CS 586 Project Report

Requirements and Design

Team 1

Liang, Bobo - A20356451
Liu, Yi - A20373003
Mei, Tian - A20359022
Zhu, Zhanjiu - A20378253

1. Project overview statement

Our project are an dashboard for the environment datasets. Users can upload datasets from local or select dataset from the category we provide in the select list. A table for the dataset will show, and User can choose to filter columns and rows for chart/plot. There are five kinds of charts we support for the selected dataset. All function are realized based on requirements and the interface are very friendly for user. We have also test manually for test cases based on the requirements, no issue found, all function works as designed. Welcome to have a try!

2. Requirements/Features List (All Requirements and Features must be numbered)

Requirements:

Functional requirements:

1. The dataset can be imported from a JSON file.
2. The user can see the selected dataset in table format in browser.
3. The user can select one/many chart types of Line, Bar, Pie, and Stacked chart.
4. The user can select/deselect the Columns to include/exclude the data from the dataset for charting the selected data.
5. The user can select/deselect Columns for inclusion/exclusion of selected data from the dataset for charting.
6. The user can filter the rows to include/exclude the data from the dataset for charting.
7. The user can plot/chart basic stat data
8. The user can make complex query, like group by, min, max, average, count, standard deviation applied on their selected dataset, and chart these static data.

Nonfunctional requirements:

1. At least 5 datasets must be present for selection and testing.
2. Need be supported by firefox, Chrome.
3. Interface of the dashboard should be simple and user-friendly.
4. Response time of each operation should not exceed maximum time limit.
5. The dashboard should be reliable.
6. Dashboard should also minimize the operation complexity.

Dashboard Feature List:

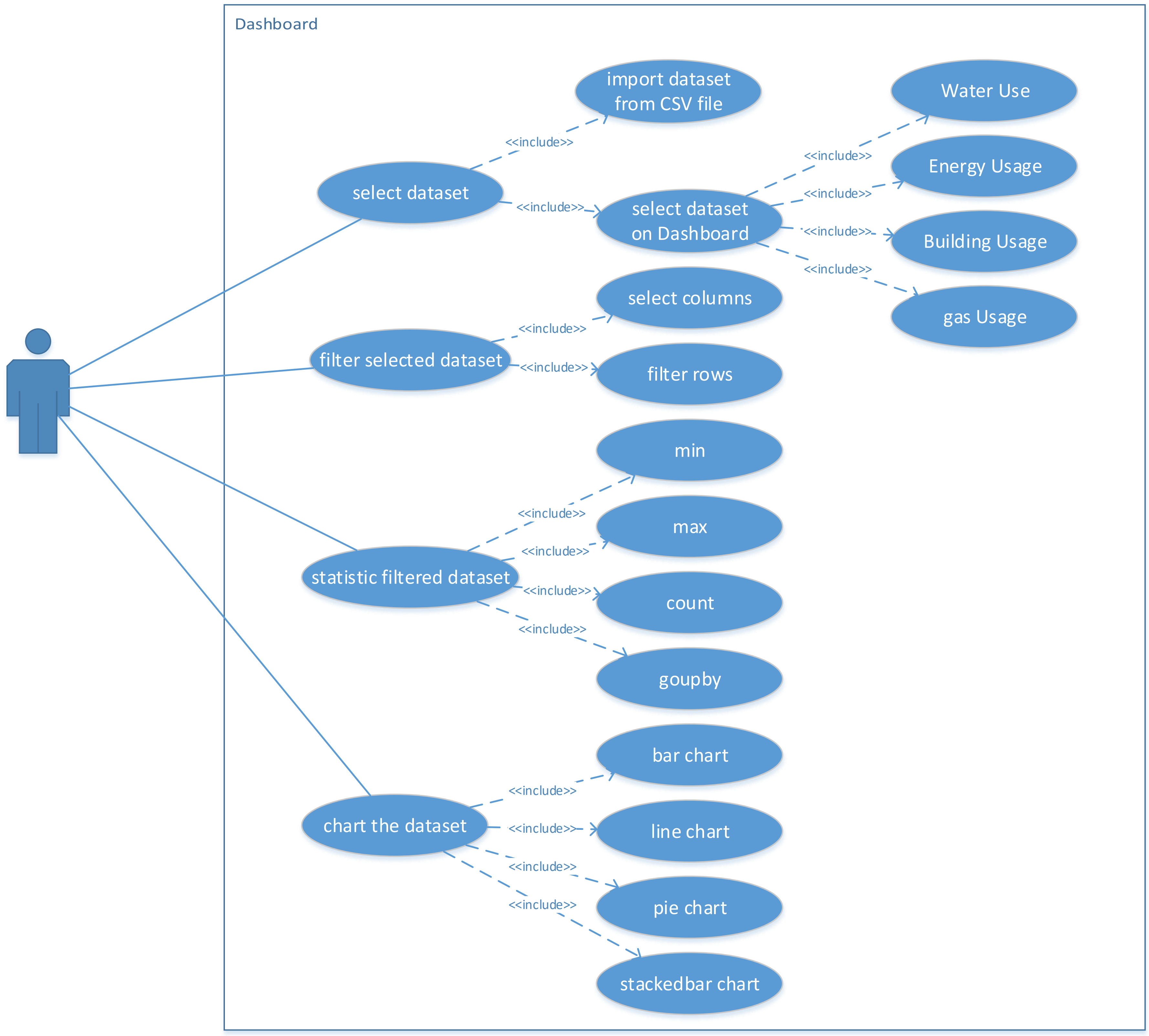
1. Import datasets from JSON files.
2. Select a dataset from dataset lists.
3. Display Dataset as table in browser.
4. Select/deselect columns in dataset.
5. Filter rows in dataset.
6. Graph/plot the data in one/many chart types.
7. Calculate and chart statistic data.
8. Enable complex analytical queries, chart the stat data.

3. Use Cases and Use Case Diagram

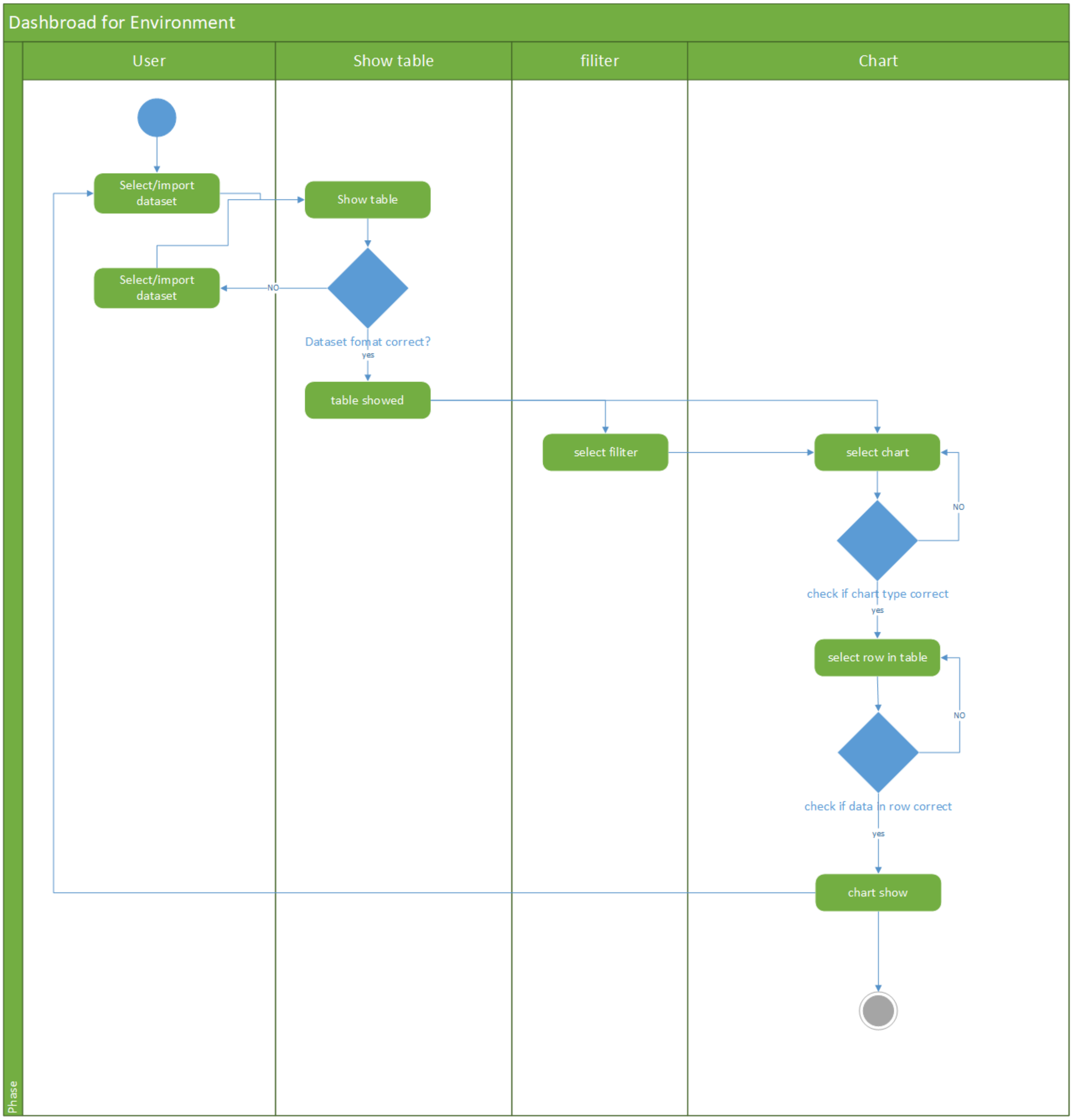
Use cases:

1. See the selected dataset in table format in browser.
2. Select one/many chart types of Line, Bar, Pie, and Stacked chart.
3. Select/deselect the Columns to include/exclude the data from the dataset.
4. Select/deselect Columns for inclusion/exclusion of selected data.
5. Filter the rows to include/exclude the data.
6. Plot/chart basic stat data.
7. Make complex query and plot the stat data.
8. Import data to Dashboard from JSON files.

Use case diagram:

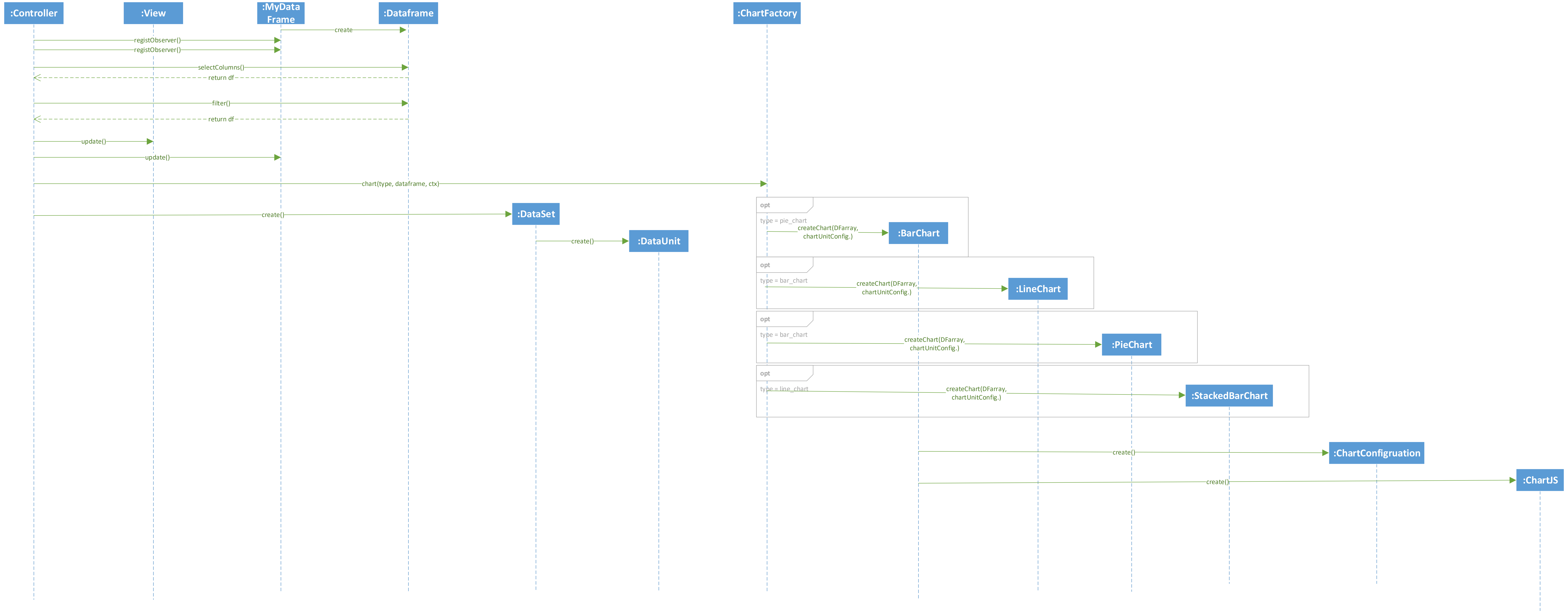

Dashboard
import dataset from CSV file
select dataset
<<include>>
<<include>>
select dataset on Dashboard
Water Use
Energy Usage
Building Usage
gas Usage
<<include>>
<<include>>
<<include>>
<<include>>
select columns
filter selected dataset
filter rows
<<include>>
<<include>>
min
max
count
goupby
statistic filtered dataset
<<include>>
<<include>>
<<include>>
<<include>>
bar chart
line chart
pie chart
stackedbar chart
chart the dataset
<<include>>
<<include>>
<<include>>
<<include>>

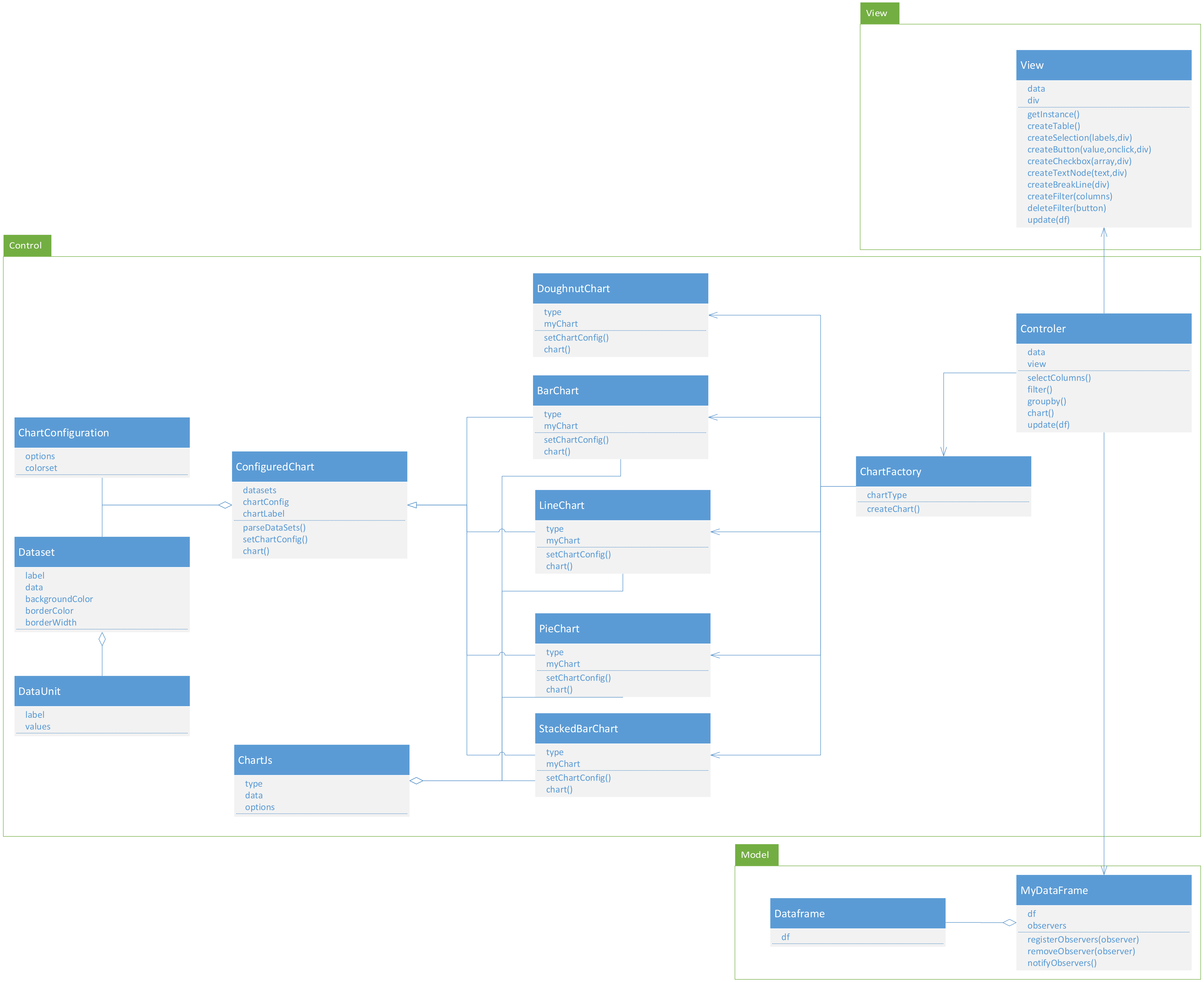
4. Activity Diagrams



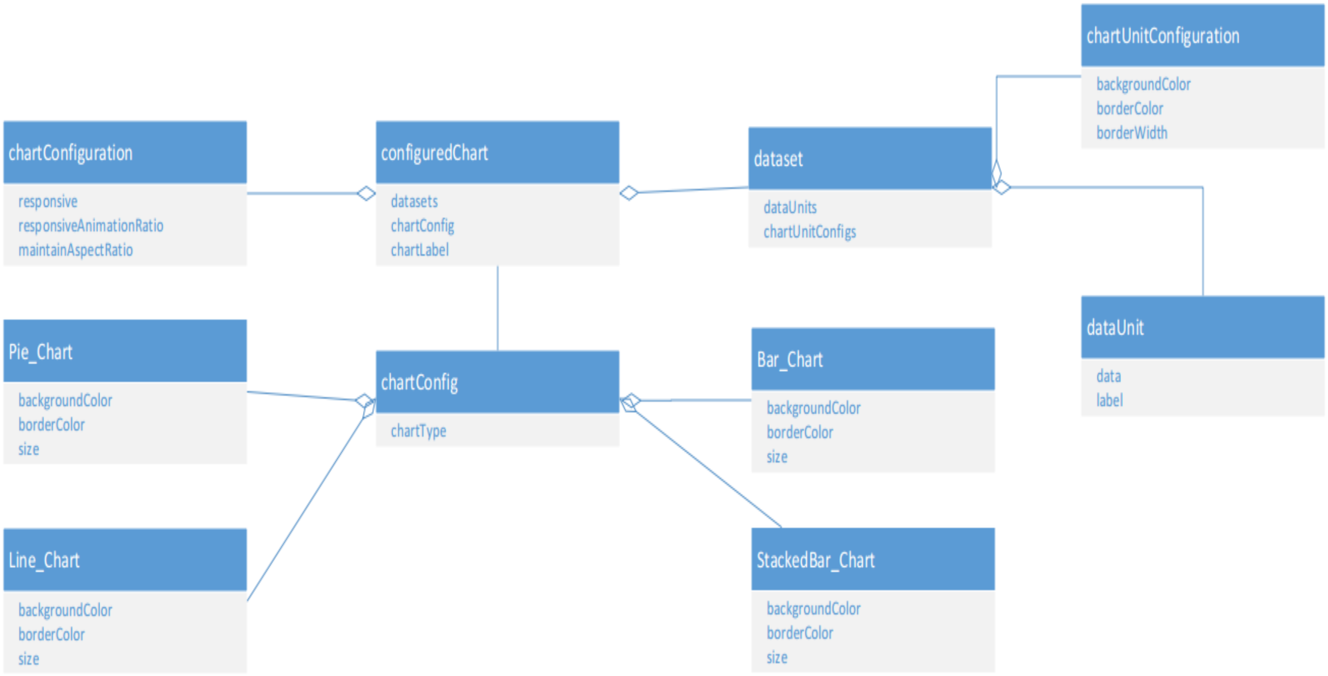
5. Sequence Diagrams

6. Design Model Class Diagram


:Controller
:View
:MyData Frame
:Dataframe
:ChartFactory
create
registObserver()
registObserver()
selectColumns()
return df
filter()
return df
update()
update()
chart(type, dataframe, ctx)
create()
:DataSet
create()
:DataUnit
opt
type = pie_chart
createChart(DFarray, chartUnitConfig.)
:BarChart
opt
type = bar_chart
createChart(DFarray, chartUnitConfig.)
:LineChart
opt
type = bar_chart
createChart(DFarray, chartUnitConfig.)
:PieChart
opt
type = line_chart
createChart(DFarray, chartUnitConfig.)
:StackedBarChart
create()
:ChartConfigruation
create()
:ChartJS


View
View
data
div
getInstance()
createTable()
createSelection(labels,div)
createButton(value,onclick,div)
createCheckbox(array,div)
createTextNode(text,div)
createBreakLine(div)
createFilter(columns)
deleteFilter(button)
update(df)
Control
DoughnutChart
type
myChart
setChartConfig()
chart()
BarChart
type
myChart
setChartConfig()
chart()
Controler
data
view
selectColumns()
filter()
groupby()
chart()
update(df)
ChartConfiguration
options
colorset
ConfiguredChart
datasets
chartConfig
chartLabel
parseDataSets()
setChartConfig()
chart()
LineChart
type
myChart
setChartConfig()
chart()
ChartFactory
chartType
createChart()
Dataset
label
data
backgroundColor
borderColor
borderWidth
PieChart
type
myChart
setChartConfig()
chart()
DataUnit
label
values
StackedBarChart
type
myChart
setChartConfig()
chart()
ChartJs
type
data
options
Model
MyDataFrame
df
observers
registerObservers(observer)
removeObserver(observer)
notifyObservers()
Dataframe
df

7. Domain Model



8. Documentation and class diagrams for Design Patterns used.

- **MVC pattern& Observer pattern**

  Applied classes:

  M: MyDataFrame class
  V: View class
  C: Controler Class and all the other related logical classes

  Usage:

  MyData is a Model class, it can register() other objects as observers. When it's data changed, it will notifyObserver()

  View is a View class, it charges for the UI generating and changes, also it is a observer of Mydata, when data of Mydata changes, it will be called by Mydata via updata()

  Controler is a C class, it has Mydata and view, control the logical business.

- **Singleton Pattern**

  Applied class: View class
  Usage: view class's constructor is a private constructor, it can be only called by its method getInstance(), this method first judge whether there is already a view exist, only create view object when there is no view exist.

- **Factory method pattern**
  Applied classes: ChartFactory, ConfiguredChart, BarChart, LineChart, PieChart, DoughountChart, StackedChart

  Usage: ChartFactory class define the factory, it will create different chart according to chart type pass to it.
- **Template method pattern**
  Applied classes: ConfiguredChart, BarChart, LineChart, PieChart, DoughountChart, StackedChart

  Usage: ConfiguredChart is abstract class of the others, it has method template of : parseDataset(), setChartConfig(), chart(). It implement the parseDataset() method, the setChartConfig() and chart() will be defined by its subclasses.