DATA DRIVEN GALLERY

Problem Definition:

Create the app **Data Driven Gallery** to showcase the graphs created by using the D3 Visualization library

The Core idea of this app is that the user can search the graph by task, find all the graphs that are feasible to his task and can adapt it. Apart, User can also add his own d3 graph to showcase to other viewers.

There are multiple types of Graphs and only one type of user i.e., Viewer

Business requirements: nouns: color verbs: color

- All the graph displayed in the Gallery View are divided into multiple graph types with description
- The Graph can have the following information:
    - Title of the graph
    - Source of the graph
    - Iframe_url of the graph
    - Type of the graph
    - Publisher of the graph
- User can search the graph by task, title, Publisher, type of the graph
- User can view the graphs by iframe property and once he clicks the details tab the user will navigate to Graph, where all the information related to graph is displayed
- The graph type can have the description about its type
- Viewer can add his own d3 graph by providing all the information Title, Source, Iframe_url, Type, Publisher of the graph in AddGraph page.
- All the newly added graphs need to be stored in the browser storage
- User can validate his iframe is working fine before adding it to the gallery in creation page.

Nouns:

1. Graph
    a. Title
    b. Source
    c. Iframe_url
    d. Type
    e. Publisher

2. Graph Type
    a. Type
    b. Description

3. Gallery View
    a. Graph Types
    b. Description

4. AddGraph
    a. Graph

Verbs:

- View the graphs
- Search the graph by task, title, Publisher, type
- Add the graphs
- Navigate to details
- Validate the iframe
- Store the data in browser storage

Target Audience:

The Target Audience for this specific d3 gallery app is the users who are working on Data Visualization and need the best graph for their task. Also, the user can showcase their graphs which will be helpful to other user who are working on the similar task

Use Cases:

- Graph View:
    - User: Viewer
    - Task: The Viewer can view all the graph divided by the graph types and he can click left and right button to view more graphs of the same type

- Graph Search:
    - User: Viewer
    - Task: A Viewer can search the graph based on the title, task, publisher, and the source

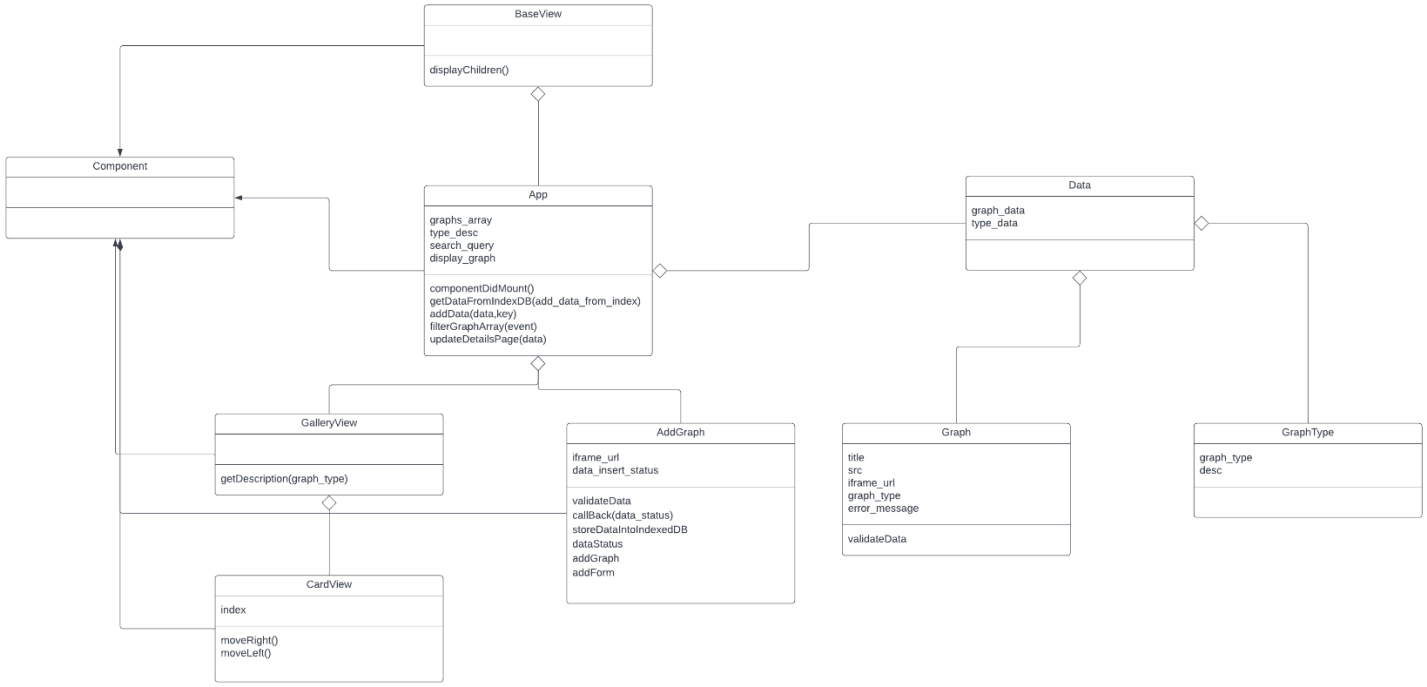
- Detailed Graph View
    - User: Viewer
    - Task: A Viewer can navigate to detailed graph view once he clicks on the details of the graph and can have detailed view

- Add the Graph:
    - User: Viewer
    - Task: A Viewer can add his d3 graph by completing the form with title, type, iframe, task, publisher, and source

- Validate the iframe:
    - User: Viewer
    - Task: A Viewer can validate his iframe before adding the graph to gallery by checking the display

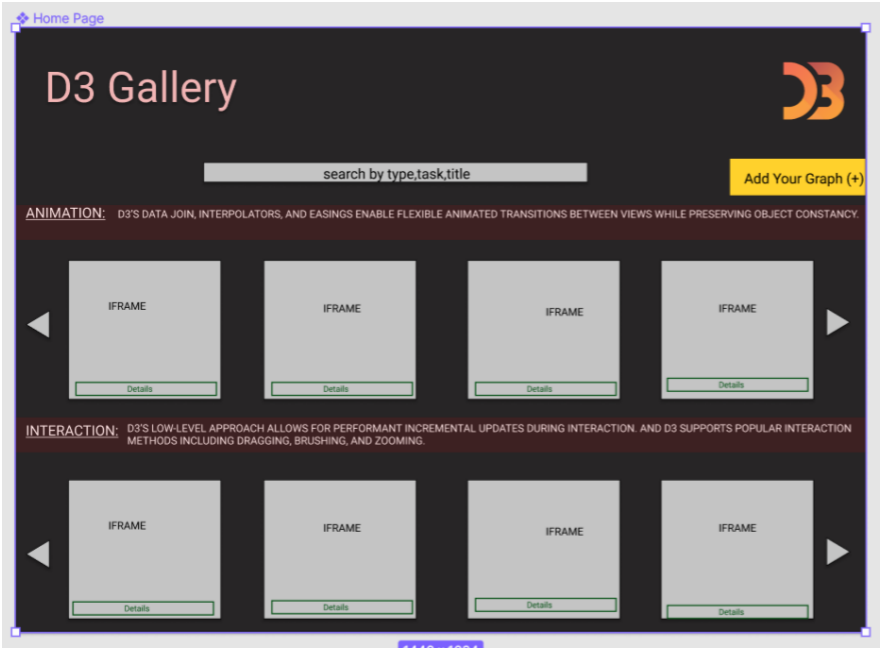
- Store the data in Brower Storage:
    - User: Viewer
    - Task: Once the user adds the graph the data needs to be stored in the browser storage. It helps to fetch the data even the user refreshed the page as we are not using the database to store the graph.

UML Diagram

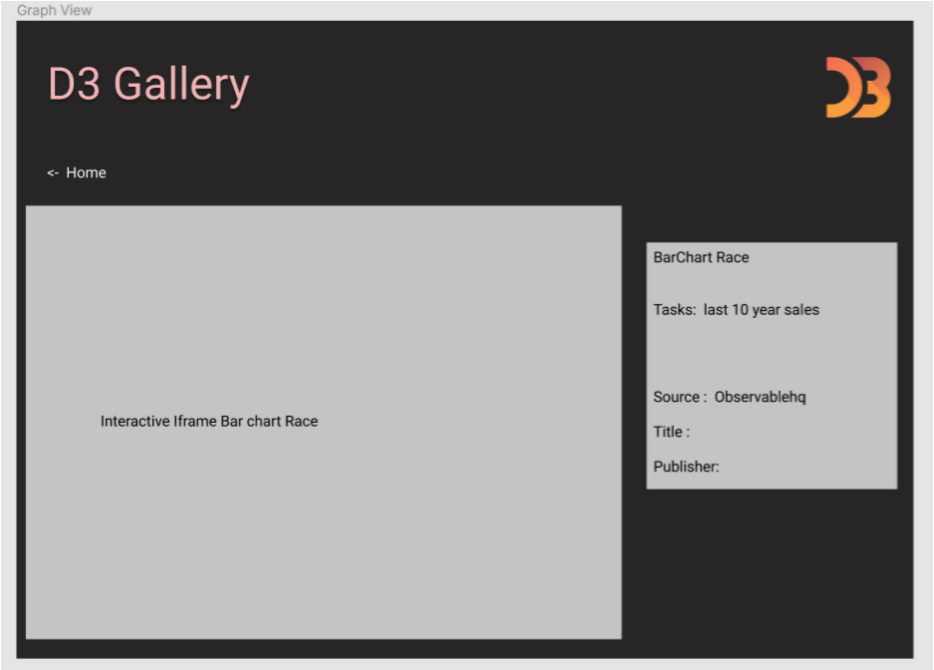

BaseView
displayChildren()
Component
App
graphs_array
type_desc
search_query
display_graph
componentDidMount()
getDataFromIndexDB(add_data_from_index)
addData(data,key)
filterGraphArray(event)
updateDetailsPage(data)
Data
graph_data
type_data
GalleryView
getDescription(graph_type)
AddGraph
iframe_url
data_insert_status
validateData
callBack(data_status)
storeDataIntoIndexedDB
dataStatus
addGraph
addForm
Graph
title
src
iframe_url
graph_type
error_message
validateData
GraphType
graph_type
desc
CardView
index
moveRight()
moveLeft()

Interface Mockups

Home Page:
The User can search the graph and view all the filtered Graphs. The User can scroll left and right if there are more graphs



GraphDetail View:



AddGraph View:

User can add the graph and store it in the index db browser storage and for the next time you can see the graph in gallery. Also once the user pastes the iframe the iframe renders and user can Verify his iframe before adding the graph details.