

Software Engineering Group Project

Deliverable 3 - Increment 2

Group 33: David Jones, Simeon Milev, Harry Brown, Iliana Hadzhiatanasova, Kiera Spencer-Hayles, Ina Li

1 - Design

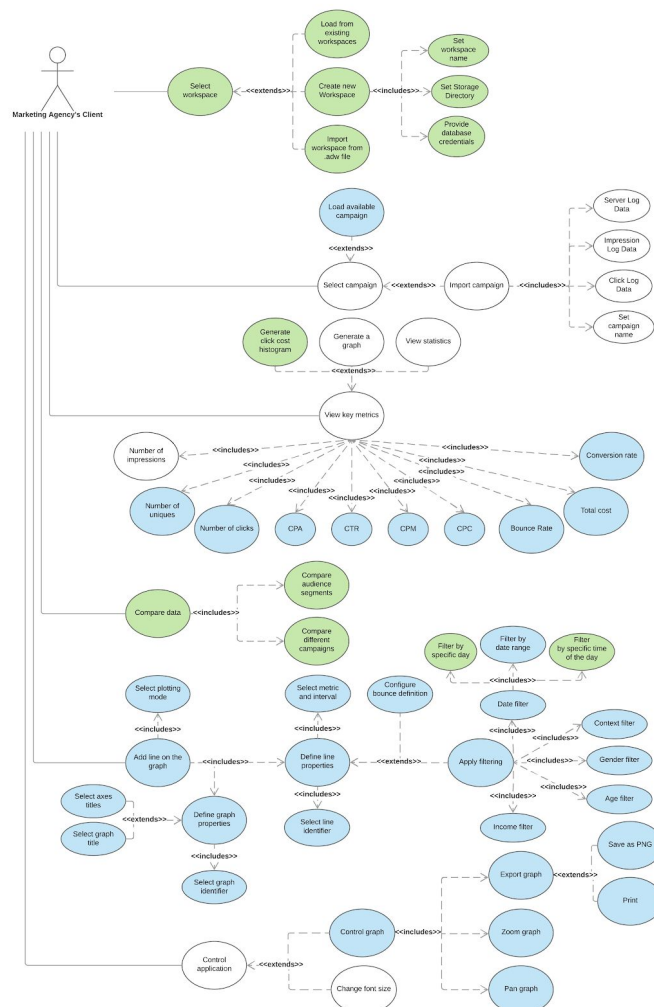
Section Summary

In this section we have included diagrams and wireframes to support our design decisions. The **Use Case Diagram** depicts all possible use cases that a user of the system might be involved in. The **Activity Diagram** outlines the various runtime scenarios the user can be involved in. Both diagrams show functionality of the system over **all three Sprints**. The **Sequence Diagram** covers the user-system-database interactions for system functionality added only in **Sprint Three**. The **Wireframes** show user interfaces added or amended in the course of this Sprint, while the **Storyboards** depict the workflow of basic scenarios.

1.1 - Key Design Artifacts

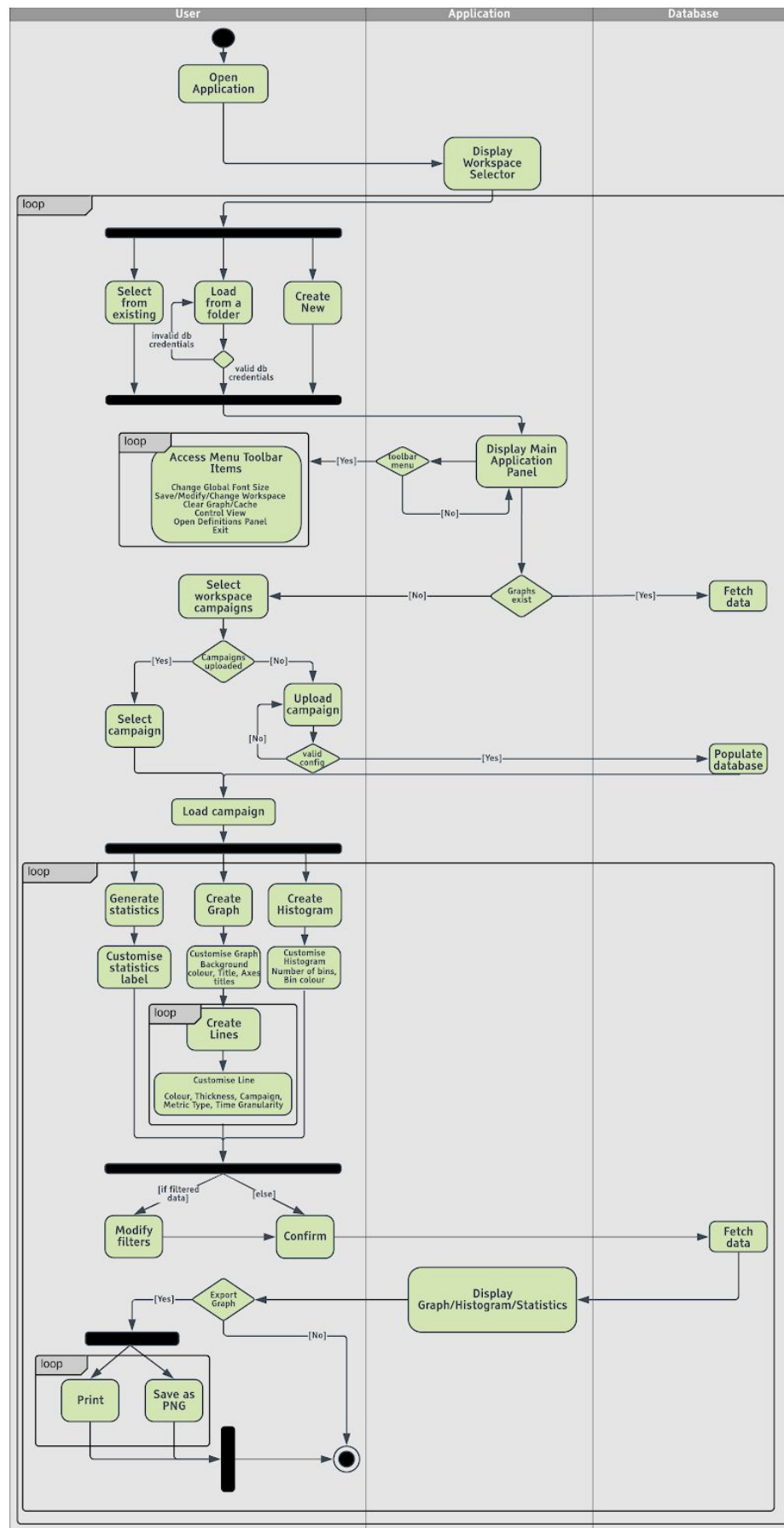
Use Case Diagram (Figure 1)

Below is the complete use case diagram based on the tasks stated in the Sprint plan for all three increments of the project. The functionality which has been added to the system during this Sprint is highlighted in green. The main focus of this last Sprint was to deliver a fully functioning system to the client that allowed them not only to view the data (as chart, histogram or statistic) but also to compare different audience segments and campaigns. We also added a *workspace* functionality, which allows the user to better manage the graphs they have created.



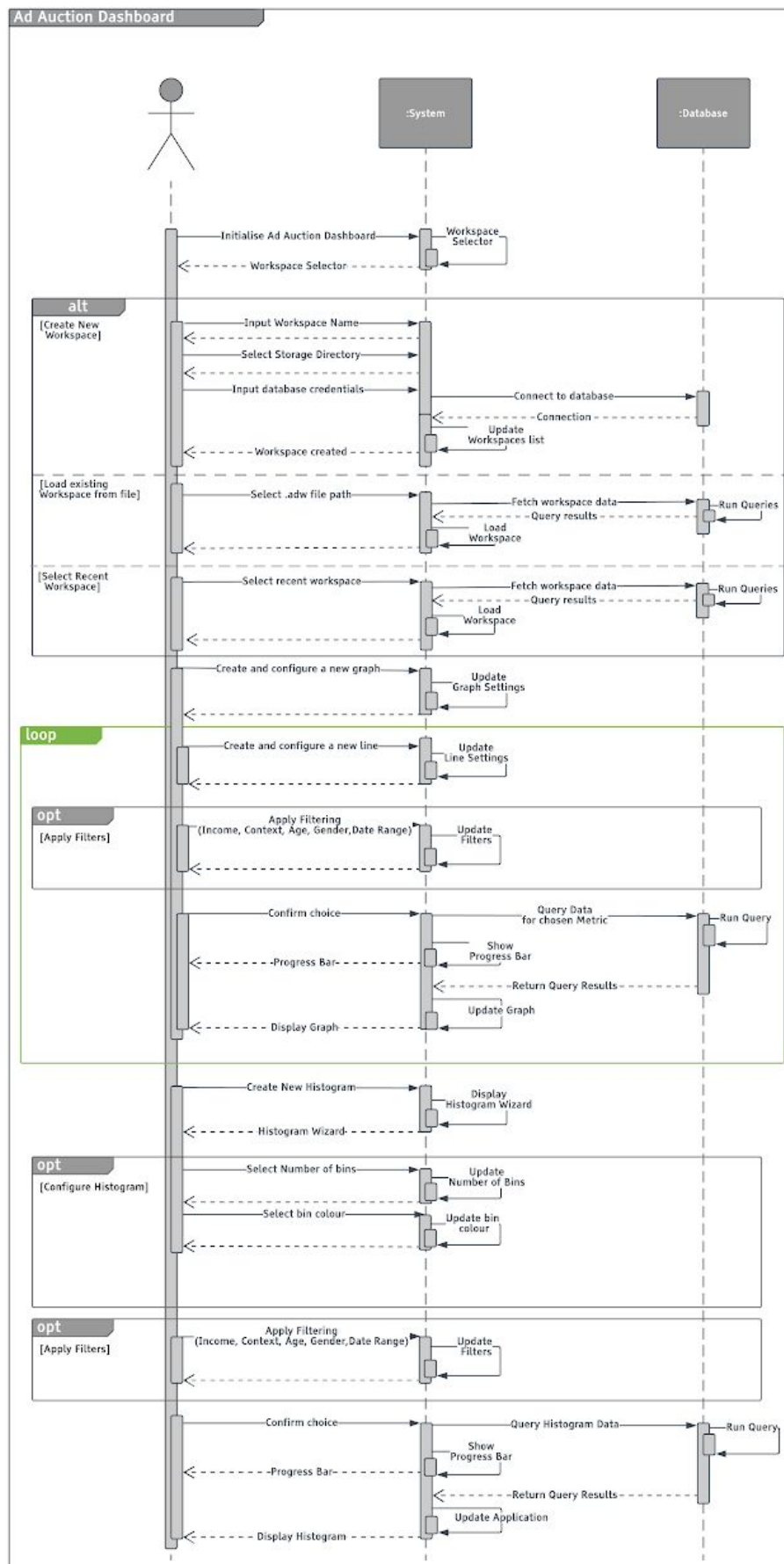
Activity Diagram (Figure 2)

Below is an activity diagram based on the functionality we promised to deliver in the Envisioning stage. The diagram itself outlines the action flows a user might follow while using the application. The loops describes (set of) actions that can be repeated multiple times, before moving on to the next action.



Sequence Diagram (Figure 3)

Below is the sequence diagram which covers the user-system-database interactions for the system functionality added during the third and final sprint. The loop field in the diagram shows that a user is able to create multiple lines specifying which campaign they refer to which allows them to compare data for different campaigns. The user is also able to configure different filters for each line and thus compare different audience segments within one or more campaigns.



Model Abstraction

Throughout the development of this application we have been strictly obeying the MVC model. The below diagram showcases (at a highly abstracted level) how this is achieved. Some key points are:

- The controller holds a private instance of 'DashboardMVC'; this holds object references to the view, model and controller. To avoid violation of the MVC model this is passed to sub-handlers (sub-controllers), but is not accessible to the model or view. This allows all sub-handlers to manipulate all objects in the view and model, and also call other sub-handlers.
- The model aspects of the application include the 'DashboardModel' and the PostgreSQL database. These have no functionality, instead just holding data (*the database handles querying but this is can be abstracted away*).
- View objects (such as the HistogramView, LineGraphView and StatisticView) must register themselves with their respective sub-handlers to avoid tight coupling.
- The visitor pattern has been used where appropriate to allow histograms and line graphs be handled correctly and safely even when passing around the superclass 'GraphConfig'.
- The workspace handler features a listening architecture so other handlers can react appropriately without tightly coupling functionality to the workspace handler.
- The view aspects of the application include the 'DashboardView'. There is a large amount of abstraction here as classes like the 'DashboardFrame' use a large number of subclasses to handle layout and functionality but these have been hidden for simplicity.
- Line direction indicates access between two classes or data flow between a class and a storage medium.
- New entities in this sprint are highlighted in green (though many other entities may have been modified also).

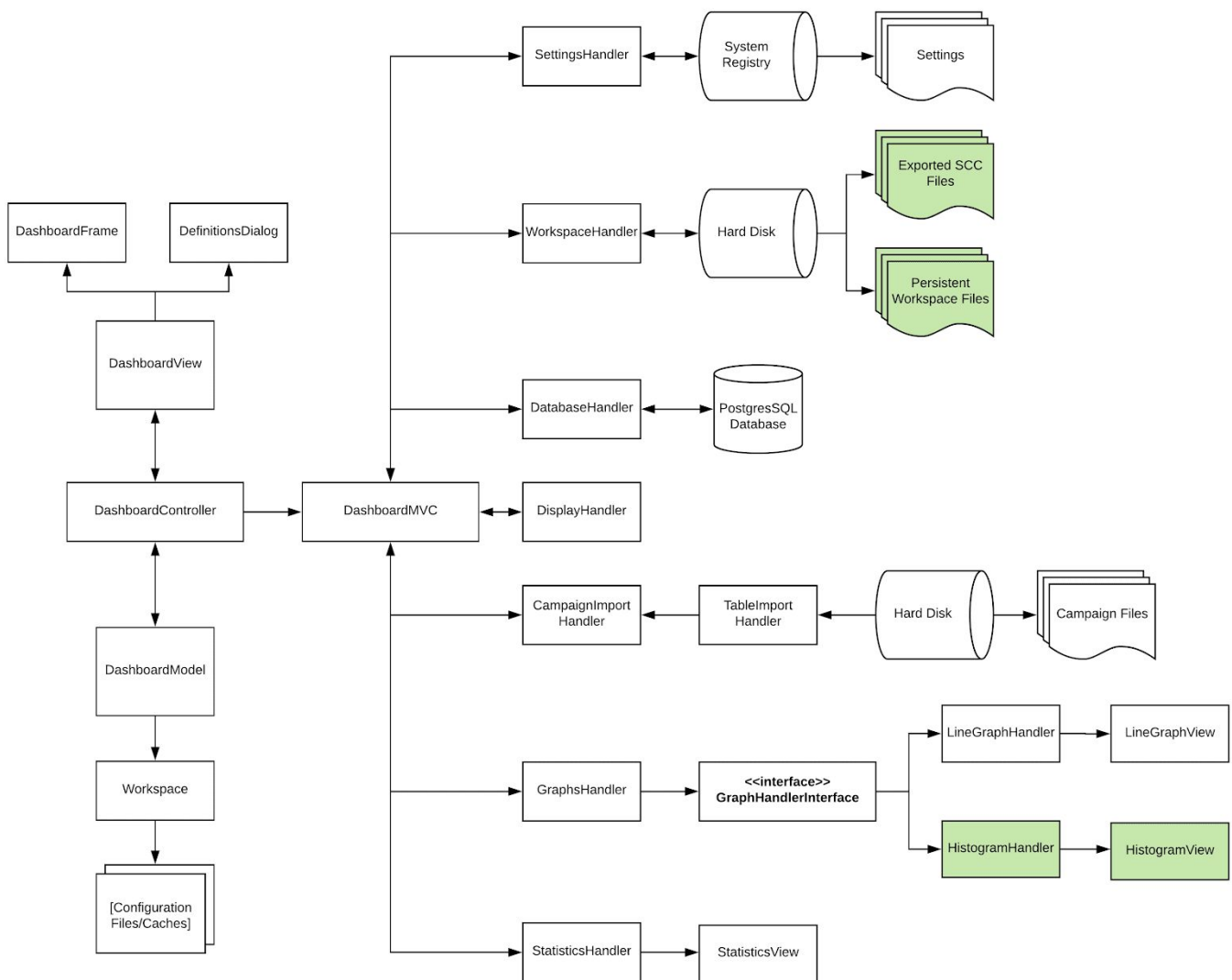


Figure 4 : Model Abstraction

1.2 - Key Design Choices

Wireframes

This section covers wireframing of major functionality additions or changes since increment 2. It contains the latest iterations of wireframes (the last created); these closely match the final application but are not necessarily direct matchups due to minor design decisions made whilst doing the implementation. If a major design decision was needed when creating the implementation, the wireframes were updated first.

Workspace Selector

The 'Workspace Selector' is now the startup window, replacing the main 'Dashboard'. It allows the user to select which workspace they would like to load into the 'Dashboard'. It features three selection modes, 'Recent', 'Open' and 'New'. The 'Recent' mode displays any workspaces that were previously loaded into the campaign (provided they still exist) as paths; this is to inform the user exactly which workspace they are opening. The 'Open' mode allows the user to locate an exact workspace on their computer, this may be helpful if they have moved the desired workspace's storage file or have not yet opened it on their current computer. The 'New' mode allows the user to create a new workspace. The workspace name and file path (directory) inform the save location of the 'adw' workspace save file. The user is given the option of loading a server configuration file or manually entering connection information. The former for general users who are provided access by a system administrator and the latter for more advanced users or system administrators.

The open state of the window depends on if there are any recent campaigns. If there is a recent campaign then 'Recent' is shown, otherwise 'New' is shown.

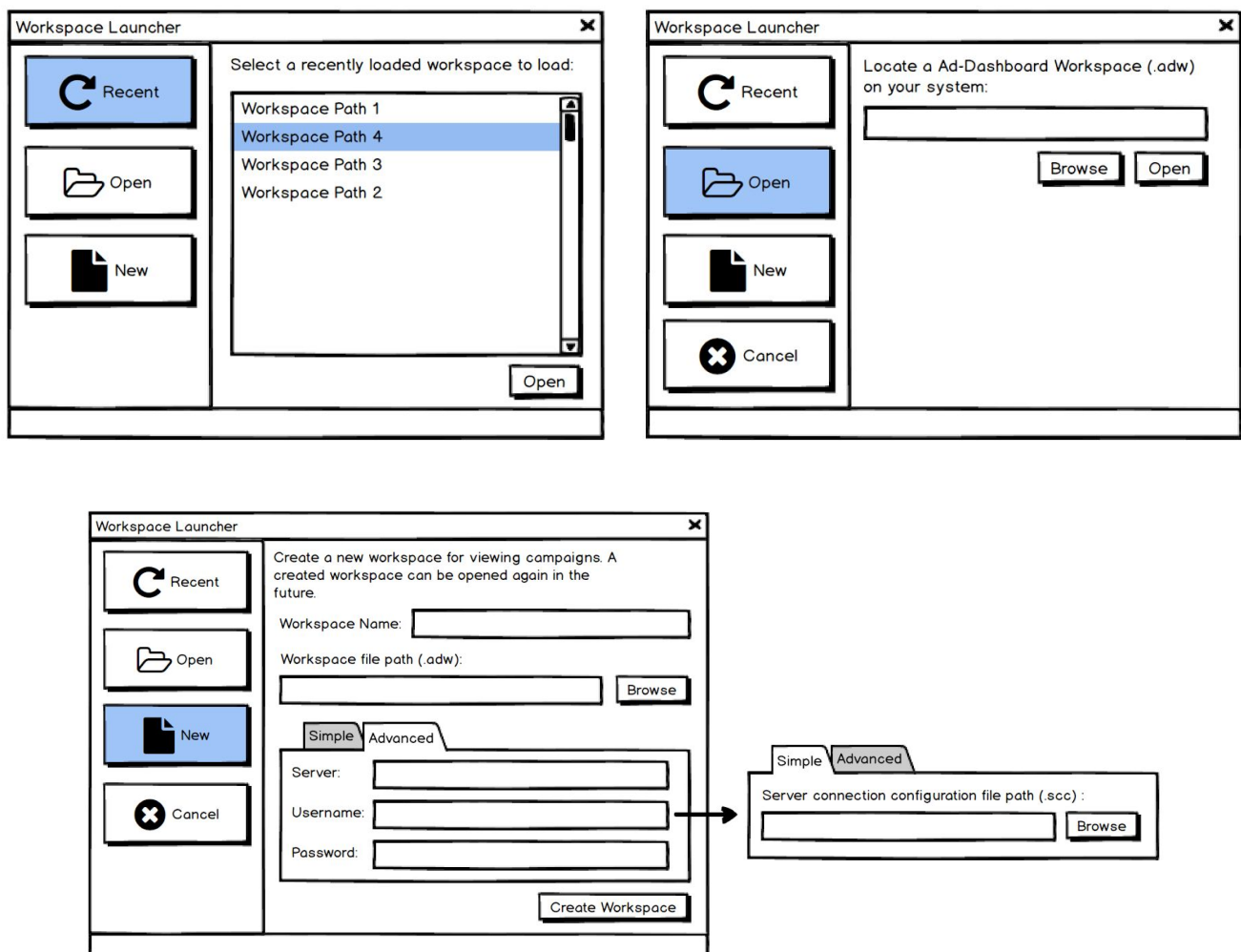


Figure 5 : Workspace Selector

Updated Campaign Selector/Manager

The campaign manager has been modified so that multiple campaigns can be loaded at any one time. It uses the consistent manager layout used throughout the application (also seen in the 'Statistic Manager' and 'Graph Manager'); this keeps the interface familiar to the user. Campaigns can either be added as an extra campaign or they can replace an existing campaign. When a campaign is added to the campaign manager it is available in data fetching interfaces (such as those in Figure ???). Replacing removes the selected campaign and replaces it with a newly selected one, it ensures every reference to the replaced campaign is also replaced so is equivalent in functionality to the 'Change Campaign'. If removal of a campaign occurs when it is referenced, the references are removed, this may invalidate some graphs/statistics.

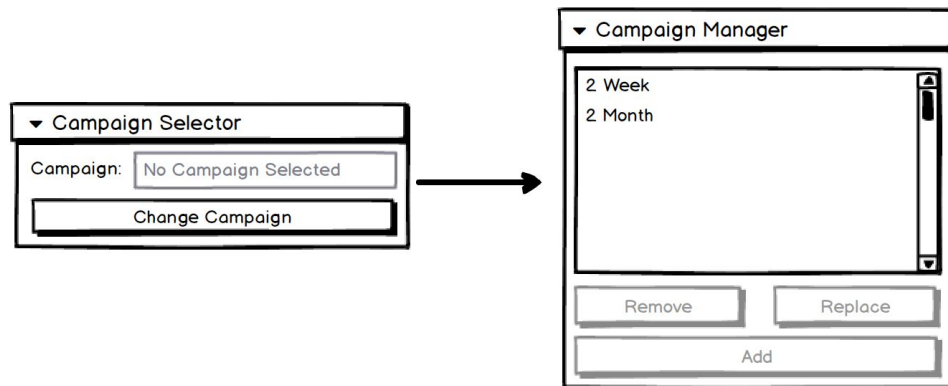


Figure 6 : New Campaign Manager

Histogram Wizard

Alongside the graph wizard, a new histogram wizard has been created. This clearly splits out the different graph functionality in this sprint. However, even though a different dialog is used, functionality and appearance is kept very similar through the use of the same modular components, i.e. the general panel and the data panel. The general panel has however been updated (in line with the line graph wizard) to autofill values so the user is less overwhelmed with options they need to fill. The data panel has also been updated to support campaign selection.

The 'Bins' panel has been added to allow multiple methods of configuration of the histogram's bins. Simple mode has been provided for beginner users or when individual weight of bins does not matter - when selected it creates the indicated number of bins of equal size. Advanced mode allows individual bins to have a weight set (the bin size being the $total\ range * weight / total\ weight$). Weights do not need to add up to anything in particular because they are normalised before being used - the normalised percentages are reflected next to the user weight input. Due to the complexity of advanced mode, reset buttons are provided so the user can return to a prior state: 'Reset Bins' loads the last valid set of applied bins, 'Use Defaults' resets to the global default.

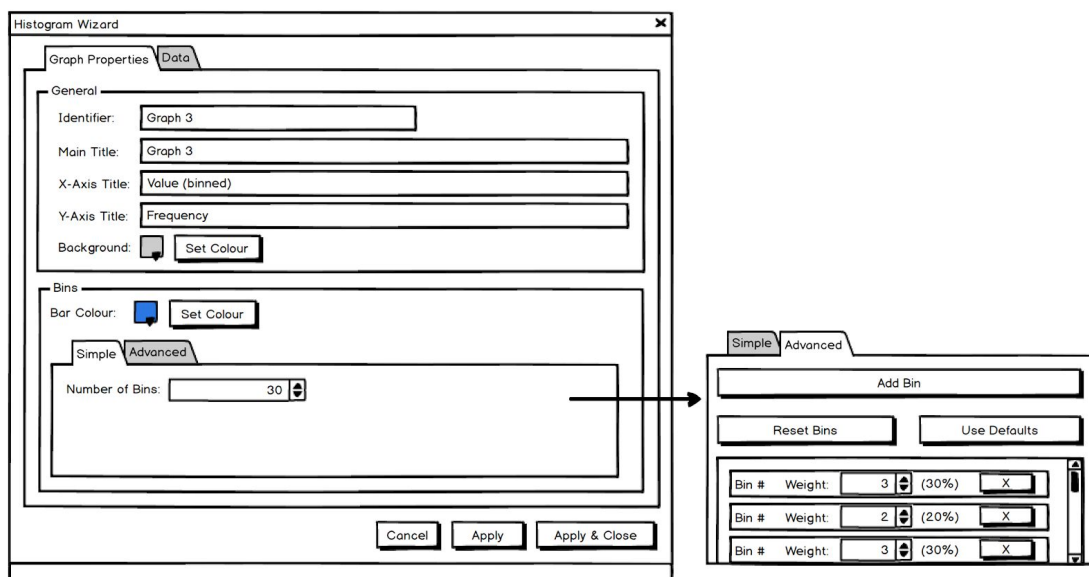


Figure 7 : Histogram Wizard [Graph Properties]

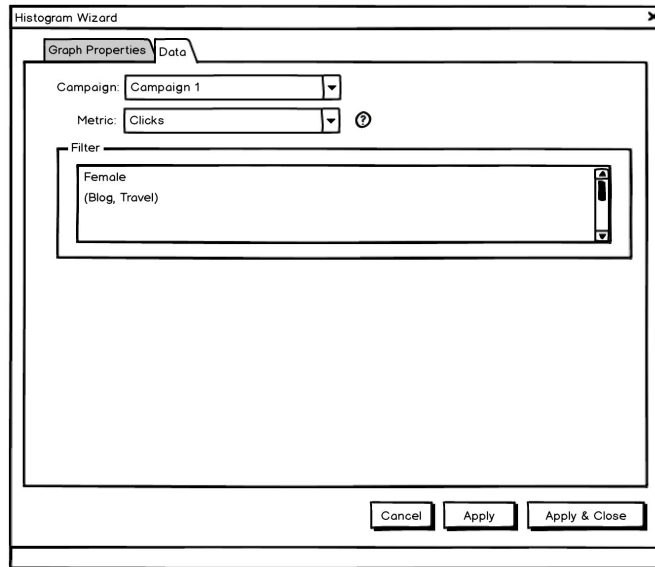


Figure 8 : Histogram Wizard [Data]

Menu Bar Modifications

To facilitate for new functionality new options were added to the 'File' and 'Help' menu bar items. A new 'Tools' menu bar was also added.

The 'File' menu bar item handles all access to workspace related functionality, this includes: changing the current workspace using Figure ???, modifying the current workspace's properties using a subset of Figure ??? and saving the current workspace. Saving functionality may also be triggered by a user action such as attempting to close a workspace that contains modifications.

The 'Tools' menu bar item holds specialised functionality that can aid a user but would otherwise clutter the workspace interface. Simple user tools such as clearing the graph are here. However, it also contains more advanced functionality, which may be for system administrators, under an 'Advanced' sub menu. This includes the 'Export Server Connection Configuration (SCC)' item, which allows exporting of the current workspace connection properties as an encrypted file; this can be loaded into another workspace using 'Simple mode seen in Figure ???.

Access to the PDF user manual is also now provided through the 'Help' menu bar item.

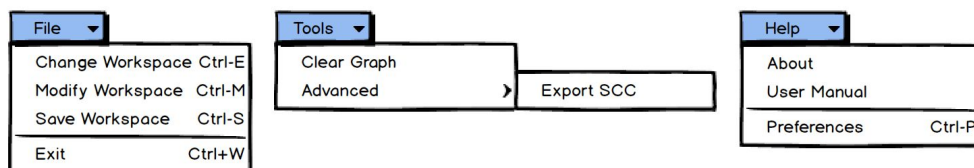


Figure 9 : Menu Bar Items [File/Tools/Help]

Storyboarding

Creating a Histogram

Below is a storyboard showing how a user would go about creating a histogram for an already imported campaign. To create a histogram the user would expand the graph manager and select the “New Graph” button. When doing so a popup will be displayed that will ask the user if they would like to create a line graph or a histogram. Upon selecting the histogram option the “Line Graph Wizard” will be displayed to the user allowing them to edit the graph and bin properties. Once the user has confirmed their selections the histogram is shown on the screen.

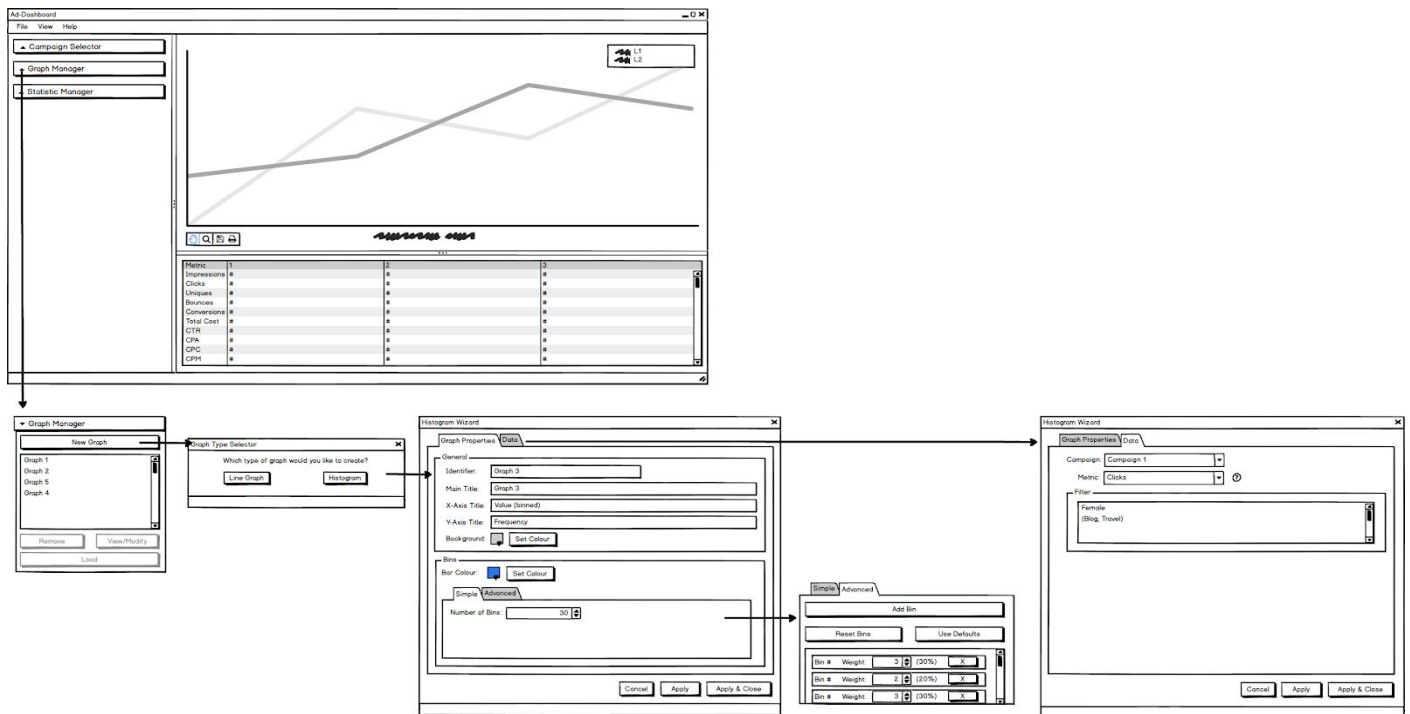


Figure 10 : Creation of a histogram

Creating a workspace

To create a new Workspace, the user should select workspace name and path to storage directory and should provide database configuration file through the *Simple* or *Advanced* mode. Once the user has clicked on the “Create Workspace” button, the application automatically loads it. From here the user can add campaigns to the workspace via the “Campaign Manager” and create graphs for them via the “Graph Manager”

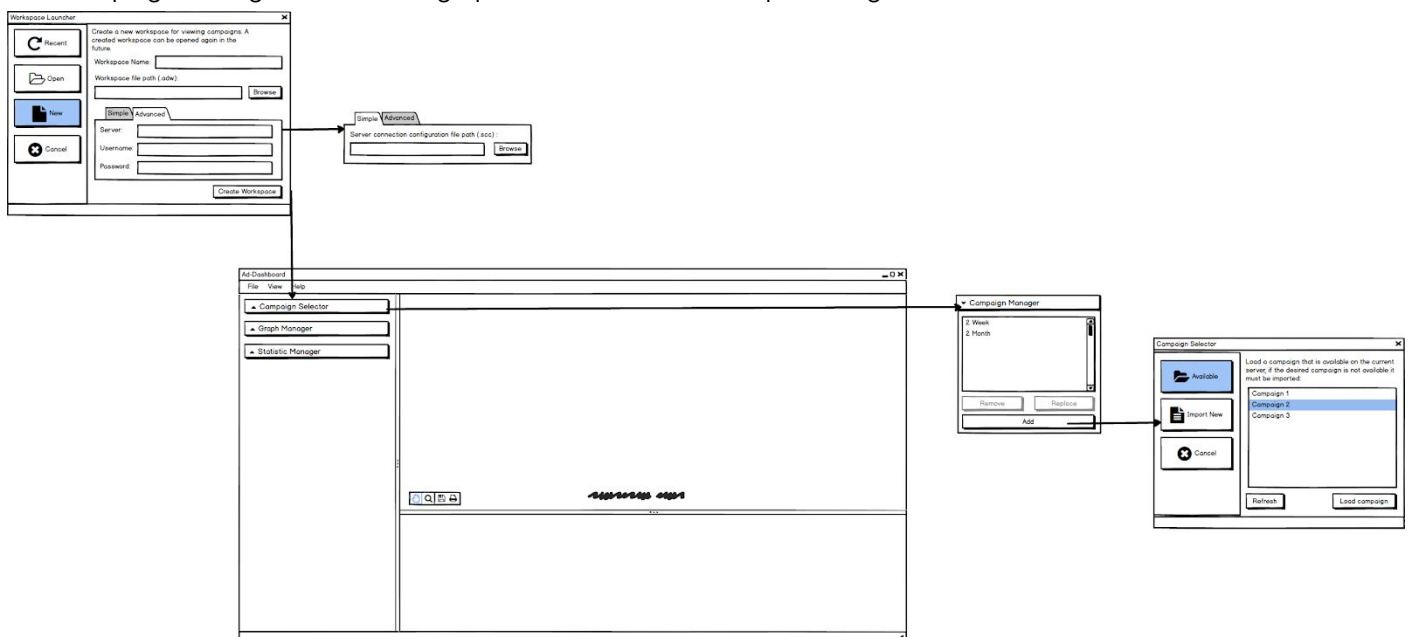


Figure 11 : Creating a workspace

2 - Scenarios and Testing

Section Summary

This first part of this section contains **Scenarios** which describe how the user would work with functionality added to the system in this final Sprint. All Scenarios cover one or several use cases from the Use Case Diagram and follow a sequence of activities from the Sequence Diagram. At the bottom of each scenario there is an explanation of the **test(s)** performed against it. The second part of this section explains all other types of testing we performed while working on developing the system. These include **Unit Tests, Integration Tests, Regression Tests, Acceptance Criteria Tests, Boundary and Partition Tests** and **Validation and Quality Control Tests**.

2.1 - Scenarios and key test outputs against them



Peter Quinn

Specific day statistics/Specific time of day statistics Scenario 1

Peter wants to get a better idea about when his add attracts more unique users.

He opens the application and is displayed with the Workspace Selector panel.

Peter has several workspaces created, so he looks through the list and finds the one he needs.

He clicks on the “Open” button.

Peter can now see the main panel of the application.

He expands the “Campaign Manager” and clicks on the “Add Campaign” button to load the campaign he needs.

On the list of campaigns he selects “New campaign 2018” and clicks the “Load Campaign” button.

Peter can now proceed to creating a graph.

Upon clicking “New Graph” button, he gets to choose between creating a Line Graph and a Histogram.

He clicks on the “Line Graph” button and is displayed with the Line Graph Wizard

He selects “Daily Statistics” as a graph identifier and configures the chart and axes titles.

After selecting “Overlaid Time Period” on the graph properties panel, he moves on to create the line configurations.

He clicks on the “Lines” tab to create the lines he wants to see on the graph.

He clicks on the “New Line” button.

He sets the identifier of the first line to be “Statistics - 1 January” and selects the metric type to “Number of Uniques” and the interval to “Hour”.

Peter wants to see data only for 1 January, so he has to apply a date range filter.

He clicks on the “Modify Filter” button.

He sets the Start Date to 1 January 2018 00:00:00 and the end date to 2 January 2018 00:00:00.

Peter clicks on the “New Line” button again and configures the settings for the new line.

However, this time, he sets the Start Date to 1 January 2018 10:00:00 and the end date to 1 January 2018 16:00:00 to only see unique users between 10am and 4pm.

He clicks on the “Apply & Close” button.

Peter forgot to put a graph identifier, so he receives an error message, saying that the graph cannot be created.

He clicks on the “OK” button and puts “Daily Statistics” as a graph identifier.

Now, he clicks on the “Apply & Close” button again and the graph is displayed on the screen.
Peter is happy with the results and he closes the application.

Corresponding User Stories:

As a <Marketing Agency Client> I want to <view charts of the performance of advertising campaigns **during a particular time of the day**> So that <I can detect when web users respond better to them> (**PbID 28**)

As a <Marketing Agency Client> I want to <view charts of the performance of advertising campaigns **at a particular day**> So that <I can detect when web users respond better to them> (**PbID 29**)

Test against scenario:

Type of test: Manual testing

Preconditions:

- Workspace exists and has valid database credentials
- Campaign data is uploaded in the application
- GUI is updated to support workspaces
- Queries are updated to support day and time selection

Actions:

- Load application
- Select desired workspace
- Select a campaign from already uploaded campaigns
- Add a new graph with the configurations described above
- Exit application

Application opens successfully.

As expected, the initial panel is Workspace Selector with a list of already created workspaces.

Selected workspace opens correctly.

Submenus opened and loaded correctly.

All panels opened correctly.

Filters are successfully configured.

Error message is shown when a graph identifier is missing.

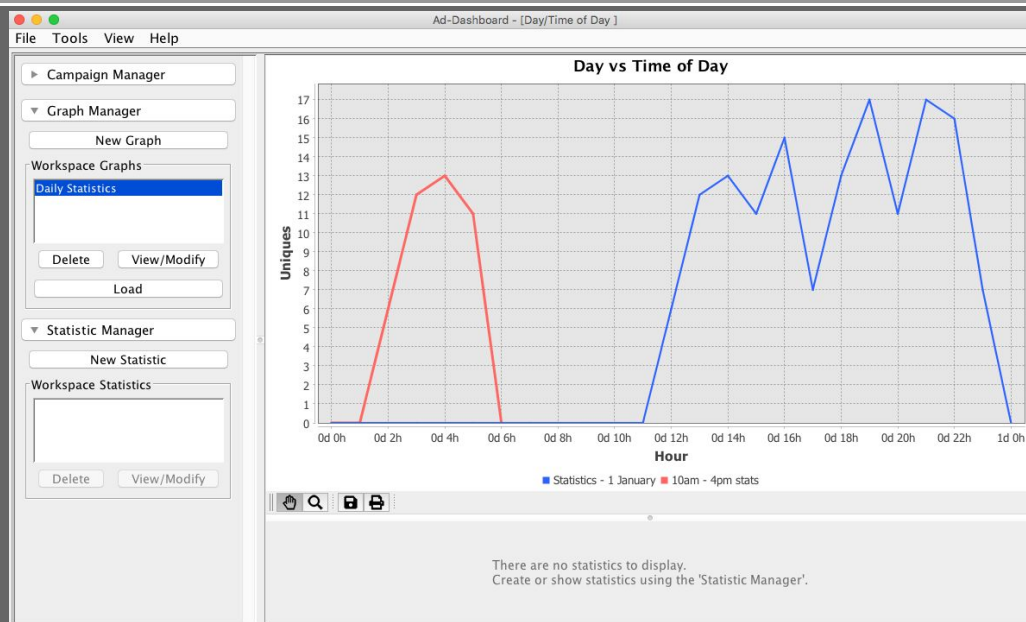
Both lines are displayed on the graph.

Data is in correct format.

All buttons work as expected.

No big delays in response time.

Test screenshot:



Click Cost Histogram Scenario 2

Peter has recently launched a new campaign and wants to see the click cost distribution on a histogram.

He launches the Ad Auction Dashboard Application.

Peter had created some graphs for the same campaign before, so he has an already existing workspace.

The campaign is already loaded into the Workspace, so Peter doesn't have to configure anything else.

He proceeds to create a new graph by clicking on the “New Graph” button.

Peter can now see the Graph Type Selector and he clicks on the “Histogram” button.

He can now see the “Histogram Wizard” on the screen.

He sets the Histogram Identifier to “Click Cost Distribution” and leaves the default titles and bin count.

Peter swaps to the “Data” tab, because he wants to only see click cost for users with “High” income. He clicks on “Modify Filter” and unticks “Low” and “Medium”. Peter clicks on “Apply & Close”. The histogram is updated.

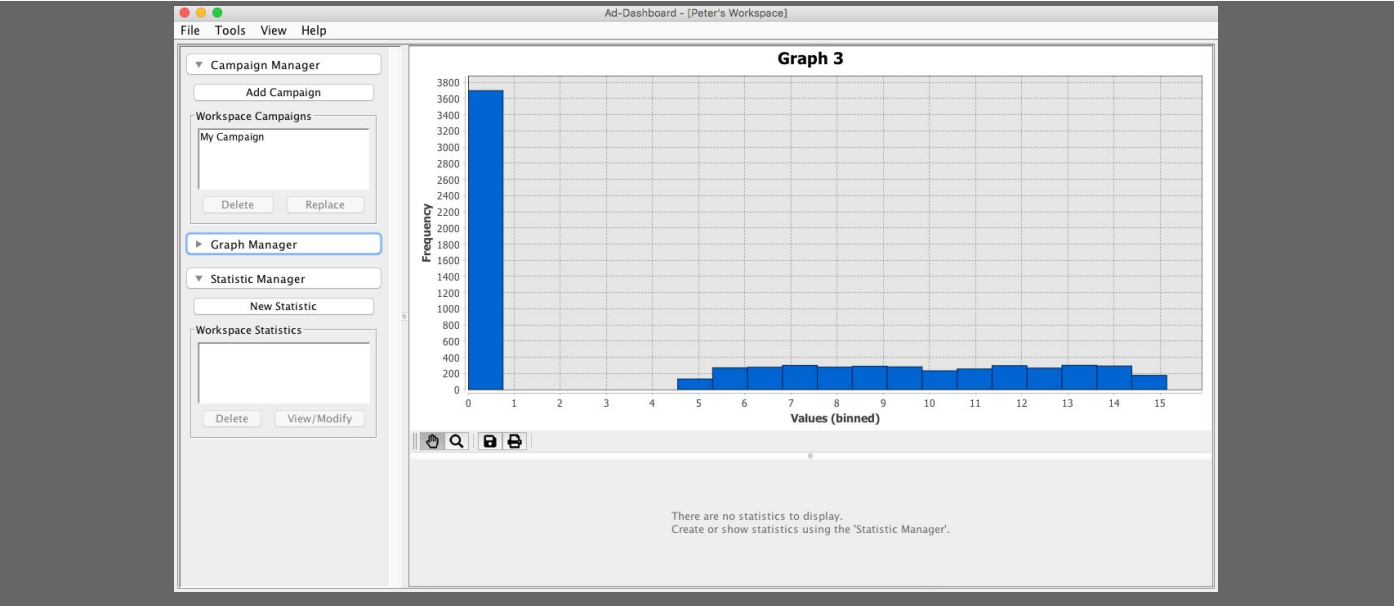
Corresponding User Story:

As a <Marketing Agency Client> I want to <see a **histogram of the click costs**> So that <I know how click cost is distributed> **(PbID 18)**

Test against scenario:

Type of test: Manual testing	Application opens successfully.
Preconditions: <ul style="list-style-type: none">Workspace exists and has valid database credentialsCampaign data is uploaded in the applicationQueries are updated to support click cost histogramGUI is updated to support click cost histogram	As expected, the initial panel is Workspace Selector with a list of already created workspaces. Selected workspace opens correctly. Submenus opened and loaded correctly. Histogram Wizard works as expected. Income filter is successfully configured. Histogram is successfully generated.
Actions: <ul style="list-style-type: none">Load applicationSelect desired workspaceAdd a new histogramModify income filterExit application	Data is in correct format. All buttons work as expected. No big delays in response time.

Test screenshot:



Audience Segments Comparison
Scenario 3

Peter is curious to know how many people with “Low” income click on his ads more often than people with “High” or “Medium” income. He clicks on the Ad Auction Dashboard application on his desktop and is displayed with the Workspace Selector panel. In the Workspaces list Peter looks for the “Audience Segments” workspace. He selects it and clicks on the “Open” button. Peter expands the “Campaign Manager” menu and clicks on the “Add Campaign” button. He selects the “Technology” campaign from the list of available campaigns and clicks on the “Load Campaign” button. He expands the “Graph Manager” menu and clicks on the “New Graph” button. Peter then selects “Line Graph” and can now see the “Line Graph Wizard” on his screen.

He puts “Income Comparison” as a graph identifier and clicks on the “Lines” tab to create the desired lines.

He clicks on the “New Line” button to create his first line.

Peter puts “Low Income” as an identifier and selects “Number of Clicks” as a Metric Type.

He selects “Day” from the “Interval” dropdown.

Peter clicks on the “Modify filter” button and deselects “Medium” and “High” in the Income filter field.

He clicks on the “New Line” button again to create his second line.

Peter puts “Medium Income” as an identifier and selects “Number of Clicks” as a Metric Type.

He selects “Day” from the “Interval” dropdown.

Peter clicks on the “Modify filter” button and deselects “Low” and “High” in the Income filter field.

He does the same for the “High Income” line, but deselects “Low” and “Medium” in the Income filter.

He clicks “Apply & Close” button and is displayed with a progress bar.

Within a few seconds the graph is updated.

Now he can compare the statistics between people with different incomes.

He gets a comprehensive view about the different audience segments

Corresponding User Story:

As a <Marketing Agency Client> I want to <compare metrics between **different audience segments**> So that <I can highlight the most engaged web users> (**PbID 19**)

Test against scenario:

Type of test: Manual testing

Preconditions:

- Workspace exists and has valid database credentials
- Campaign data is uploaded in the application
- Queries are updated to support filtering

Actions:

- Load application
- Select desired workspace
- Select a campaign from already uploaded campaigns
- Add a new graph with 3 lines each filtered by the 3 types of income
- Exit application

Application opens successfully.

As expected, the initial panel is Workspace Selector with a list of already created workspaces.

Selected workspace opens correctly.

Submenus opened and loaded correctly.

All panels opened correctly.

Income filter is successfully configured.

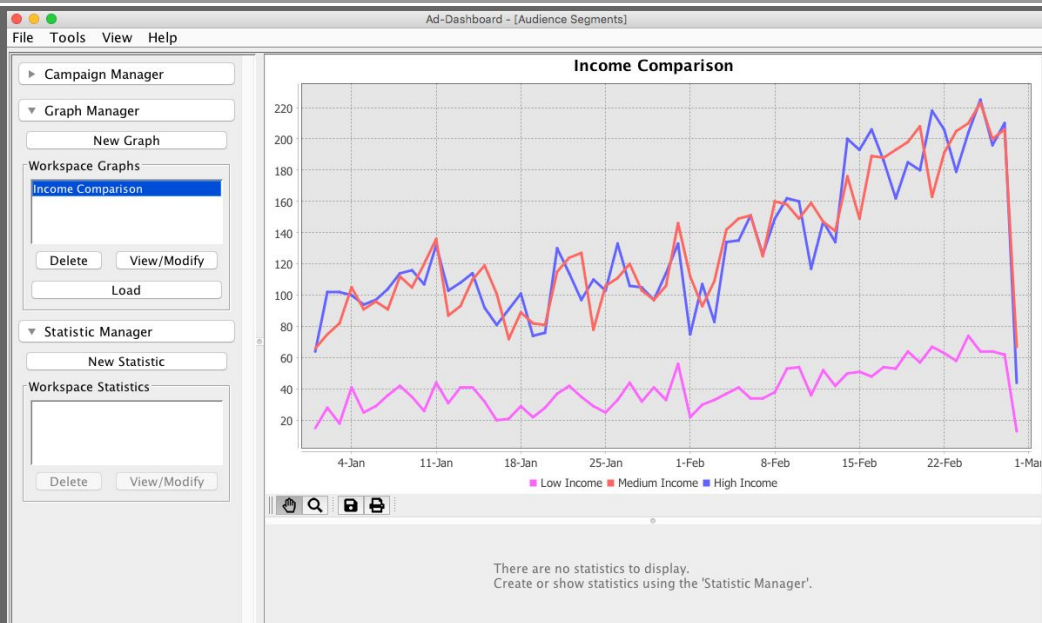
All three lines are displayed on the graph.

Data is in correct format.

All buttons work as expected.

No big delays in response time.

Test screenshot:





Jessica Pierce

Campaign Comparison Scenario 4

Jessica's manager created 2 kids' toys campaigns last month and wants to know which one had a lower bounce rate. To do this Jessica is going to use the Ad Auction Dashboard Application. She opens the application, and selects the "Toys Campaigns 2018" workspace. She expands the "Campaign Manager" menu and adds the 2 toy campaigns. Jessica clicks on the "New Graph" button and selects "Line Graph". She configures the graph properties and moves on to create 2 lines for the 2 campaigns. For the first line she selects "Toy Campaign 1" from the "Campaign" dropdown. She selects "Bounce Rate" as a Metric Type and "Day" as an Interval. Jessica then changes the bounce definition to "Pages Viewed" and sets the value to 2. Similarly, she adds a second line, but this time choosing "Toy Campaign 2" from the "Campaign" dropdown. She sets the same metric type, interval and bounce definition. Jessica clicks on the "Apply & Close" button. She can see a progress bar and within a few seconds the graph is updated. Jessica decides she also wants to view statistics for the two campaigns. She expands the "Statistic Manager" menu and clicks on the "New Statistics" button. First Jessica selects "Toy Campaign 1" and configures the bounce definition as above. She clicks on the "Apply & Close" button, but gets an error for not providing Statistics Identifier. Jessica puts "TC 1" as an identifier and clicks on the button again. After less than 20 seconds she can see all the statistics below the graph. She repeats the same for "Toy Campaign 2", but this time she doesn't forget to put "TC 2" as an identifier. She is pleased with her findings about the difference between those two advertising campaigns and goes on to give the results to her manager.

Corresponding User Story:

As a <Marketing Agency Client> I want to <be able to **compare data of different campaigns**> So that < I have statistics of what type of ads are more appealing to web users > (PbID 27)

Test against scenario:

Type of test: Manual testing

Preconditions:

- Workspace exists and has valid database credentials
- Data for 2 campaigns is uploaded in the application
- Queries are updated to support campaign comparison
- GUI is updated to support campaign comparison

Actions:

- Load application
- Select desired workspace

Application opens successfully.

As expected, the initial panel is Workspace Selector with a list of already created workspaces.

Selected workspace opens correctly.

2 campaigns are successfully added to the workspace.

All panels opened correctly.

Lines for different campaigns are successfully added.

Bounce definition section appears upon "Bounce Rate" metric selection.

Both lines are displayed on the graph.

- Load 2 campaigns into the workspace
- Configure 2 lines for the 2 different campaigns
- Create statistics for the 2 different campaigns
- Exit application

Statistics for both campaigns are added and are available below the main graph.

Error is shown when Statistics Identifier is missing.

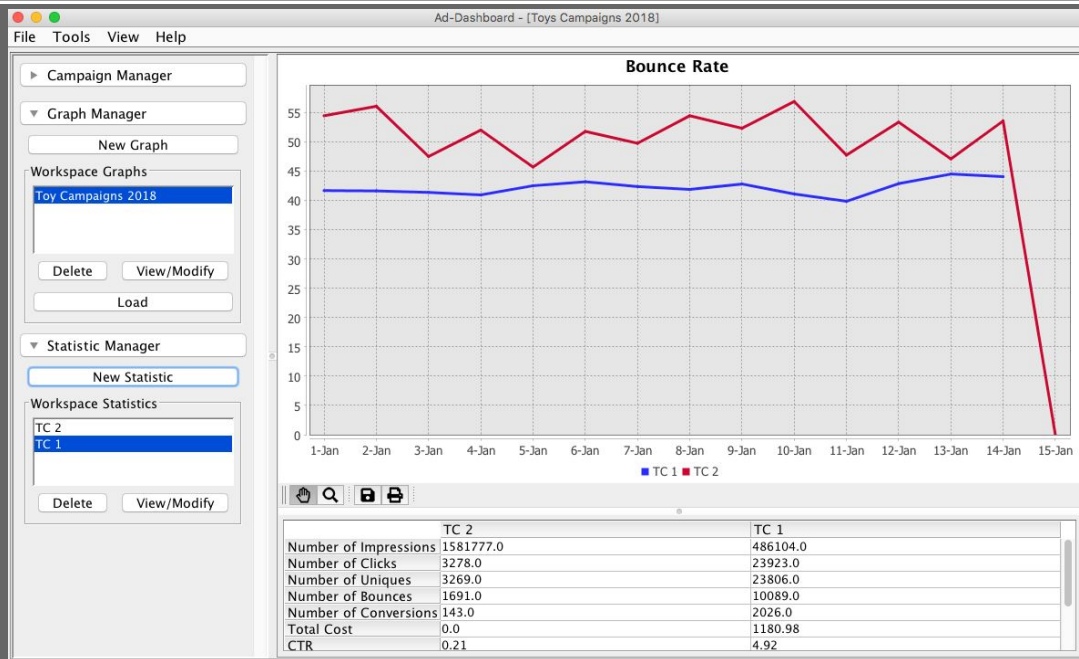
Statistics are successfully generated when identifier is added.

Data is in correct format.

All buttons work as expected.

No big delays in response time.

Test screenshot:



Workspaces Scenario 5

Jessica works in a big company that manages a lot of campaigns at the same time.

To make it easier, when evaluating and comparing campaigns, Jessica has created several Workspaces, containing different graphs.

When she opens the application for the first time, she sees the “Workspace Selector” panel.

She has several options: to select a Workspace from a list of already existing workspaces, to import an Ad-Dashboard Workspace (.adw) file from her computer or to create a New Workspace.

Jessica decides to create a new workspace, so she clicks on the “New” button.

She inputs “Campaigns April - May 2018” in the “Workspace Name” field.

Jessica now has to specify a directory where her .adw files will be stored.

She clicks on the “Browse” button and navigates to the “Workspaces” folder in her computer.

She confirms her choice by clicking on the “Choose” button.

Jessica can choose “Simple” or “Advanced” mode to connect to the database.

She clicks on the “Advanced” tab and inputs her Server, Username and Password.

She clicks on the “Create New” button.

The application automatically loads the new workspace.

Jessica realised she had put the wrong Workspace name and she has to change it.

She clicks on “File” in the toolbar menu and selects “Modify Workspace”.

She can now see the “Workspace Modifier” panel.

Jessica changes the name of the workspace and clicks on the “Apply Changes” button.

She then proceeds to add already existing campaigns to the workspace and create several graphs.

When she is done she tries to exit the application without saving the workspace changes.

Jessica sees a prompt asking her if she wants to save the workspace before exiting.
She clicks on the “Yes” button and the application is closed successfully.
Every time she opens the application, she has the option to choose any of the previously created workspaces.

Corresponding User Story:

As a <Small Marketing Agency Owner> I want <my client to be able to **change workspaces**> So that <the application is more flexible and easy to use> (**UsID 40**)

Test against scenario:

Type of test: Manual testing

Preconditions:

- GUI is updated to support campaign comparison
- Workspace storage method is created

Actions:

- Load application
- Create a workspace with relevant configuration
- Modify a workspace
- Add campaigns to workspace
- Create Graphs for added campaigns
- Try to exit without saving workspace
- Save workspace
- Exit application

Application opens successfully.

As expected, the initial panel is Workspace Selector with a list of already created workspaces.

Workspace created successfully.

Database credentials are correct and connection to the database server is established.

Campaigns are added to the workspace.

Graphs are generated correctly.

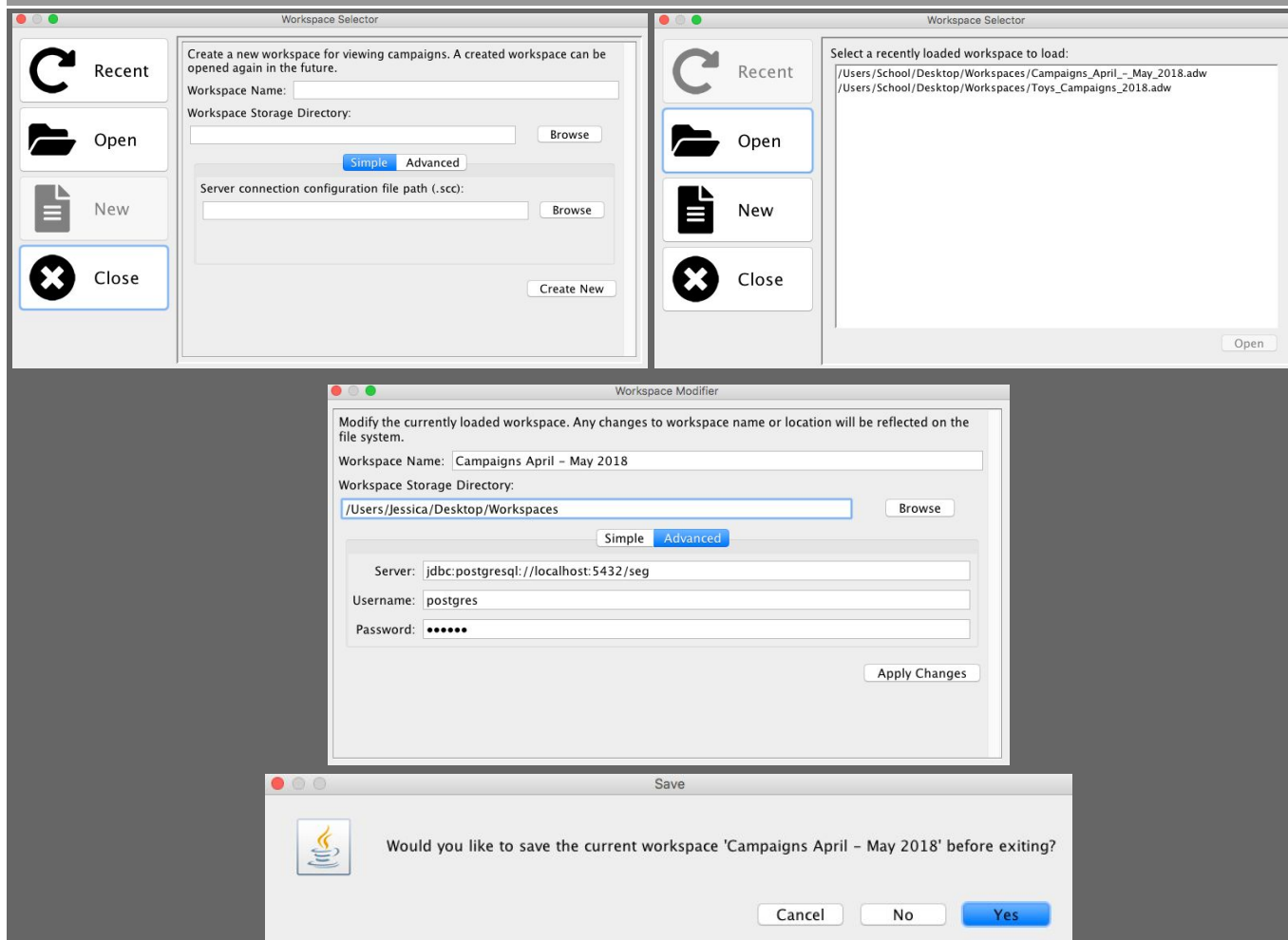
Warning received when trying to exit without saving workspace.

Application automatically closed after choosing if workspace should be saved.

The workspace is now available in the Workspaces List and all the graphs and statistics that were previously generated are available to view and/or modify.

No big delays in response time.

Test screenshot:



2.2 - Testing


Unit Testing

To automatically run the tests use the 'mvn test' command from the root directory of the repository.

Preconditions	Unit Tests	Pass/Fail
The config.properties includes user Postgres credentials.	Database Tests <ul style="list-style-type: none"> Tests are run automatically through TravisCI every time code is pushed to any branch in the repository This set of tests check if the database is connected properly and if the tables are created correctly. In addition the grouping tests checks if an invalid time interval has been added. 	<div> <div> DatabaseTest219ms </div> <div> <div>connectionArgsTest49ms</div> <div>clickCreateTest2ms</div> <div>groupingTest93ms</div> <div>configTest0ms</div> <div>clearTableTest74ms</div> <div>impressionCreateTest1ms</div> <div>serverCreateTest0ms</div> </div> </div>
Integration between TravisCI and Postgres is set up. Example dataset that of the correct format is available. The test.config file includes user Postgres credentials (DB_HOST, DB_USER, DB_PASSWORD) Queries are available in the <i>DatabaseQueryFactory</i> Class.	Query Correctness Tests/ Filter Tests/ Compare Segments Tests <ul style="list-style-type: none"> Tests are run automatically through TravisCI every time code is pushed to any branch in the repository Before tests are run, a test database is populated with the example dataset. Query results are compared with precalculated statistics from the smaller dataset. After the tests, all tables are dropped to avoid test failure in case of corrupted data The QueryCorrectnessTest class also utilises integration testing, as the tests there check how the The tests run both statistic and graph queries on a small dataset. Some of the tests check for boundary behaviour - e.g. setting the end date before the start date, date range not existing in the dataset, or non-existent campaign_id attribute in the logs 	<div> <div> QueryCorrectnessTest357ms </div> <div> <div>uniquesTest18ms</div> <div>notNegativeTest274ms</div> <div>clicksTest3ms</div> <div>totalCostTest6ms</div> <div>impressionsTest5ms</div> <div>dateRangeTest7ms</div> <div>cpaTest6ms</div> <div>cpcTest8ms</div> <div>cpmTest5ms</div> <div>conversionTest5ms</div> <div>ctrTest5ms</div> <div>checkDatabaseHandler0ms</div> <div>bounceRateTest11ms</div> <div>bouncesTest4ms</div> </div> </div> <div> <div> FilterTest115ms </div> <div> <div>ageTest45ms</div> <div>incomeTest19ms</div> <div>genderTest3ms</div> <div>emptyFilterTest2ms</div> <div>contextTest37ms</div> <div>startDateAfterEndDateTest7ms</div> <div>dateRangeNotExistingTest2ms</div> </div> </div> <div> <div> CompareTest54ms </div> <div> <div>compareGender18ms</div> <div>compareIncome14ms</div> <div>compareAges6ms</div> <div>nonExistentCampaign2ms</div> <div>compareContext14ms</div> </div> </div>
	Serialisation Tests <ul style="list-style-type: none"> Testing different password configurations 	<div> <div> SerializationTest160ms </div> <div> <div>plaintextSerialisationTest32ms</div> <div>encryptedSerialisationTest124ms</div> <div>wrongPasswordTest4ms</div> </div> </div>

Travis CI

All unit tests are run through TravisCI in order to detect problems in the code as early as possible.

IlianaHadzhiatanasova / SEG  build passing

Current Branches Build History Pull Requests More options

Default Branch

✓ master 68 builds	# 460 passed a day ago	c502fe2 Iliana Hadzhiatanasova	✓	✓	✓	✓	✓
-----------------------	---------------------------	-----------------------------------	---	---	---	---	---

Active Branches

✓ javadoc-and-readme 2 builds	# 459 passed a day ago	e9f8dc5 Iliana Hadzhiatanasova	✓	✓			
✓ testing 2 builds	# 427 passed 4 days ago	5ffb985 Iliana Hadzhiatanasova	✓	✓			

Inactive Branches

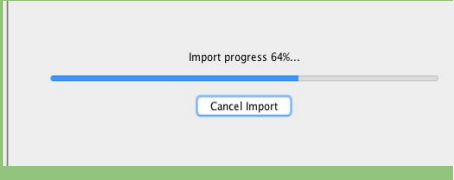
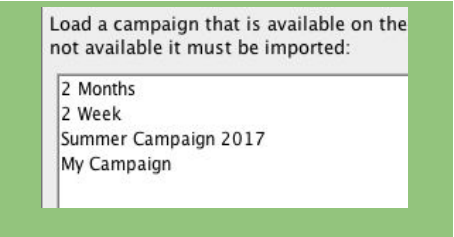
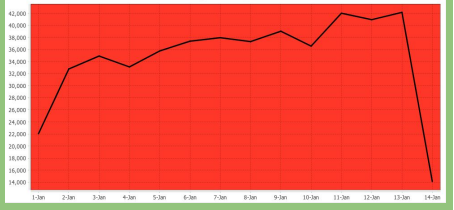
✓ caching 13 builds	# 456 passed 3 days ago	9630ab7 David Jones	✓	✓	✓	✓	!
------------------------	----------------------------	------------------------	---	---	---	---	---

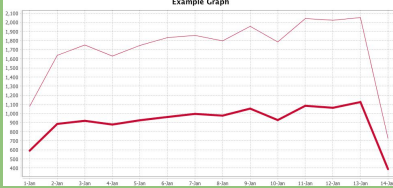


Integration Testing

Test Case ID	Test	Test Description/Result	Pass/Fail
1	Integration between font size changing and the application	Manual tests were performed to detect if there are defects when setting font size. Actions: set preferred font size, check if it has been applied globally Results: font changes affect the whole application - all panels and submenus	Pass
2	Integration between tooltip disabling and the application	Manual tests were performed to detect if there are defects when disabling tooltips Actions: disable tooltips, check if it has been applied globally Results: tooltips are disabled everywhere in the application	Pass
3	Integration between keyboard shortcuts and application	Manual tests were performed to detect if there are defects when using keyboard shortcuts on different OSs Actions: test all keyboard shortcuts on Linux, Windows and MacOS Results: all shortcuts work as expected and open the correct panels	Pass
4	Importing data into database via files	Manual and Unit tests were performed to detect if database is correctly populated given three logs	Pass

		Actions: upload three logs, open pgAdmin and check existing tables in database; create a small dataset, upload dataset, check if all rows are imported Results: both manual and unit tests passed - data is imported in the database	
5	Integration between GUI, graph, database	Manual tests + Unit tests were performed to determine correct integration between GUI, graph and database Actions: set preferred font size, check if it has been applied globally Results: font changes affect the whole application - all panels and submenus	Pass

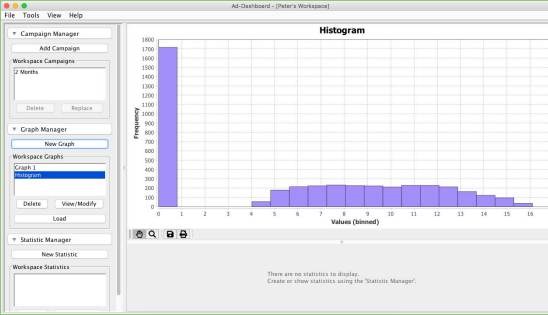
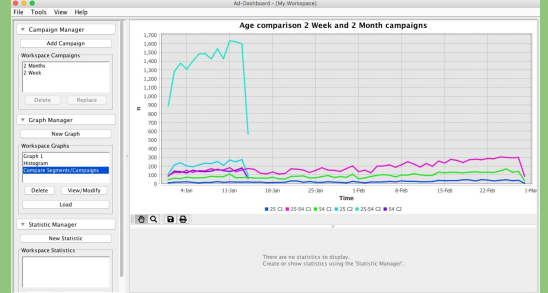
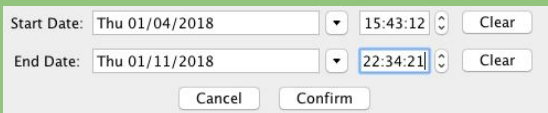
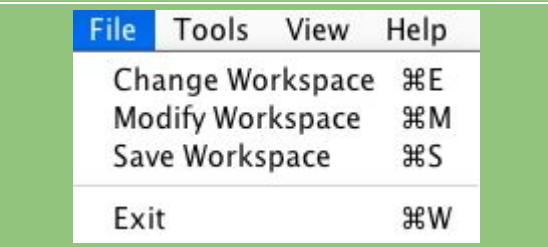

Regression Testing

Old functionality	Testing	Pass/Fail
Sprint 1 functionality		
Import campaign	Campaign is imported correctly through both “Advanced” and “Simple” Mode. Database is populated. The new campaign table is updated and all logs include a campaign_id attribute. The new functionality of the application has not affected campaign importing.	Pass
Global change of font size	Font size is configurable by the user. The new UI components are affected by the font change.	Pass
Generate graph	The graph generation options are extended - more metrics, graph and line customisation are available, but this does not affect previous functionality. Graphs are generated correctly.	Pass
Generate statistic	Statistics for a campaign can be queried and generated. (See Sprint 2 functionality - statistic for all metrics)	Pass
Import progress	The user can see import progress (in %). Once the campaign is imported the user gets a confirmation message.	
Sprint 2 functionality		
Load Campaign from a list of existing campaigns	A list of already uploaded campaigns is available within the “Available” tab in the Campaign Selector panel. The introduction of a workspace does not affect loading a campaign, however, once loaded, the campaign is only loaded within a particular workspace, rather than along the whole application. (which is the desired behaviour)	
Configure chart background colours	None of the new functionality affected the chart background colour setting.	

Configure line colour and thickness	Line colour and thickness are still configurable and the new application features did not affect the functionality.																									
The user can see statistics for all metrics	<p>Statistics for different campaigns are calculated and displayed in the table under the graph.</p> <p>When the campaign is changed, all the statistics are recalculated to match with the data of the new campaign.</p> <p>The results of the queries are tested on a smaller dataset for correctness.</p> <p>Some of the queries are tested using <i>awk</i>.</p> <p>In Sprint 3 data format is updated, which did not affect the correctness of the statistics.</p>	<table><thead><tr><th colspan="2">All Statistics</th></tr></thead><tbody><tr><td>Number of Impressions</td><td>8828248.0</td></tr><tr><td>Number of Clicks</td><td>17754.0</td></tr><tr><td>Number of Uniques</td><td>17708.0</td></tr><tr><td>Number of Bounces</td><td>310.0</td></tr><tr><td>Number of Conversions</td><td>684.0</td></tr><tr><td>Total Cost</td><td>1135.46</td></tr><tr><td>CTR</td><td>0.2</td></tr><tr><td>CPA</td><td>1.66</td></tr><tr><td>CPC</td><td>0.06</td></tr><tr><td>CPM</td><td>0.01</td></tr><tr><td>Bounce Rate</td><td>1.75</td></tr></tbody></table>	All Statistics		Number of Impressions	8828248.0	Number of Clicks	17754.0	Number of Uniques	17708.0	Number of Bounces	310.0	Number of Conversions	684.0	Total Cost	1135.46	CTR	0.2	CPA	1.66	CPC	0.06	CPM	0.01	Bounce Rate	1.75
All Statistics																										
Number of Impressions	8828248.0																									
Number of Clicks	17754.0																									
Number of Uniques	17708.0																									
Number of Bounces	310.0																									
Number of Conversions	684.0																									
Total Cost	1135.46																									
CTR	0.2																									
CPA	1.66																									
CPC	0.06																									
CPM	0.01																									
Bounce Rate	1.75																									
Configure Bounce Definition	<p>The user can select bounce configuration per line.</p> <p>Query results are updated according to the selected definition and value.</p>	<div><p>Bounce Definition</p><p><input checked="" type="radio"/> Time (/s): <input type="text" value="3"/></p><p><input type="radio"/> Page Count: <input type="text" value="0"/></p></div>																								
Filter data (Age, Gender, Income, Context, Date Range)	<p>Initially all data is fetched from the database. Every time a user creates a graph or a new statistic, they have the option to apply filters.</p> <p>Filter configurations are given as checkbox items to avoid invalid user inputs.</p> <p>All queries had to be updated to support specific day (and time of day) statistics. This improved the date range filter and did not affect the rest of the filters.</p>	<div><p>Filter Modifier</p><p>Age</p><ul style="list-style-type: none"><input checked="" type="checkbox"/> <25<input checked="" type="checkbox"/> 25-34<input checked="" type="checkbox"/> 35-44<input checked="" type="checkbox"/> 45-54<input checked="" type="checkbox"/> > 54<p>Gender</p><ul style="list-style-type: none"><input checked="" type="checkbox"/> Male<input checked="" type="checkbox"/> Female<p>Income</p><ul style="list-style-type: none"><input checked="" type="checkbox"/> Low<input checked="" type="checkbox"/> Medium<input checked="" type="checkbox"/> High<p>Context</p><ul style="list-style-type: none"><input checked="" type="checkbox"/> Shopping<input checked="" type="checkbox"/> Blog<input checked="" type="checkbox"/> Social Media<input checked="" type="checkbox"/> News<input checked="" type="checkbox"/> Travel<input checked="" type="checkbox"/> Hobbies<p>Start Date: <input type="text"/> 00:00:00 <input type="button" value="Clear"/></p><p>End Date: <input type="text"/> 00:00:00 <input type="button" value="Clear"/></p><p><input type="button" value="Cancel"/> <input type="button" value="Confirm"/></p></div>																								
Export graph (save/print)	Generated graph can be saved as .PNG in a destination folder chosen by the user and can be printed if there is a printing service available.	<div><div></div><div>Export the currently displayed graph as an image.</div><div></div><div>Print the currently displayed graph.</div></div>																								

Acceptance criteria testing

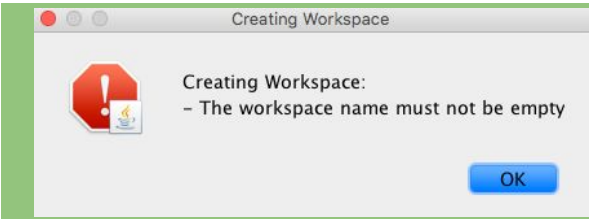
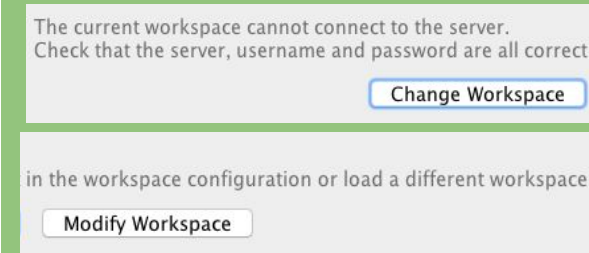
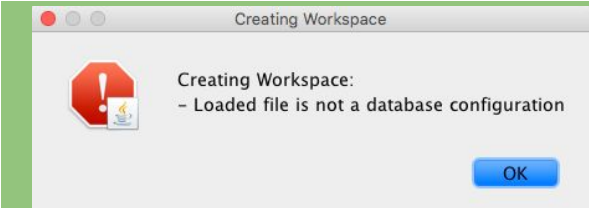

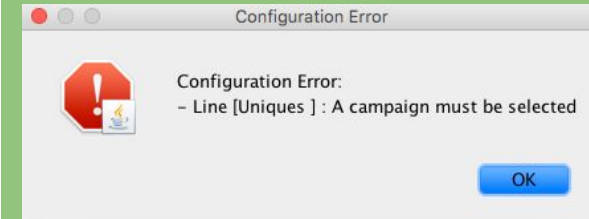
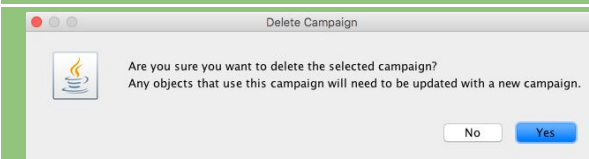
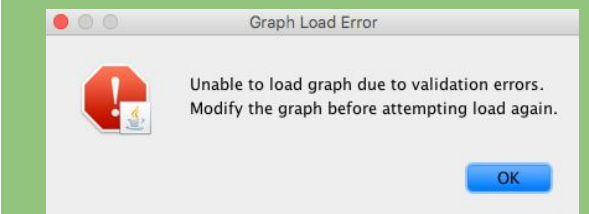
As in this sprint we are delivering a fully developed system, we ran acceptance criteria tests for all user stories that we included in our Envisioning document. In addition we made sure we have complied with the additional non-functional requirements of the system such as fast response and large dataset support.

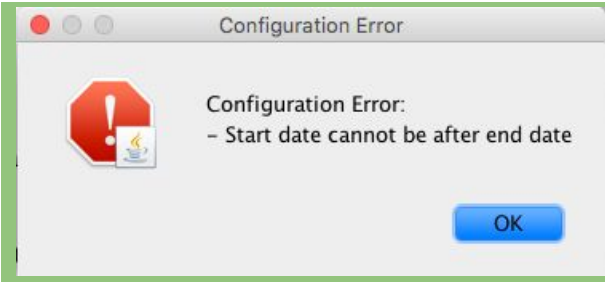
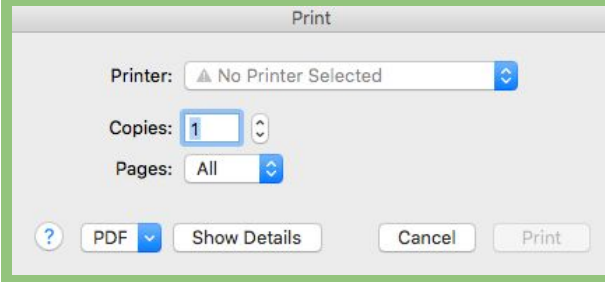
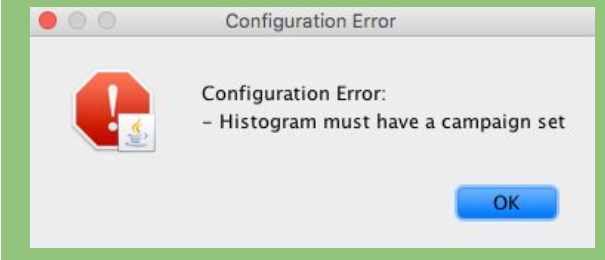
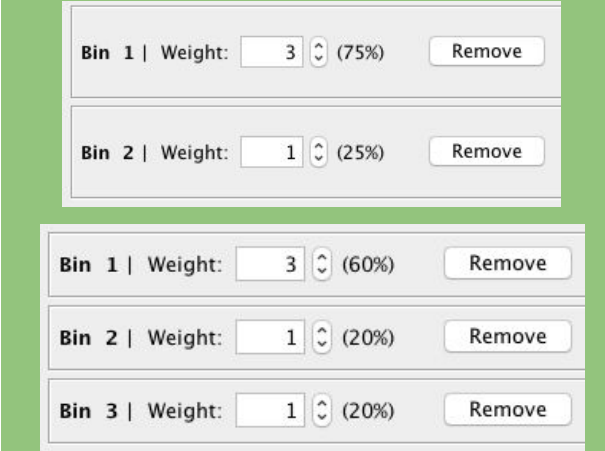
PbID	Criteria	Notes	Pass/Fail
18	The user can see a click cost histogram	<p>An overview of click cost distribution is available through creating a new histogram.</p> <p>The data can be filtered by date range, age, gender, income and/or context.</p>	
19	The user is able to compare different audience segments	Each line on the graph can have its own filters and campaign it corresponds to. This enables the user to have multiple lines with different configurations on one graph.	
27	The user is able to compare two or more campaigns		
29	The user is able to view time of day statistics	Filters allow for setting a 24 hour span, within which a key metric will be evaluated. Additionally, a span of only several hours can be set.	
30	The user is able to view specific day statistics	For invalid inputs, See Boundary Testing	
40	The user can change workspaces	Workspaces allow the user to have persistent "storage" for graphs and statistics. These can be changed/modified at any point in time.	
New	Tooltips Disabling	<p>Tooltips can be globally disabled to allow more experienced users of the system work comfortably.</p> <p>This setting can be toggled.</p>	
User Stories from Sprints 1 & 2			
1	User friendly dashboard	The system is designed mainly for trained users, but is extremely flexible and allows the user to configure numerous.	Pass
2	Generate a chart	Charts for all key metrics can be generated.	Pass
3	Gather data from 3 logs	A campaign is uploaded only via the 3 log files. If one or more files are	Pass

		missing an error is displayed (See Boundary Testing - Sprint 2)	
4	Select time granularity	Time granularity can be selected via the “Modify filter” option when creating graph/histogram or statistic. Appropriate error messages are displayed for incorrect inputs.	Pass
5-14	Statistics for all key metrics	Statistics can be generated for all metrics via the “Statistic Manager” panel.	Pass (See Regression Testing for Screenshots)
20,21	Configurable bounce definition	Bounce definition is configurable for “Bounce Rate” and “Number of Bounces” metrics. In addition to bounce definition, the user is able to set a desired bounce value.	Pass (See Regression Testing for Screenshots)
22-26	Filtering (date range, age, context, gender, income)	Filtering is available via the “Modify filter” option when creating a graph/histogram or statistic. Appropriate error messages are shown for empty filters.	Pass (See Regression Testing for Screenshots)
28	Font size changing	Font sizing is configurable by the user and can be changed more than once. When setting the font size the user is able to see a preview of the text.	Pass (See Integration Testing)
31	Save graph	A user is able to save a graph as a PNG file, or print it via a printing device. In case no printer is available, the graph can be exported as a PDF.	Pass (See Regression Testing for Screenshots)
34	Print graph		
NFRs			
35	The system responds within a reasonable time	Response time is tested along with scenario testing. The system responds within reasonable time, given the size of the dataset. As expected, queries for large datasets are slightly slower, however there is no major delays.	Pass
36	The system supports large datasets	The application supports large datasets. A slight delay in response time is noticed for more complex queries for large datasets, however the workflow is still smooth.	Pass
41	The system supports chart background and line colour configuration	Colour changing is an accessibility that is supported by the application to enhance user experience for customers, who might have visual impairments.	Pass (See Regression Testing for Screenshots)

Boundary and Partition Testing

In order to ensure that the application is consistent and a wide range of cases is covered, apart from performing the boundary tests from **Sprint 2**, we performed some additional tests, mainly focused on the new functionality of the system. (Please refer to Increment 2 Documentation for a full list of previous boundary tests)

Requirement	Testing	Result
Workspace		
Workspace Name should be more than 1 characters in length	Error message is displayed to the user if no workspace name is selected.	
Valid Server, Username and Password credentials are provided when creating a workspace in "Advanced" mode	Error message is displayed to the user if any of the fields have incorrect input.	
Valid server connection configuration file path is provided when creating a workspace in "Simple" mode	Error message is displayed to the user if a wrong path to a server connection configuration (.scc) file is given.	
Campaign		
Campaign must be selected to add a line	<p>This Sprint introduced the comparing campaigns feature. Therefore, when adding a new line the user must specify which campaign the line refers to.</p> <p>A list of available campaigns is provided via a dropdown menu to avoid invalid campaign name inputs.</p> <p>If the workspace contains at least one campaign the Campaign field is automatically populated.</p> <p>An error message is shown to the user if campaign is not specified (there are no campaigns in the workspace).</p>	 
Deleting a campaign with existing statistics/graphs	<p>A warning message is displayed when a user tries to delete a campaign, for which graphs/statistics exist.</p> <p>If the user decides to delete the campaign, an error message is shown, reminding the user that if they want to see graphs with the same configuration, they will have to change the Campaign field within the Lines tab in the Line Graph Wizard.</p>	 

Graphs		
Filtering by specific time of the day	An error message is displayed to the user if the end hour is before the start hour of the hour interval. A statistic is not generated until the filter is correctly configured.	 A screenshot of a 'Configuration Error' dialog box. It features a red octagonal warning icon with a white exclamation mark and a small document icon. The text reads: 'Configuration Error: - Start date cannot be after end date'. There is an 'OK' button at the bottom right.
Printing feature	If no printer device is available, the user is able to save the file as a PDF	 A screenshot of a 'Print' dialog box. It has a 'Printer:' dropdown menu showing 'No Printer Selected'. Below it are 'Copies:' (set to 1) and 'Pages:' (set to 'All') fields. At the bottom, there are buttons for '?', 'PDF', 'Show Details', 'Cancel', and 'Print'.
Histograms		
Histogram Campaign	An error message is shown if there is no campaign set, when creating a histogram. A histogram is not generated until campaign is selected.	 A screenshot of a 'Configuration Error' dialog box. It features a red octagonal warning icon with a white exclamation mark and a small document icon. The text reads: 'Configuration Error: - Histogram must have a campaign set'. There is an 'OK' button at the bottom right.
Bin count and size	Bin count and weight (in %) are configurable by the user. To avoid incorrect inputs when the weight of one bin is increased/decreased, the rest of the weight is distributed among the rest of the bins. If a new bin is added the weight is adjusted according to the custom weights.	 A screenshot of the histogram configuration interface. It shows two sections. The top section has 'Bin 1 Weight: 3 (75%)' and 'Bin 2 Weight: 1 (25%)', each with a 'Remove' button. The bottom section has 'Bin 1 Weight: 3 (60%)', 'Bin 2 Weight: 1 (20%)', and 'Bin 3 Weight: 1 (20%)', each with a 'Remove' button. The weights are shown in a dropdown menu.

Validation & Defect Testing (Software Quality Control)

To ensure that the application is stable and bug-free, we made sure to track the software quality through the following steps:

- When a new feature is added:
 - Regression Tests are performed to ensure old functionality is still stable
 - Boundary Tests are performed to detect unusual behaviour of the system
 - In case of a bug, code is edited and retested
- When old functionality is modified:
 - Validation tests are performed to ensure that the improved functionality doesn't affect any part of the system
 - In case of a bug, code is edited and retested

3 - Responses to Increment 2 Feedback

Section Summary

This section describes the actions we took after receiving feedback in the marking meeting for **Sprint 2**. Following the recommendations of our client, we have now included a **Section Summary** that outlines main points in the section, **Boundary Unit Tests** that run often in order to catch bugs on time, and lastly we have made the **connections between** our **Storyboards** more clear.

3.1 - Personalised Feedback

Boundary Unit Testing

We have included boundary unit tests as part of our handin on the request of the client. This allowed us to detect unusual behaviour in the system early. Boundary unit tests explanation and screenshots are available in **Section 2.2**.

Summary Sections

In our meeting the client suggested that we have a small paragraph describing main points mentioned in each section of our documentation. This would allow the client to get a quick read through of the document and understand project progress and development in less detail. This would be beneficial for the small amount of time we have to conduct a meeting. We followed this feedback and a summary section is included for each part of this report.

Disabling Tooltips

In one of our meetings, the client raised the concern that despite being useful for a new user of the system, for a more advanced user, the tooltips might only be getting in the way of an otherwise smooth workflow. Therefore, the application now supports tooltip disabling.

Text fields automatically populated

Another request from the client was to automatically populate text fields requiring user input, which would make the process of creating graphs, histograms and statistics quicker.

Connection between storyboards

One of the concerns raised by our client was the lack of connections between the storyboards we had. For this reason, in **Section 1.2** we have included wireframes with all intermediate links between them. This would allow the client to have a better understanding of the application and the existing transitions between the main panels.

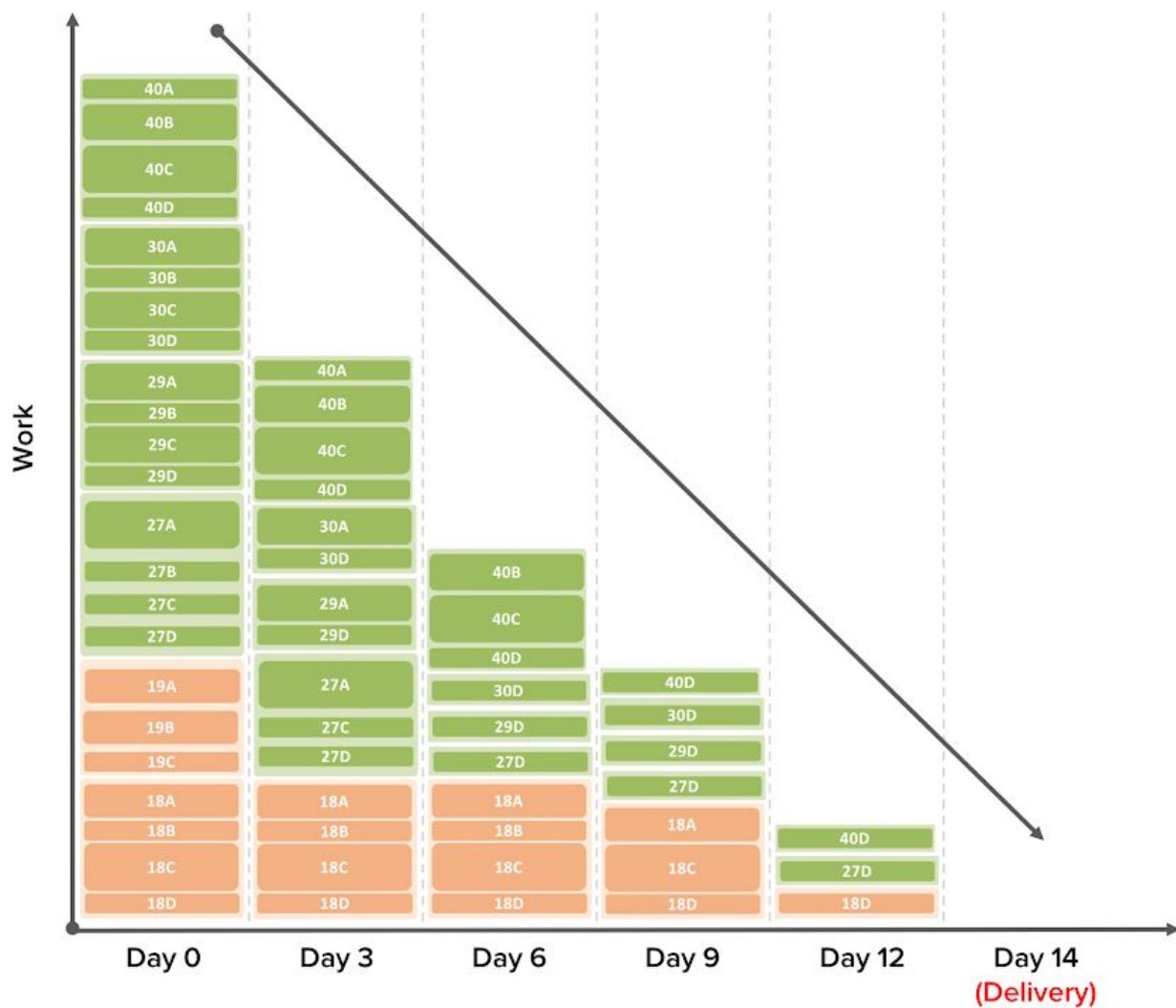
4 - Planning

Definition of Done

The **DoD** we have formulated for is as follows:

- All tasks from the Sprint are completed
- All unit tests pass
- Regression tests are performed
- Acceptance criteria for **all User Stories** is met
- Functional and Non-functional requirements are met
- Code is reviewed by team members
- Client is satisfied with the development of the project

4.1 - Sprint 3 Burndown Chart



5 - Conclusions

Section Summary

This final section summarises how our initial task estimations turned out to be and shows what tools we utilised to distribute tasks and track project development process.

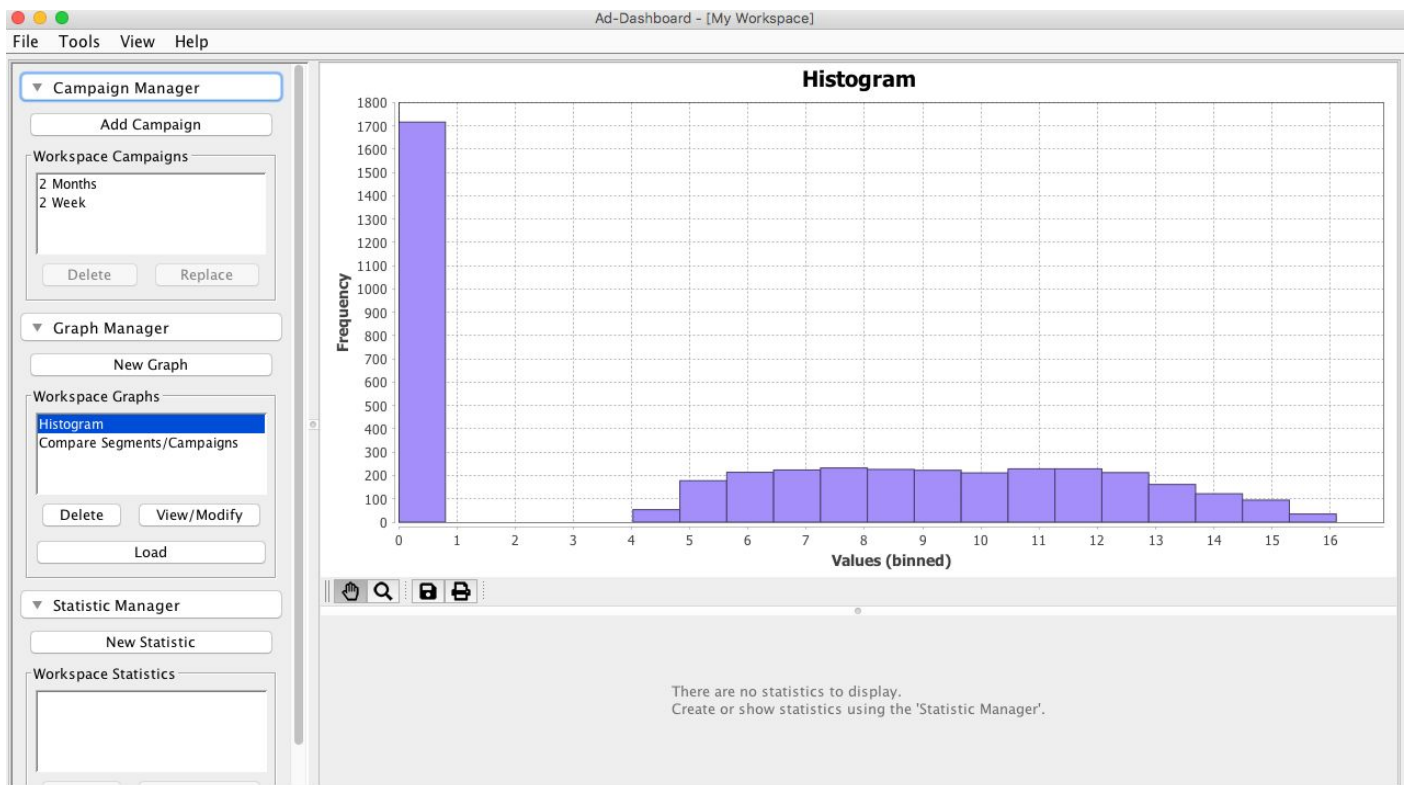
Key	
	Estimated correctly
	Estimated Higher
	Estimated Lower

Product backlog ID	User Story Heading	Initial Estimation	Actual
01	User-friendly dashboard (GUI)	XL	XXL
02	View metrics as chart	XL	XL
03	Data gathered from 3 logs	L	M
04	Time interval	M	M
05	Number of Impressions	M	S
06	Conversion rate summary	M	M
07	Total cost statistic	M	M
08	CPA statistics	M	M
09	CPC statistics	M	M
10	CPM statistics	M	M
11	CTR statistics	M	M
12	Bounce rate summary	M	M
13	Number of clicks	S	M
14	Number of uniques	S	M
15	Database schema design	S	XS
16	Overall design of the dashboard	M	Removed
17	Research Java libraries	M	S
18	Click costs histogram	L	M
19	Compare audience segments	L	M
20	Bounce definition - time	S	S
21	Bounce definition – number of pages	S	S
22	Date range filter	S	L
23	Context filter	S	S
24	Gender filter	S	S
25	Age filter	S	S
26	Income filter	S	S
27	Compare different campaigns	XL	M
28	Change font size option	L	M
29	Time of day statistics	M	M
30	Specific day statistics	M	M
31	Saving option	M	S

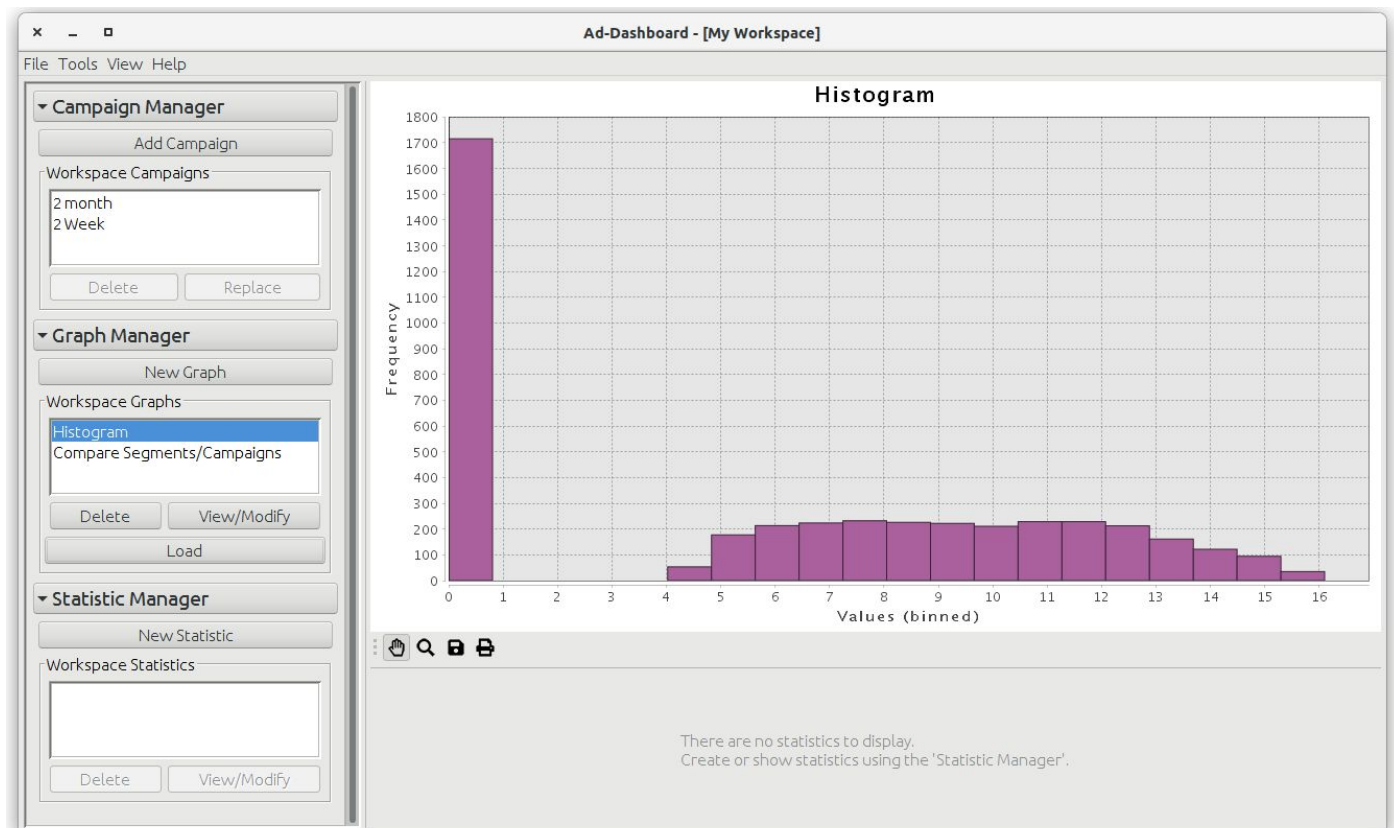
Appendices

Sprint 3 Application UI

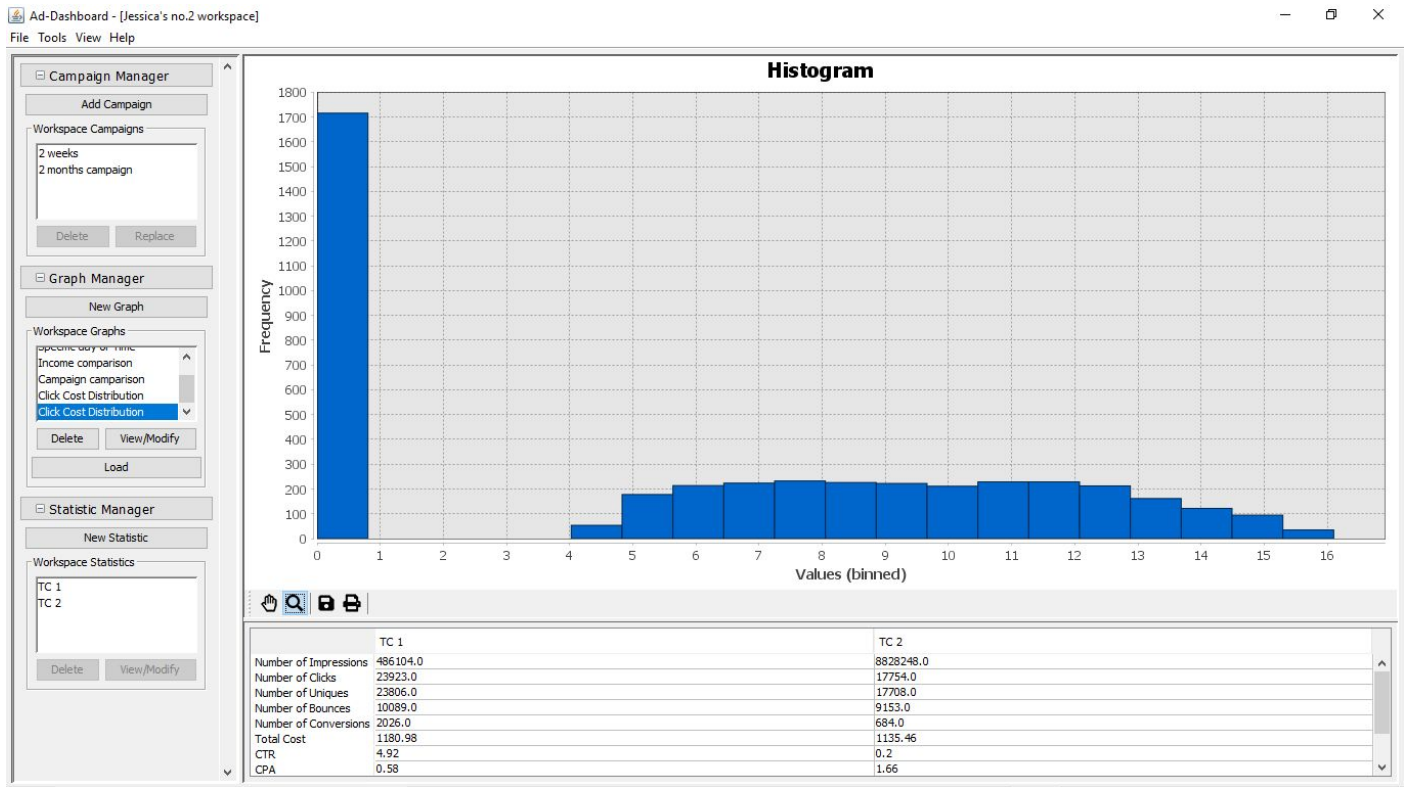
MacOS



Ubuntu



Windows



Notes

All screenshots and diagrams are available in the documentation folder in the zip file.