Rotman

# INTRO TO DATA VISUALIZATION

## Part IV Build Dashboards with Quarto and Plotly Express

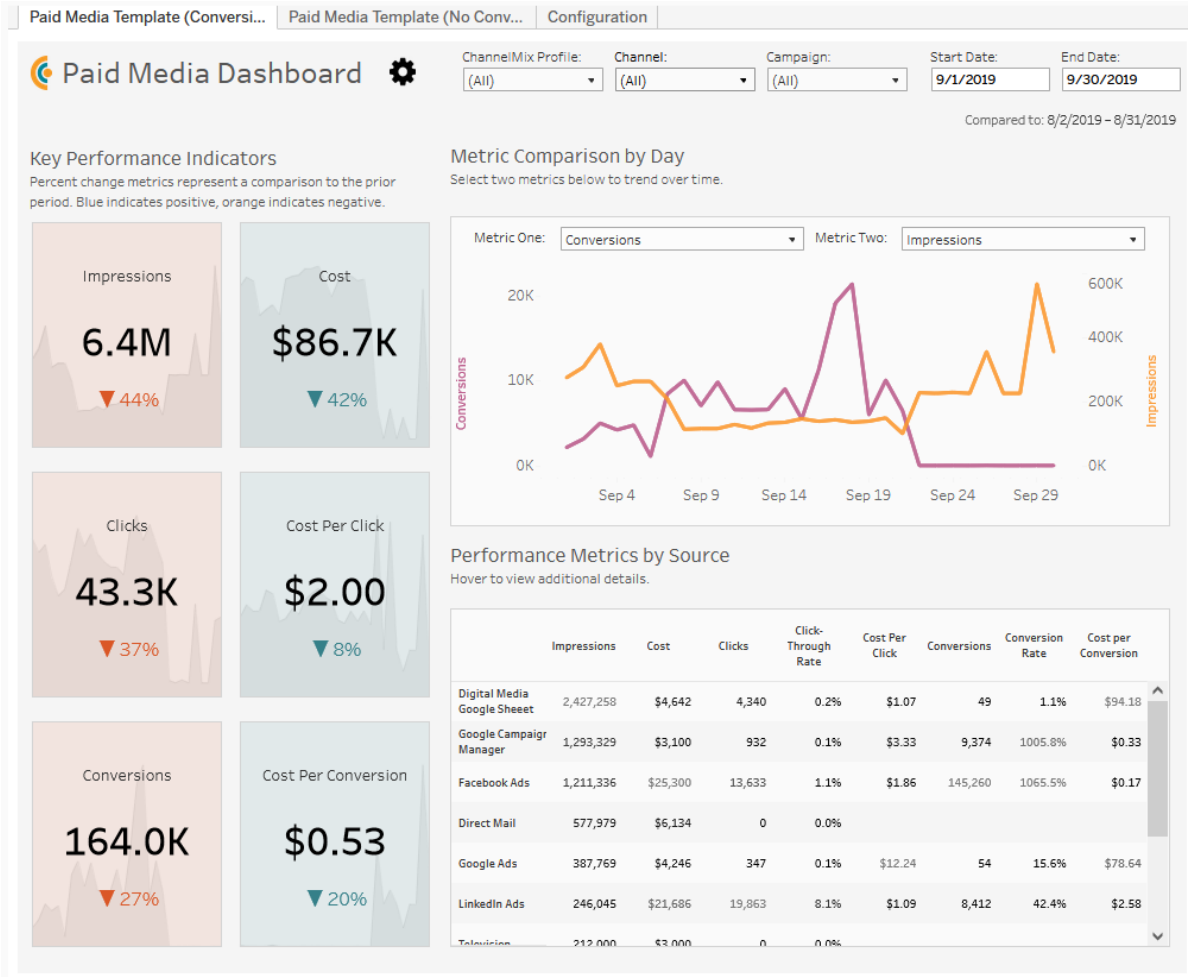
September 8, 2024 Prepared by Jay Cao / TDMDAL

Website: https://tdmdal.github.io/dv-2024/

Rotman School of Management
UNIVERSITY OF TORONTO

# What is a Dashboard

- A way to display related data visualization and summary in one place
- Usually contains interactive and dynamic features
- Usually accessible via a web browser
- Market campaign performance example
  - Key Performance Indicators (KPIs)
  - Metric comparison line plot
  - Aggregated data table
  - Interactivity

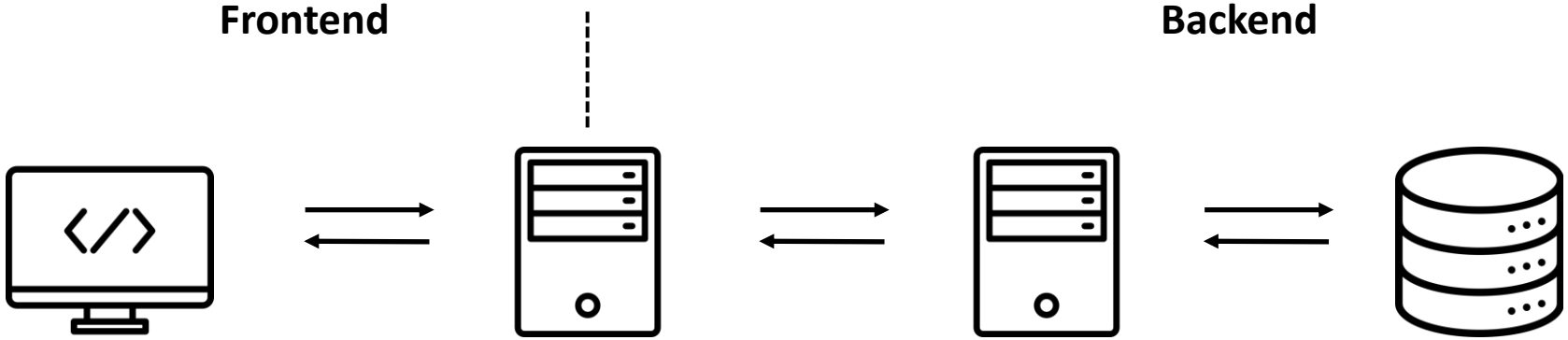



Source: Paid Media Dashboard

Ref: https://www.tableau.com/learn/articles/dashboards/what-is

# Most Dashboards are Web Apps

**Frontend** **Backend**




Render a web page
- HTML
- CSS
- JAVASCRIPT

Deliver static content
- HTML, CSS, JAVASCRIPT
- Images, Videos, etc.

Handle App Logic
Interact with DB
Generate dynamic content

e.g. a new HTML page based on user input, DB query and some computation logic.

Run-time language: Python, Javascript (Node.js), R (Shiny), etc.

Storage and manage data

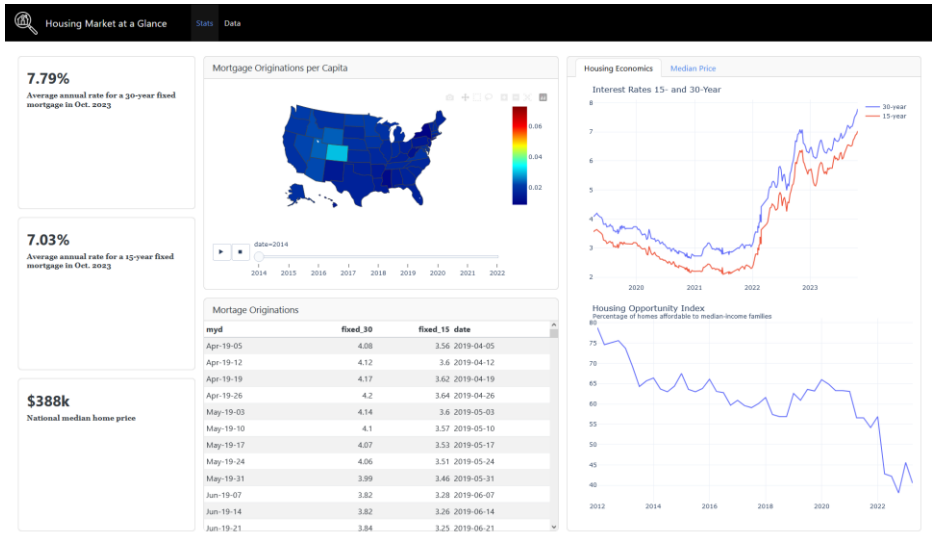
# Python Dashboard Tooling

- Purpose of those tools
  - Make dashboard building easy
  - Require minimum knowledge on web technology (html, css, javascript, etc.)
- Landscape
  - Dash (by Plotly)
  - Panel (by Anaconda)
  - Voila (by Quantstack)
  - Streamlit
  - Gradio
  - Shiny for Python (by Posit)
  - Quarto Dashboards (by Posit)

Ref: 1) https://pyviz.org/dashboarding/index.html; 2) https://posit.co/blog/why-shiny-for-python/

# Quarto Dashboards

- Easy to use
  - IMO, truly minimum knowledge of web stack
- Support Python and R
- Support simple interactivity (via Javascript)
  - can be deployed as static web pages
- Support enhanced interactivity (via Shiny for Python)
  - need to be deployed with Shiny Server
- Downside
  - Fairly new and in active dev; hence bugs and feature-incomplete




An example of Quarto dashboard with simple interactivity
https://ivelasq.github.io/mortgage-dashboard/

Ref: https://quarto.org/docs/dashboards/

# Simple Interactivity Dashboard

- Dashboard that need no backend to run (i.e. no app server, DB, etc.)
  - Sometimes called “static” dashboard, i.e., no app server dependency
- Need only a web server to make it available to the internet
  - For example, you can host it on Github for free
- Use Javascript for basic interactivity
  - You don’t need to code in Javascript
- Quarto Dashboard examples
  - https://jjallaire.github.io/stock-explorer-dashboard/

# Enhanced Interactivity Dashboard

- Dynamically retrieve data and display/visualize information
- Support complex interactivity
- Require App server, DB, etc. in addition to a Web server
- Quarto Dashboard Examples
  - https://jjallaire.shinyapps.io/penguins-dashboard/

# Simple Db – Code Skeleton 1

````
---
title: "Superstore"
format:
  dashboard:
    logo: super.png
---

```{python}
# load dataset, prepare it for display and plot

```

# Sales (`{python} year`)

## Row {height=15%}

```{python}
#| content: valuebox
#| title: "Total Sales"

```

```{python}
#| content: valuebox
#| title: "Total Profit"

```
````

# Simple Db – Code Skeleton 2

````
## Row {height=35%}

```{python}
#| title: Sales by State
#| padding: 0

# sales by state plot
```

```{python}
#| title: Sales by Segment
#| padding: 0

# sales by segment plot
```

## Row {height=35%}

```{python}
#| title: Sales by Category
#| padding: 0

# sales by category plot
```

```{python}
#| title: Sales by Sub-Category
#| padding: 0

# sales by sub-category plot
```
````

# Simple Db – Code Skeleton 3

```
## Row {height=15%}

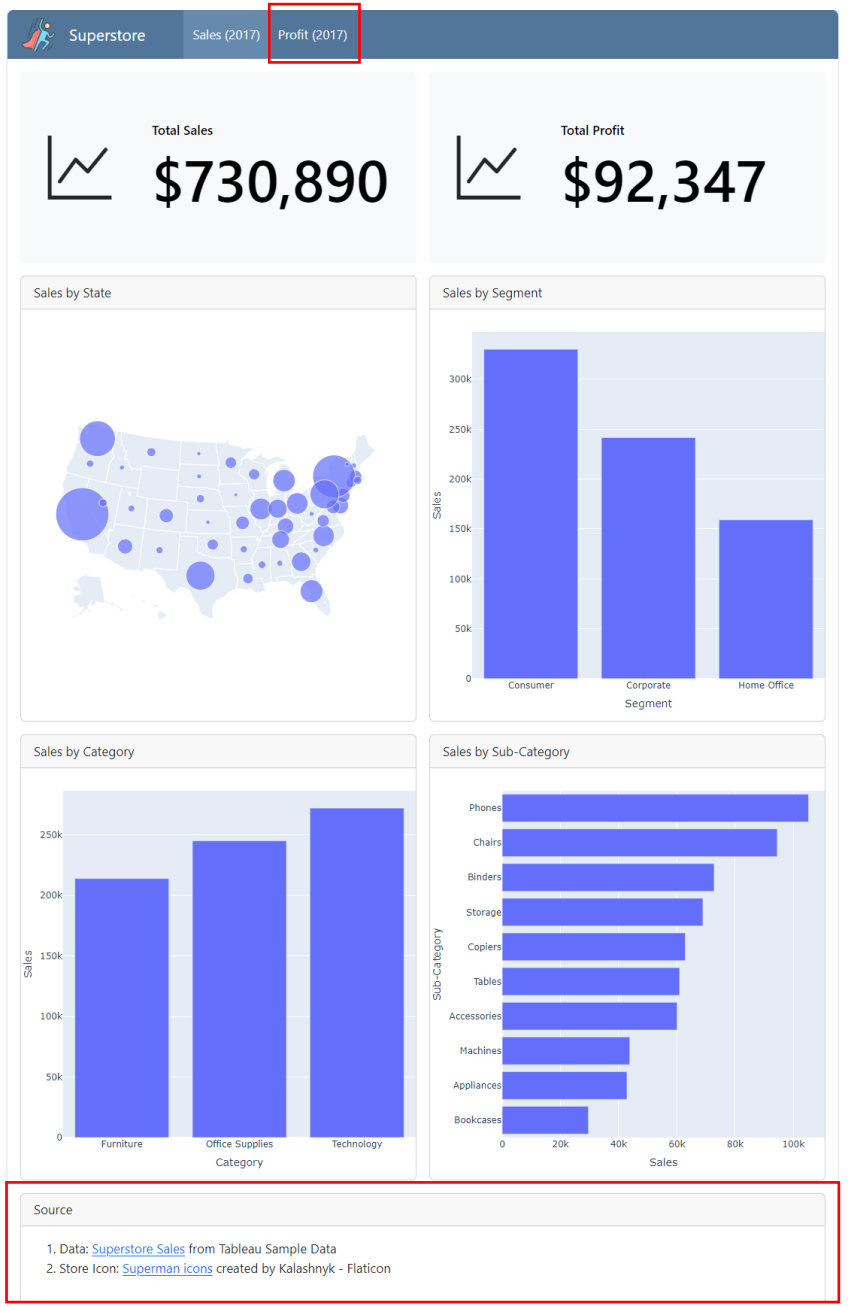
::: {.card title="Source"}

:::

# Profit (`{python} year`)

**It's Your Turn to Build.**
```

```
> quarto render superstore.qmd --to dashboard
```



# Enhanced Dashboard – Code Skeleton 1

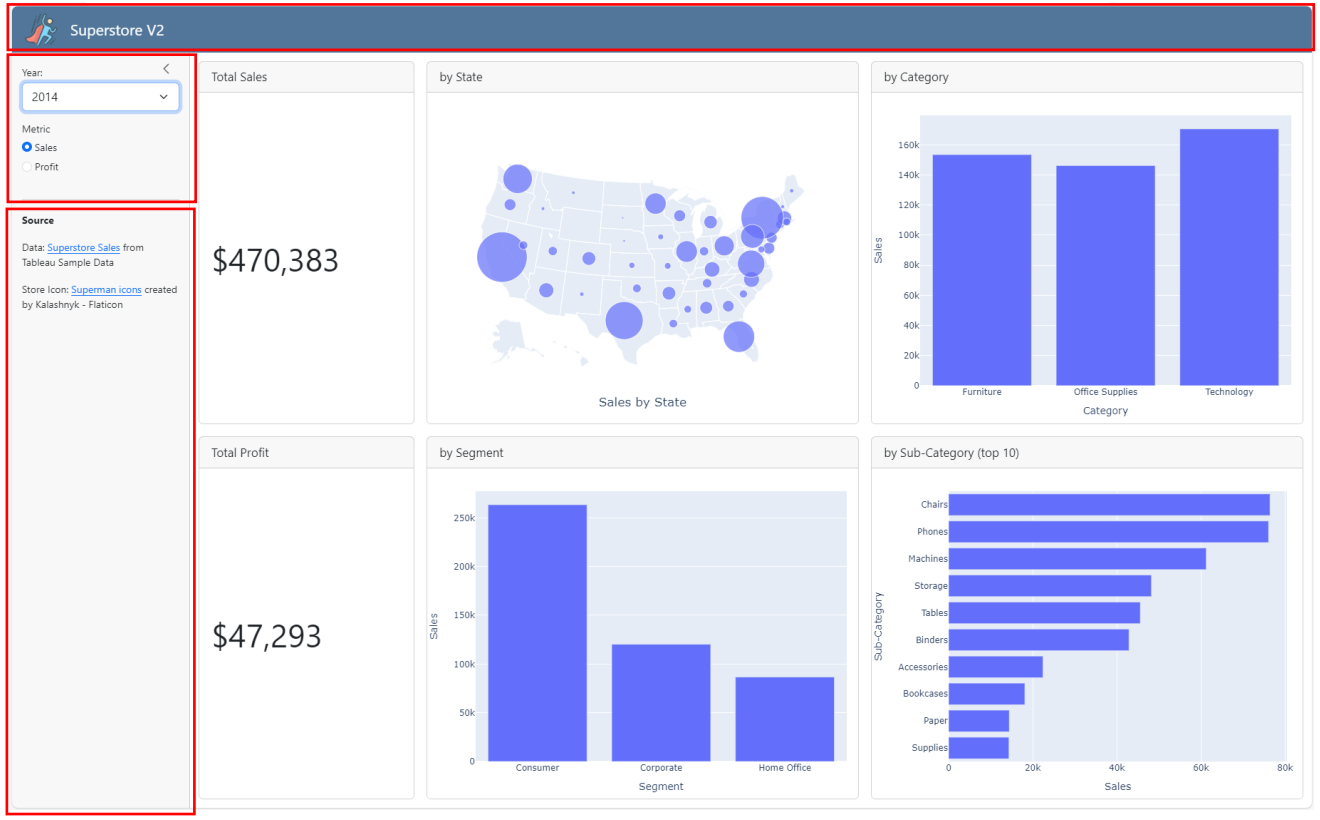
```
---
title: "Superstore V2"
format:
  dashboard:
    logo: super.png
server: shiny
---

```{python}
# load python library (shiny, plotly.express, etc.)

# load and prepare data

```

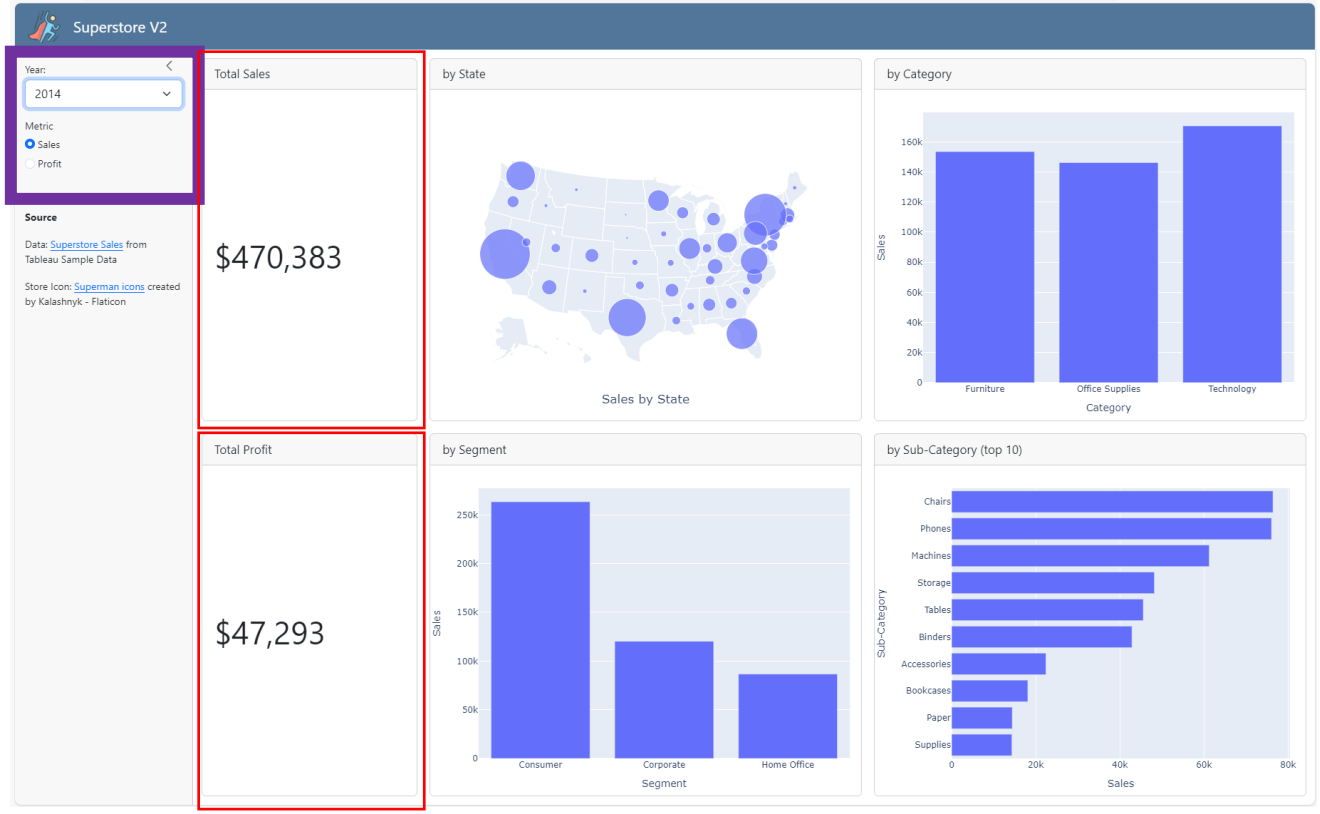
## {.sidebar}
```{python}
ui.input_select()

ui.input_radio_buttons()
```

---

**Source**

Data: [Superstore Sales](https://public.tableau.com/app/learn/sample-data) from Tableau Sample Data
```



# Enhanced Dashboard – Code Skeleton 2

```
```{python}
@reactive.calc
def ss_year():
   return ss[ss["Ship
Date"].dt.year==int(input.year())]

@reactive.calc
def sales():
   return ss_year()["Sales"].sum()

# more reactive calculation
```

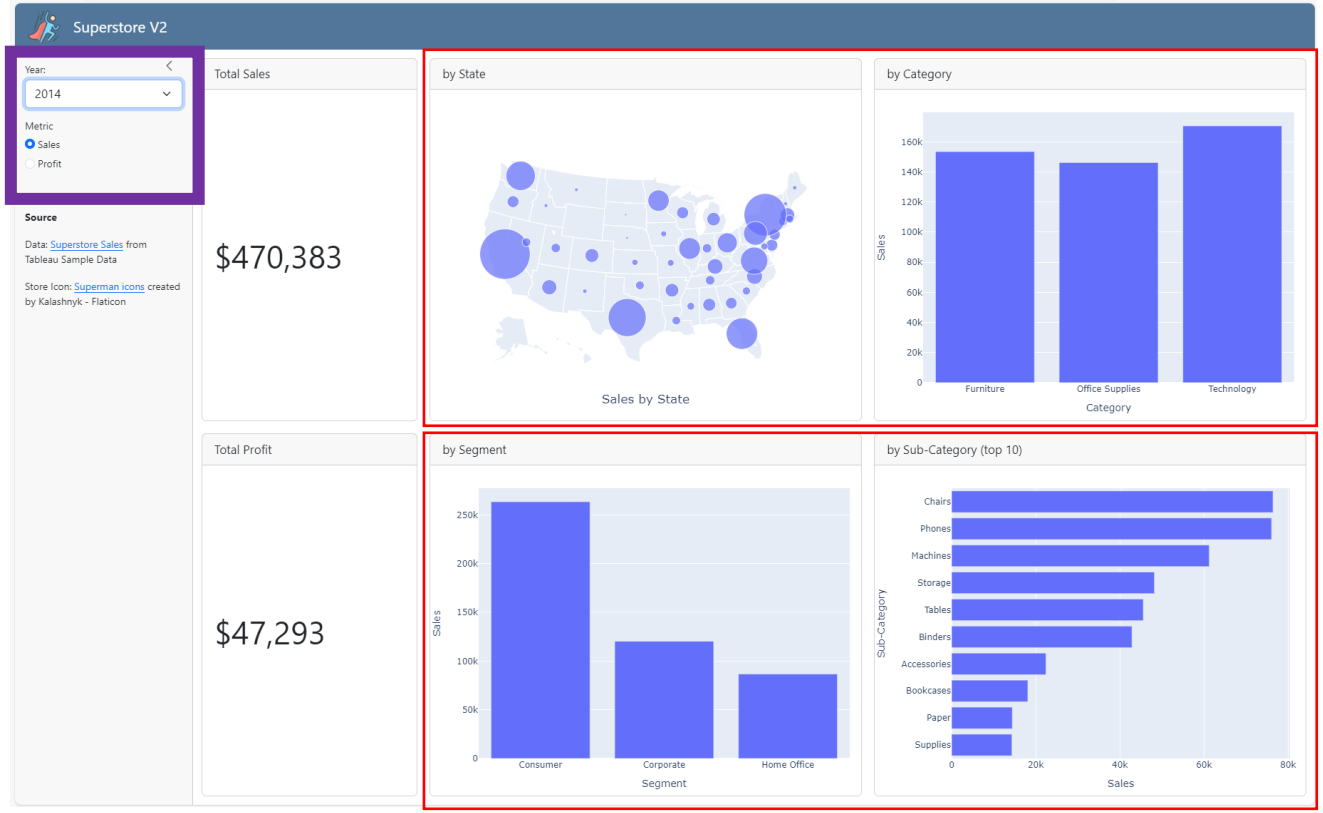
## Column {width=20%}
```{python}
#| title: Total Sales
#| padding: 0
@render.ui
def sales_value_box():
   return shiny.ui.value_box()
```

```{python}
#| title: Total Profit
#| padding: 0
@render.ui
def profit_value_box():
   return shiny.ui.value_box()
```
```



# Enhanced Dashboard – Code Skeleton 3

````
## Column {width=40%}
```{python}
#| title: by State
#| padding: 0

@render_widget
def plot_by_state():
   fig = px.scatter_geo()
   fig.layout.update()
   return fig
```

```{python}
#| title: by Segment
#| padding: 0

@render_widget
def plot_by_segment()
```

## Column {width=40%}
```{python}
#| title: by Category
#| padding: 0
```

```{python}
#| title: by Sub-Category (top 10)
#| padding: 0
```
````




```
> quarto render superstore-v2.qmd --to dashboard
> shiny run app.py
```

# Deploy Your Dashboard

- Simple (static) dashboard
  - Any public services that can host website (e.g., Quarto Pub, Github, etc.)
    - Document: Deploy to Quarto Pub or Github
  - Of course you can setup your own web server too
  - You can use `quarto publish` command to make the deployment easy

- Enhanced (shiny) dashboard
  - Public services such as shinyapps.io and Shiny on Space from Hugging Face
    - Document: Deploy to shinyapps.io or Shiny on Space
  - You can setup your own shiny server