedback">
             {% for error in loginForm.username.errors %}
                 <span>{{ error }}</span>
             {% endfor %}
         </div>
     {% else %}
         {{ loginForm.username(class="form-control") }}
     {% endif %}
   </div>
   <div class="form-group">
     <label>Password</label>
     {% if loginForm.password.errors %}
         {{ loginForm.password(class="form-control is-invalid") }}
         <div class="invalid-feedback">
             {% for error in loginForm.password.errors %}
                 <span>{{ error }}</span>
             {% endfor %}
         </div>
     {% else %}
         {{ loginForm.password(class="form-control") }}
     {% endif %}
   </div>
   <div class="form-group">
     {{ loginForm.submit(class="btn btn-primary") }}
   </div>
 </form>
</div>
{% endblock %}
```

auth/register.html

```
{% extends "layout.html" %}
{% block content %}
<div class="auth">
 <form method="POST" action="">
   {{ loginForm.hidden_tag() }}
   <h4>Login</h4>
   <hr>
   <div class="form-group">
     <label>Username</label>
     {% if loginForm.username.errors %}
         {{ loginForm.username(class="form-control is-invalid") }}
         <div class="invalid-feedback">
             {% for error in loginForm.username.errors %}
                 <span>{{ error }}</span>
             {% endfor %}
         </div>
     {% else %}
         {{ loginForm.username(class="form-control") }}
     {% endif %}
   </div>
   <div class="form-group">
     <label>Password</label>
     {% if loginForm.password.errors %}
         {{ loginForm.password(class="form-control is-invalid") }}
         <div class="invalid-feedback">
             {% for error in loginForm.password.errors %}
                 <span>{{ error }}</span>
             {% endfor %}
         </div>
     {% else %}
         {{ loginForm.password(class="form-control") }}
     {% endif %}
   </div>
   <div class="form-group">
     {{ loginForm.submit(class="btn btn-primary") }}
   </div>
 </form>
</div>
{% endblock %}
```

card.html

```
{% extends "layout.html" %}
{% block content %}
{% if card %}
<div class="row">
 <div class="col-3">
   <img src="{{ card['image_url'] }}" style="width: 100%;">
 </div>
 <div class="col-6">
   <h3 class="title">{{ card["name"] }}</h3>
   <p class="set">{{ card["set"] }}</p>
   <hr>
   <div class="row">
     <div class="col-4">
       <p class="section-title">Rarity:</p>
       <p class="section-title">Card Type:</p>
       {% if "Creature" in card["type"] %}
         <p class="section-title">P / T:</p>
       {% endif %}
       <p class="section-title">Description:</p>
       <p class="section-title">Flavor:</p>
     </div>
     <div class="col-8">
       <p class="section-title-content">{{ card["rarity"].title() }}</p>
       <p class="section-title-content">{{ card["type"] }}</p>
       {% if "Creature" in card["type"] %}
         <p class="section-title-content">{{ card["power"] + ' / ' + card["toughness"] }}</p>
       {% endif %}
       <p class="section-title-content">{{ card["text"] }}</p>
       <p class="section-title-content">{{ card["flavor"] }}</p>
     </div>
   </div>
 </div>
 <div class="col-3">
   <div class="content-section">
     <h3 class="price">$37.75</h3>
     <form method="POST" action="">
       {{ form.hidden_tag() }}
       <div class="form-group">
         {% if form.condition.errors %}
         {{ form.condition(class="form-control is-invalid") }}
         <div class="invalid-feedback">
           {% for error in form.condition.errors %}
             <span>{{ error }}</span>
           {% endfor %}
         </div>
         {% else %}
           {{ form.condition(class="form-control") }}
         {% endif %}
         <p style="margin-top: 0.2rem; font-size: 0.78rem;"><a
href="https://www.mtgseattle.com/card_condition_guide" target="_blank"
style="color: #145488;">What is my cards condition?</a></p>
       </div>
       <div class="form-group">
         {% if form.quantity.errors %}
         {{ form.quantity(class="form-control is-invalid") }}
         <div class="invalid-feedback">
           {% for error in form.quantity.errors %}
             <span>{{ error }}</span>
           {% endfor %}
         </div>
         {% else %}
           {{ form.quantity(class="form-control", placeholder="Quantity")
}}
         {% endif %}
       </div>
       <div class="form-group">
         {{ form.add(class="btn btn-success add-btn") }}
       </div>
     </form>
   </div>
 </div>
</div>
{% else %}
 <h1>Not Valid Card</h1>
{% endif %}
{% endblock %}
```

collection.html

```
{% extends "layout.html" %}
{% block content %}
{% if card %}
<div class="row">
 <div class="col-3">
   <img src="{{ card['image_url'] }}" style="width: 100%;">
 </div>
 <div class="col-6">
   <h3 class="title">{{ card["name"] }}</h3>
   <p class="set">{{ card["set"] }}</p>
   <hr>
   <div class="row">
     <div class="col-4">
       <p class="section-title">Rarity:</p>
       <p class="section-title">Card Type:</p>
       {% if "Creature" in card["type"] %}
         <p class="section-title">P / T:</p>
       {% endif %}
       <p class="section-title">Description:</p>
       <p class="section-title">Flavor:</p>
     </div>
     <div class="col-8">
       <p class="section-title-content">{{ card["rarity"].title() }}</p>
       <p class="section-title-content">{{ card["type"] }}</p>
       {% if "Creature" in card["type"] %}
         <p class="section-title-content">{{ card["power"] + ' / ' +
card["toughness"] }}</p>
       {% endif %}
       <p class="section-title-content">{{ card["text"] }}</p>
       <p class="section-title-content">{{ card["flavor"] }}</p>
     </div>
   </div>
 </div>
 <div class="col-3">
   <div class="content-section">
     <h3 class="price">$37.75</h3>
     <form method="POST" action="">
       {{ form.hidden_tag() }}
       <div class="form-group">
         {% if form.condition.errors %}
         {{ form.condition(class="form-control is-invalid") }}
         <div class="invalid-feedback">
           {% for error in form.condition.errors %}
             <span>{{ error }}</span>
           {% endfor %}
         </div>
         {% else %}
           {{ form.condition(class="form-control") }}
         {% endif %}
         <p style="margin-top: 0.2rem; font-size: 0.78rem;"><a
href="https://www.mtgseattle.com/card_condition_guide" target="_blank"
style="color: #145488;">What is my cards condition?</a></p>
       </div>
       <div class="form-group">
         {% if form.quantity.errors %}
         {{ form.quantity(class="form-control is-invalid") }}
         <div class="invalid-feedback">
           {% for error in form.quantity.errors %}
             <span>{{ error }}</span>
           {% endfor %}
         </div>
         {% else %}
           {{ form.quantity(class="form-control", placeholder="Quantity")
}}
         {% endif %}
       </div>
       <div class="form-group">
         {{ form.add(class="btn btn-success add-btn") }}
       </div>
     </form>
   </div>
 </div>
</div>
{% else %}
 <h1>Not Valid Card</h1>
{% endif %}
{% endblock %}
```

index.html

```
{% extends "layout.html" %}
{% block content %}
 <div class="splash">
   <img src="{{ url_for('static', filename='img/mtg.png') }}" style="height: 30vh; margin-bottom: 4rem;">
   <h1 class="display-4">MagicLogger</h1>
   <p class="lead">Your Magic The Gathering card database solution.</p>
 </div>
{% endblock %}
```

layout.html

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

   <!-- Local copy of Bootstrap stylesheet -->
   <link rel="stylesheet" text="text/css" href="{{ url_for('static', filename='styles/bootstrap.min.css') }}">

   <!-- Custom stylesheets -->
   <link rel="stylesheet" text="text/css" href="{{ url_for('static', filename='styles/fonts.css') }}">
   <link rel="stylesheet" text="text/css" href="{{ url_for('static',
filename='styles/main.css') }}">

   <!-- Bootstrap jquery libraries -->
   <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
integrity="sha384-J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYf
oRSJoZ+n" crossorigin="anonymous"></script>
   <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVv
oxMfooAo" crossorigin="anonymous"></script>
   <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.j
s"
integrity="sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3
Og8ifwB6" crossorigin="anonymous"></script>

   <!-- Font Awesome icon kit -->
   <script src="https://kit.fontawesome.com/010964ebfa.js"
crossorigin="anonymous"></script>

   <title>MagicLogger | {{ title }}</title>
 </head>
 <body>
   <nav class="navbar navbar-expand-lg navbar-light top-results">
     <a class="navbar-brand" href="{{ url_for('home') }}">Magic Logger</a>
     <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
       <span class="navbar-toggler-icon"></span>
     </button>

     <div class="collapse navbar-collapse" id="navbarSupportedContent">
       <ul class="navbar-nav mr-auto">
         <li class="nav-item active">
```

```
          <a class="nav-link" href="{{ url_for('home') }}">Home <span
class="sr-only">(current)</span></a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Info</a>
        </li>
        {% if not current_user.is_authenticated %}
          <li class="nav-item dropdown">
            <a class="nav-link dropdown-toggle" href="#"
id="navbarDropdown" role="button" data-toggle="dropdown"
aria-haspopup="true" aria-expanded="false">
              Auth
            </a>
            <div class="dropdown-menu" aria-labelledby="navbarDropdown">
              <a class="dropdown-item" href="{{ url_for('login')
}}">Login</a>
              <a class="dropdown-item" href="{{ url_for('register')
}}">Register</a>
            </div>
          </li>
        {% else %}
          <li class="nav-item">
            <a class="nav-link" href="{{ url_for('collection')
}}">Collection</a>
          </li>
        {% endif %}
      </ul>
      <form method="POST" action="" class="form-inline my-2 my-lg-0">
        <div class="form-group">
            <input type="text" placeholder="Enter a card name"
class="form-control mr-sm-2">
        </div>
        <div class="form-group">
          <button class="btn btn-outline-primary">Search</button>
        </div>
    </form>
    </div>
```

```
   </nav>
   <main>
     <div class="message-area">
       {% with messages = get_flashed_messages(with_categories=true) %}
         {% if messages %}
           {% for category, message in messages %}
             <div class="alert alert-{{ category }} alert-dismissible fade
show">
               {{ message }}
               <button type="button" class="close" data-dismiss="alert"
aria-label="Close">
                   <span aria-hidden="true">&times;</span>
               </button>
             </div>
           {% endfor %}
         {% endif %}
       {% endwith %}
     </div>
     {% block content %}
     {% endblock %}
   </main>
 </body>
</html>
```

nav.html

```
<nav class="navbar navbar-expand-lg navbar-light top-results">
 <a class="navbar-brand" href="{{ url_for('home') }}">Magic Logger</a>
 <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
   <span class="navbar-toggler-icon"></span>
 </button>

 <div class="collapse navbar-collapse" id="navbarSupportedContent">
   <ul class="navbar-nav mr-auto">
     <li class="nav-item active">
```

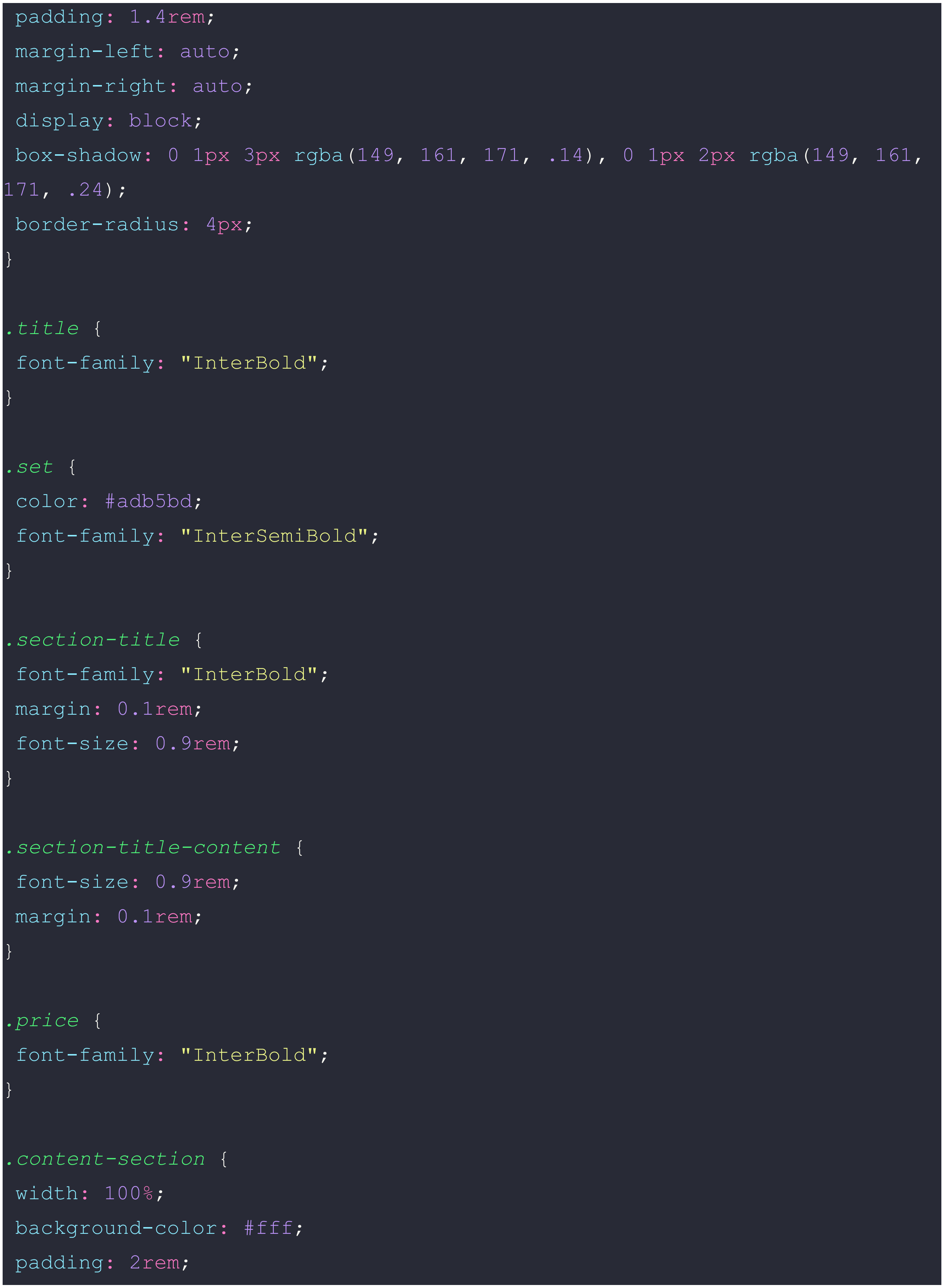
```
       <a class="nav-link" href="{{ url_for('home') }}">Home <span
class="sr-only">(current)</span></a>
     </li>
     <li class="nav-item">
       <a class="nav-link" href="#">Info</a>
     </li>
     <li class="nav-item dropdown">
       <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
role="button" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
         Auth
       </a>
       <div class="dropdown-menu" aria-labelledby="navbarDropdown">
         <a class="dropdown-item" href="#">Login</a>
         <a class="dropdown-item" href="#">Registers</a>
       </div>
     </li>
   </ul>
   <form method="POST" action="" class="form-inline my-2 my-lg-0">
     {{ form.hidden_tag() }}
     <div class="form-group">
         {% if form.query.errors %}
             {{ form.query(class="form-control is-invalid") }}
             <div class="invalid-feedback">
                 {% for error in form.query.errors %}
                     <span>{{ error }}</span>
                 {% endfor %}
             </div>
         {% else %}
             {{ form.query(class="form-control mr-sm-2",
placeholder="Enter a Card Name") }}
         {% endif %}
     </div>
     <div class="form-group">
         {{ form.search(class="btn btn-outline-primary") }}
     </div>
 </form>
 </div>
</nav>
```

results.html

```
{% extends "layout.html" %}
{% block content %}
 <div class="row row-cols-4">
   {% for i in cards %}
     <div class="col">
       <a href="{{ url_for('card', multiverse_id=i[2]) }}">
         <div class="card-result">
           <img src="{{ i[3] }}">
           <div class="card-result-content">
             <p class="game-name">Magic: The Gathering</p>
             <p class="set-name">{{ i[1] }}</p>
             <h5 class="card-name">{{ i[0] }}</h5>
           </div>
         </div>
         </a>
     </div>
   {% endfor %}
 </div>
{% endblock %}
```

main.css

```
html, body {
 background-color: #f8f9f9;
 overflow-x: hidden;
}

.splash {
 text-align: center;
}

.top-results {
 background-color: #fff;
 padding: 1.25rem 1.5rem;
 box-shadow: 0 1px 3px rgba(149, 161, 171, .14), 0 1px 2px rgba(149, 161,
171, .24);
}

.top-results p {
 font-family: "Inter";
 margin: 0;
}

.top-results span {
 font-family: "InterBold";
 font-size: 1.1rem;
}

main {
 width: 96%;
 margin-left: auto;
 margin-right: auto;
 margin-top: 2rem;
 display: block;
 min-height: 70vh;
}

.card-result {
 background-color: #fff;
 padding: 0.1rem 0.2rem;
 margin-bottom: 1rem;
 box-shadow: 0 1px 3px rgba(149, 161, 171, .14), 0 1px 2px rgba(149, 161,
171, .24);
 border-radius: 4px;
}

.card-result img {
 width: 10rem;
 height: auto;
 margin: 0 auto;
 padding: 0.5rem;
 display: block;
}

.card-result-content {
 padding: 0.5rem 2.5rem;
}

.game-name {
 font-family: "InterBold";
 font-size: 0.666rem;
 color: #171717;
 margin: 0;
}

.set-name {
 font-size: 0.666rem;
 font-family: "InterSemiBold";
 margin: 0;
 color: #374047;
}

.card-name a {
 color: #171717;
 font-size: 17px;
 font-family: "InterBold";
 margin-top: 0.4rem;
}

.card-add {
 color: #145488;
 font-family: "InterSemiBold";
 font-size: 0.9rem;
}

.auth {
 width: 60%;
 background-color: #fff;
```



```css
 padding: 1.4rem;
 margin-left: auto;
 margin-right: auto;
 display: block;
 box-shadow: 0 1px 3px rgba(149, 161, 171, .14), 0 1px 2px rgba(149, 161, 171, .24);
 border-radius: 4px;
}

.title {
 font-family: "InterBold";
}

.set {
 color: #adb5bd;
 font-family: "InterSemiBold";
}

.section-title {
 font-family: "InterBold";
 margin: 0.1rem;
 font-size: 0.9rem;
}

.section-title-content {
 font-size: 0.9rem;
 margin: 0.1rem;
}

.price {
 font-family: "InterBold";
}

.content-section {
 width: 100%;
 background-color: #fff;
 padding: 2rem;
 margin-left: auto;
 margin-right: auto;
 display: block;
 box-shadow: 0 1px 3px rgba(149, 161, 171, .14), 0 1px 2px rgba(149, 161,
171, .24);
 border-radius: 4px;
}

.add-btn {
 font-family: "InterSemiBold";
 font-size: 0.9rem;
 width: 100%;
}

.collection-card {
 padding: 0.25rem 0.5rem;
 border-left: 1px solid #333;
 border-right: 1px solid #333;
 border-bottom: 1px solid #333;
}

.collection-card p {
 margin: 0;
 font-size: 0.87rem;
}

.collection-card-header {
 padding: 0.25rem 0.5rem;
 border: 1px solid #333;
}

.collection-card-header p {
 font-weight: bold;
 margin: 0;
}

.btn-xsm {
 font-size: 0.75rem;
 padding:.125rem .25rem;
}
```

fonts.css

```
@font-face {
 font-family: "Inter";
 src: url(../fonts/Inter/Inter-Regular-slnt=0.ttf);
}

@font-face {
 font-family: "InterSemiBold";
 src: url(../fonts/Inter/Inter-SemiBold-slnt=0.ttf);
}

@font-face {
 font-family: "InterBold";
 src: url(../fonts/Inter/Inter-Bold-slnt=0.ttf);
}
```