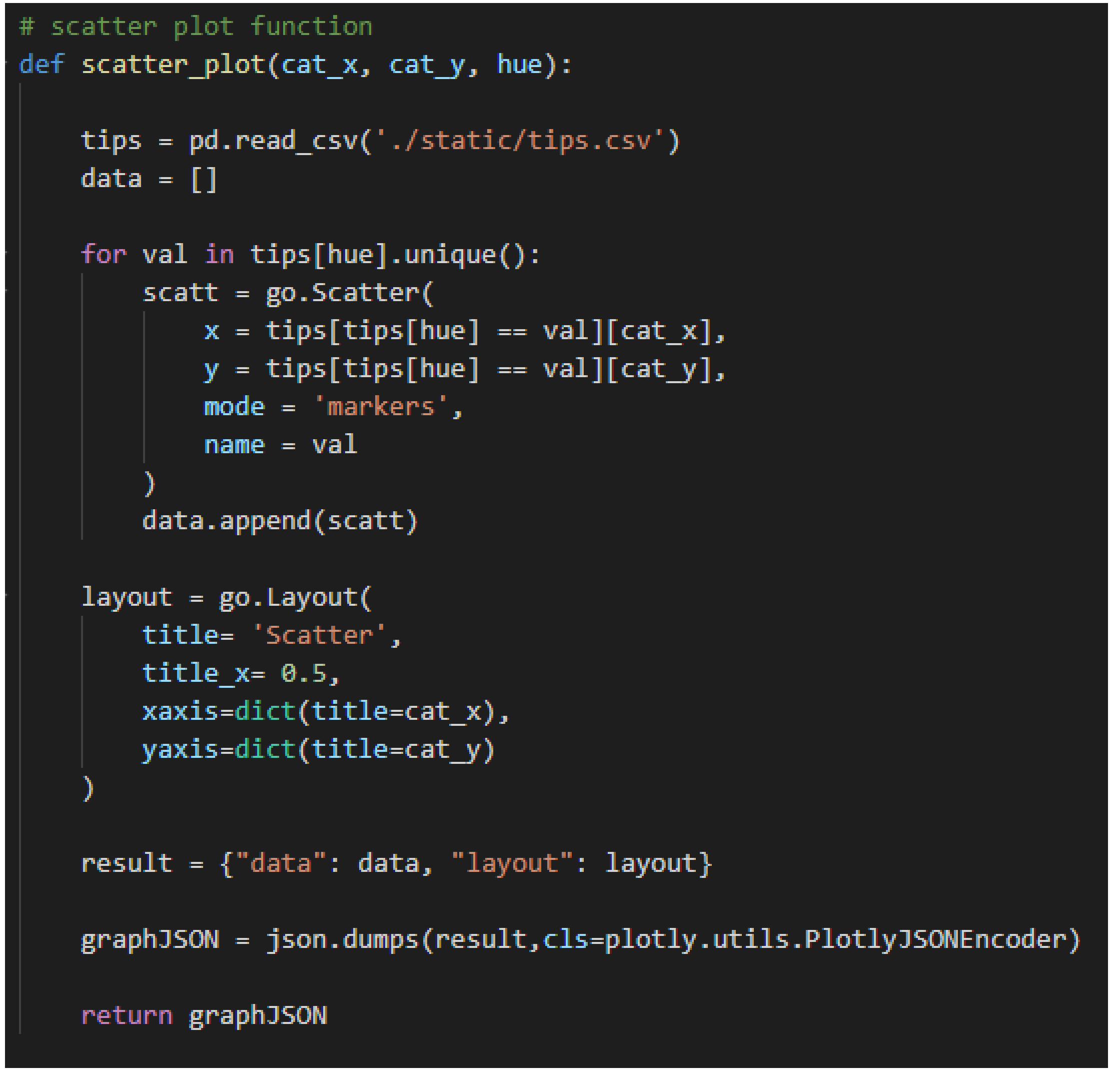
SESSION 8

Flask Dashboard

Create Scatter Plot Menu

Purwadhika
Startup and Coding School

SCATTER PLOT PAGE (app.py)

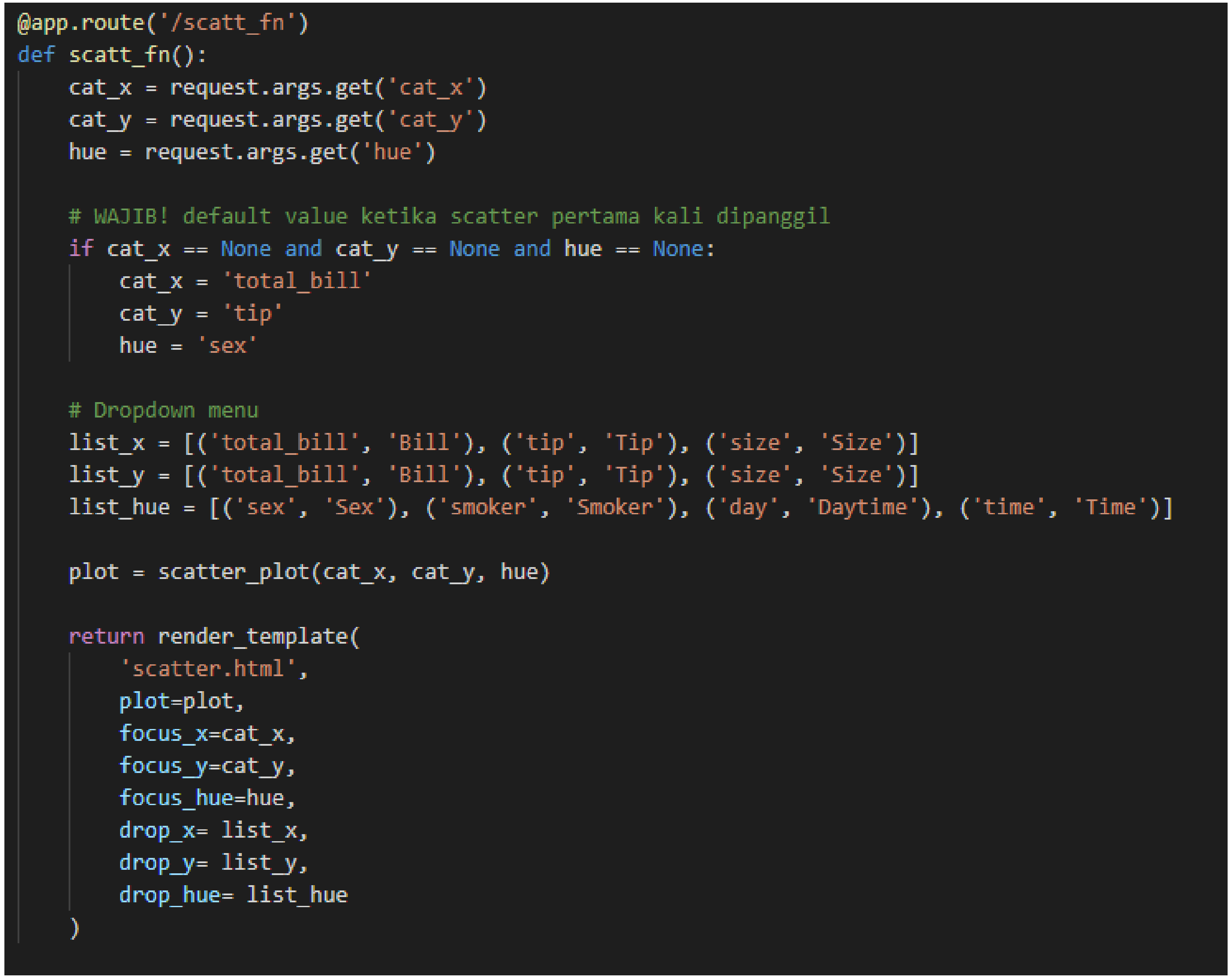


```
# scatter plot function
def scatter_plot(cat_x, cat_y, hue):

    tips = pd.read_csv('./static/tips.csv')
    data = []

    for val in tips[hue].unique():
        scatt = go.Scatter(
            x = tips[tips[hue] == val][cat_x],
            y = tips[tips[hue] == val][cat_y],
            mode = 'markers',
            name = val
        )
        data.append(scatt)

    layout = go.Layout(
        title= 'Scatter',
        title_x= 0.5,
        xaxis=dict(title=cat_x),
        yaxis=dict(title=cat_y)
    )

    result = {"data": data, "layout": layout}

    graphJSON = json.dumps(result,cls=plotly.utils.PlotlyJSONEncoder)

    return graphJSON
```

Define the function for scatter plot. In this function, load the data and the rest is code to make the plot. Not like Category plot function, this scatter plot function only has a plot, that is scatter plot itself. Here is the code

SCATTER PLOT PAGE (app.py)

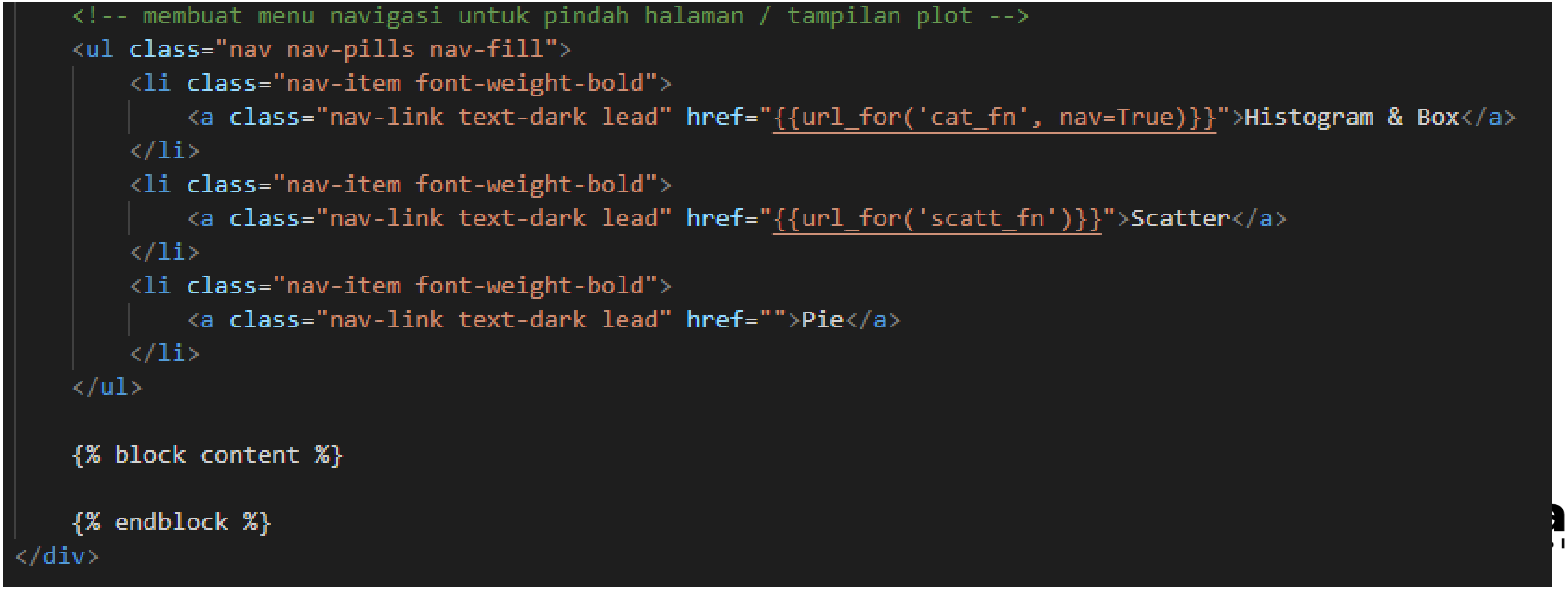
Create a router called ‘scatt_fn’. This router will be used in landing page of scatter plot menu/page. It only has 3 components, they are X axis which consist of ‘total_bill’, ‘tip’, and ‘size’ variable from data ; Y axis consist of same variable as X axis, and Hue consist of ‘sex’, ‘smoke’, ‘day’, and ‘time’ variable. So, in html page of this scatter plot, there will be 3 drop down menu.



```python
@app.route('/scatt_fn')
def scatt_fn():
    cat_x = request.args.get('cat_x')
    cat_y = request.args.get('cat_y')
    hue = request.args.get('hue')

    # WAJIB! default value ketika scatter pertama kali dipanggil
    if cat_x == None and cat_y == None and hue == None:
        cat_x = 'total_bill'
        cat_y = 'tip'
        hue = 'sex'

    # Dropdown menu
    list_x = [('total_bill', 'Bill'), ('tip', 'Tip'), ('size', 'Size')]
    list_y = [('total_bill', 'Bill'), ('tip', 'Tip'), ('size', 'Size')]
    list_hue = [('sex', 'Sex'), ('smoker', 'Smoker'), ('day', 'Daytime'), ('time', 'Time')]

    plot = scatter_plot(cat_x, cat_y, hue)

    return render_template(
        'scatter.html',
        plot=plot,
        focus_x=cat_x,
        focus_y=cat_y,
        focus_hue=hue,
        drop_x= list_x,
        drop_y= list_y,
        drop_hue= list_hue
    )
```

SCATTER PLOT PAGE (index.html)

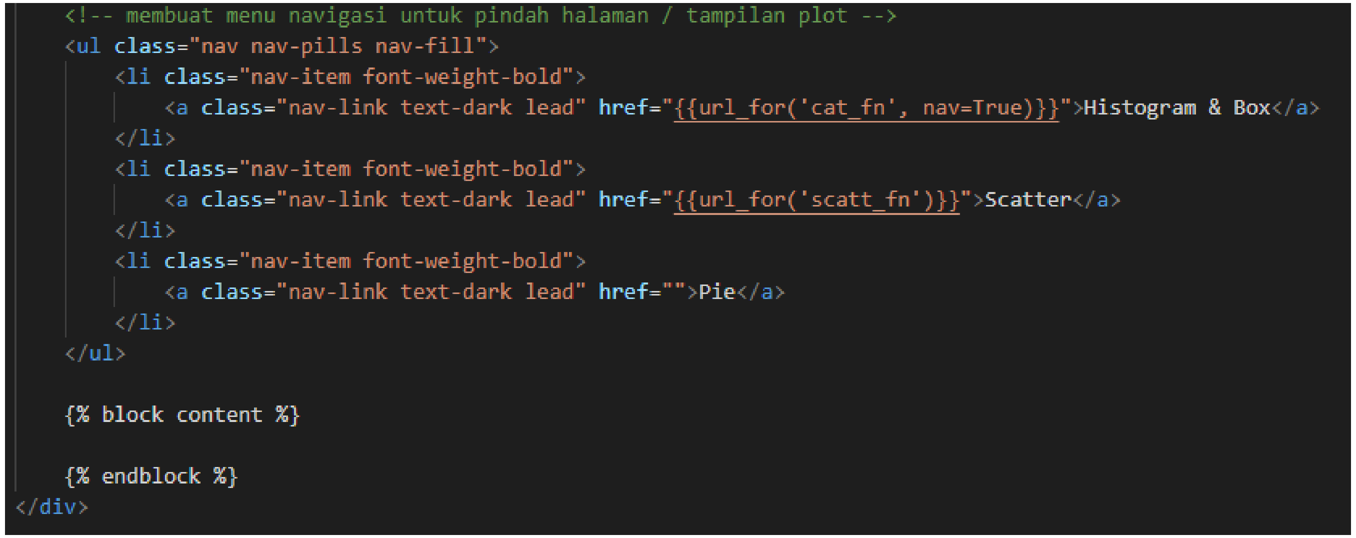
Lets move to ‘index.html’ file. In this file, same as Histogram & Box, add a ‘href’ of Scatter, fill with the name of scatter plot router. Write the code by using jinja (`href="{{url_for('scatt_fn')}}`)



```
    <!-- membuat menu navigasi untuk pindah halaman / tampilan plot -->
    <ul class="nav nav-pills nav-fill">
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('cat_fn', nav=True)}}">Histogram & Box</a>
        </li>
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('scatt_fn')}}">Scatter</a>
        </li>
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="">Pie</a>
        </li>
    </ul>

    {% block content %}

    {% endblock %}
</div>
```

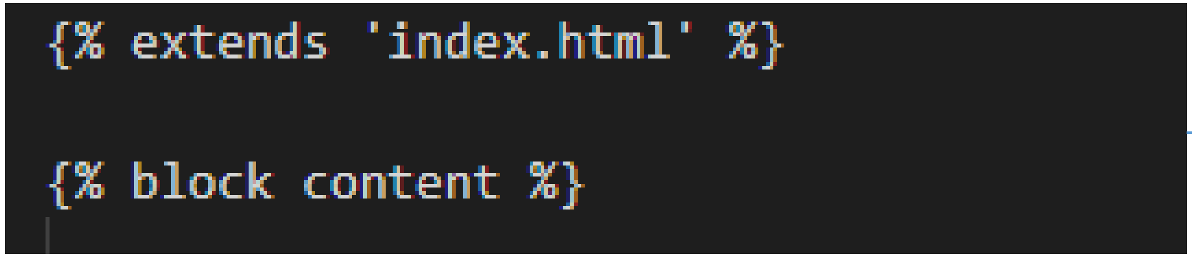
SCATTER PLOT PAGE (scatter.html)

In this page, create an extends and block content. This code is to connect from ‘index.html’ file and continue the code start from the place where the code written.



```
    <!-- membuat menu navigasi untuk pindah halaman / tampilan plot -->
    <ul class="nav nav-pills nav-fill">
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('cat_fn', nav=True)}}">Histogram & Box</a>
        </li>
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('scatt_fn')}}">Scatter</a>
        </li>
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="">Pie</a>
        </li>
    </ul>

    {% block content %}

    {% endblock %}
</div>
```

From ‘index.html’



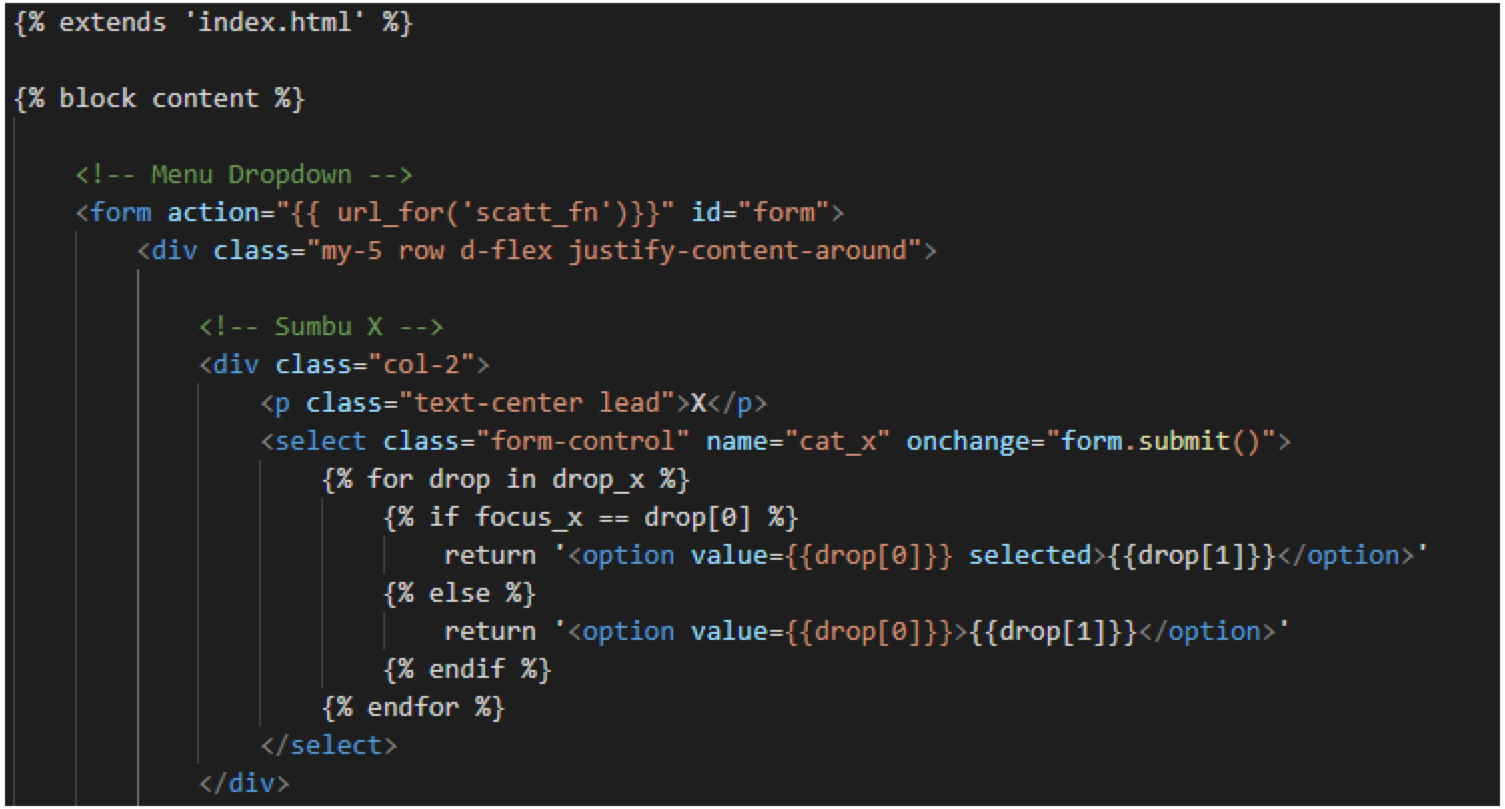
```
{% extends 'index.html' %}

{% block content %}
```

From ‘scatter.html’

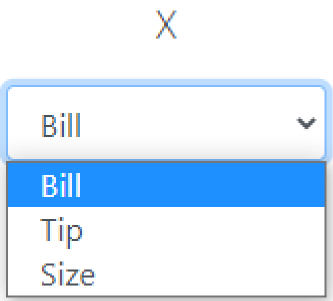
Purwadhika
Startup and Coding School

SCATTER PLOT PAGE (scatter.html)

Because this page only consist of a plot (scatter plot), so there isn't any drop down for the plot type. It will make X axis as the first drop down menu. Value of 'drop_x' from 'app.py' in 'scatt_fn' router will be looped. Every element of 'drop_x' be showed in X axis drop down menu. Here is the code and its output

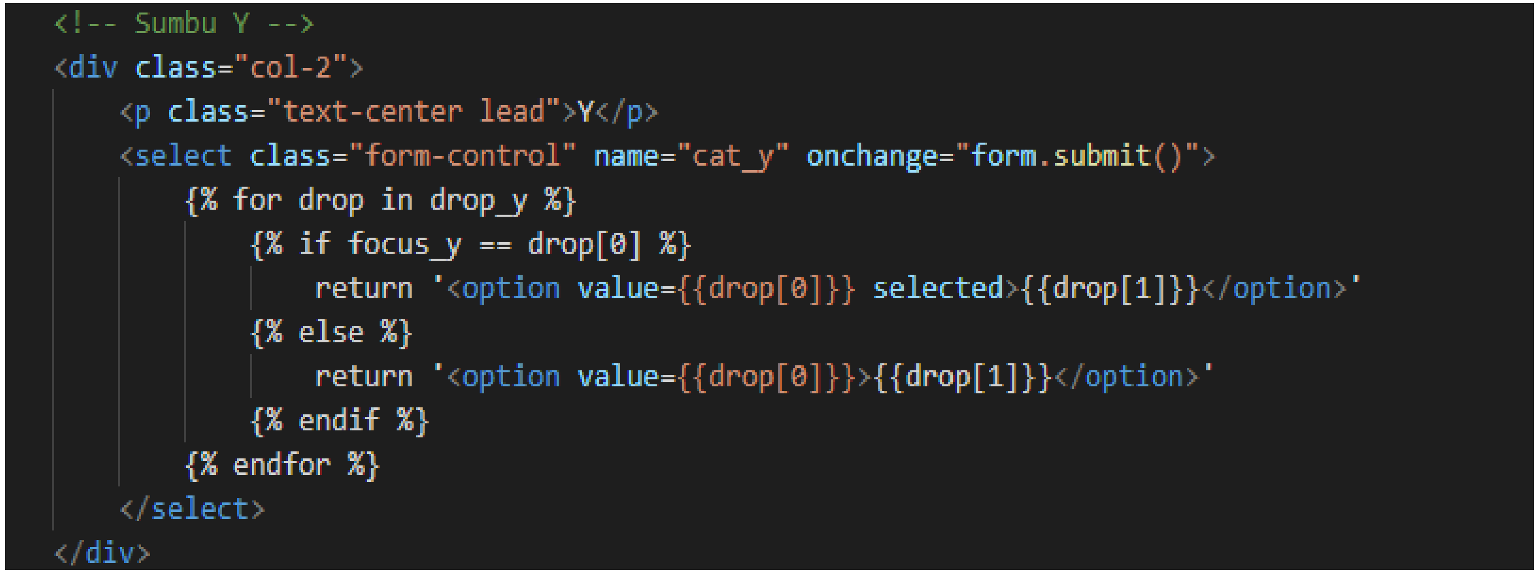


```
{% extends 'index.html' %}

{% block content %}

    <!-- Menu Dropdown -->
    <form action="{{ url_for('scatt_fn')}}" id="form">
        <div class="my-5 row d-flex justify-content-around">

            <!-- Sumbu X -->
            <div class="col-2">
                <p class="text-center lead">X</p>
                <select class="form-control" name="cat_x" onchange="form.submit()">
                    {% for drop in drop_x %}
                        {% if focus_x == drop[0] %}
                            return '<option value={{drop[0]}} selected>{{drop[1]}}</option>'
                        {% else %}
                            return '<option value={{drop[0]}}>{{drop[1]}}</option>'
                        {% endif %}
                    {% endfor %}
                </select>
            </div>
```

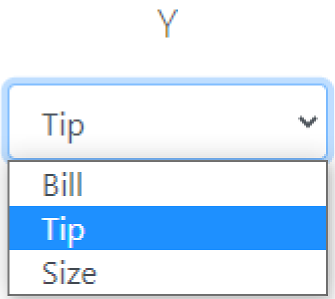


SCATTER PLOT PAGE (scatter.html)

In Y axis drop down menu of this page, the option same as X axis. Value of ‘drop_y’ from ‘app.py’ in ‘scatt_fn’ router will be looped. Every element of ‘drop_y’ be showed in Y axis drop down menu. Here is the code and its output

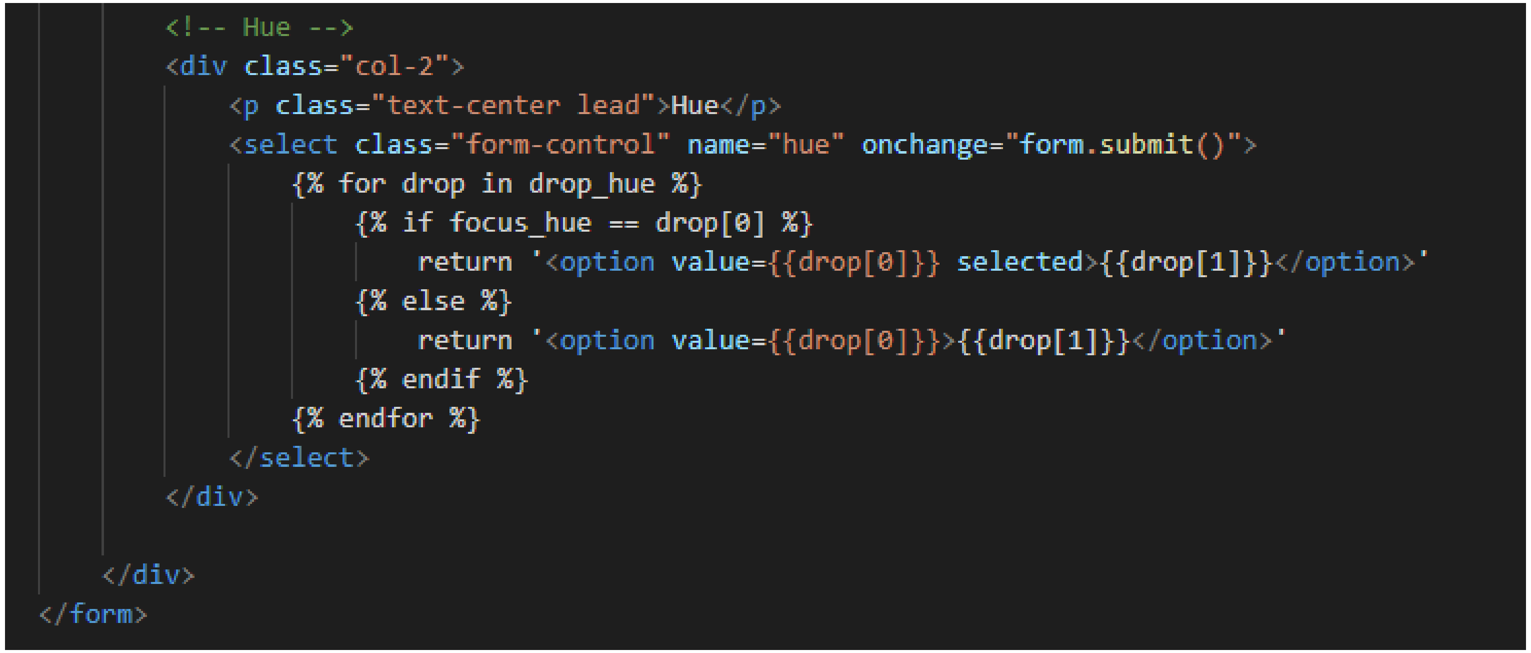


```
<!-- Sumbu Y -->
<div class="col-2">
    <p class="text-center lead">Y</p>
    <select class="form-control" name="cat_y" onchange="form.submit()">
        {% for drop in drop_y %}
            {% if focus_y == drop[0] %}
                return '<option value={{drop[0]}} selected>{{drop[1]}}</option>'
            {% else %}
                return '<option value={{drop[0]}}>{{drop[1]}}</option>'
            {% endif %}
        {% endfor %}
    </select>
</div>
```

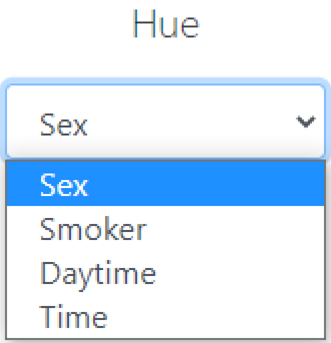


SCATTER PLOT PAGE (scatter.html)

Last is Hue drop down menu. This menu show categorical variable so the plot will divided into category in this hue menu. Options that showed in hue menu are from the value in ‘drop_hue’ variable in ‘app.py’ file, of course ‘drop_hue’ from ‘scatt_fn’ router. Here is the code for Hue drop down menu

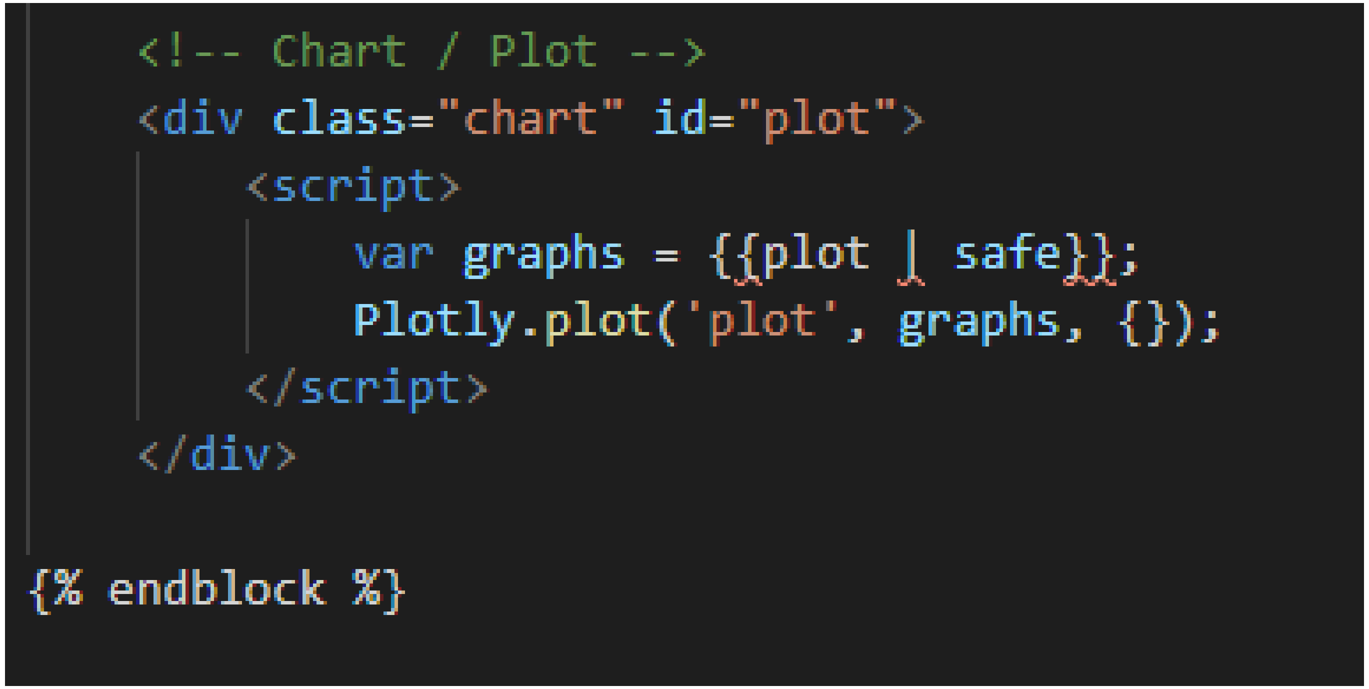


```
        <!-- Hue -->
        <div class="col-2">
            <p class="text-center lead">Hue</p>
            <select class="form-control" name="hue" onchange="form.submit()">
                {% for drop in drop_hue %}
                    {% if focus_hue == drop[0] %}
                        return '<option value={{drop[0]}} selected>{{drop[1]}}</option>'
                    {% else %}
                        return '<option value={{drop[0]}}>{{drop[1]}}</option>'
                    {% endif %}
                {% endfor %}
            </select>
        </div>

    </div>
</form>
```



SCATTER PLOT PAGE (scatter.html)

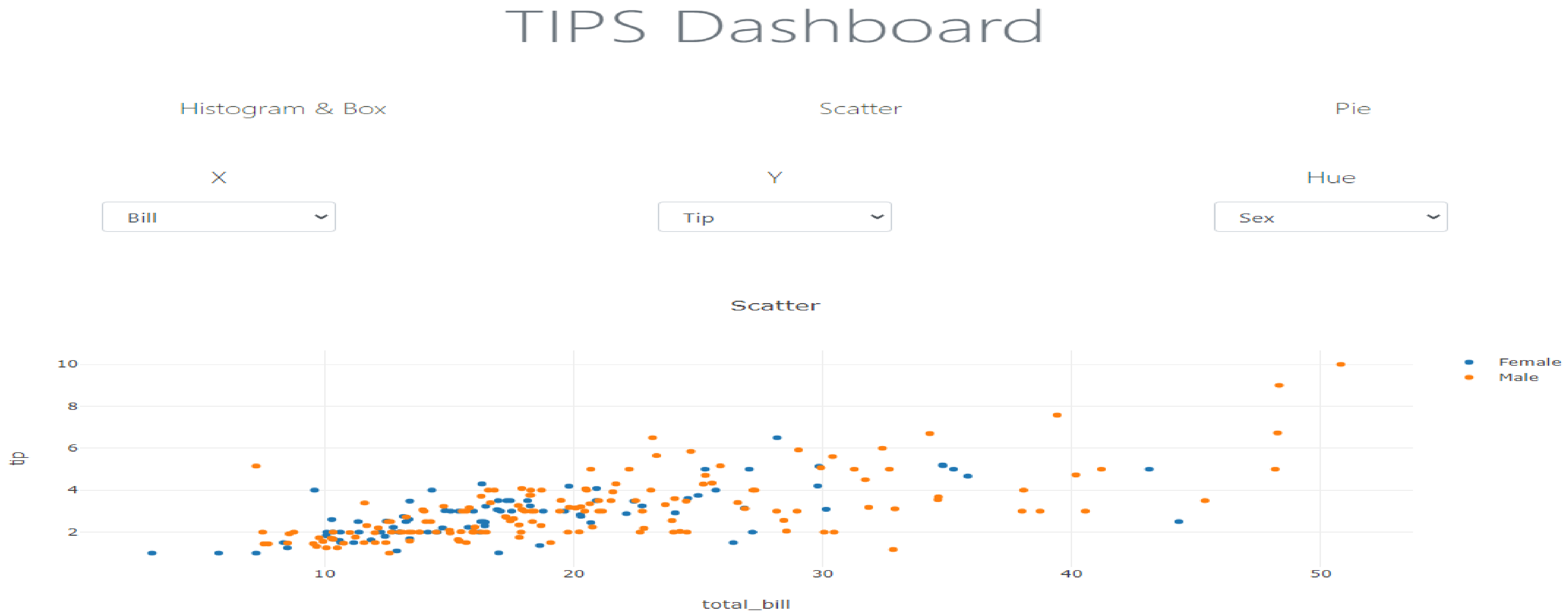
Last from this scatter page is to show the graph and endblock content. Below the code of this page



```
    <!-- Chart / Plot -->
    <div class="chart" id="plot">
        <script>
            var graphs = {{plot | safe}};
            Plotly.plot('plot', graphs, {});
        </script>
    </div>

{% endblock %}
```

SCATTER PLOT MENU PREVIEW

To show the dashboard, run ‘app.py’ in terminal or command prompt. After that, open http://localhost:5000 in browser.



TIPS Dashboard

X: Bill

Y: Tip

Hue: Sex

SESSION 8

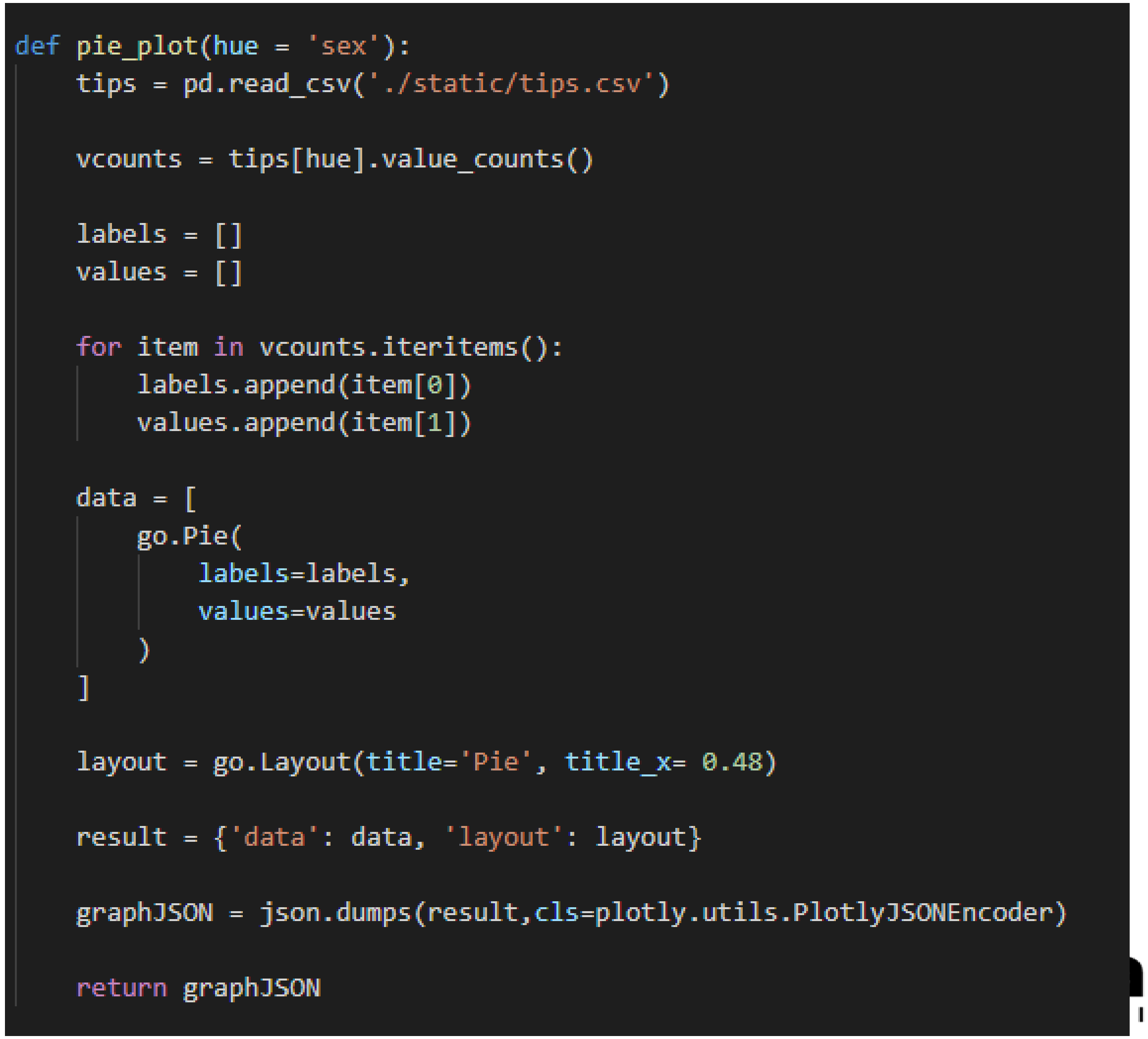
Flask Dashboard

Create Pie Plot Menu

Purwadhika
Startup and Coding School

PIE CHART PAGE (app.py)

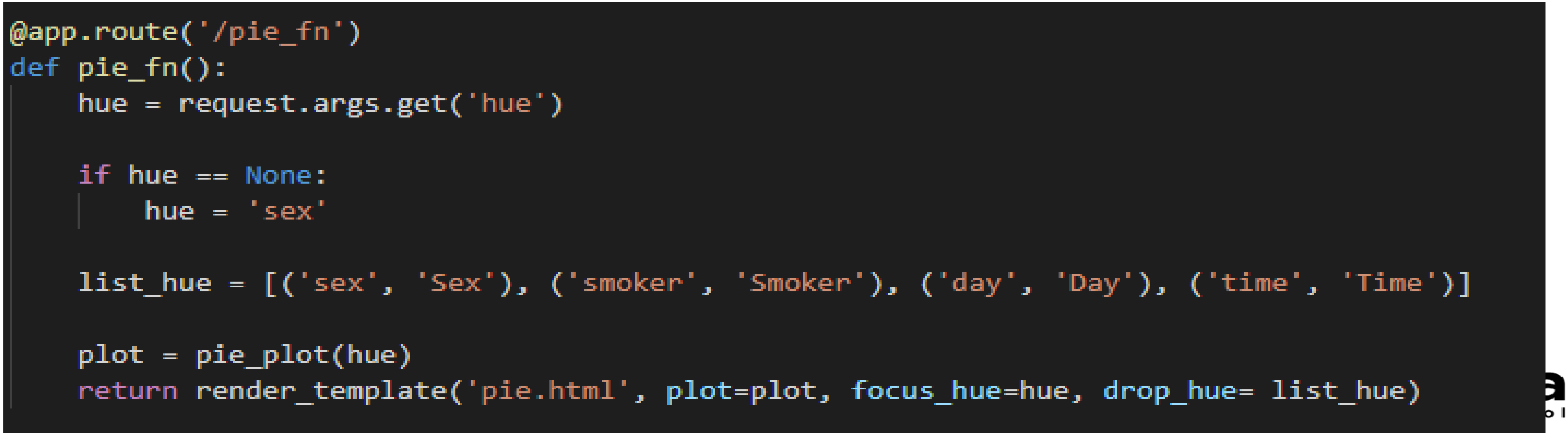
Define the function for pie plot. In this function, load the data and the rest is code to make the plot. Not like Category plot function, this pie chart function only has a plot, that is pie chart itself. Here is the code



```
def pie_plot(hue = 'sex'):
    tips = pd.read_csv('./static/tips.csv')

    vcounts = tips[hue].value_counts()

    labels = []
    values = []

    for item in vcounts.iteritems():
        labels.append(item[0])
        values.append(item[1])

    data = [
        go.Pie(
            labels=labels,
            values=values
        )
    ]

    layout = go.Layout(title='Pie', title_x= 0.48)

    result = {'data': data, 'layout': layout}

    graphJSON = json.dumps(result,cls=plotly.utils.PlotlyJSONEncoder)

    return graphJSON
```

PIE CHART PAGE (app.py)

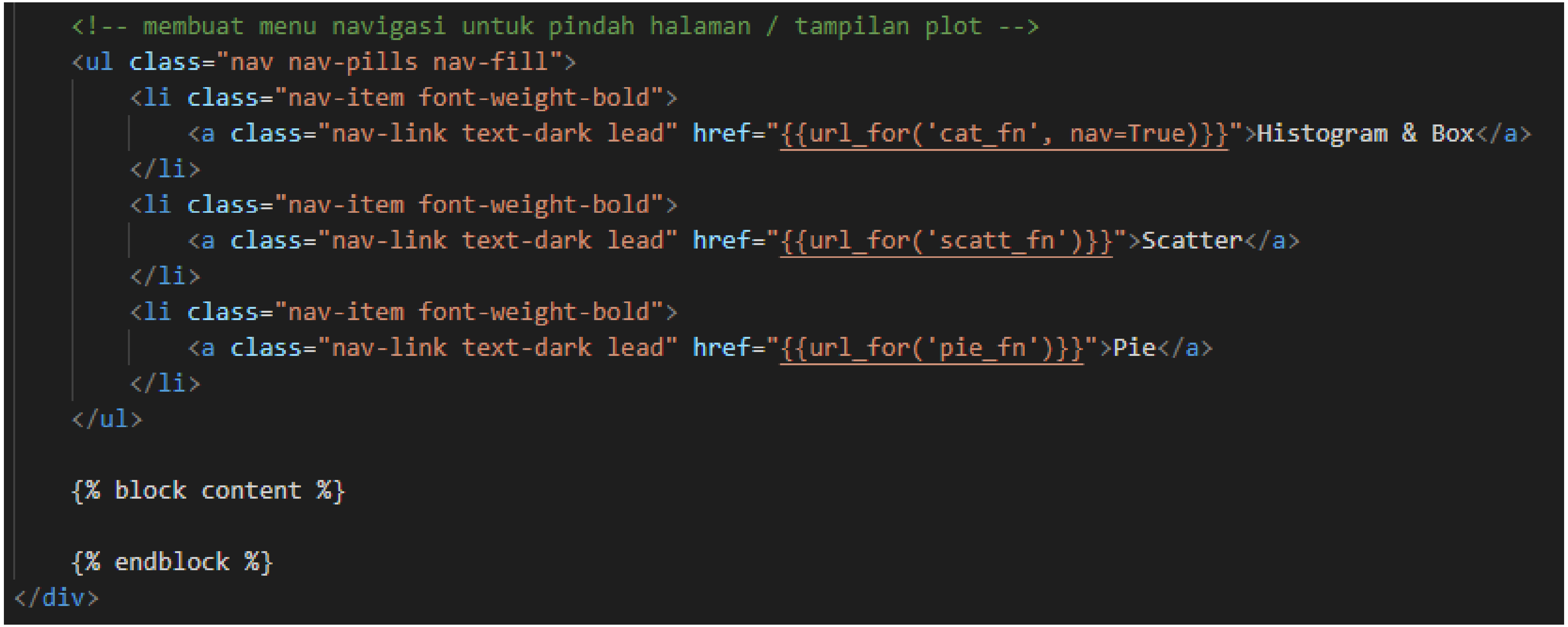
Create a router called ‘pie_fn’. The landing page of ‘Pie’ menu will use this router. It only has one component, that is Hue which consist of ‘sex’, ‘day’, ‘smoker’, and ‘time’ variable from the dataset. So, in html page of this scatter plot, there will be only a drop down menu.



```
@app.route('/pie_fn')
def pie_fn():
    hue = request.args.get('hue')

    if hue == None:
        hue = 'sex'

    list_hue = [('sex', 'Sex'), ('smoker', 'Smoker'), ('day', 'Day'), ('time', 'Time')]

    plot = pie_plot(hue)
    return render_template('pie.html', plot=plot, focus_hue=hue, drop_hue= list_hue)
```

PIE CHART PAGE (index.html)

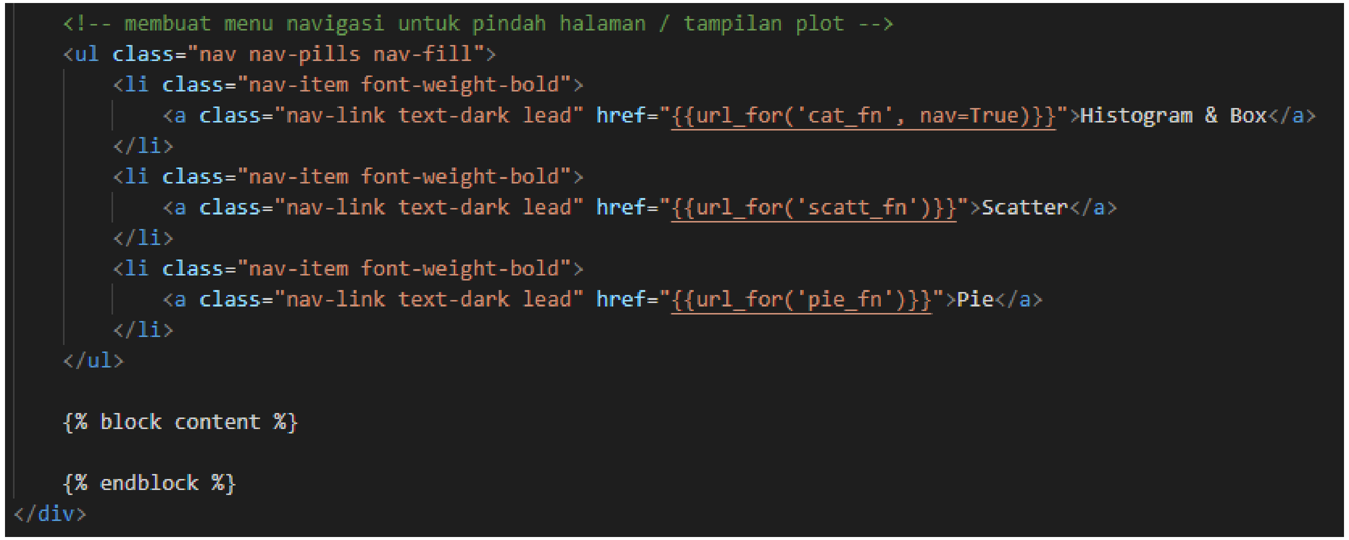
Lets move to ‘index.html’ file. In this file, same as Histogram & Box, add a ‘href’ of Pie, fill with the name of scatter plot router. Write the code by using jinja (href="{{url_for('pie_fn')}}")



```
    <!-- membuat menu navigasi untuk pindah halaman / tampilan plot -->
    <ul class="nav nav-pills nav-fill">
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('cat_fn', nav=True)}}">Histogram & Box</a>
        </li>
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('scatt_fn')}}">Scatter</a>
        </li>
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('pie_fn')}}">Pie</a>
        </li>
    </ul>

    {% block content %}

    {% endblock %}
</div>
```

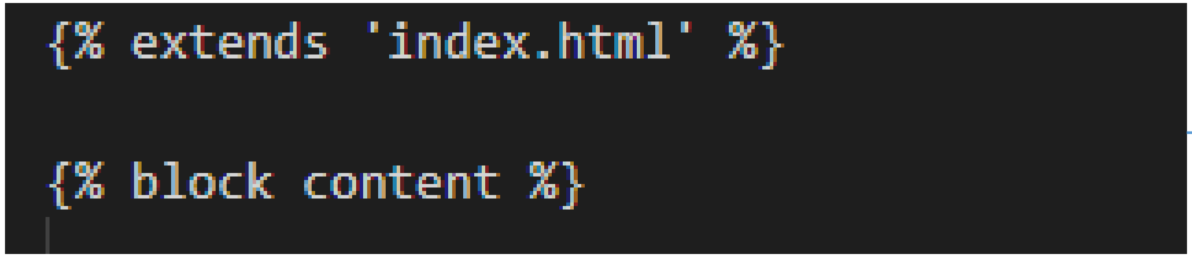
PIE CHART PAGE (pie.html)

In this page, create an extends and block content. This code is to connect from ‘index.html’ file and continue the code start from the place where the code written.



```
    <!-- membuat menu navigasi untuk pindah halaman / tampilan plot -->
    <ul class="nav nav-pills nav-fill">
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('cat_fn', nav=True)}}">Histogram & Box</a>
        </li>
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('scatt_fn')}}">Scatter</a>
        </li>
        <li class="nav-item font-weight-bold">
            <a class="nav-link text-dark lead" href="{{url_for('pie_fn')}}">Pie</a>
        </li>
    </ul>

    {% block content %}

    {% endblock %}
</div>
```

From ‘pie.html’

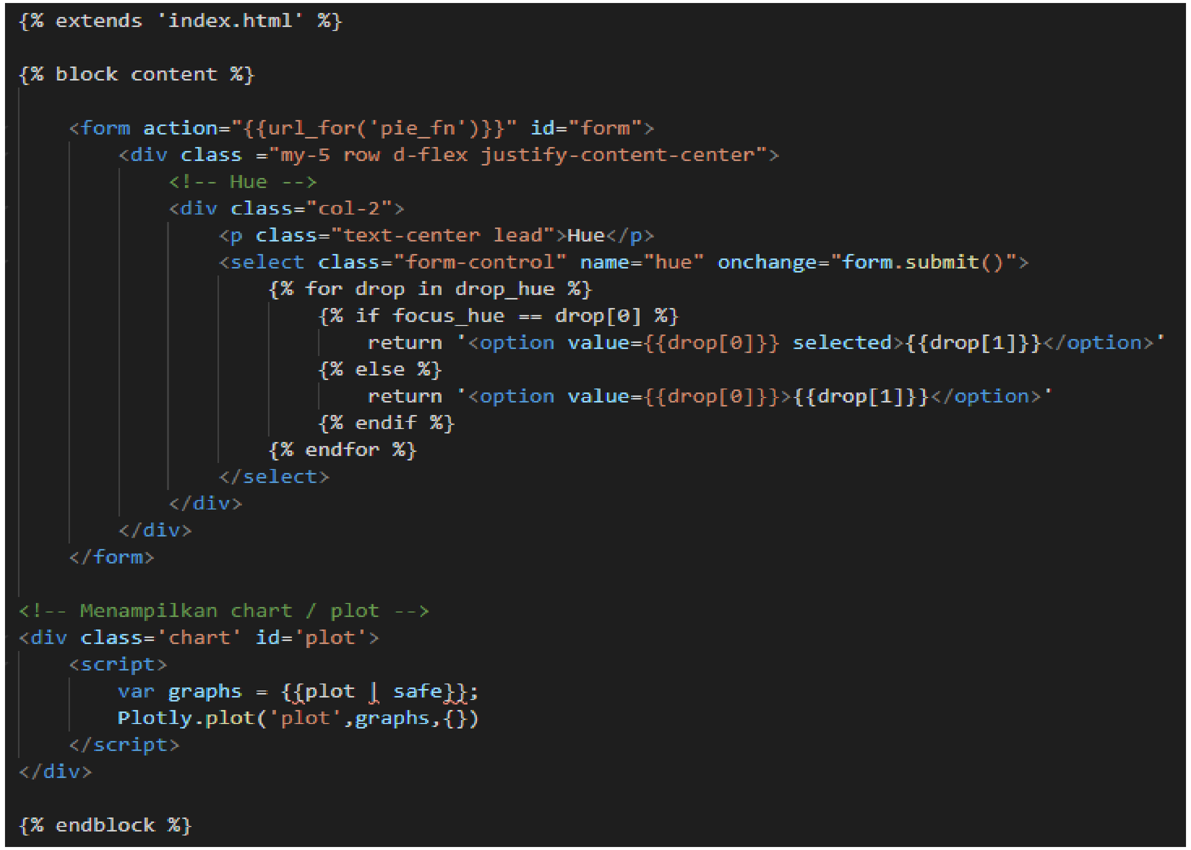


```
{% extends 'index.html' %}

{% block content %}
```

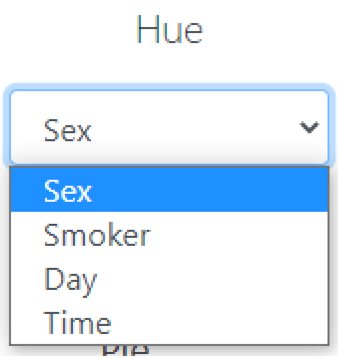
From ‘pie.html’

PIE CHART PAGE (pie.html)

Pie chart page only has one drop down menu, that is Hue drop down menu. Options that showed in hue menu are from the value in ‘drop_hue’ variable in ‘app.py’ file, of course ‘drop_hue’ from ‘pie_fn’ router. Here is the code for Hue drop down menu. Don’t forget to make extend block content connected to ‘index.html’ page.

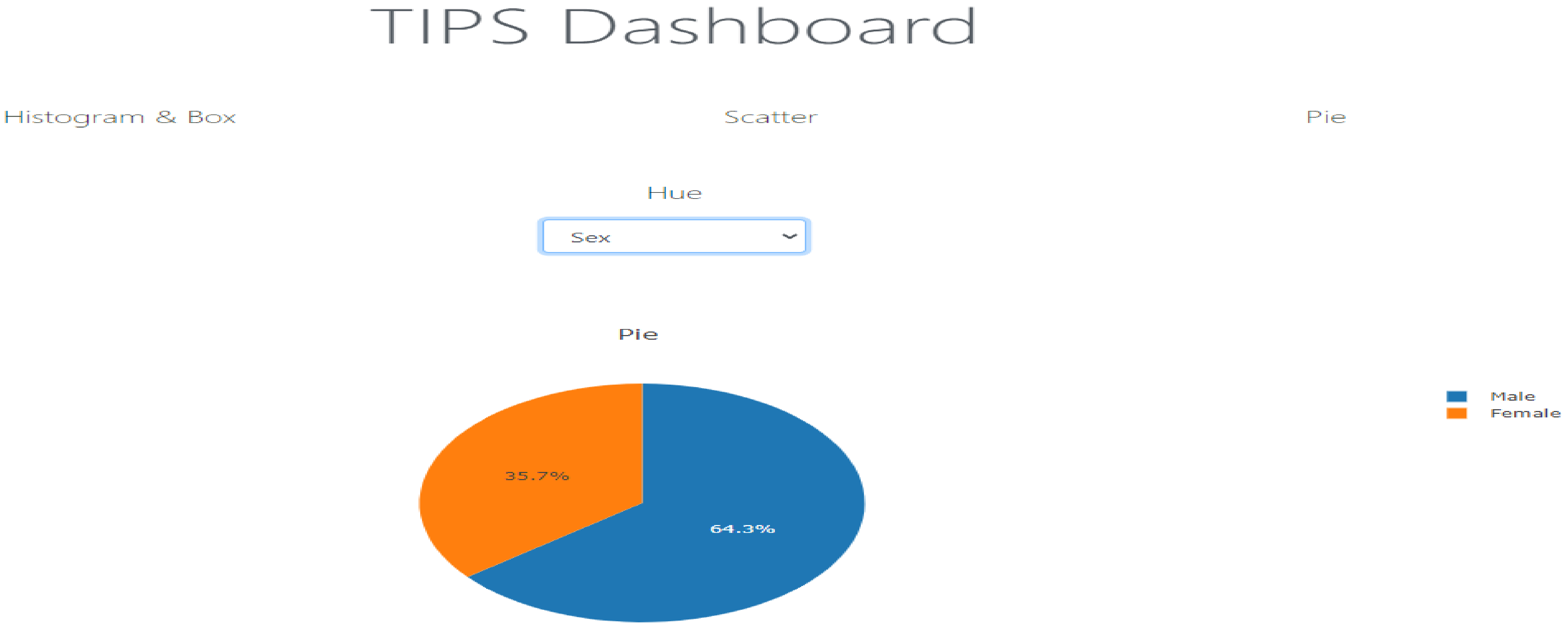


```
{% extends 'index.html' %}

{% block content %}

    <form action="{{url_for('pie_fn')}}" id="form">
        <div class ="my-5 row d-flex justify-content-center">
            <!-- Hue -->
            <div class="col-2">
                <p class="text-center lead">Hue</p>
                <select class="form-control" name="hue" onchange="form.submit()">
                    {% for drop in drop_hue %}
                        {% if focus_hue == drop[0] %}
                            return '<option value={{drop[0]}} selected>{{drop[1]}}</option>'
                        {% else %}
                            return '<option value={{drop[0]}}>{{drop[1]}}</option>'
                        {% endif %}
                    {% endfor %}
                </select>
            </div>
        </div>
    </form>

<!-- Menampilkan chart / plot -->
<div class='chart' id='plot'>
    <script>
        var graphs = {{plot | safe}};
        Plotly.plot('plot',graphs,{})
    </script>
</div>

{% endblock %}
```



PIE CHART PREVIEW PAGE

To show the dashboard, run ‘app.py’ in terminal or command prompt. After that, open http://localhost:5000 in browser.



TIPS Dashboard

Hue

Sex