Software Engineering Group Project

Deliverable 3 - Increment 2

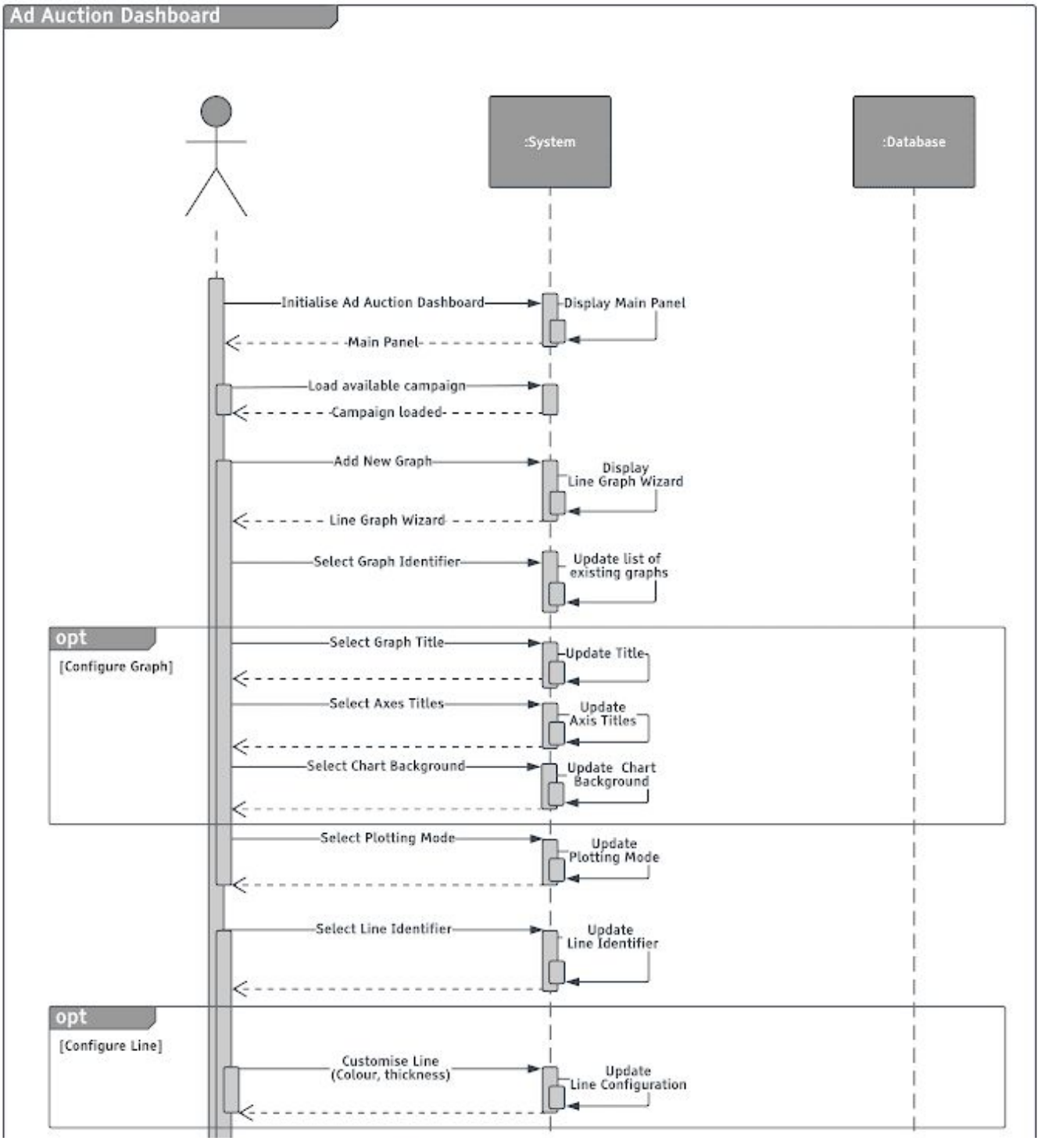
Group 33: David Jones, Simeon Milev, Harry Brown, Iliana Hadzhiatanasova, Kiera Spencer-Hayles, Ina Li

1 - DESIGN

1.1 - Key Design Artifacts

Sequence Diagram

The sequence diagram below shows the functionality we added to the system during this sprint based on the sprint backlog. The diagram shows the interactivity between the user and the system and allows us to visualise and validate various runtime scenarios. Similar activities (e.g. applying filters) are generalised to keep the diagram as simple as possible. For the same reason, the diagram doesn't include detailed functionality from Sprint 1.



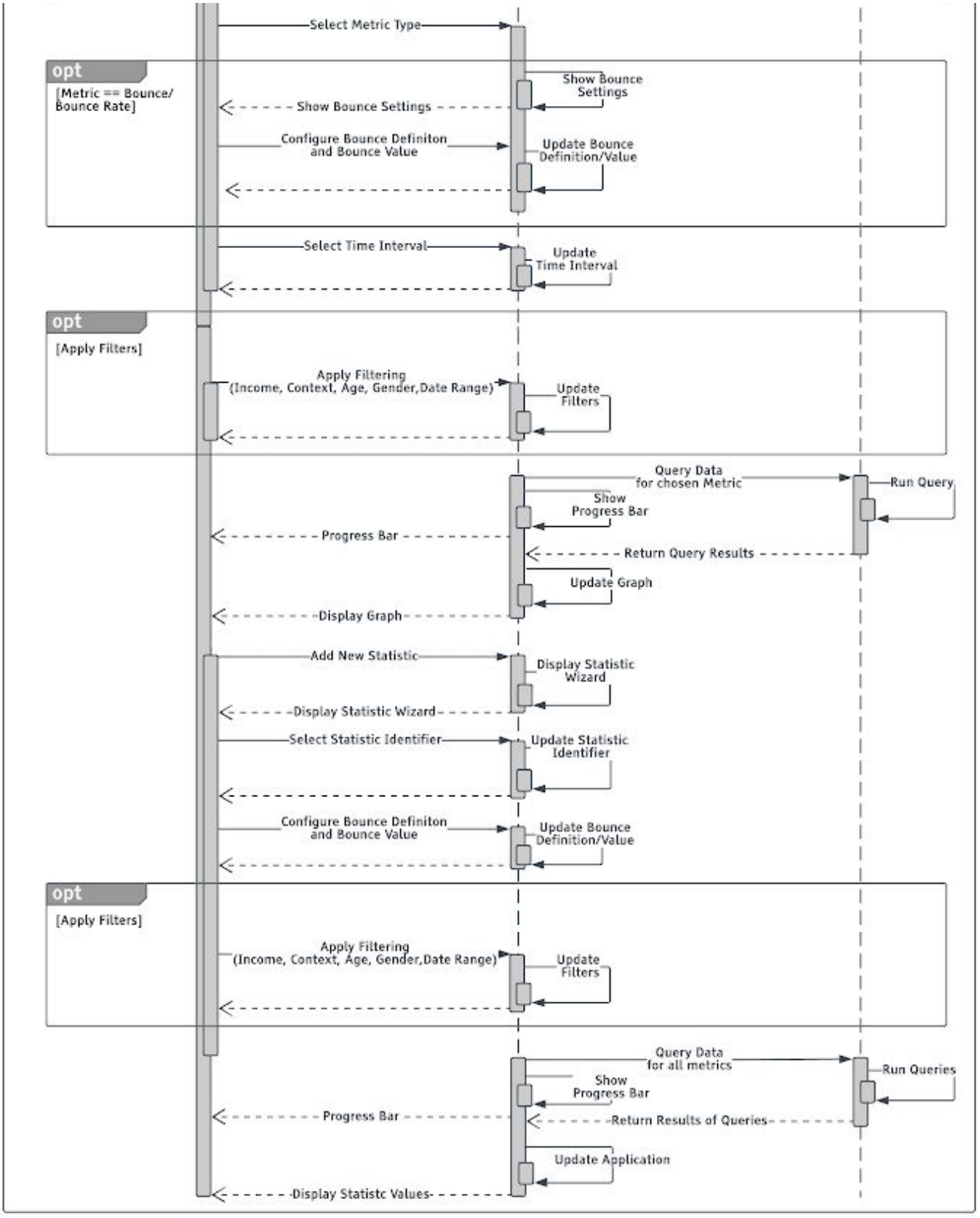


Figure 1: Sequence Diagram

Use Case Diagram

Below is the use case diagram based on the tasks stated in the sprint plan for the first two increments of the project. The functionality which has been added to the system during this sprint is highlighted in blue. Our main focus of this sprint was to make graphs and statistics available for the majority of the key metrics a user of the system may want to see. We also included a filtering option, which provides the user with the flexibility to view data based on income, gender, context, age and date range. Lastly, we added a saving option, which allows the user to save a generated graph as a file.

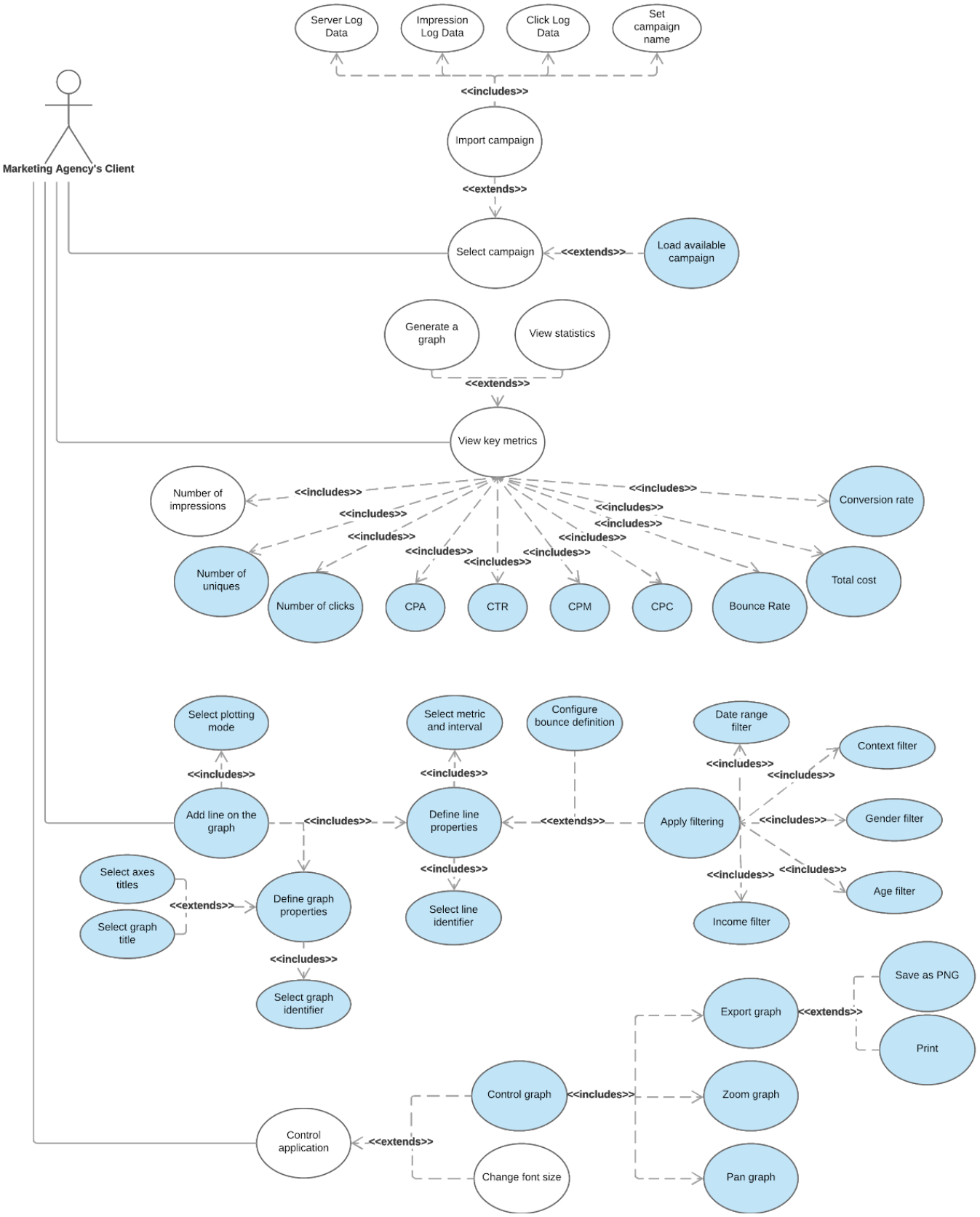


Figure 2: Use Case Diagram

Entity Relationship Diagram

The ERD below shows the relationships between the tables stored in the Ad Auction Dashboard database. In addition to the three tables - ***server_log***, ***impression_log*** and ***click_log***, which are populated with data from CSV files, we have included a **campaign** table. In this sprint, the campaign table is used so that multiple campaigns are available at the same time and the user is able to switch between them. The primary key in the campaign table is set to be of type *SERIAL* so that the id value is auto incremented every time a new campaign is imported into the system.

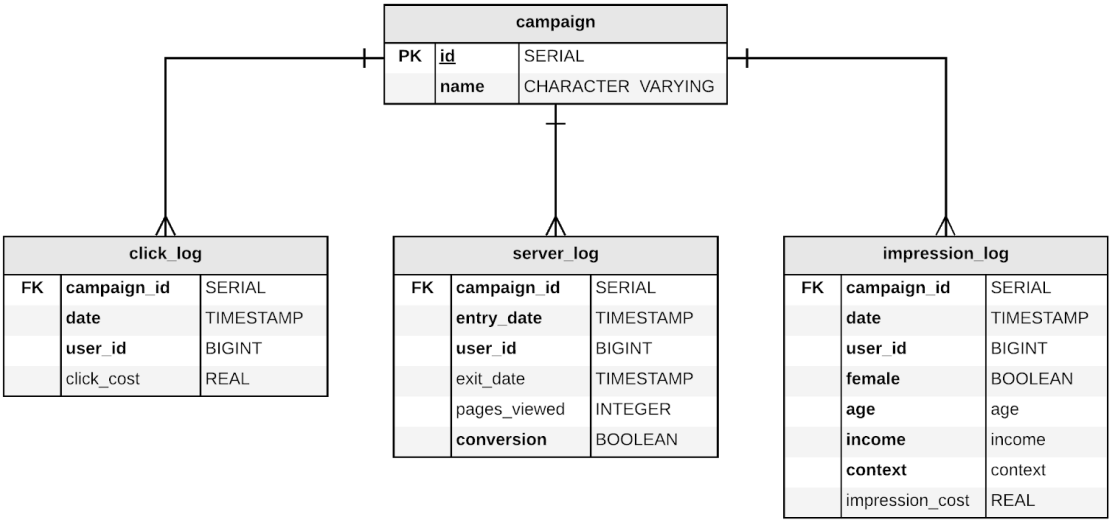


Figure 3: Entity Relationship Diagram

1.2 - Design Architecture (MVC Approach)

To ensure our application was easy to maintain, in increment 1, we implemented an extension of the active model similar to that of iOS (known as iOS MVC). Using this approach, the model and views are separate entities with all interactions passing through a controller. This means the view and model are interchangeable; provided a suitable controller is provided. Because this project is making use of Swing we adapted this model with another layer of abstraction. This is because Swing closely couples the view and controller by handing the graphical interface and the corresponding capture of user events, such as button presses. We have kept this as the view/controller, making it responsible for getting data from the view into a state understandable by the controller.

This design architecture has proved to be robust as all new functionality has been added with little logic changed. There is however a large amount of functionality changes, which have been achieved by adding or modifying ‘sub-handlers’ in the controller logic. Handlers are usually split into serving one of two purposes: updating the model or updating the view’s. The view handlers make use of a listener infrastructure that gets updates when the model is changed (and in what way its changed) to ensure model updates are always reflected in the view appropriately.

Future Proofing

For increment 3, we should be able to utilise the design architecture similarly to this increment. One feature of the next increment is adding a new graph type. In anticipation of this, where appropriate, we have implemented the visitor pattern. This means that if a graph type were to be added, it can be handled by a common interface with safety guaranteed at compile time.

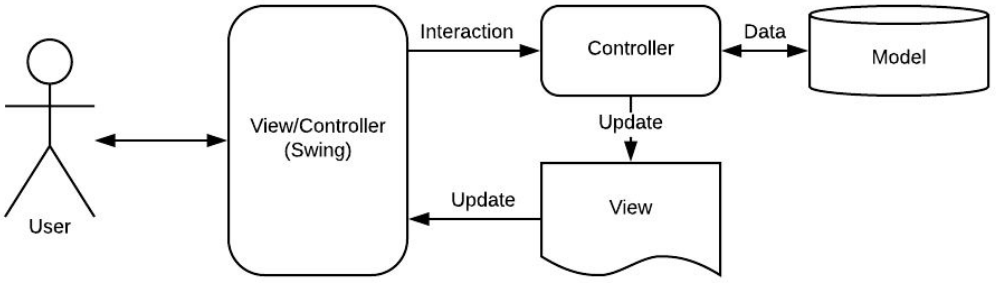


Figure 4: MVC Model

1.3 - Key Design Choices (Storyboarding)

This section covers storyboarding of functionality added or changed since increment 1. 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.

Dashboard Structure

The ‘Dashboard’ is the startup window and contains the structure of the application. The interface is separated into four main sections: the ‘Dashboard Menu Bar’ (along the top of the frame), the ‘Dashboard Controls’ (on the left hand side), the ‘Graph View’ (on the top right hand side) and the ‘Campaign Statistics View’ (on the bottom right hand side). The latter three sections are all adjustable in size and collapsible so the view can be modified to suit the user. The window as a whole is fully resizable with the ‘Graph Panel’ getting the preference of extra space. Like any normal application window, it can be minimised, maximised and closed using the frame header.

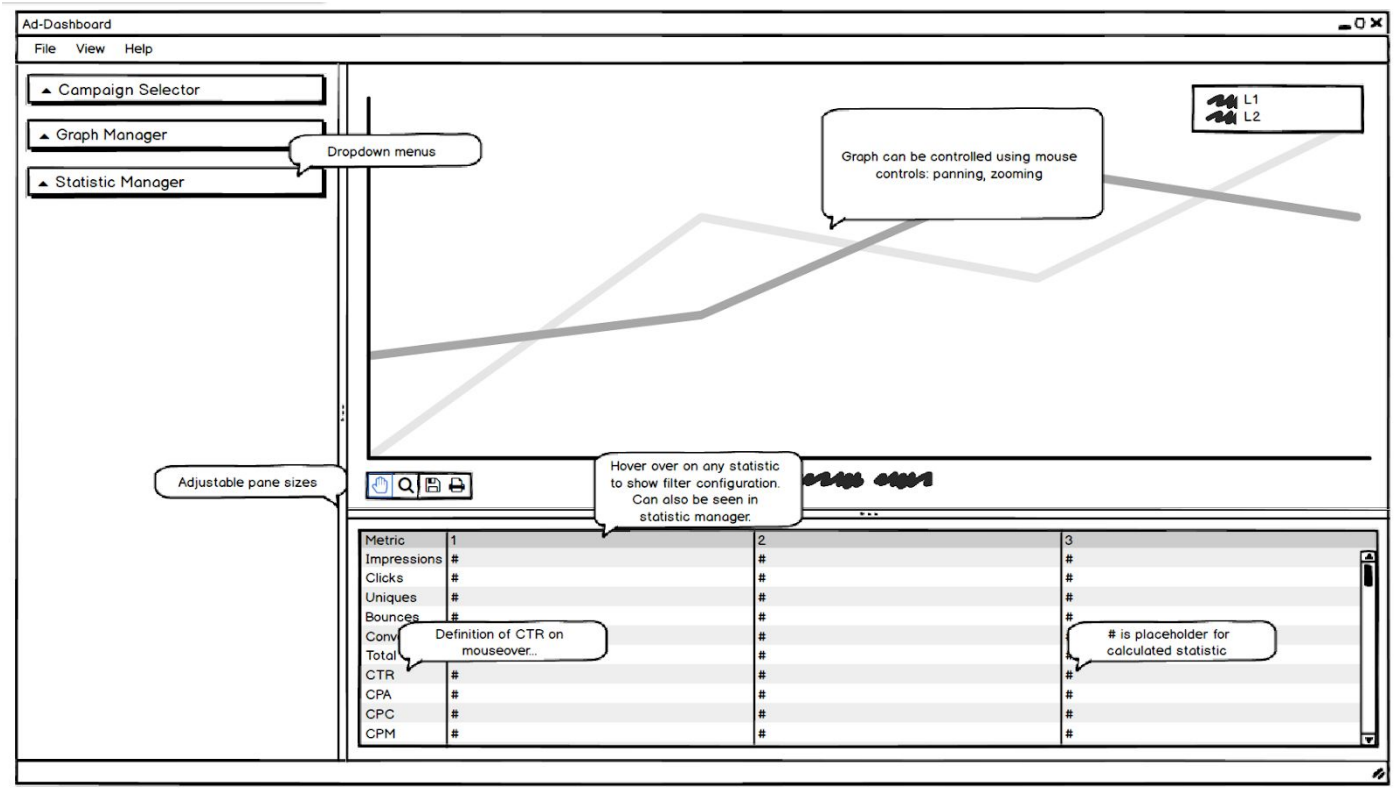


Figure 5: Dashboard

Dashboard Menu Bar

The ‘Dashboard Menu Bar’ provides access to one-click features that do not have a place in a sub-hierarchy. Some of the menu items are also given global keyboard shortcuts for ease of use. As of this sprint, there are three menus (seen in Figure 6). The ‘File Menu’ provides a clear method to exit the application. The ‘View Menu’ provides three toggles for hiding the various sections of the frame; these are convenient if the user knows for example that they will not be using any statistics or want to view a graph in full screen. It also provides global access to the ‘Definitions Dialog’. The ‘Help Menu’ provides global access to the ‘About Dialog’ and the ‘Preferences Dialog’; these are not accessible from anywhere else.

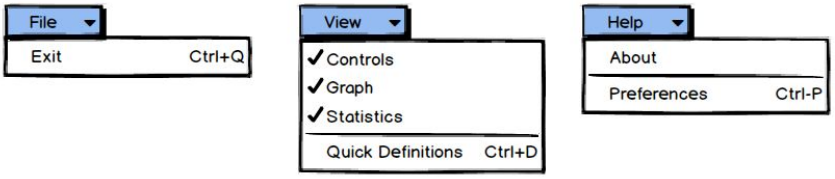


Figure 6: Menu Bar Items

Dashboard Controls

The ‘Dashboard Controls’ contains a set of expandable managers; shown in their collapsed state in Figure 5. These panels can be toggled to display/hide with a single click (on their respective header). The three required for this sprint are the ‘Campaign Selector’, the ‘Graph Manager’ and the ‘Statistic Manager’.

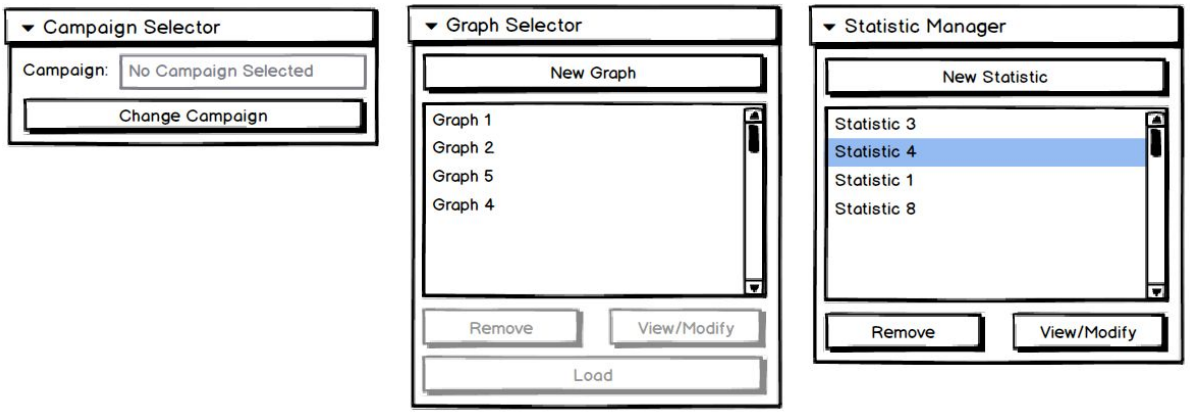


Figure 7: Campaign Selector, Graph Manager, Statistic Manager

Campaign Selector

The ‘Campaign Selector’ displays and allows the setting of the current campaign in the workspace. This campaign is used for all queries as a search basis. Pressing the ‘Change Campaign’ button shows the ‘Campaign Importer/Selector Dialog’.

Graph Manager

The ‘Graph Manager’ displays and allows modification of the workspace’s graph configurations; this includes addition, modification and removal. It also allows loading of a graph into the ‘Graph View’. To make disallowed behaviour clear, appropriate buttons are disabled when no graph is selected in the graph list. Modification or creation of graphs makes use of the ‘Graph Wizard’. The configurations of existing graphs are available as tooltips when hovered over in the graph list.

Statistic Manager

The Statistic Manager’ displays and allows modification of the workspace’s statistic configurations; this includes addition, modification and removal. Modifications here are reflected in the ‘Campaign Statistic View’. To make disallowed behaviour clear, appropriate buttons are disabled when no statistic is selected in the statistic list. Modification or creation of statistics makes use of the ‘Statistic Wizard’. The configurations of existing statistics are available as tooltips when hovered over in the statistic list.

Campaign Importer/Selector Dialog

Since increment 1, campaign selection has been added to the ‘Campaign Importer/Selector Dialog’. Import/selection behaviour can be chosen using the large navigation buttons on the left-hand side. Campaign selection exists as a list of available campaigns on the connected server; this includes any campaigns imported using the import functionality. There is also has a refresh button to allow the user to update the list in case of external updates.

There are no graphical changes to the campaign import process.

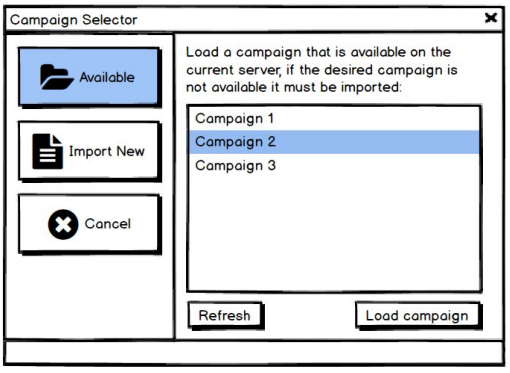


Figure 8: Campaign Selector

Line Graph Wizard

The ‘Line Graph Wizard’ is a dialog that can be used to generate or modify line graph configurations. It consists of two tabs: ‘Graph Properties’ and ‘Lines’. The ‘Graph Properties’ tab can be used to configure the global behaviour of the graph configuration, that is behaviour that applies either to every line or the single graph instance. The ‘Lines’ tab has support for modification of the set of lines belonging to the graph configuration. This is achieved through a subset of tabs, each representing a line. These can be created from scratch using the ‘New’ button or initialised with information such as filters, from an existing statistic, using the ‘Import’ button. The currently displayed line can also be removed using the ‘Remove’ button; this is disabled if no line is selected. When modifying a line in the line wizard, all configuration information is first loaded. Changes are discarded and the wizard is closed if the ‘Cancel’ button is pressed. Whereas, the line wizard updates the workspace’s graph list when either ‘Apply’ or ‘Apply & Close’ are pressed (the latter also closing the dialog).

Aethstetics

Each graph and its lines are highly modifiable in terms of aesthetics, this is to avoid any issues with contrast and allows for large amounts of personal preference. This is in the form of setting the graph background colour (with appropriate grid-line colours chosen automatically) and allowing each line to have its colour and thickness modified independently.

Hiding

A line can temporarily be hidden (whilst data is still fetched) to allow for quick modification of displayed data in the graph.

Line Data

The data a line uses is configured using the universal ‘Filter Selection Dialog’ and if the appropriate metric is set, the universal ‘Bounce Rate Definition Panel’ is also shown. When selecting a metric a helpful tooltip is available to display the definition (over the ‘?’ button); this is to avoid any confusion over what each metric is displaying. Pressing the ‘?’ button will bring up the full ‘Definitions Dialog’.

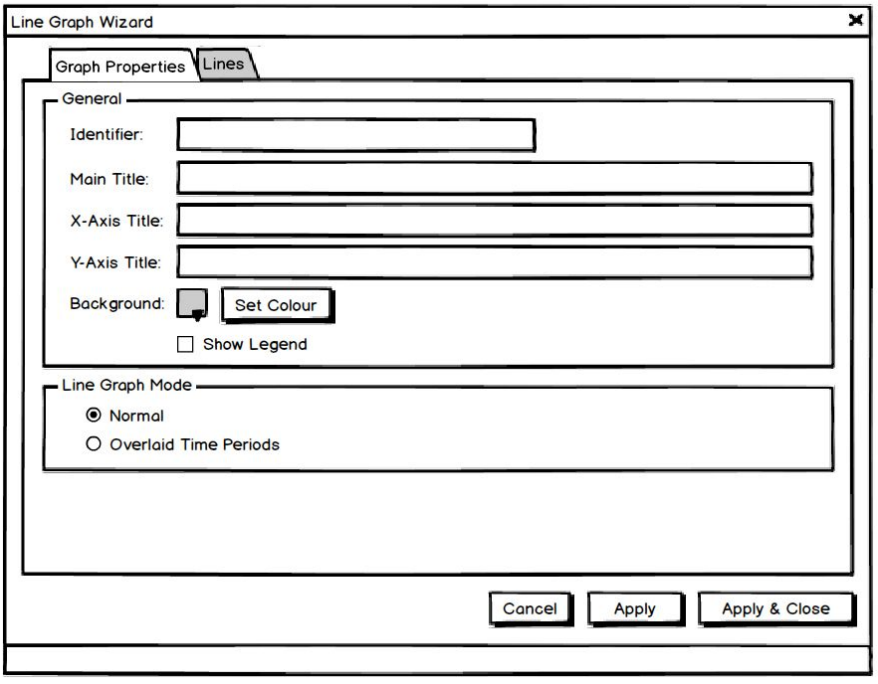


Figure 9: Line Graph Wizard - Graph Properties

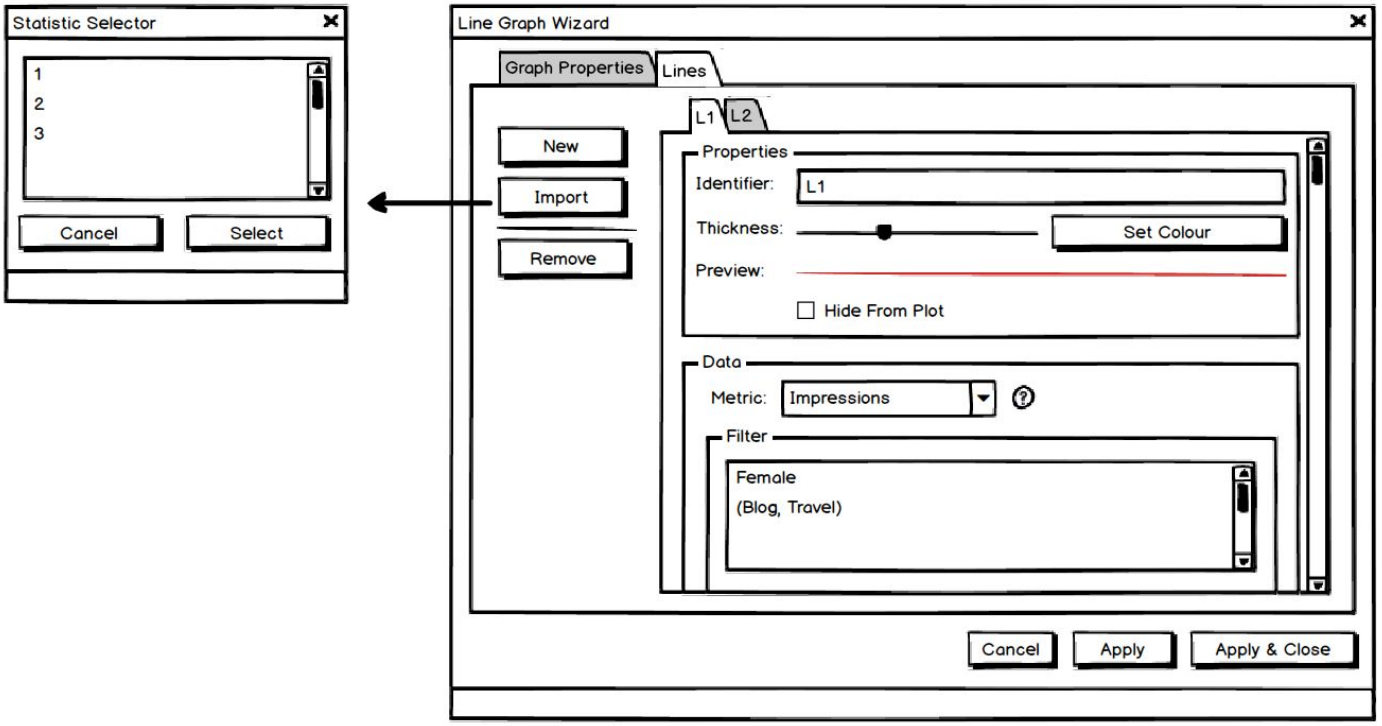


Figure 10: Line Graph Wizard - Lines

Statistic Wizard

The Statistic Wizard' is a dialog that can be used to generate or modify statistic configurations. A statistic is defined as a query for all metric types using a single uniform configuration. The query used is configured using the universal 'Filter Selection Dialog' and as all metrics are queried the universal 'Bounce Rate Definition Panel' is also shown. When loading the wizard, it can be loaded with pre-existing data; this allows the user to modify an existing statistic or create a new statistic from scratch. Changes are discarded and the wizard is closed if the 'Cancel' button is pressed. Whereas, the statistic wizard updates the workspace's statistic list when either 'Apply' or 'Apply & Close' are pressed (the latter also closing the dialog).

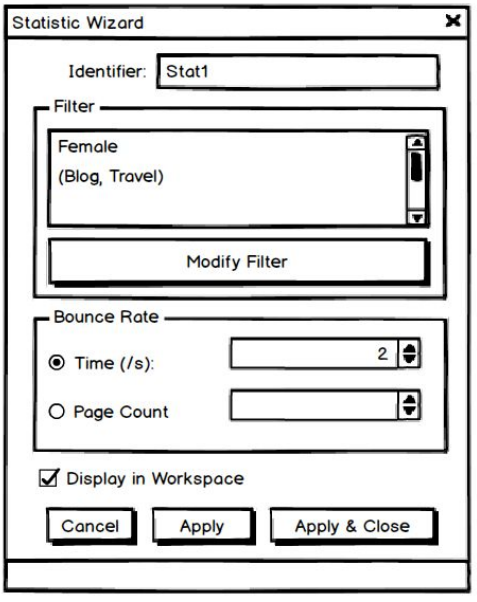


Figure 11: Statistic Wizard

Filter Selection Dialog

The ‘Filter Selection Dialog’ is an intuitive method for creating a filter for queries. For most filter types it makes use of a tree structure where each branch relates to a certain field to filter on. By default all items are selected as there is no filter. To remove an item from queries (filtering it out), the item can be unticked. Filtering by start and end date can also be configured using calendar dropdowns. The dialog is self validating so if the filter is not valid (e.g. All values filtered out for a certain field will always return no data), an error will display when the ‘Confirm’ button is pressed. The ‘Cancel’ button discards any changes to the filter.

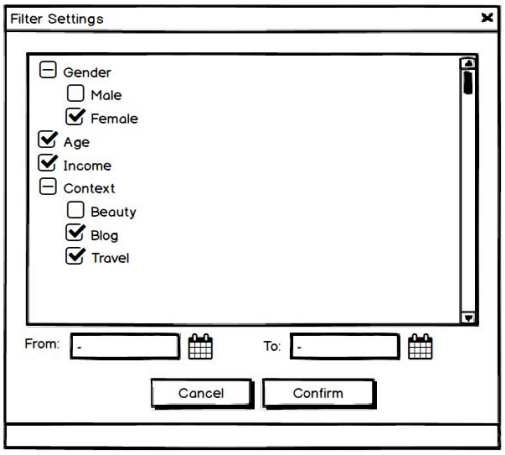


Figure 12: Filter Selection Dialog

Graph View

The ‘Graph View’ is the method of displaying any line graph configuration and the respective data created in the ‘Line Graph Wizard’. It can be interacted with directly by the user using one of two control schemes: Pan mode and Zoom mode. Pan mode utilises an intuitive, quick to use, scheme: mouse dragging to pan the view, zooming in and out with the scroll wheel and resetting the view with a double mouse click. Zoom behaviour can be modified to affect either range, domain or both with system-dependent modifiers. Zoom mode is added for precision, its scheme uses: a drag box to zoom the range and domain to a specific area and dragging with a modifier to pan around. These modes are togglable using the toolbox buttons. The toolbox buttons also allow for saving and printing of the current view.

To ensure the user knows what each line represents the line configuration options are visible as a tooltip when over the respective line’s legend entry (if enabled).

See Figure 5 for wireframing.

Campaign Statistics View

The ‘Campaign Statistics View’ is the method of displaying all statistics in the workspace. The queried values for each metric are displayed in a table where each statistic has a column and each metric has a row. To ensure metric definitions are clear a tooltip will appear when hovering over the metric types. To ensure the user knows what each statistic represents the statistic configuration options are visible as a tooltip when over the respective column. There is no interaction with this view other than manipulating column positioning and size.

See Figure 5 for wireframing.

Definitions Dialog

The ‘Definitions Dialog’ is a floating dialog that displays all helpful definitions in a scrollable list; no wireframe was required for this.

About Dialog

The ‘About Dialog’ is used to display general application information and more importantly the licenses used to create the application; no wireframe was required for this.

1.3 - Amendments

Changes in User Stories

Look and Feel

In the Increment 1 meeting, the clients raised a concern related to the look and feel of the application when executed on different operating systems. Having received this feedback, we decided to include a new user story (**ID 39**). This user story is under the *"NFR"* section, as it does not state a specific behaviour of the system, but rather a criteria for its functionality.

New User Story:

39	As a <Small Marketing Agency Owner> I want <the UI of the application to look good on **different operating systems** > So that <my clients can work comfortably with it >

Bounce Values

While working on Increment 2 we noticed that in our user stories we have set a specific bounce value. However, we believe that the user should have the flexibility to not only change the bounce definition but also the bounce value. Therefore, we made some amendments to user stories 16 and 17:

Initial User Stories:

16	As a <Marketing Agency Client> I want to <define that a bounce is registered as time spent on the advertising campaign website> So that <I can create statistics of web users who have not interacted with the website **for more than 10 seconds** >
17	As a <Marketing Agency Client> I want to <define that a bounce is registered as number of pages visited> So that <I can create statistics of web users who have not visited **more than 1 page** of the advertising campaign website>

User Stories After Changes:

16	As a <Marketing Agency Client> I want to <define that a bounce is registered as time spent on the advertising campaign website> So that <I can create statistics of web users who have not interacted with the website **within a certain time** >
17	As a <Marketing Agency Client> I want to <define that a bounce is registered as number of pages visited> So that <I can create statistics of web users who have not visited **more than a certain number of pages** of the advertising campaign website>

Changes in Tasks

Printing

When we created the backlog, we decided that the printing user story would be considered a WON'T HAVE however now we have implemented this functionality because JFreeChart has printing support, greatly simplifying the task. Therefore, we have included a task (**PbID 34**) in our burndown chart for this sprint, which can be seen in **section 4.1**

Initial task:

34	Printing option	L	WON'T HAVE

Current task:

34	Printing option	S	COULD HAVE

Campaign Storage

In the documentation for Sprint 1, we included a campaign table in our Entity-Relationship Diagram. We thought it would be best to create it in the third sprint as this was when we were going to introduce the functionality of comparing multiple campaigns. However, we have now decided to place it into this sprint. This is due to the fact that it would be more user-friendly to maintain a list of campaigns that are already uploaded so that the user doesn't have to upload the logs every time the application is launched. In order to do this, we needed to add the campaign table so that the database can keep multiple clicks, server and impression logs, where each row would have a link to a *campaign_id* attribute. The task is included in our burndown chart in **section 4.1**

2 - Scenarios and Testing

2.1 - Scenarios and key test outputs against them

Peter Quinn

Viewing Key Statics
(Number of conversions / clicks / uniques, CTR)
Scenario 1

Peter has been on holiday for two weeks and now is back in the office.

He wants to view the development of different statics of his latest campaign while he was gone.

Peter opens the application and his most recent campaign is loaded in the application.

He is interested in the number of conversions, clicks, and uniques, as well as in the click-through rate of the campaign.

Peter opens the "Statistic Manager" drop-down menu as he wants to view the number of clicks and number of uniques.

He clicks on the “Add New Statistic” button.

He wants to fetch all data, so he doesn’t change the default configuration.

He clicks the “Apply & Confirm” button.

He can see all the statistics in the table below the graph.

He looks for the statistics he wants to see.

Peter is happy with the good campaign statistics for the time he was out of office.

Corresponding User Stories:

As a <Marketing Agency Client> I want to <view the **Conversion Rate** of an ad> So that <I can evaluate how many web users have shown further interest in a campaign> (**PbID 06**)
As a <Marketing Agency Client> I want to <view the **Click-through-rate (CTR)** of an ad> So that <I know if the advertising campaign is of interest to web users> (**PbID 11**)
As a <Marketing Agency Client> I want to <view the **number of clicks** on the campaigns> So that <I know how frequently a web user shows interest in an ad> (**PbID 13**)
As a <Marketing Agency Client> I want to <know how many **unique web users** have clicked on an ad> So that <I can create monthly statistics about how many new users have been attracted by an advertising campaign> (**PbID 14**)

Test against scenario:

Type of test: Manual testing

Preconditions:

- Database credentials are provided and campaign is uploaded in the application.
- GUI exists and is interfaced with the database.
- Queries for metrics are available.

Actions:

- Select a campaign from the list of available campaigns.
- Expand "Statistic Manager".
- Add new statistics.
- Observe if all buttons act as expected.

Application opened successfully.
A list of all campaigns uploaded in the application is available.
The desired campaign is successfully selected.
Submenu opened and loaded correctly.
Successfully queried statistic data.
Statistic values are shown in the table below the graph.
Data is in correct format.
All buttons work as expected.
Application closes successfully.
No big delays in response time.

Finance management
(Total cost, CPA, CPC, CPM)
Scenario 2

Peter was asked by his manager to collect financial information about this year's Easter campaign and share it with his new colleague in the Financial Department.

Peter clicks on the Ad Auction Dashboard icon on his desktop to launch the application.

He notices that there is no campaign set under the "Campaign Manager" collapsible panel.

He clicks on the "Change Campaign" button.

Peter uploaded the campaign last time he was using the application, so he sees it in the list of already loaded campaigns under the "Available" section.

He clicks on "Easter Campaign" and confirms his selection by clicking on the Load Campaign button.

He looks at the email from his manager and sees that he has to check statistics for Total cost, CPA, CPC, CPM.

He expands the "Statistic Manager" menu and clicks on "New Statistic".

Peter wants to see data for people with Low income, so he clicks on the "Modify filter" button.

He deselects *High* and *Medium* in the "Income" field.

He then clicks the "Apply & Confirm" button.

Peter can now see statistic calculations in the table below the graph.

He includes all the information he has gathered in an email and sends it to his new colleague.

Corresponding User Stories:

As a <Marketing Agency Client> I want to <know the the **total cost** for an advertising campaign> So that <I can make better management of outgoing finances for each campaign> (**PbID 07**)
As a <Marketing Agency Client> I want to <view the **Cost-Per-Acquisition (CPA)** of an ad> So that <I have statistics of average outgoing finances of for every time a web user clicks and acts on an ad> (**PbID 08**)
As a <Marketing Agency Client> I want to <view the **Cost-Per-Click (CPC)** metrics of an ad> So that <I have statistics of average outgoing finances for every time a web users clicks on an ad campaign> (**PbID 09**)
As a <Marketing Agency Client> I want to <view the **Cost-Per-Thousand Impressions (CPM)** metrics of an ad> So that <I have statistics of average outgoing finances for every one thousand times an advertising campaign is shown to a web user> (**PbID 10**)

Test against scenario:

Type of test: Manual testing

Preconditions:

- Database credentials are provided and campaign is uploaded in the application.
- GUI exists and is interfaced with the database.
- Queries for metrics are available and working.

Actions:

- Select a campaign from the list of available campaigns.
- Expand "Statistic Manager".
- Add new statistic.

Application opened successfully.
A list of all campaigns uploaded in the application is available.
Submenu opened and loaded correctly.
Successfully queried statistic values.
Filter is successfully configured.
Statistic values are shown in the table below the graph.
Data is in correct format.
All buttons work as expected.
No big delays in response time.

Jessica Pierce

Filtering

Scenario 3

Jessica's company has launched a campaign about women health over March and she is given the task to see the people it attracts.

She opens the Ad Auction application and uploads the "Women health" campaign data.

She wants to see click data for women that are between 25 and 34 years old and wants to compare the group of women having "High" and "Low" incomes.

She clicks on "Graph Manager" and selects "New Graph".

She puts "Women health" as graph identifier and clicks on the "Line" tab.

From the Line panel, she selects the line preferences, as well as "Clicks" as the metric type.

She applies the desired age and gender filters and selects the "High" income filter for the first line.

She sets the start date to 1/03/2018 and the end date to 15/03/2018.

Jessica clicks on the "Add new line" button.

She selects name, thickness and colour for the second line.

She applies the same age and gender filters, but selects "Low" income filter.

She sets the start date to 15/03/2018 and the end date to 31/03/2018.

She clicks on the "Apply and Close" button.

Jessica can now see a graph with the 2 lines created over the two time periods.

She now wants to see overlaid data, so she clicks on the "View/Modify" button.

Jessica can see the Line Graph Wizard and she clicks on the tick box next to "Overlay Mode"

She clicks on the "Apply and Close" button.

Jessica can now see the two overlaid lines on the graph.

Corresponding User Story:

As a <Marketing Agency Client> I want <to **filter** performance metrics **by date range**> So that <I can detect common trends amongst web users> (**PbID 22**)

As a <Marketing Agency Client> I want to <**filter** performance metrics **by context**> So that <I have statistics of the websites that referred the most users > (**PbID 23**)

As a <Marketing Agency Client> I want to <view performance metrics **filtered by gender**> So that <I can identify trends amongst web users of different gender> (**PbID 24**)

As a <Marketing Agency Client> I want to <view performance metrics **filtered by age**> So that < I can identify trends amongst web users of different age> (**PbID 25**)

As a <Marketing Agency Client> I want to <view performance metrics **filtered by income**> So that < I can identify trends amongst web users with different incomes> (**PbID 26**)

Test against scenario:

Type of test: Manual testing

Preconditions:

- Database credentials are provided.
- The campaign is not uploaded in the application.
- GUI exists and is interfaced with the database.
- Queries for metrics are available and working.

Application opened successfully.

Import of campaign was successful - database is populated.

Graph Manager menu expanded and loaded correctly.

Line settings are configurable.

Graph displays multiple lines.

Actions: • Upload new campaign. • Create new graph. • Apply appropriate filtering to the 2 lines. • Check if there are big delays in system responses. • Observe if all buttons act as expected.	**All lines are on the graph with corresponding line preference settings. [Filtering and graphs behave as expected]** **Overlay mode can be enabled / disabled.** **Data is in correct format.** **All buttons work as expected.** **No big delays in response time.**

Bounce statistics

(Bounce rate, Bounce definition, Total bounces)

Scenario 4

Jessica is given the task to see bounce statistics for the most recent campaign that was created for her toy company by the Marketing Agency.

Jessica has to check statistic for number of bounces and bounce rate for users who have visited less than 2 pages, as well as users, who have spent less than 10 seconds on the webpage.

She expands the "Graph Manager" panel and clicks on the "New Graph" button.

Jessica puts "Bounces summary" as the Graph identifier and clicks on the "Line" tab to add the desired lines.

She clicks on the "New Line" button.

Jessica selects the line properties and selects "Number of Bounces" from the dropdown menu next to the "Metric Type" label.

An additional section for Bounce Definition is now open.

Jessica selects "Page Count" and sets the value to 2.

She clicks "Apply & Close".

The system displays an error, because Jessica hasn't set a time interval.

Jessica clicks on "Ok" and selects HOUR from the interval drop down.

She clicks on the "New line" button again to add a second line with bounce statistics for users who have spent less than 10 seconds on the webpage.

She follows the same steps before, but she selects "Time" in the Bounce Definition section and sets the value to 10.

Jessica clicks on the "Apply & Close" button.

She is now displayed with a graph having two lines for the two types of bounce definition.

Jessica remembers she had to view bounce rate statistics as well, so she selects the correct graph from the "Existing Graphs" list and clicks on the "View/Modify" button.

She selects the "Lines" tab, configures her line preferences and selects "Bounce Rate" from the dropdown next to the "Metric Type" label.

An additional section for Bounce Definition is now open and she selects "Pages" with value 2.

She repeats the steps for creating a new line and selects "Times" with value 10 in the Bounce definition section.

She clicks on "Apply & Confirm".

Jessica can now see all the metrics on the graph.

She closes the application.

Corresponding User Story:

As a <Marketing Agency Client> I want to <view the **Bounce Rate** of an ad> So that < I can determine if an advertising campaign hasn't kept the attention of the web user for a long time> (**PbID 12**)
As a <Marketing Agency Client> I want to <define that a **bounce** is registered as **time spent** on the advertising campaign website> So that <I can create statistics of web users who have not interacted with the website within a certain time > (**PbID 20**)
As a <Marketing Agency Client> I want to <define that a **bounce** is registered as number of **pages visited**> So that <I can create statistics of web users who have not visited more than a certain number of pages of the advertising campaign website> (**PbID 21**)
As a <Marketing Agency Client> I want to <see **total bounce statistics**> So that <I know how often a user fails to interact with my campaign> (**PbID 39**)

Test against scenario:	
Type of test: Manual testing **Preconditions:** • Database credentials are provided and campaign is uploaded in the application. • GUI exists and is interfaced with the database. • Queries for metrics are available and working. **Actions:** • Upload new campaign. • Follow scenario steps. • Check if there are big delays in system responses. • Observe if all buttons act as expected.	**Application opened successfully.** **Graph Manager menu expanded and loaded correctly.** **Bounce definition appears only when appropriate metric is selected.** **Bounce type and value are configurable by the user.** **Values below 0 are not allowed.** **Appropriate error message is displayed when the time interval is omitted.** **Line settings are configurable.** **Graph displays multiple lines.** **All lines are on the graph with corresponding line preference settings.** **Data is in correct format.** **All buttons work as expected.** **No big delays in response time.**

Saving and Printing

Scenario 5

Jessica has been at her desk all day and wants to take a break and get a coffee from the shop next to her office.

She does not want to leave all her work on the computer as someone else may want to use it while she's away, so she has decided to save the graphs she's generated.

The Ad Auction Dashboard is open on her Desktop and has multiple graphs loaded in the "Existing Graphs" list.

She clicks on the "Export" button under the loaded graph.

She navigates to her workspace folder and clicks on the "OK" button to confirm.

Jessica also decided to print a copy of the graph to look over during her coffee break.

She clicks on the "Print" button.

Jessica selects the size and the source for printing and confirms by clicking the "OK" button.

The printer on her desk prints the graph.

Jessica takes the copy and heads out to the near cafe to have a break.

Corresponding User Story:

As a <Marketing Agency Client> I want to <have the option to **save** advertising campaign **summary charts** in different formats> So that <I can have offline access to the data at any point in time> (**PbID 31**)

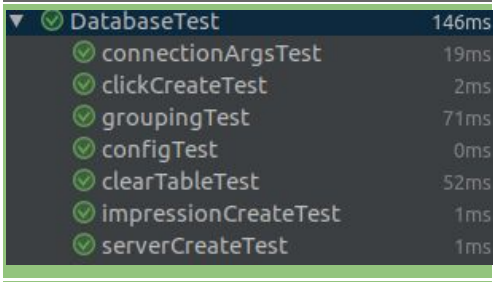
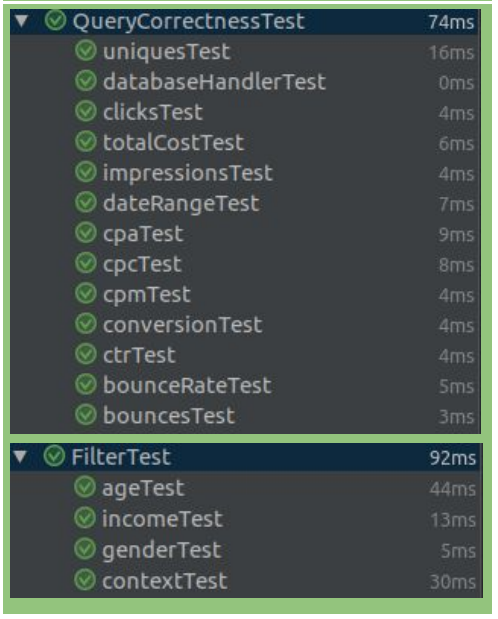
As a <Marketing Agency Client> I want to <have the option to **print** advertising campaign charts> So that <I have all the data on paper, as well as on a computer> (**PbID 34**)

Test against scenario:	
Type of test: Manual testing **Preconditions:** • Database credentials are provided and campaign is uploaded in the application. • GUI exists and is interfaced with the database. • Queries for metrics are available and working. **Actions:** • Upload new campaign. • Follow scenario steps. • Check if there are big delays in system responses. • Observe if all buttons act as expected.	**Application opened successfully.** **All lines are on the graph with corresponding line preference settings.** **Graph is saved in the specified location and can be reopened.** **Graph is printed successfully.** **All buttons work as expected.** **No big delays in response time.**

2.2 - Testing

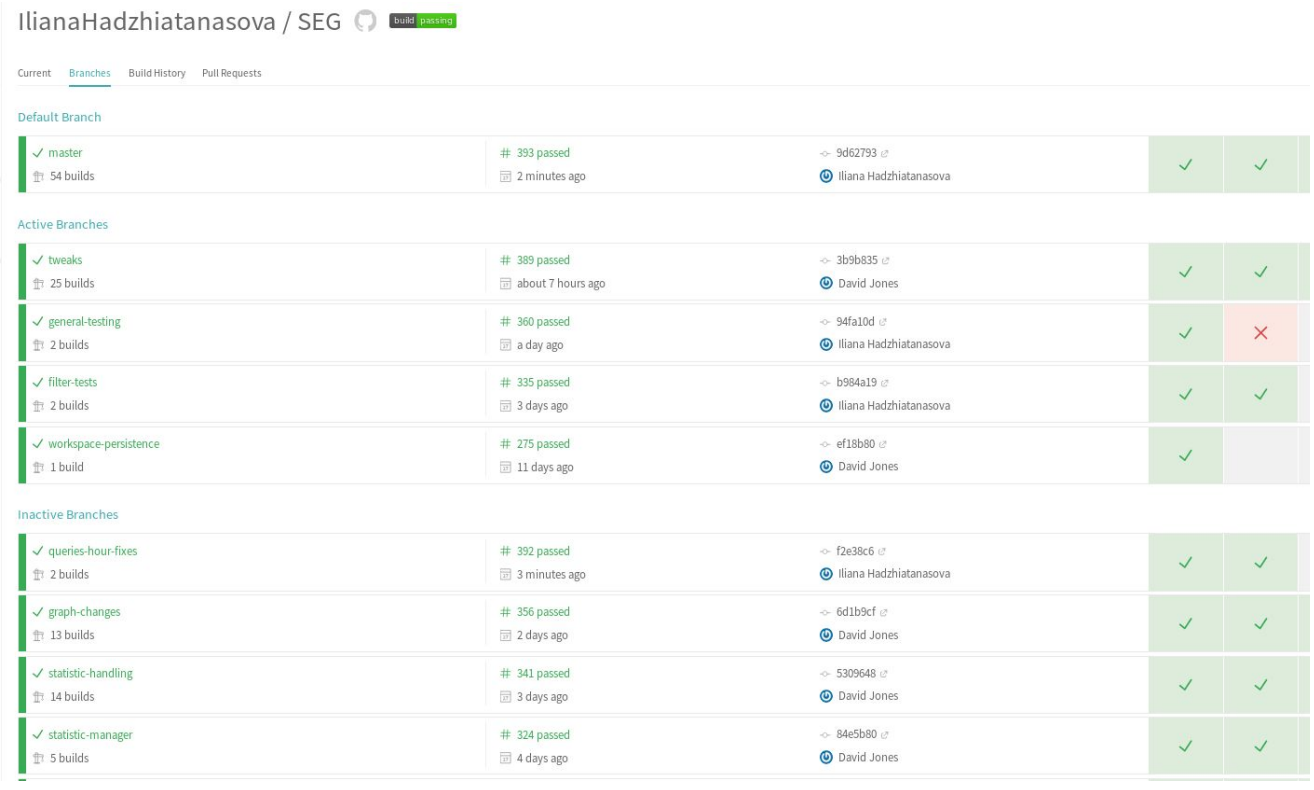
Unit Testing

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

Preconditions	Unit Test	Pass/Fail
The config.properties includes user Postgres credentials.	Database Tests • Tests are run automatically through TravisCI every time code is pushed to any branch in the repository	
Integration between TravisCI and Postgres is set up. Example dataset that is 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 *DatabaseQueryFactory* Class.	Query Correctness/ Filter Tests • 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	

TravisCI

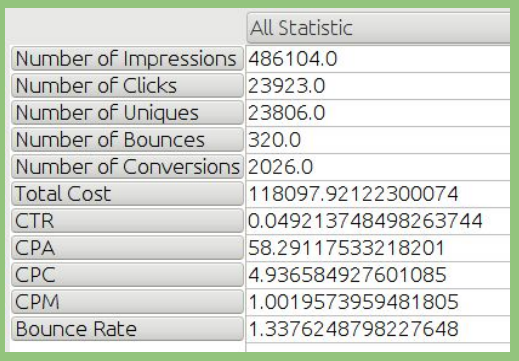
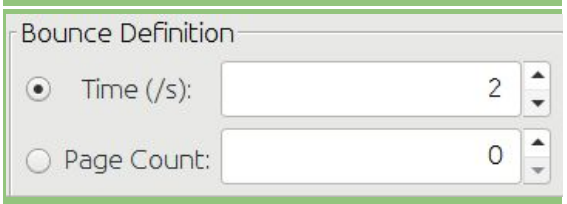
All unit tests are run through TravisCI in order to detect problems in the code as early as possible.

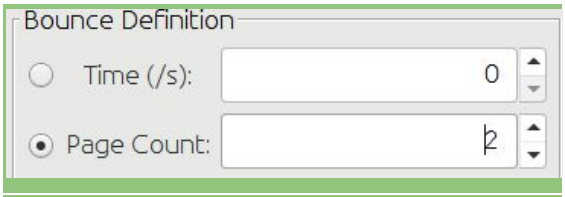
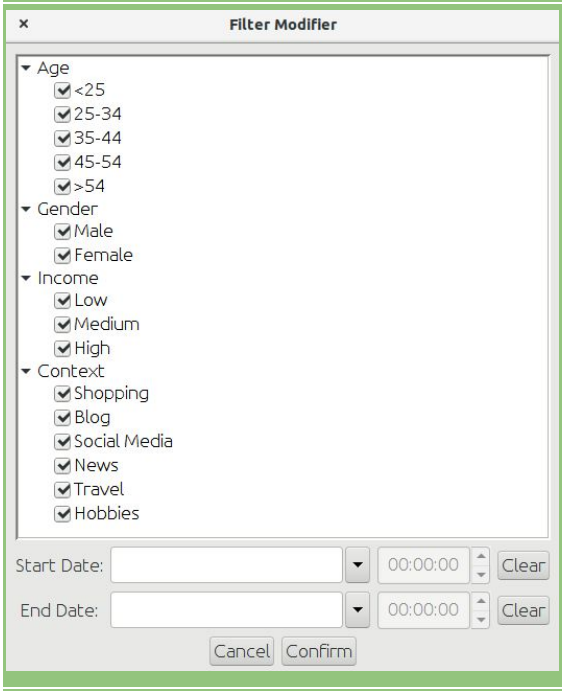
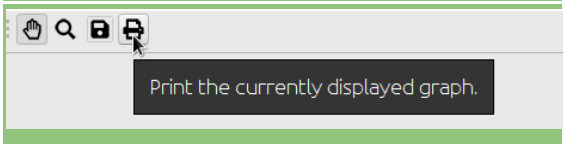


Regression Testing

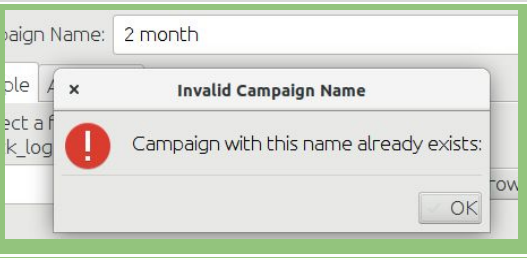
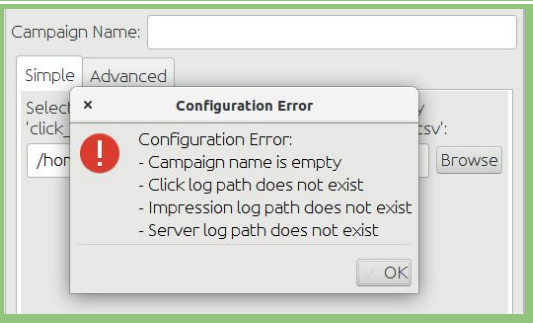
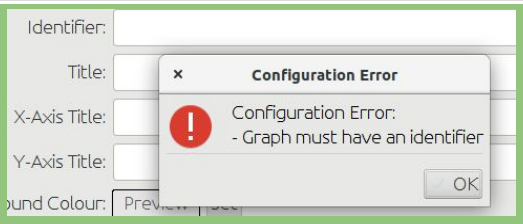
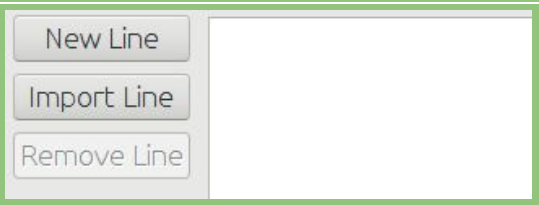
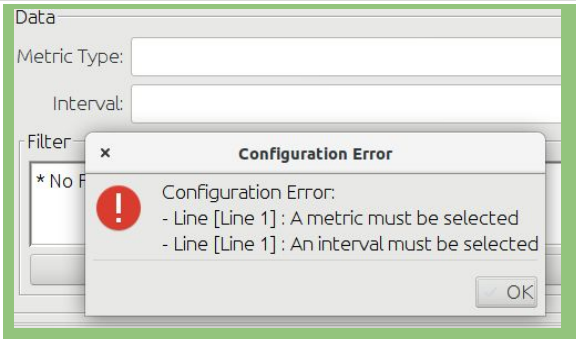
Old functionality	Testing	Pass/Fail
Import campaign	Campaign is imported correctly through both “Advanced” and “Simple” Mode. Database is populated. The new campaign table is updated and all logs include a campaign_id attribute. The new functionality of the application has not affected campaign importing.	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.	Pass
Import progress	The user can see import progress (in %). Once the campaign is imported the user gets a confirmation message.	Pass

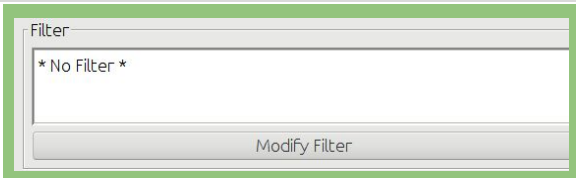
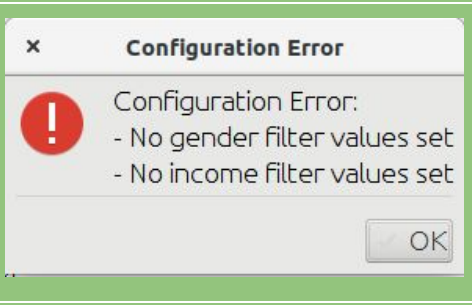
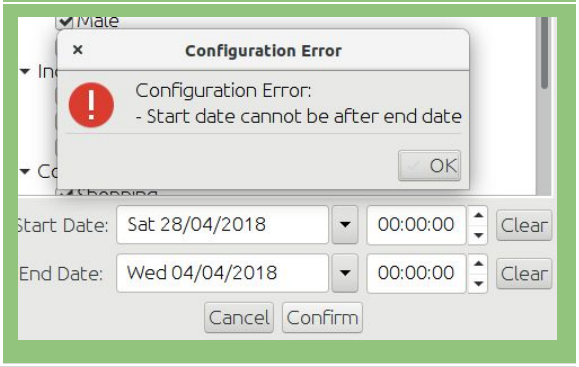
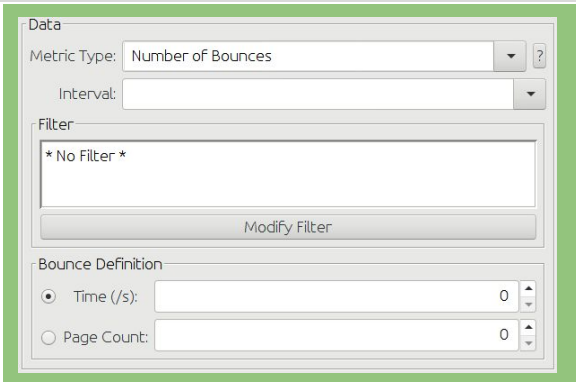
Acceptance criteria testing

PbID	Criteria	Notes	Pass/Fail
06	The user can see total conversions summary	Statistics for different campaigns are calculated and displayed in the table under the graph. When the campaign is changed, all the statistics are recalculated to match with the data of the new campaign. The results of the queries are tested on a smaller dataset for correctness. Some of the queries are tested using *awk*.	
07	The user can see total cost statistic		
08	The user can see cost per acquisition		
09	The user can see cost per click statistic		
10	The user can see cost per a thousand impressions		
11	The user can see click through rate statistic		
12	The user can see bounce rate values		
13	The user can see number of clicks statistic		
14	The user can see number of uniques statistic		
20	The user can select the bounce definition to be time spent on a webpage	The user can select bounce configuration per line. Query results are updated according to the selected definition and value.	

21	The user can select the bounce definition to be pages viewed		
22	The user can filter results by date range	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. Filter configurations are given as checkbox items to avoid invalid user inputs.	
23	The user can filter results by context		
24	The user can filter results by gender		
25	The user can filter results by age		
26	The user can filter results by income		
31	The user can save a graph	Generated graph can be saved as *.PNG* in a destination folder chosen by the user.	
34	The user can print a graph	Generated graph can be printed if there is a print service available.	

Boundary and Partition Testing

Requirement	Testing	Result
	Upload Campaign	
[Upload Campaign] 2 campaigns cannot have the same name	An error message is shown to the user if they are trying to upload a new campaign and use a name already existing in the workspace. The upload does not progress until the user has set a valid name.	
A campaign name should be of length at least 1	An error message is shown to the user if no Campaign Name is set. The upload proceeds only if there is at least 1 symbol in the name field.	
Paths to logs should be provided for “Advanced” and “Simple” mode	An error message is shown to the user if 1 or more file path inputs are not populated. Campaign cannot be uploaded until paths to all files are provided.	
	Configure Graph	
Graph Identifier must be selected and should be of length 1 or more	An error message is shown if the user tries to create a graph, without specifying an identifier, which is essential when swapping between already created graphs. The application doesn’t generate a graph until the condition is satisfied.	
If there are no lines added, the Remove button is disabled	The “Remove Line” button is disabled until there is at least one created line. Removing line is temporary and the action cannot be undone. Alternatively, a user can temporarily “hide” a line from the graph.	
	Configure Line	
Metric must be selected	A line cannot be generated if the system doesn’t have knowledge of the metric for which to query data. An error message is shown to the user if the metric field is empty. Line is not generated, until this condition is satisfied.	
Interval must be selected	A line cannot be generated if the user hasn’t set an Interval. Available options are *hour, day, week, month, year.* An error message is shown to the user if the interval field is empty. Line is not generated, until this condition is satisfied.	

Filtering		
No initial filters applied	The default state of a line is without any filters applied. Filters are configurable, but not obligatory. A graph can still be generated, without applying any filtering.	
Filters cannot be empty	An error message is shown to the user if one of the filter fields is empty. Filters are not applied until this condition is satisfied.	
Start Date cannot be after End Date	Optional date range filter is available. THe user can set a *Start* and an *End Date*. An error message is shown to the user if the Start Date is before the End Date. The filter is not applied, until this condition is satisfied.	
Bounces		
Bounce definition field appears only for *Number Of Bounces*/*Bounce Rate*	Bounce definition field is available if the selected metric is *Number of Bounces* or *Bounce Rate*. This is an optional field. The default bounce definition is *Time,* and the default value is 0. Line is generated without any user configuration.	
Bounce definition cannot be set below 0	Negative bounce values are not allowed. Once 0 is reached, the down arrow is disabled.	

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 Feedback on Increment 1

3.1 - Personalised Feedback

Application on Different OSs

One piece of feedback that we received was to be able to show the client the application running on different operating systems. This is due to the fact that the appearance may be altered slightly based on the OS. Based on this feedback, we have created a new user story (User Story 35) which is mentioned in section **1.3 Amendments**.
Screenshots of the application running on different OSs can be seen in the **Appendices.**

Include Test Outputs

Part of the feedback we received for the previous increment was that the client would like to see various test outputs to gain better insight on the types of testing we have performed. We have now included this in section **2.2 Testing**.

Changing Graph Colours

Another piece of feedback we were given was that the default colour of the line was not appropriate and it would be nice to allow the user to change the line colours. This option is now available to the user when adding a new line to the graph. In addition to colour preference, the user can select thickness and line identifier.

The background of a graph can also be changed to avoid any contrast issues with user's preferences for line colour.

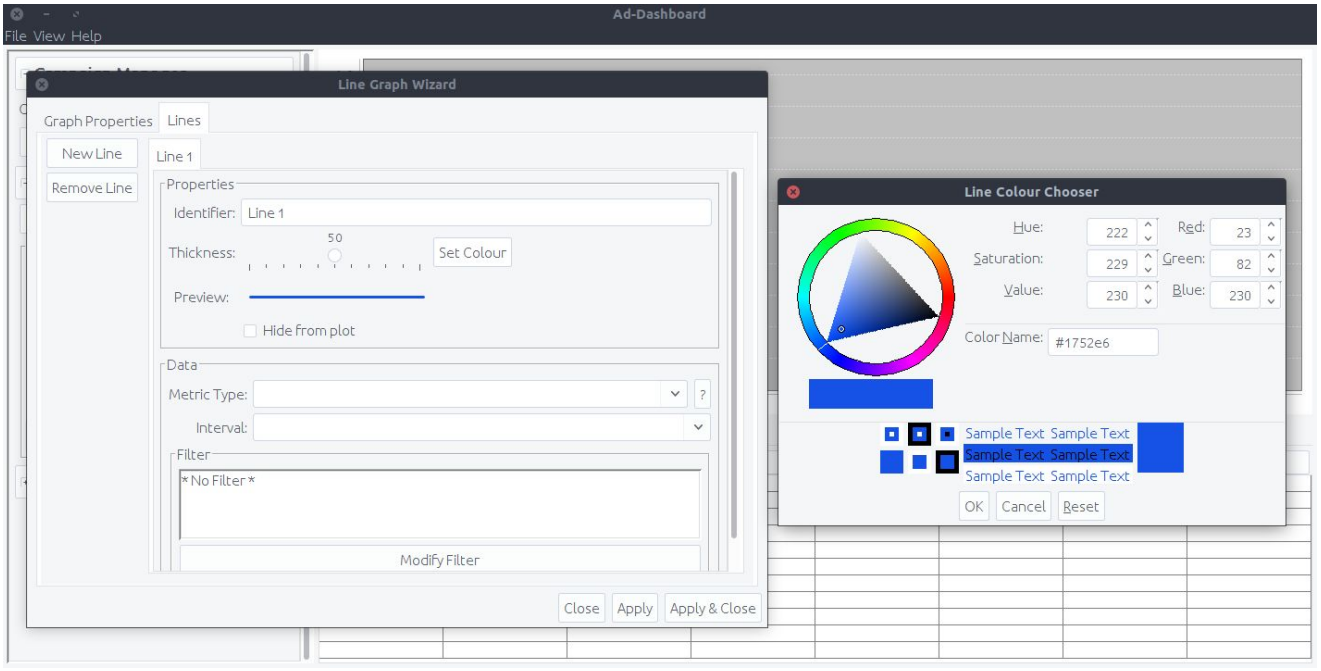


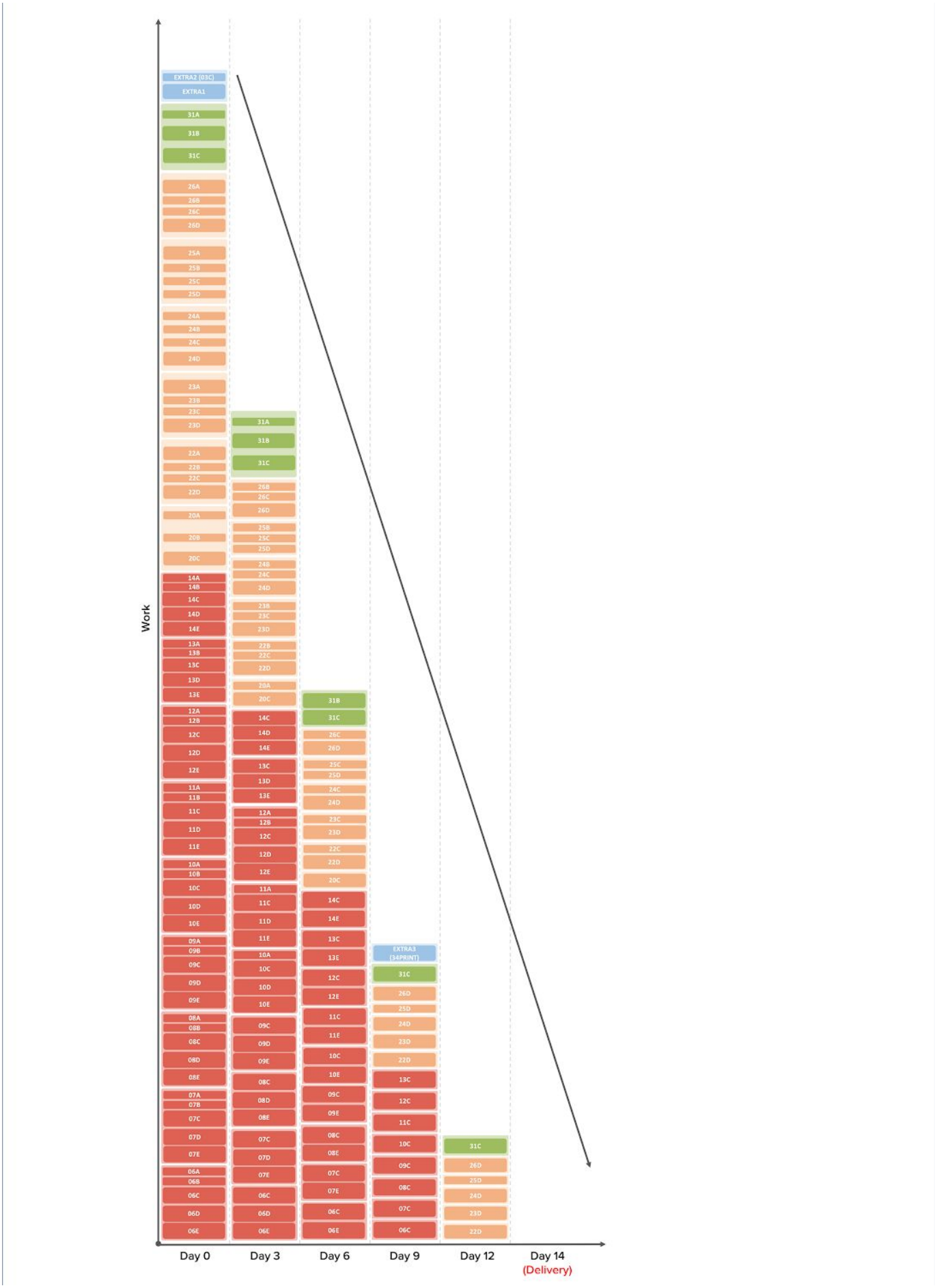
Figure 13: Line Colour Selection

Misconstrued Information

Whilst envisioning we were asked how we would ensure users, who are new to the system and are unfamiliar with the terminology used did not misconstrue information provided. To address this problem we have now implemented an inbuilt definition viewer ('Definitions Dialog') alongside help tooltips that appear next to metric types.

4 - Planning

4.1 - Sprint 2 Burndown Chart (Figure 14)

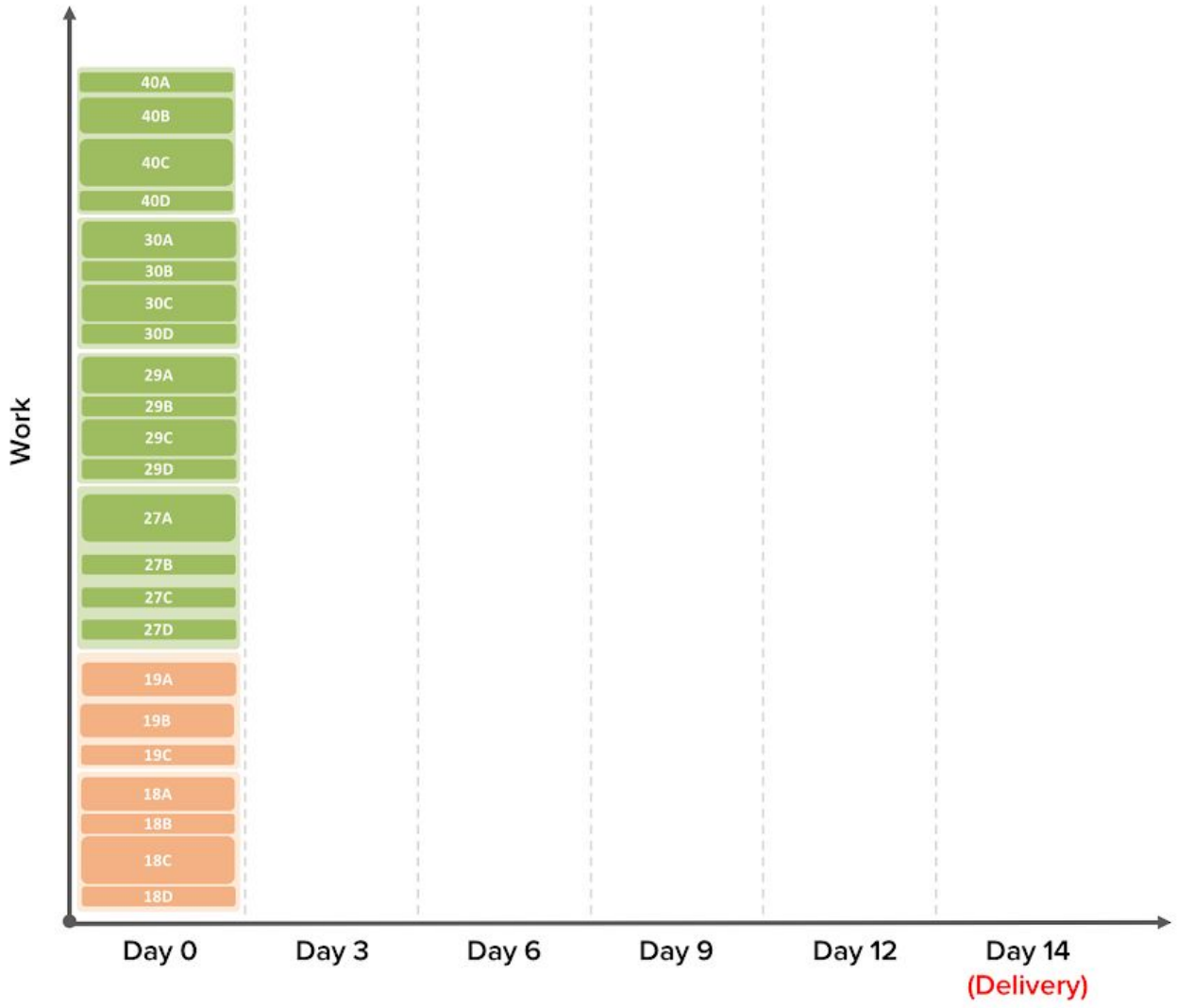


Task 03C - Initial indexes
Task 03C was related to creating indexes on the tables to speed up the queries. It was part of our backlog for Spring 1 when we only implemented *Number of Impressions* statistic. We concluded that the queries were fast enough without having any indexes and decided to leave the task for Sprint 2.
While writing the queries for Sprint 2, it was easier to identify which rows from the tables needed to be indexed, so that queries are fast enough to not cause long delays in response time.

Definition of Done
The **DoD** we have formulated is as follows:

- All tasks from the Sprint are completed
- All unit tests pass
- Regression tests are performed
- Acceptance criteria is met
- Functional and Non-functional requirements are met
- Code is reviewed by team members
- Application is programmed with thought of future sprints to enable easier further work
- Client is satisfied with the development of the project

4.1 - Sprint 3 Burndown Chart (Day 0) (Figure 15)



4.2 - Sprint 3 Backlog

In the initial plan for increment 3, we included task **PbID32**, which was adding colour changing functionality to enhance application accessibility. However, after a conversation with our client, we came to an agreement not to include this feature. The decisions behind this included: 3rd party application solutions would be more flexible and that the mandatory GUI framework is not designed to permit this so solutions would likely reduce future compatibility. Instead, we added a new User Story (**ID 40**) that allows the user to change workspaces, which would make the program more user-friendly.

40	As a <Small Marketing Agency Owner> I want <my client to be able to change between workspaces> So that <the application is more flexible and easier to use>

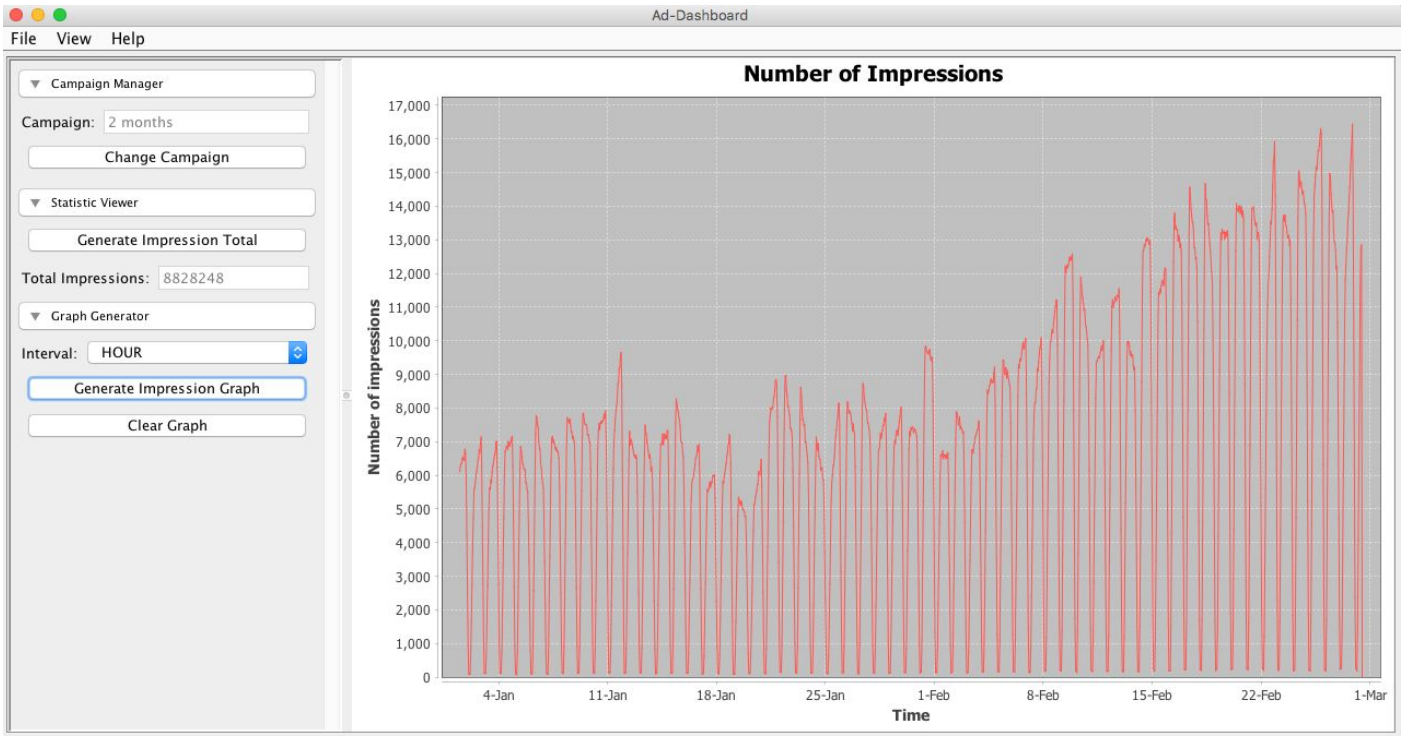
The value we are aiming to deliver in Sprint 3 is fully functioning and user-friendly dashboard that complies with the client's requirements.

Sprint Backlog				
PbID	TID	Task	Size	Dependencies
18 [Click costs histogram]	18A	Extend GUI to support click costs histogram	M	**-**
	18B	Add query for click cost histogram	S	**-**
	18C	Update graphs to support click cost histogram	L	18B
	18D	Test the implementation	S	18A, 18B, 18C
19 [Compare audience segments]	19A	Extend GUI to support comparison between segments	M	**-**
	19B	Update graphs to support segments comparison	M	-
	19C	Test the implementation	S	19A, 19B
27 [Compare different campaigns]	27A	Extend GUI to support campaign comparison	L	**-**
	27B	Update queries to support different campaignID	S	-
	27C	Update graphs to support campaign comparison	S	27B
	27D	Test the implementation	S	27A, 27B, 27C
29 [Time of day statistics]	29A	Extend GUI to support time of day selection	M	-
	29B	Update queries to support specific time of day queries	S	-
	29C	Update graphs to support time of day statistics	M	29B
	29D	Test the implementation	S	29A, 29B, 29C
30 [Specific day statistics]	30A	Extend GUI to support specific day selection	M	-
	30B	Update queries to support specific day queries	S	-
	30C	Update graphs to support specific day statistics	M	30B
	30D	Test the implementation	S	30A, 30B, 30C
40 [Workspace change option]	40A	Link all persistent resources to workspace	S	-
	40B	Create workspace storage method	M	-
	40C	Extend GUI to support workspace change option	L	-
	40D	Test the implementation	S	40A, 40B, 40C

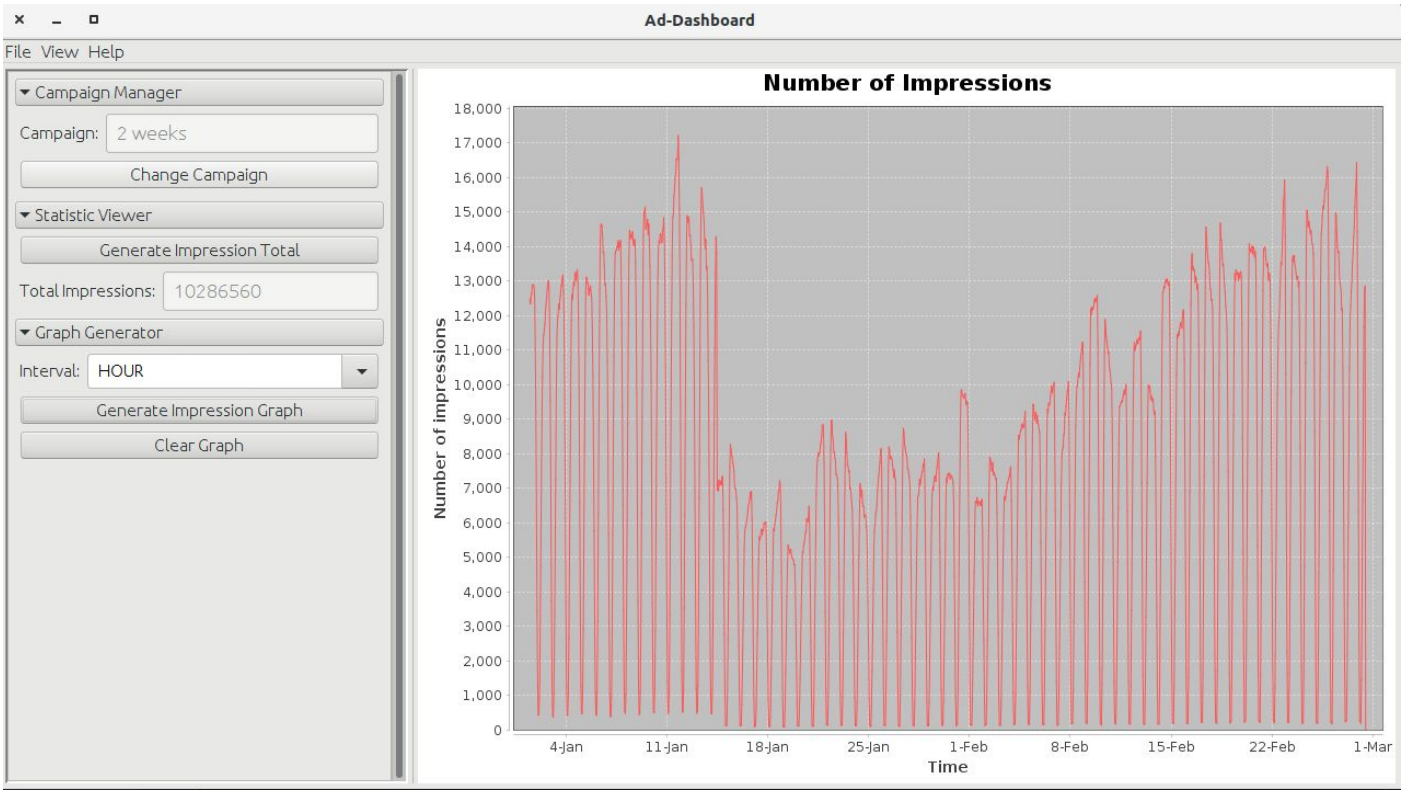
Appendices

Sprint 1 Application UI

MacOS

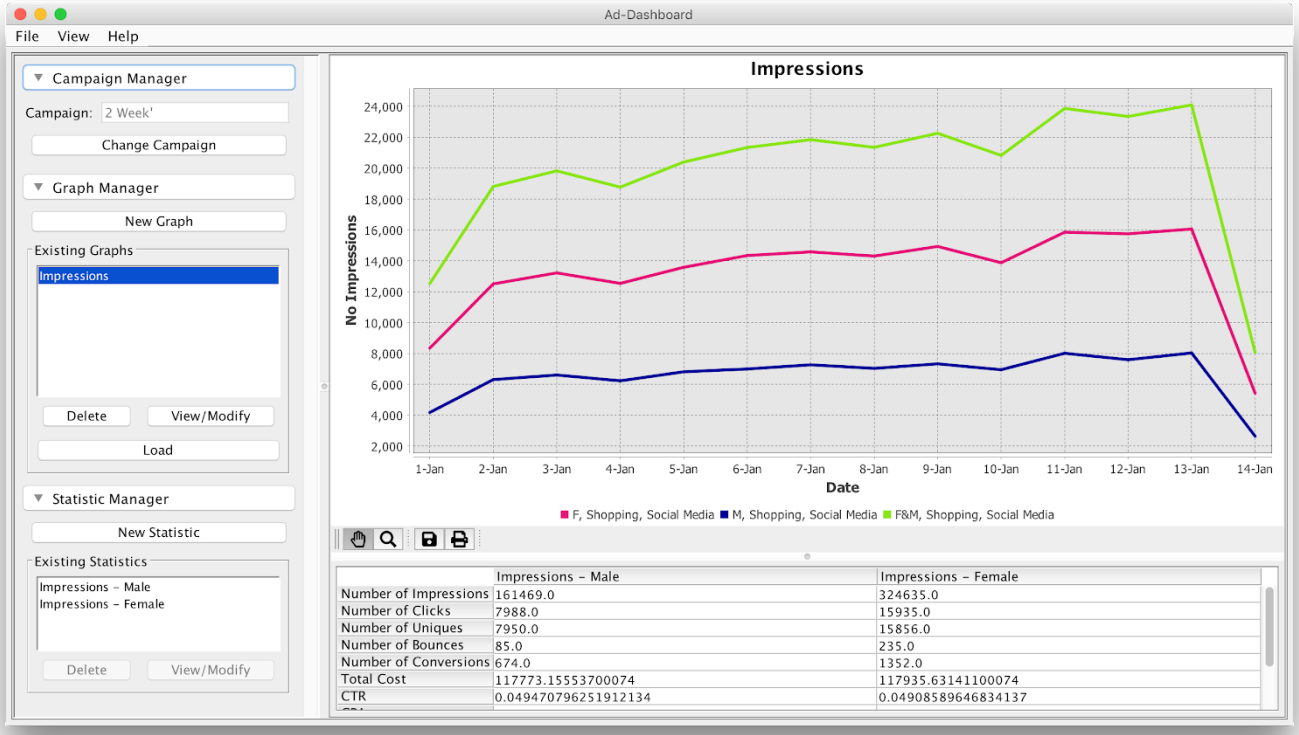


Ubuntu

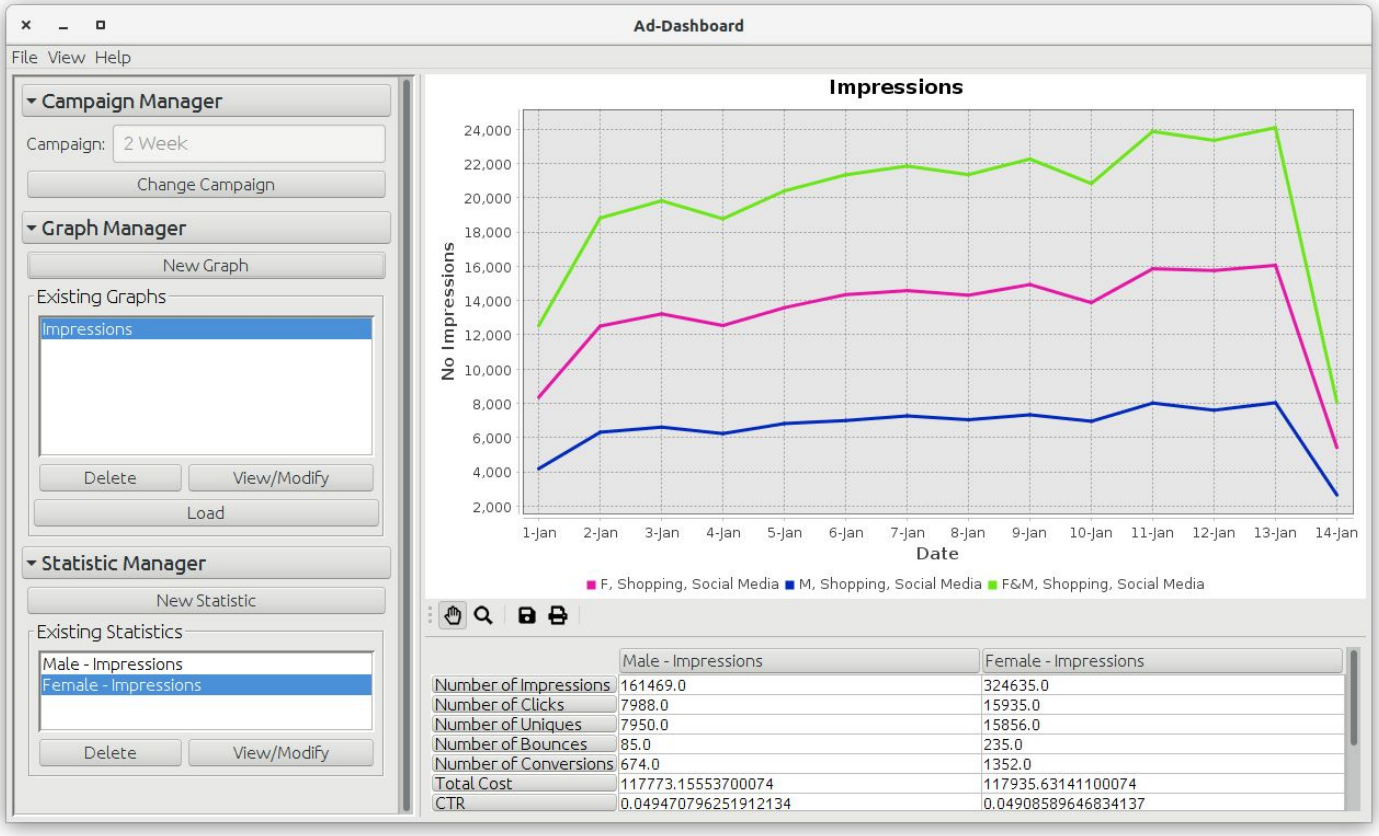


Sprint 2 Application UI

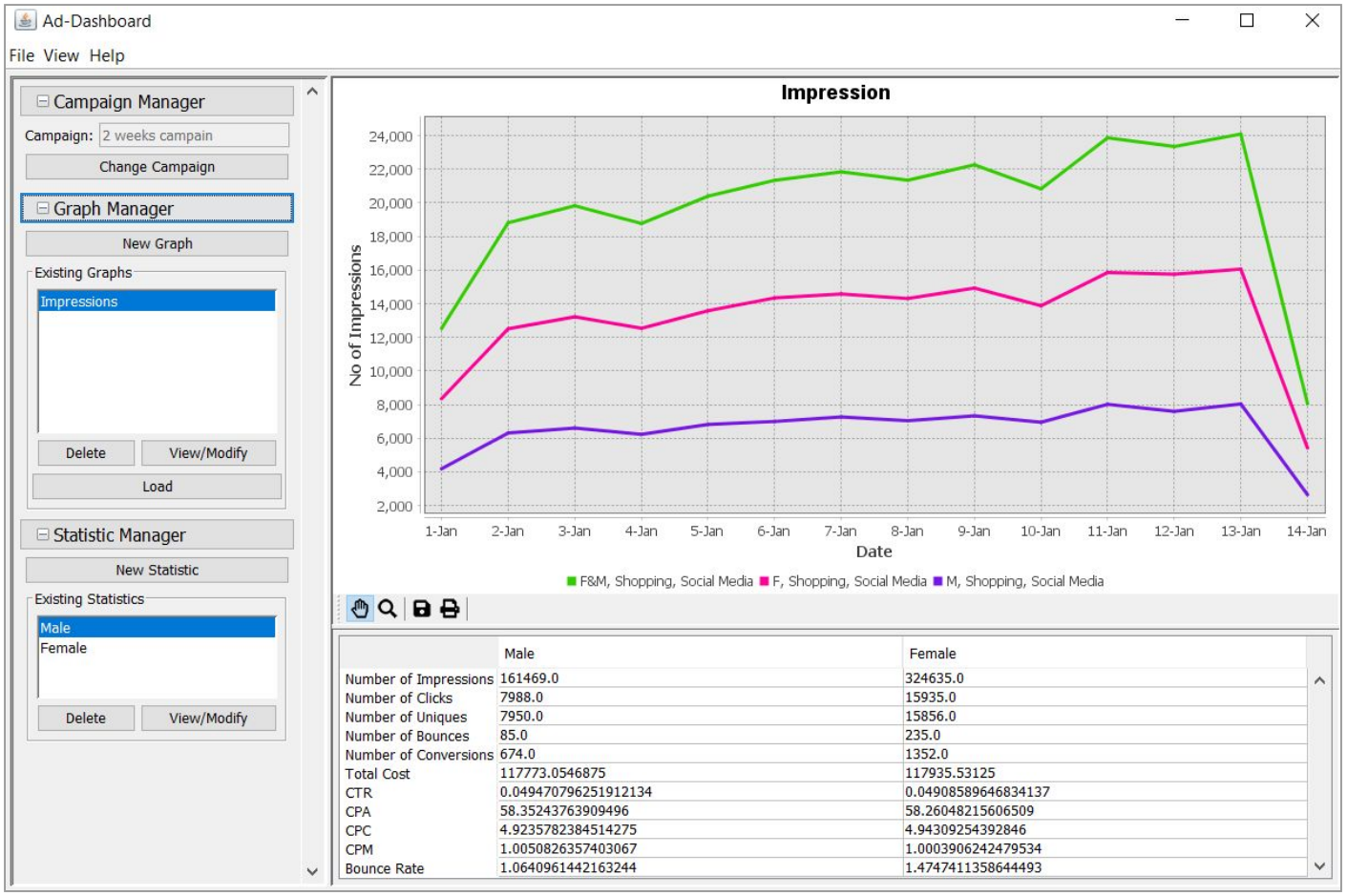
MacOS



Ubuntu



Windows



	Male	Female
Number of Impressions	161469.0	324635.0
Number of Clicks	7988.0	15935.0
Number of Uniques	7950.0	15856.0
Number of Bounces	85.0	235.0
Number of Conversions	674.0	1352.0
Total Cost	117773.0546875	117935.53125
CTR	0.049470796251912134	0.04908589646834137
CPA	58.35243763909496	58.26048215606509
CPC	4.9235782384514275	4.94309254392846
CPM	1.0050826357403067	1.0003906242479534
Bounce Rate	1.0640961442163244	1.4747411358644493

Notes

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