CompSci 316: Intro to Databases

HW-4: A basic flask-based “Beers” application

Score: 100

Release Date: Wed, 02/26/2020

Due Date: Wed, 03/04/2020 11:59PM

Unlike the other homeworks, this has to be done by each project group together. Only one submission per group is needed and every group member will receive the same score.

1. Sample Flask Application

Work through the instructions here:
https://www2.cs.duke.edu/courses/spring20/compsci316/instructions/flask/

After this, you should have some idea of the structure of a Flask project. Read through the code and learn what each component does. Understand how each component individually handles one aspect of the application, and how they work together.

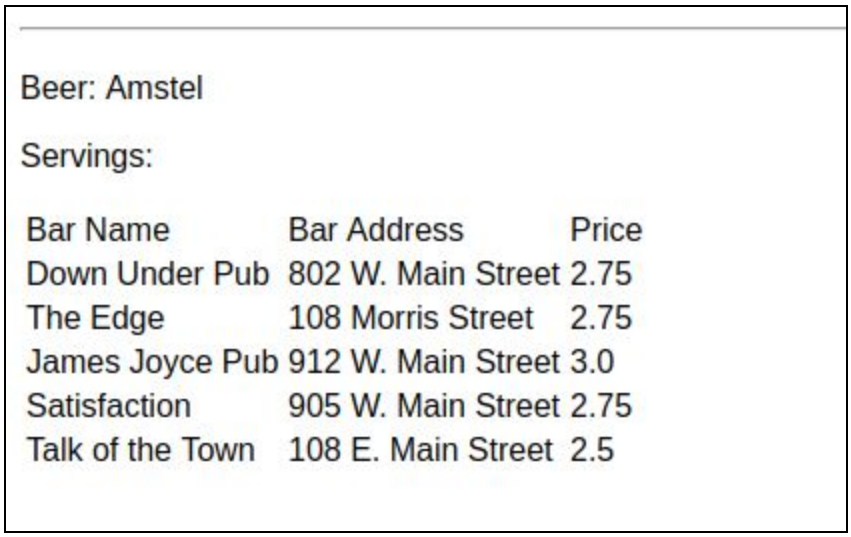
2. Implement Your Own Feature

Now it’s your time to add some features to this example website. Modify the example from the previous section, so it has a new page such that

1. You can visit it through ‘<your website baseurl>/serves’, which means the route is ‘/serves’
2. The page has a drop down menu, where you can choose beer names.
3. After you choose a beer, it will navigate to a new page ‘<your website baseurl>/servings/<the beer name>’.
4. The new page shows a list of bars below, with each row including the bar name, the bar address, and the price for that beer.

The new pages should follow the examples below:

Page 1 Example (http://vcm-xxxx.vm.duke.edu:5000/serves)



Page 2 Example (http://vcm-xxxx.vm.duke.edu:5000/servings/Amstel)

3. Sample Codes

You can finish this task through all possible approaches. Here we also provide some sample codes with some placeholders to help you finish this task. You can modify the existing files or create new files, and replace "@TODO" with your codes:

app.py	@app.route('/serves', methods=['GET', 'POST'])
	def serves():
	  beer_names = @TODO
	  form = forms.ServingsFormFactory.form(@TODO)
	  if form.@TODO():
	  return @TODO('/servings/' + form.beer_sel.data)
	  return render_template('serves.html', form=form)
	
	@app.route('/servings/<beer_name>')
	def servings_for(beer_name):
	  results = db.session.query(models.Serves, models.Bar) \
	  .filter(@TODO) \
	  .join(@TODO).all()
	  return render_template('servings_for.html', beer_name=beer_name, data=results)

forms.py	from wtforms import SelectField, SubmitField
	
	class ServingsFormFactory:
	  @staticmethod
	  def form(beer_names):
	  class F(@TODO):
	  beer_sel = SelectField('Beer Name', choices= @TODO)
	  submit = SubmitField('Submit')
	  return F()

templates/serves.html	{% extends 'layout.html' %}
	{% block content %}
	 Select Beer
	<form method="@TODO">
	  {{form.csrf_token}}
	  {{form.beer_sel.label}}
	 @TODO
	  @TODO
	</form>
	{% endblock %}

templates/servings_for.html	{% extends 'layout.html' %} {% block content %} Beer: {{beer_name}} Servings: <table> <tr> <td>Bar Name</td> <td>Bar Address</td> <td>Price</td>

	</tr> {% for serve, bar in data %} <tr> <td> @TODO </td> <td> @TODO </td> <td> @TODO </td> </tr> {% endfor %} </table> {% endblock %}

Submission Instructions

1. **Submit all the code** of your Flask application as a zip file on Gradescope. This is a group submission. Each group only needs to submit once. Make sure all the members are included in the Gradescope submission.

2. **Keep your Flask server running and provide the URL** to your website in your team's private Piazza post. The URL should be like http://vcm-xxxxx.vm.duke.edu:5000.