Software engineering group project

Deliverable 3 - Increment 2

	Name	ID
1	Marwan Altamimi	32489005
2	Juzheng Bai	32410964
3	Henry York	33113556
4	Jack Breslin	33055521
5	Aditya Bansal	33335052

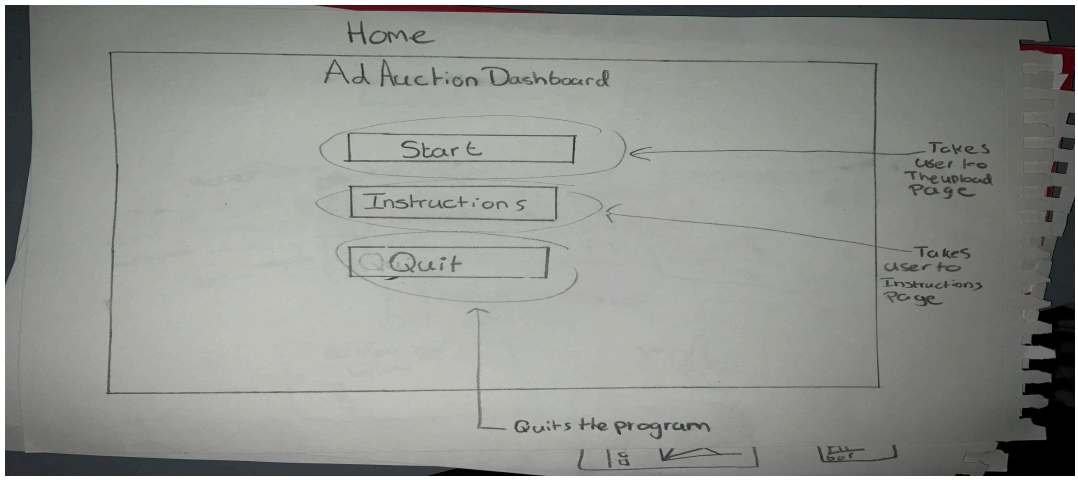
Supervised by
Gerasim Tsonev

Project Design

Storyboards

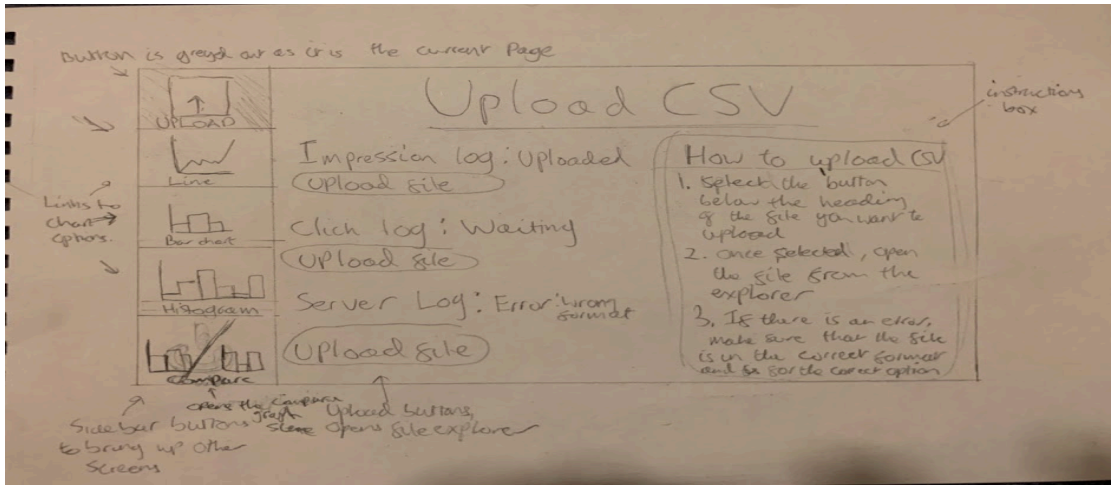
We have made some changes to our storyboards from last increment, we will show the new storyboards and explain what the changes are, and why we made them.

Figure 1 - Shows the Home screen of the application, we haven't made any changes to this screen, so it is the same as it was in the last increment.



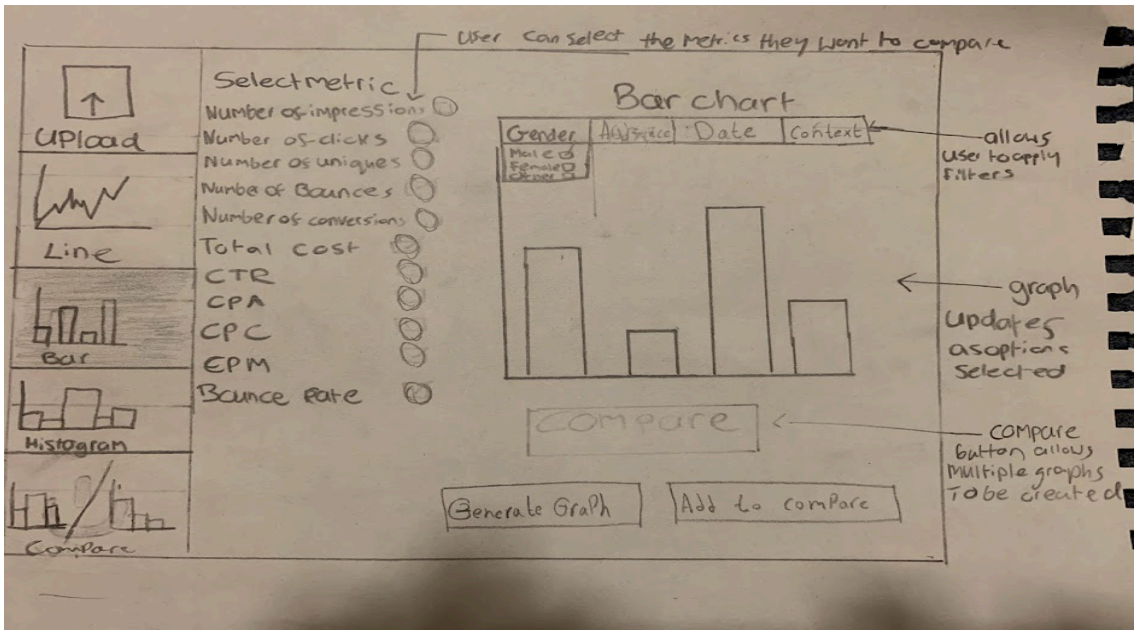
(Figure1)

Figure 2 - Shows Our upload page, the only change we made to this page was to remove the quit button from the sidebar, and instead replace it with a button that allows the user to navigate to the compare page. We made this change because after the addition of the compare page we needed to add a way for the user to navigate to it. Due to us originally planning to have a sidebar it was only natural that we decided to add a compare button to the sidebar. After further review we decided to replace the compare button with the quit button, instead of just adding an additional button, because we found it to be redundant due to having a quit button on the homepage, and the default exit button for the application is provided by the window.



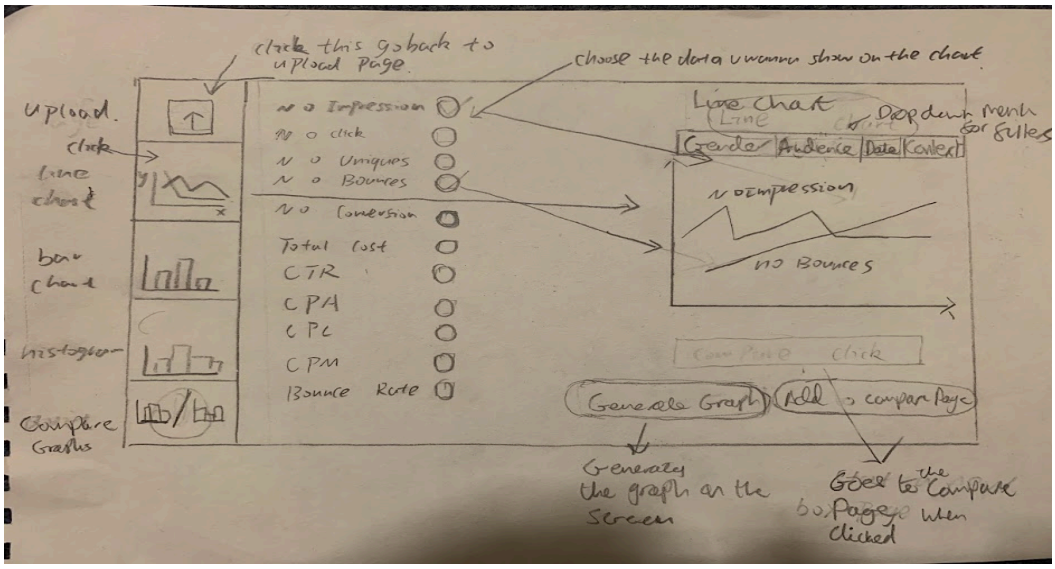
(Figure2)

Figure 3 - Shows the changes that have been made to the barchart page. This page has the same changes made to the sidebar as the other pages, as well as changing from selection tick boxes to radio buttons. This is because the original plan was to be able to display multiple metrics on the same graph, however after implementation, the range of values per metric was quite large with impressions being upwards of forty thousand and other metrics being way below one hundred. This greatly hurt the graph's readability, when metrics with vastly different values were present on the same graph. It also would indirectly cause the user to have to reduce the date range of the graph as by having two metrics present on one graph would double the amount of bars compared to one metric. We also thought it was unnecessary to include multiple metrics on one graph as when it comes to graph comparison the user is more likely to want to compare the same metrics but with different filters applied, and/or date ranges. In addition to this change we have also changed the compare button to be more intuitive by displaying “add to compare page” rather than just compare. We also added a “generate graph button” which will apply the selected filters and metric when it is pressed. We have also removed the filter page that appears when the user clicks the filters button, and instead added a filter bar above the graph which has a drop down menu to apply the filters they want on their graphs, except for the date range filter this instead drops down a calendar for the user where they can instead select a start and end date for the filter, or alternatively the dates can be typed in in the format DD-MM-YYYY , we made this change to make the application more seamless and reduce the amount of screen changes for the user.



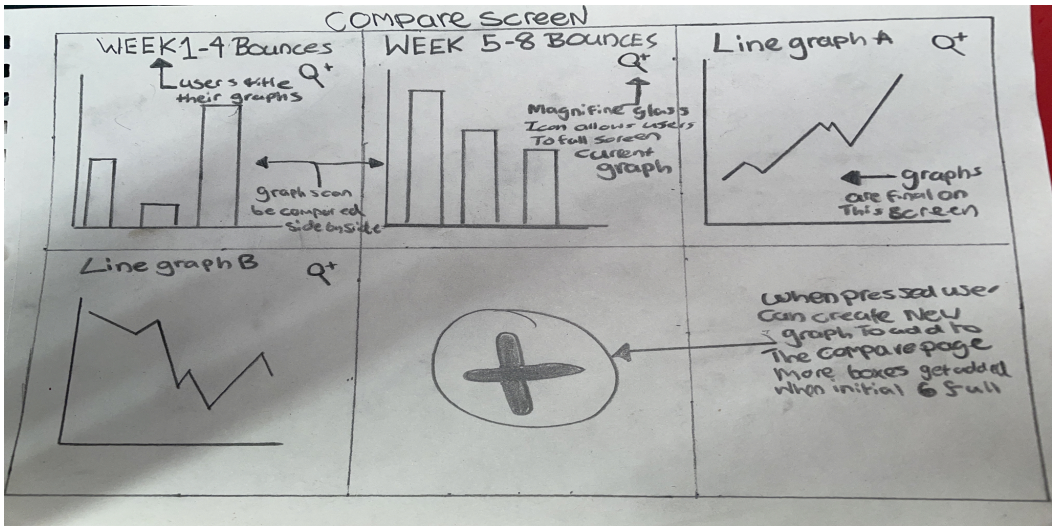
(Figure3)

Figure 4 - The line chart page has the same changes that we made to the barchart page with the same reasoning.



(Figure4)

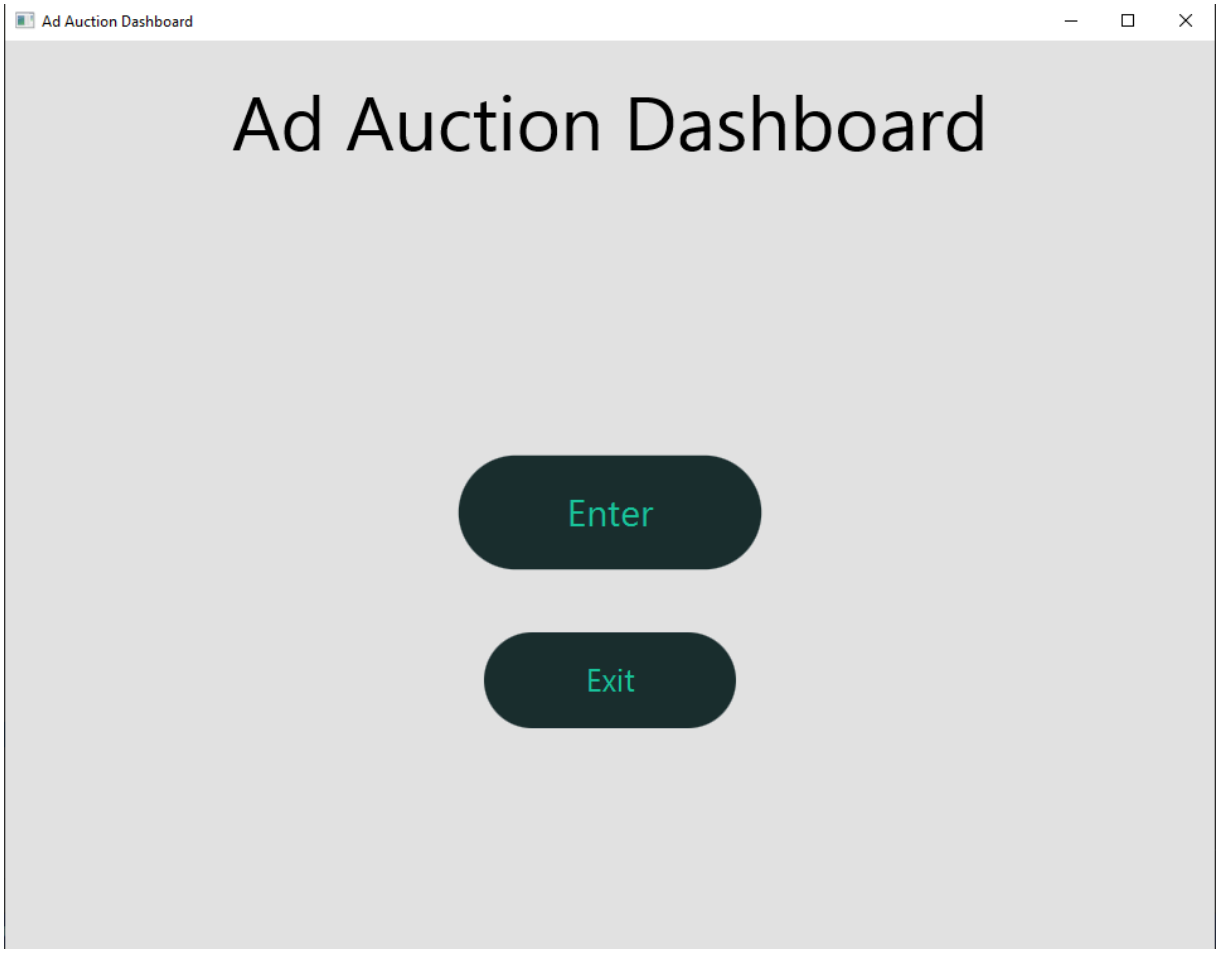
Figure 5 - Shows our new compare screen. On this screen the user can see their saved graphs, This allows the users to make side by side comparisons of many graphs showing different metrics, different time date ranges, different time intervals, context, and audience segments. The user can also click the button on each graph which causes that particular graph to be maximised so that it fills the application window, this allows for the graphs to be analysed further, as it improves readability. The user can also name the graphs, this will aid analysis because then the user will know what the day range, and filters are on the graph. We made this change due to the feedback we received from our customers.
Originally the way we decided to compare graphs was to make them “pop-out” in a separate window, instead of having a dedicated compare page, our customer informed us it would be better to not make any additional windows, and instead try to display the graphs within one window.



(Figure5)

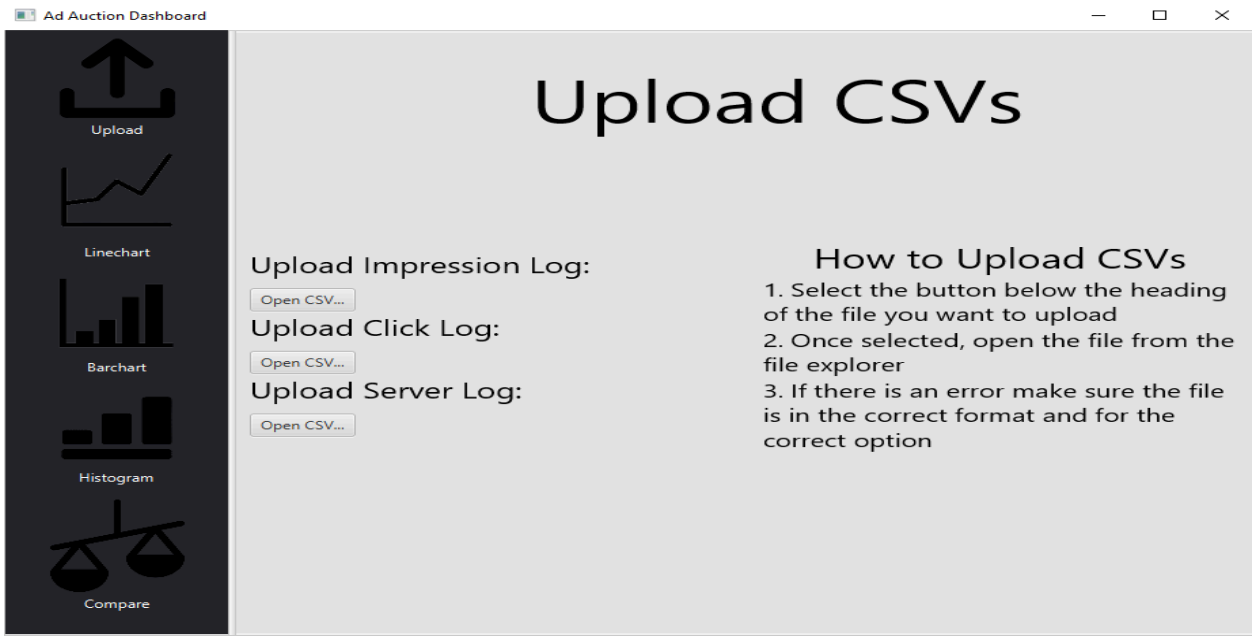
Storyboard Comparisons

Main Menu



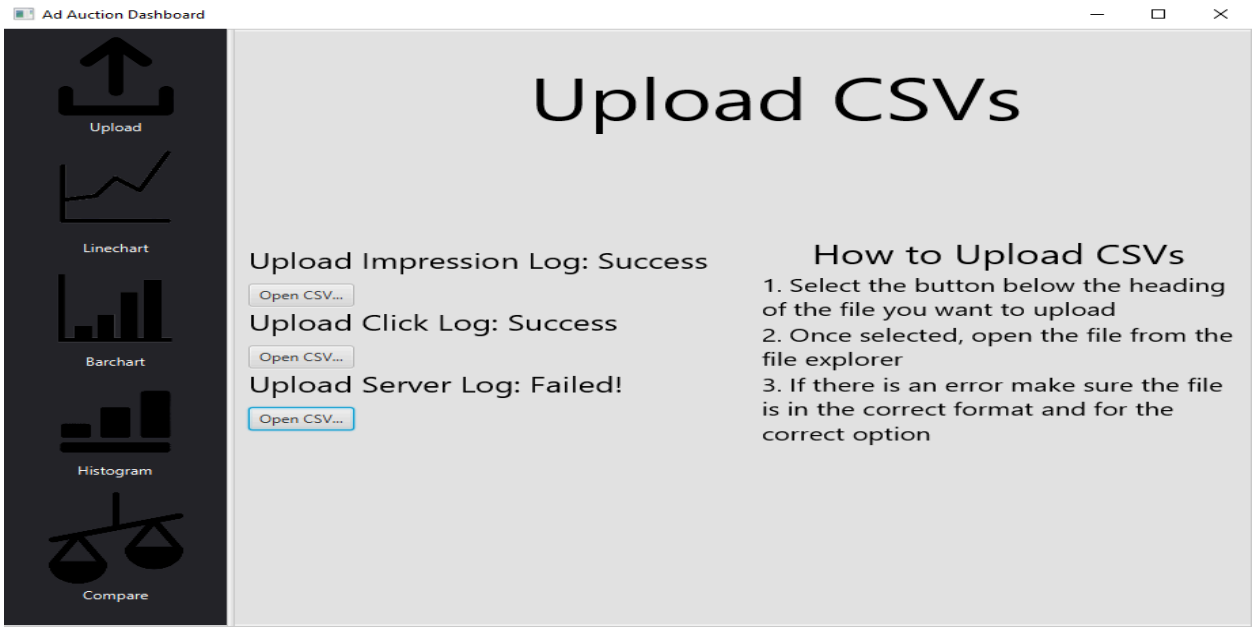
Similar in design to our storyboard for the main menu, however there is no instructions button as we believe that later in development we can add an instruction page for user guidance once the system is finished. Additionally, from our previous GUI designs we have changed the background to a grey gradient and darker buttons which we think looks more suitable for our system.

Upload Screen



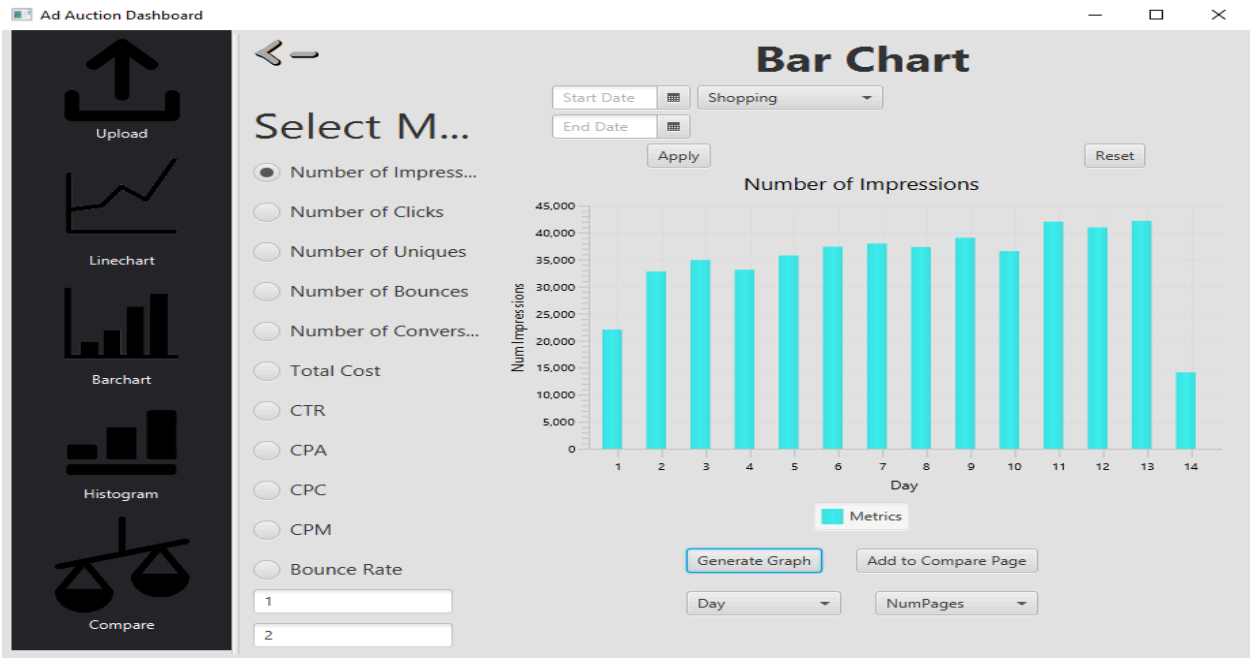
The upload page is also similar to our storyboarded version, the only difference is that the instructions are not in a box which could make the GUI look nicer if implemented. However we will focus more on improving the GUIs design in our third increment.

Upload Screen with uploaded files



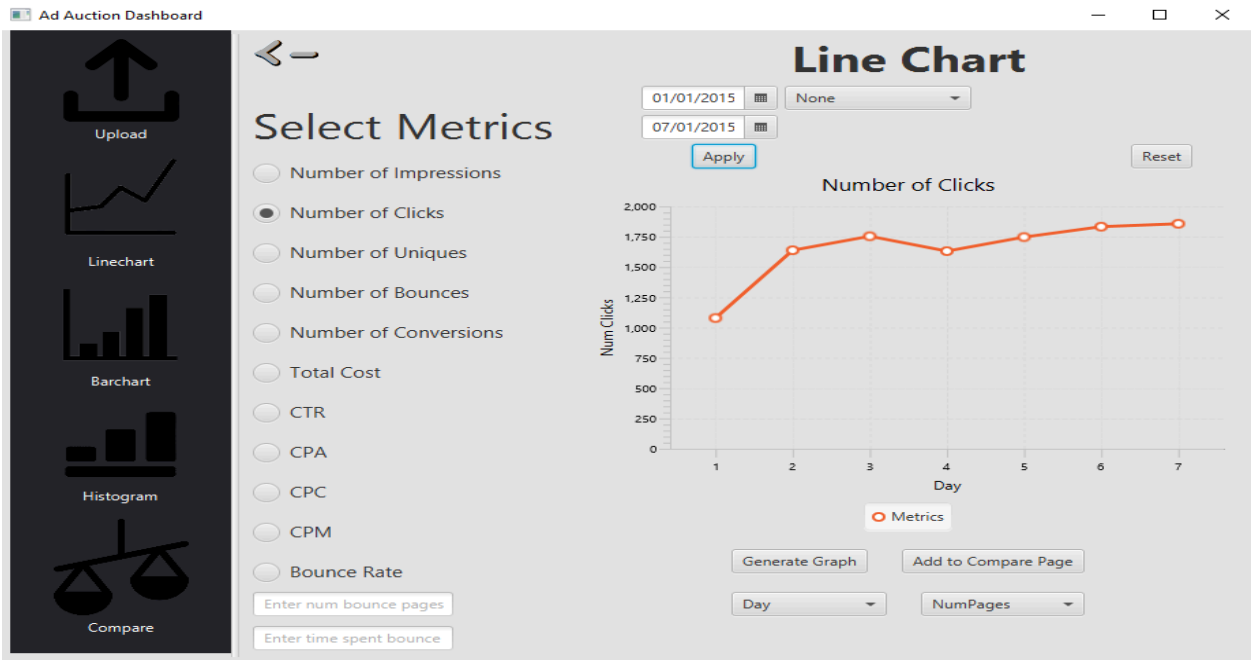
The error messages have been simplified but still give enough information for the user to understand, along with the instructions to let them know what they can do.

Bar chart Screen



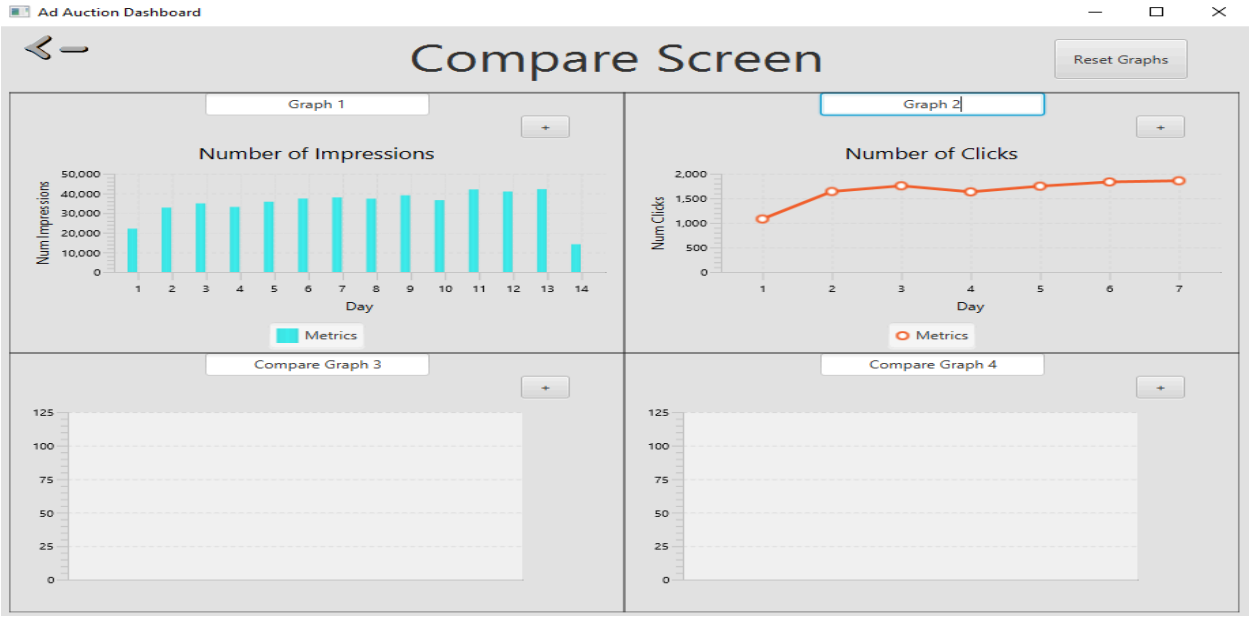
Similar to our storyboard but we have added text fields to define bounces and drop down boxes for defining the time granularity and the type of bounce. Additionally there are apply and reset buttons to make it easier to use the filters and apply them to the graph. The filters have been modified to not be a single bar at the top of the graph but rather a small section above it

Line Chart Screen



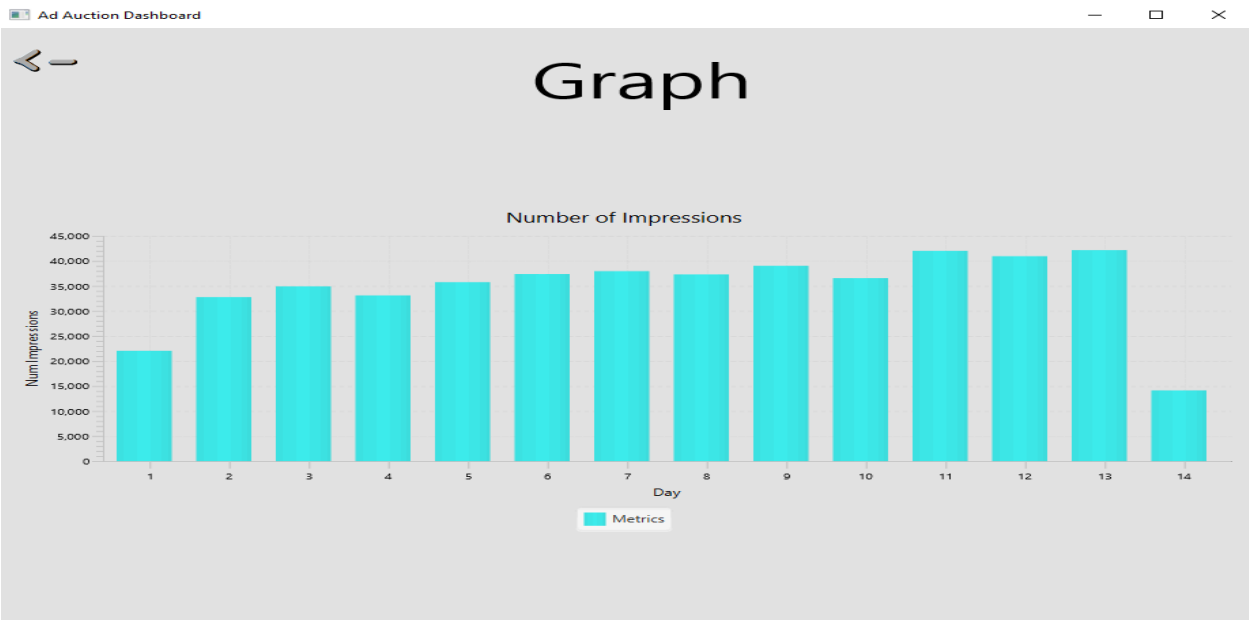
Changes made are similar to the bar graph page.

Compare Screen



The most notable difference here is the change of having 6 graph slots to compare in the storyboards while we have implemented four. We did this as we think any more than 4 will make the graphs hard to read and clutter the screen. Instead of a slot to add more graphs, placeholder graphs are simply put in the grid, and the user can navigate to add more graphs by pressing the back button. Additionally we added a reset graphs button so the user can remove the currently compared graphs and add new ones as they generate a new set.

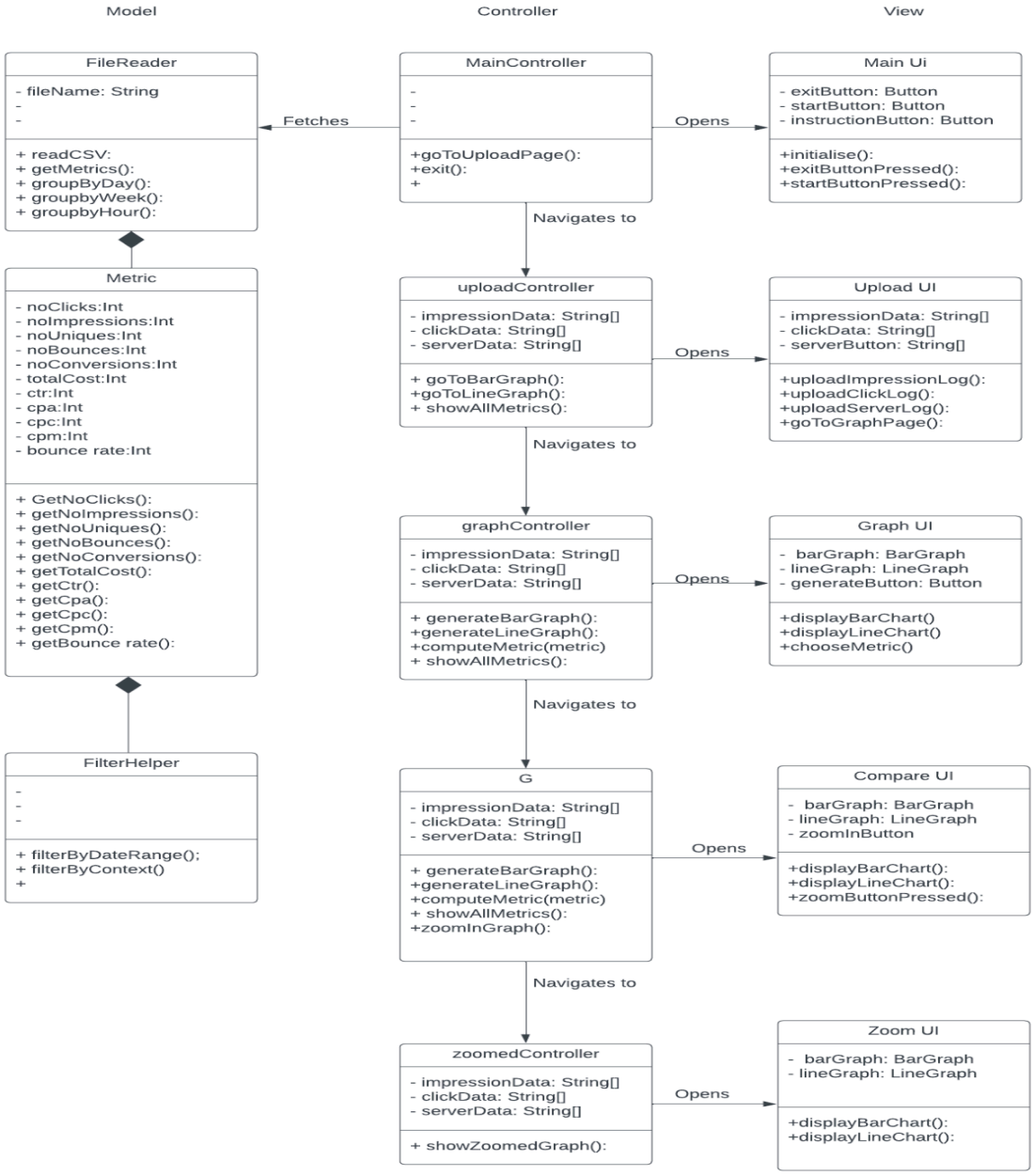
Zoom In Graph



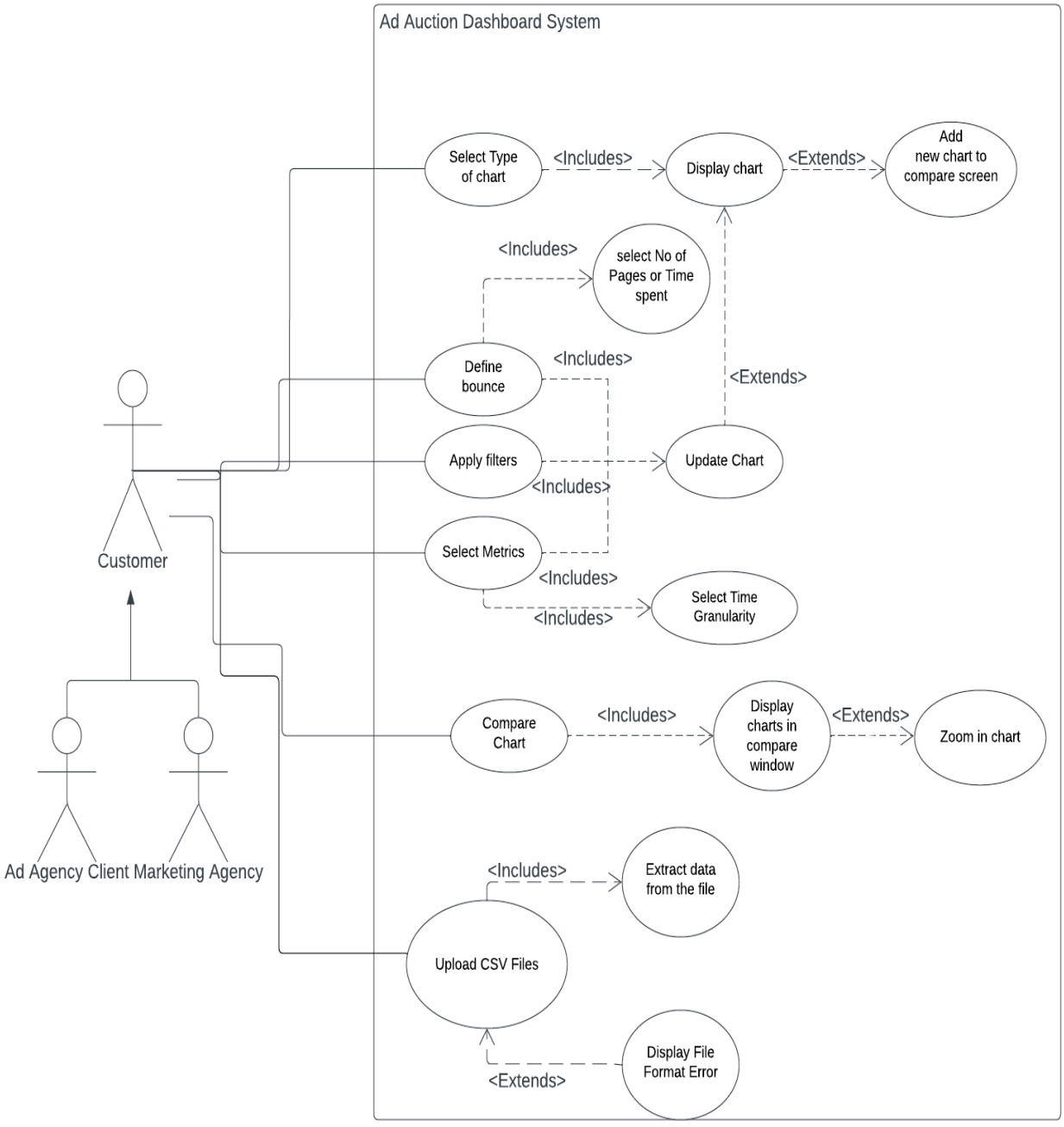
The zoomed in graph from the compare page allows one to see one of the graphs in more detail. This was not shown in the storyboards.

UML Diagrams

Class Diagram - **Changed slightly to support our change in design**



Use Case Diagram - Changed slightly to support our change in design



Scenarios -**These are new scenarios to support our design choices for this increment**

Scenario 1: Ad agency Client - Brian

1. Brian opens the application and presses the start button
2. He selects his logs from the latest ad campaign and uploads them to the system on the upload screen
3. He navigates to the bar chart and selects the number of impressions metric
4. He selects two date range filters to only include data from the first 4 days and clicks apply.
5. A bar graph is displayed showing 4 bars for each day
6. He then wishes to compare the graph with data from other weeks and adds the graph to the compare page.
7. He titles the original graph “Data for 1-4 days” and then he clicks the back icon on the compare page and adds a new graph for data from 5-9 days.
8. He compares the data from both of these graphs and then notes down some conclusions he deduces from them.

Scenario 2: Marketing Agency - Joyce

1. Joyce starts the system and navigates through the home screen to the upload screen.
2. She uploads three CSV files, The impressions, clicks and server log.
3. She navigates to the line chart screen and selects the number of bounces metric.
4. She defines what her client wishes to classify as a bounce for their use case, which is the number of pages.
5. She inputs the bounce limit as 2 pages
6. She presses generate graph
7. The application then updates the charts and shows her an updated chart with bounce rates for the selected time period based on her definition of a bounce.

Scenario 3: Software developer

1. Marwan is alerted of a bug in the application by a client when they try to update their bounce rates classifications.
2. He tries to replicate the bug in his development application.
3. After locating the bug, he spends some time fixing it.
4. Once he is sure it is fixed and there are no side effects, he deploys the new code to the live application and tells the client it is fixed.

Scenario 4: Ad agency Client - Brian

1. Brian opens the application and presses the start button
2. He selects his logs from the latest ad campaign and uploads them to the system on the upload screen
3. He navigates to the bar chart and selects the number of bounces metric and chooses to view the data over weekly increments.
4. He defines the bounce classifications based on his needs (he chooses to use time spent as he believes it is more accurate for his use case)
5. He presses the generate graph button and the graph appears on the display.
6. He notes down the data from the graph and any conclusions he comes to.

Response to Feedback

Following our feedback from our customer, we have implemented various changes to our document and program.

Application Feedback

1. Firstly, we followed the advice of changing our method of comparing our graphs to have its own individual compare screen, where instead of having the graphs show up in multiple different windows which would make the screen cluttered, we have a single screen with multiple graphs together, which made it easier for the user to compare the graphs. Additionally, we took the advice of adding a zoom in feature in the compare screen to individually zoom in on the graphs.
2. Since our customer found the program difficult to navigate through the line graph and bar graph buttons on the upload page, we added a sidebar which contains all the buttons in an easy to spot place and on each key screen so the user can navigate more easily to these screens.

Design Feedback

1. We also added small descriptions below our GUI screenshots to show any differing design choices from our storyboards which allowed us to see the contrast between our original idea and final design.

Testing feedback

1. We added JUnit tests for our code created in this increment and the previous increment to allow a more structural way of testing.

2. For the next increment we will also put integration testing to allow more test methods to our code.

3. We also added scenario testing by running through each step of our scenarios and checking that the steps can be followed in real usage of the program.

Planning Feedback

1. We added reasoning to any changes to our original plan if needed, such as when changing the increment plan or the sprint backlog.

2. Additionally, we added a diagonal line to our second burndown chart to show the trend in our progress, and added actual days instead of an index of days to make it more readable and comprehensible.

Project Testing

Scenario Testing

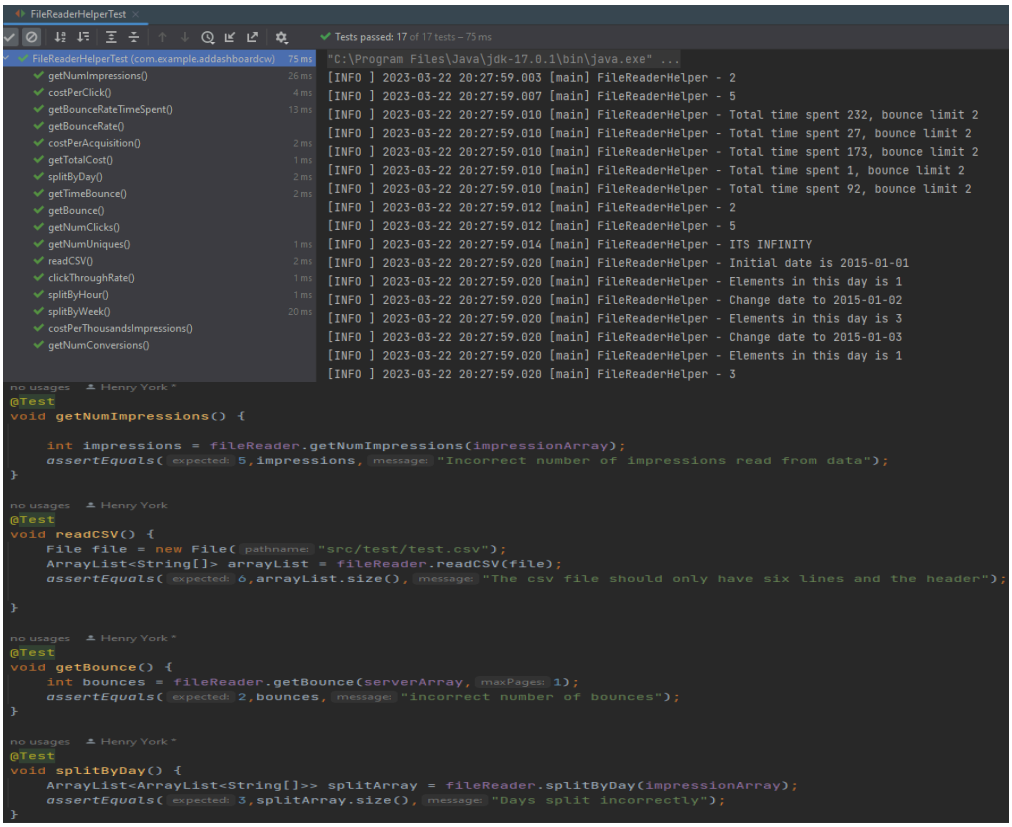
For our scenario testing we ran through each step of our scenarios and simulated it in the actual program, ticking off the steps as we went along.

Scenario ID	Steps Matched
1	1. Brian opens the application and presses the start button ✓ 2. He selects his logs from the latest ad campaign and uploads them to the system on the upload screen ✓ 3. He navigates to the bar chart and selects the number of impressions metric ✓ 4. He selects two date range filters to only include data from the first 4 days and clicks apply ✓ . 5. A bar graph is displayed showing 4 bars for each day ✓ 6. He then wishes to compare the graph with data from other weeks and adds the graph to the compare page. ✓ 7. He titles the original graph “Data for 1-4 days” and then he clicks the back icon on the compare page and adds a new graph for data from 5-9 days. ✓ 8. He compares the data from both of these graphs and then notes down some conclusions he deduces from them. ✓

2	1. Joyce starts the system and navigates through the home screen to the upload screen. ✓ She uploads three CSV files, The impressions, clicks and server log. ✓ 2. She navigates to the line chart screen and selects the number of bounces metric. ✓ 3. She defines what her client wishes to classify as a bounce for their use case, which is the number of pages. ✓ 4. She inputs the bounce limit as 2 pages ✓ 5. She presses generate graph ✓ 6. The application then updates the charts and shows her an updated chart with bounce rates for the selected time period based on her definition of a bounce. ✓
3	1. Marwan is alerted of a bug in the application by a client when they try to update their bounce rates classifications. ✓ 2. He tries to replicate the bug in his development application. ✓ 3. After locating the bug, he spends some time fixing it. ✓ 4. Once he is sure it is fixed and there are no side effects, he deploys the new code to the live application and tells the client it is fixed. ✓
4	1. Brian opens the application and presses the start button ✓ 2. He selects his logs from the latest ad campaign and uploads them to the system on the upload screen ✓ 3. He navigates to the bar chart and selects the number of bounces metric and chooses to view the data over weekly increments. ✓ 4. He defines the bounce classifications based on his needs (he chooses to use time spent as he believes it is more accurate for his use case) ✓ 5. He presses the generate graph button and the graph appears on the display. ✓ 6. He notes down the data from the graph and any conclusions he comes to. ✓

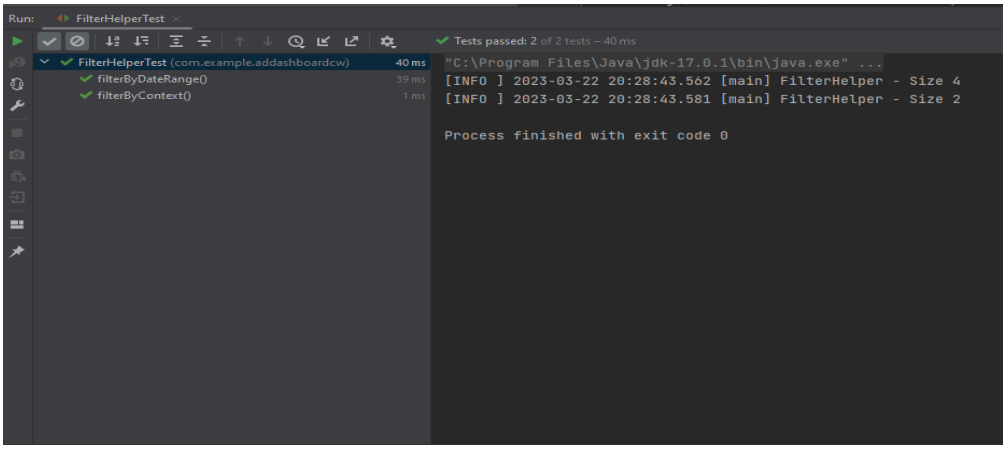
Unit Testing

In conjunction to scenario testing we have also produced some unit tests for our application. To do this we have used JUnit. This allowed us to test our program's logic. An example of this logic is, we needed to test the file reading, and the grouping of the data. For example, after reading the file every record each field has to be added to their respective ArrayLists which then need to be made into sublists for weeks, days or hours. JUnit allows us to make sure that the logic of our program works for different inputs.



File Reader test

File Reader test code



Filter Helper Test

User Story Acceptance Testing

Tests code: Green - Passed, Yellow - Ongoing, Red - Failed

User Story ID	Test Criteria for User story	Expected Outcome	Actual Outcome
2	Can charts be made of different time intervals?	The user needs to be able to switch the x-axis of the barcharts to see changes over longer or shorter periods of time.	The user can change the x-axis intervals to be hours, days or weeks
9	Can bounces be defined?	The users must define bounces before creating a graph.	If bounces are not defined an alert is presented to the user informing that they need to define them before creating a bounce graph.
5	Can the minimum values for a bounce be set?	When bounces are calculated the user can change what the minimum values are for the calculation.	The user has two fields present which allows them to input the minimum value for the number of pages and the time spent.
10	Can graphs be filtered by a date range?	The graphs created can be filtered by a date range so particular entries in the csv files spanning a certain time can be analysed easier.	When a graph is generated the user has the option to select a start and end date they want the graph to span.
8	Can graphs be filtered by context?	The data uploaded can be filtered so that the graphs created only represent certain context groups	Above the graph the user has a drop down box which allows them to select various contexts from News, Shopping, Social, Media, Blog, Hobbies, Travel
12	Is there a sidebar for navigation?	The user needs to be able to navigate the application seamlessly and with ease to make the application feel more professional.	A sidebar was created which allows the user to navigate between the important pages of the application, Upload page, bar chart, line chart, Histogram, and the compare page.
13	Can the barchart's time period be switched?	The graphs need to have options for multiple time intervals so data can be analysed at a finer detail, and at a larger scale.	When a graph is made the user can change the time interval from hourly, daily, and weekly. So the data can be analysed as fit
14	Can a user create a report comparing two graphs?	The users need to be able to see two graphs at once so that they can create reports on them	Once a user has made a graph they can make as many other graphs they would like and final graphs can be added to the compare page.
21	Can multiple graphs be seen side by side?	The users would like to compare multiple graphs side by side to see, while still being readable.	The compare page can have a maximum of four graphs can be added to the compare page which can all be named, so the user can tell them apart easier. The user can also full screen each graph to increase readability.

Project Planning

Time estimation per increment

Increment	XS	S	M	L	XL
1	2	3	2	1	N/A
2	N/A	1	3	2	N/A
3	2	2	4	1	N/A

Increment Plan

Colour coded to match the priority of the user story, number represents the ID and the symbol represents the time estimate.
We have shortened down each user story to its main goal.
We revised our increment 3 to make user story 18 a MUST priority instead of a SHOULD priority as we have found it is a mandatory requirement for the system and must be completed to give value to the customer.

Increment 1	Increment 2	Increment 3
3-M: ad campaign data	2-L: charts of data over different time intervals	6-M: histogram relating to the click costs of the campaign
7-S: key metric from the campaign	9-S: option to define bounce	18-M: filter audience segments by gender, age range, or income etc
4-M: key metrics of my campaign displayed in system	5-S: set the minimum number of pages or time spent for a bounce	17-S: graphs updated quickly with new data
11-XS: view the timeframe of my metrics	10-XS: filter graphs in the system by date ranges	19-XS: system to scale size
1-L: chart representation of the data	8-S: filter graphs in the system by ad context	20-L: could customise the application
15-S: code efficient in reading large data	12-XS: sidebar with different types of graphs	23-S: colour themes for people with colorblindness
16-S: display error messages for incompatible files or irrelevant data	13-M: switch the time periods of the barcharts	22-M apply multiple audience segments filters at once
	14-S: comparison reports comparing two graphs	24-XS: charts to be stylised.
	21-M: overlay two graphs on the system	25-M: tool to save the graphs as an image

Value to customer in second increment

Now that the second increment has been completed, the user has an easier way of navigating through the screens using the sidebar, a single area that can be used to select each separate screen. The user also has access to various filters on the graph screens, allowing them to filter by a set of start and end dates to look more closely at a range of days in the data and filter by ad contexts such as shopping or social media so the user can see which type of ads obtained the most attention. The user can also define a bounce to the number of pages visited or the time spent on the ad website, giving them more freedom over this metric. There is also a new compare screen which can be navigated to by the sidebar or by adding a graph on the graph screens, which allows the user to have 4 graphs of any kind next to each other for comparison. The user can also zoom in to each of these graphs.

Sprint backlog for increment 2

ID User stories	Task	Member doing task	Hour Estimate	Actual Hours
2	Charts of data over different time intervals	Aditya	4	3
8	Filter input data by the selected audience filters	Marwan	4	3
10	Filter data by date ranges	Juzheng Bai	3	2
All	Write JUnit tests	Aditya	2	2
14, 21	Allow graph to appear in new window for comparison	Henry	2	#
13	Buttons to switch between different time intervals	Juzheng Bai	2	1
13	Compute metrics over different time periods	Henry	2	2
8,10	Update the graph when filters are applied	Jack	2	3
14, 21	Optimise generating multiple graphs	Marwan	2	1
8,10	Optimising filters being applied	Jack	1	1
5	Minimum number of pages or time spent for a bounce can be set	Henry	1	3
9	Bounces can be defined by number of pages or time spent on the system	Henry	1	2
12	Add sidebar which allows the user to Navigate between graph types	Jack	1	1
12	Save file data between all scenes	Marwan	1	1

Revised Sprint backlog for increment 2

We reworded the task: **"**Allow graph to appear in new window for comparison**"**
To be: **"**Allows graphs to be added to the compare page, with ability to name the graph**"**
We added the task: "Implementation of zoom in function**"**
The new task supports User Stories 14, and 21, and added an hour to our total time estimate.

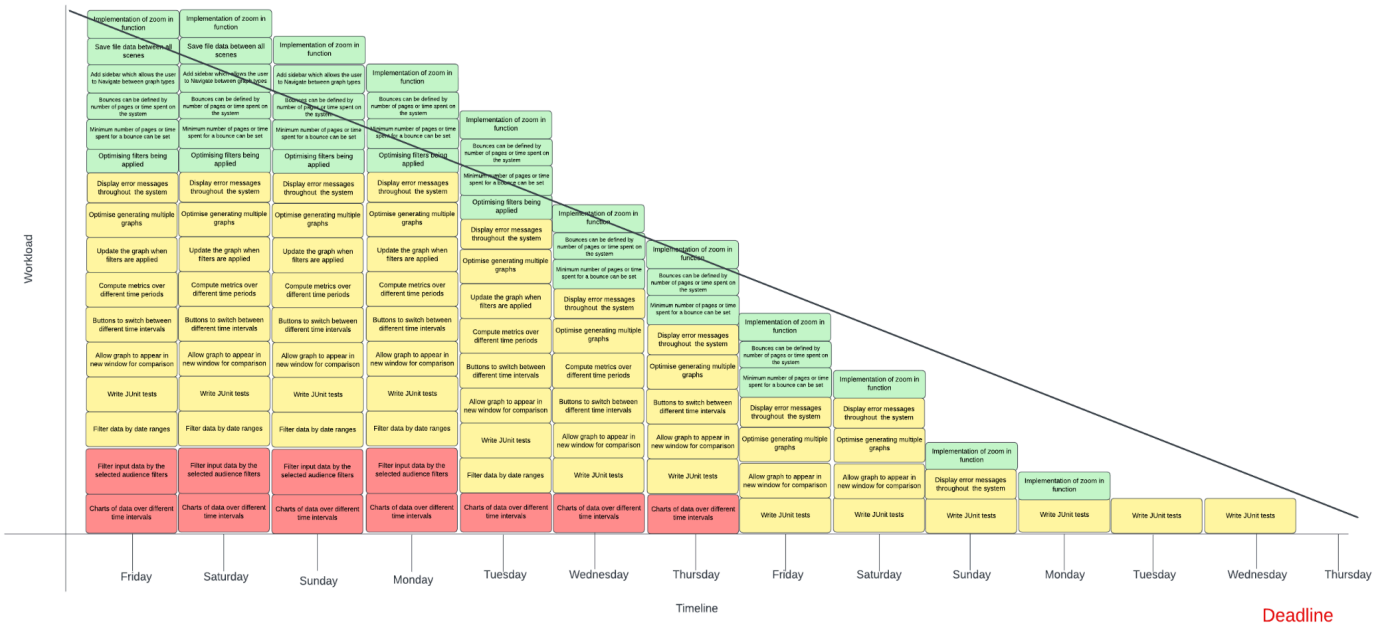
ID User stories	Task	Member doing task	Hour Estimate	Actual Hours
2	~~Charts of data over different time intervals~~	Aditya	4	3
8	~~Filter input data by the selected audience filters~~	Marwan	4	3
10	~~Filter data by date ranges~~	Juzheng Bai	3	2
All	~~Write JUnit tests~~	Aditya	2	2
14, 21	~~Allows graphs to be added to be added to the compare page, with ability to name the graphs~~	Juzheng Bai	3	4
13	~~Buttons to switch between different time intervals~~	Juzheng Bai	2	1
13	~~Compute metrics over different time periods~~	Henry	2	2
8,10	~~Update the graph when filters are applied~~	Jack	2	3
14, 21	~~Optimise generating multiple graphs~~	Marwan	2	1
All	~~Display error messages throughout the system~~	Jack	2	2
8,10	~~Optimising filters being applied~~	Jack	1	1
5	~~Minimum number of pages or time spent for a bounce can be set~~	Henry	1	3
9	~~Bounces can be defined by number of pages or time spent on the system~~	Henry	1	2
12	~~Add sidebar which allows the user to Navigate between graph types~~	Jack	1	1
12	~~Save file data between all scenes~~	Marwan	1	1
14,21	~~Implementation of zoom in function~~	Aditya	1	2

Sprint backlog for increment 3

ID User stories	Task	Member doing task	Hour Estimate	Actual Hours
All	Create User Guide	Aditya	3	#
6	Compute values for click cost histogram	Marwan	3	#
6	New window for the histogram page	Juzheng Bai	2	#
22	Apply multiple audience filter at once	Aditya	2	#
18	Filters for gender, age range and income	Henry	2	#
19	Adjust screen scaling	Juzheng Bai	2	#
23	Colour themes for with colorblindness	Juzheng Bai	2	#
23,20	Interface themes setting	Jack	2	#
25	Allow graph to be saved as an image	Marwan	2	#
ALL	Write JUnit and Integration tests	Jack	2	#
6	Display histogram graph on screen	Juzheng Bai	1	#
24	Add style to the charts	Henry	1	#
17	Optimise reading data into graphs	Aditya	1	#
All	FAQ	Henry	1	#
All	Video walk through the application	Jack	1	#
25	Save graph as image button	Marwan	1	#

Burn Down Chart

Current Burndown Chart



Third Increment Burndown Chart