Collaboration Diagrams

Initialize

Object

1. Search an item
   1. :Initialize

Clothing

* 1. :Clothing
     1. searchClothing() 🡨 Database
     2. returnClothes() 🡨 Memory Stored
  2. :GUI

GUI

* + 1. displayList() 🡨 Info Stored
  1. :EndOfProcess

Front End/JQuery

EndOfProcess

Initialize

1. Filter the Search
   1. :Initialize
   2. :Clothing

Object

* + 1. readSearch() 🡨 Message

Clothing

* + 1. searchClothing() 🡨 Database
  1. :GUI
     1. displayList() 🡨 Info Stored

GUI

* 1. :EndOfProcess

Front End/JQuery

EndOfProcess

Initialize

1. Alter Dashboard 1
   1. :Initialize
   2. :GUI

GUI

* + 1. newData() 🡨 CSV or SVG

Front End/JQuery

* + 1. googleChartsUpdate() 🡨 GoogleTalk
    2. displayChart() 🡨 GraphUpdate
  1. :EndOfProcess

EndOfProcess

Initialize

1. Alter Dashboard 2
   1. :Initialize

Front End/JQuery

* 1. :GUI

GUI

* + 1. newData() 🡨 CSV or SVG
    2. googleChartsUpdate() 🡨 GoogleTalk
    3. displayChart() 🡨 GraphUpdate

EndOfProcess

* 1. :EndOfProcess

Initialize

1. Alter Dashboard 3
   1. :Initialize
   2. :GUI

Front End/JQuery

* + 1. newData() 🡨 CSV or SVG

GUI

* + 1. googleChartsUpdate() 🡨 GoogleTalk
    2. displayChart() 🡨 GraphUpdate
  1. :EndOfProcess

EndOfProcess

Detailed Design

1. Search an item

searchClothes(String)

clothes = []

String url = "jdbc:weblogic:oracle";

String user= "java\_developer";

String pwd = "oracle";

Connection con = DriverManager.getConnection(url, user, pwd);

SQLTalk database = new SQLTalk()

clothes = database.getMatch(String)

//getMatch queries the database for all partial and complete matches to the string entered

//this is how

/\* String = Tokenize(string)

if database[i] contains tokens

store database[i]

return array of stored variables

\*/

return clothes

1. Filter the search

filterSearch(String, Filter)

clothes = []

String url = "jdbc:weblogic:oracle";

String user= "java\_developer";

String pwd = "oracle";

Connection con = DriverManager.getConnection(url, user, pwd);

SQLTalk database = new SQLTalk()

clothes = database.getFilteredMatch(String, Filter)

//getMatch queries the database for all partial and complete matches to the string entered

//this is how

/\* String = Tokenize(string)

if database[i] in Filter contains tokens

store database[i]

return array of stored variables

\*/

return clothes

1. Alter Dashboard 1

alterDB1(newData (CSV || SVG))

//JQuery

<html>

<head>

<!--Load the AJAX API-->

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>

<script type="text/javascript">

// Load the Visualization API and the piechart package.

google.charts.load('current', {'packages':['corechart']});

// Set a callback to run when the Google Visualization API is loaded.

google.charts.setOnLoadCallback(drawChart);

// Callback that creates and populates a data table,

// instantiates the pie chart, passes in the data and

// draws it.

function drawChart() {

// Create the data table.

var data = new google.SVG.visualization(newData)

// Set chart options

var options = {'title':'How Much Pizza I Ate Last Night',

'width':400,

'height':300};

// Instantiate and draw our chart, passing in some options.

var chart = new google.visualization.PieChart(document.getElementById('chart\_div'));

chart.draw(data, options);

}

1. Alter Dashboard 2

alterDB2(newData)

//JQuery

<html>

<head>

<!--Load the AJAX API-->

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>

<script type="text/javascript">

// Load the Visualization API and the piechart package.

google.charts.load('current', {'packages':['corechart']});

// Set a callback to run when the Google Visualization API is loaded.

google.charts.setOnLoadCallback(drawChart);

// Callback that creates and populates a data table,

// instantiates the pie chart, passes in the data and

// draws it.

function drawChart() {

// Create the data table.

var data = new google.SVG.visualization(newData)

// Set chart options

var options = {'title':'How Much Pizza I Ate Last Night',

'width':400,

'height':300};

// Instantiate and draw our chart, passing in some options.

var chart = new google.visualization.PieChart(document.getElementById('chart\_div'));

chart.draw(data, options);

}

1. Alter Dashboard 3

alterDB3(newData)

//JQuery

<html>

<head>

<!--Load the AJAX API-->

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>

<script type="text/javascript">

// Load the Visualization API and the piechart package.

google.charts.load('current', {'packages':['corechart']});

// Set a callback to run when the Google Visualization API is loaded.

google.charts.setOnLoadCallback(drawChart);

// Callback that creates and populates a data table,

// instantiates the pie chart, passes in the data and

// draws it.

function drawChart() {

// Create the data table.

var data = new google.SVG.visualization(newData)

// Set chart options

var options = {'title':'How Much Pizza I Ate Last Night',

'width':400,

'height':300};

// Instantiate and draw our chart, passing in some options.

var chart = new google.visualization.PieChart(document.getElementById('chart\_div'));

chart.draw(data, options);

}