Software Engineering Group Project

Deliverable 2 - Increment 1

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

1 - DESIGN

1.1 - Key Design Artifacts

Use Case Diagram

The use case diagram below is created based on the user stories that we included in our first sprint. The Marketing Agency's Client represents the primary stakeholder we recognised in the envisioning section of the project, as it is the only stakeholder directly communicating with the system. We also included the Marketing Agency (representing a secondary stakeholder), since they provides the custom-made campaigns, whose metrics are evaluated.

The *includes* relationships in the diagram are used to show the connection between the base and additional use cases. These will be updated as we go through further sprints and add extra functionality to the system.

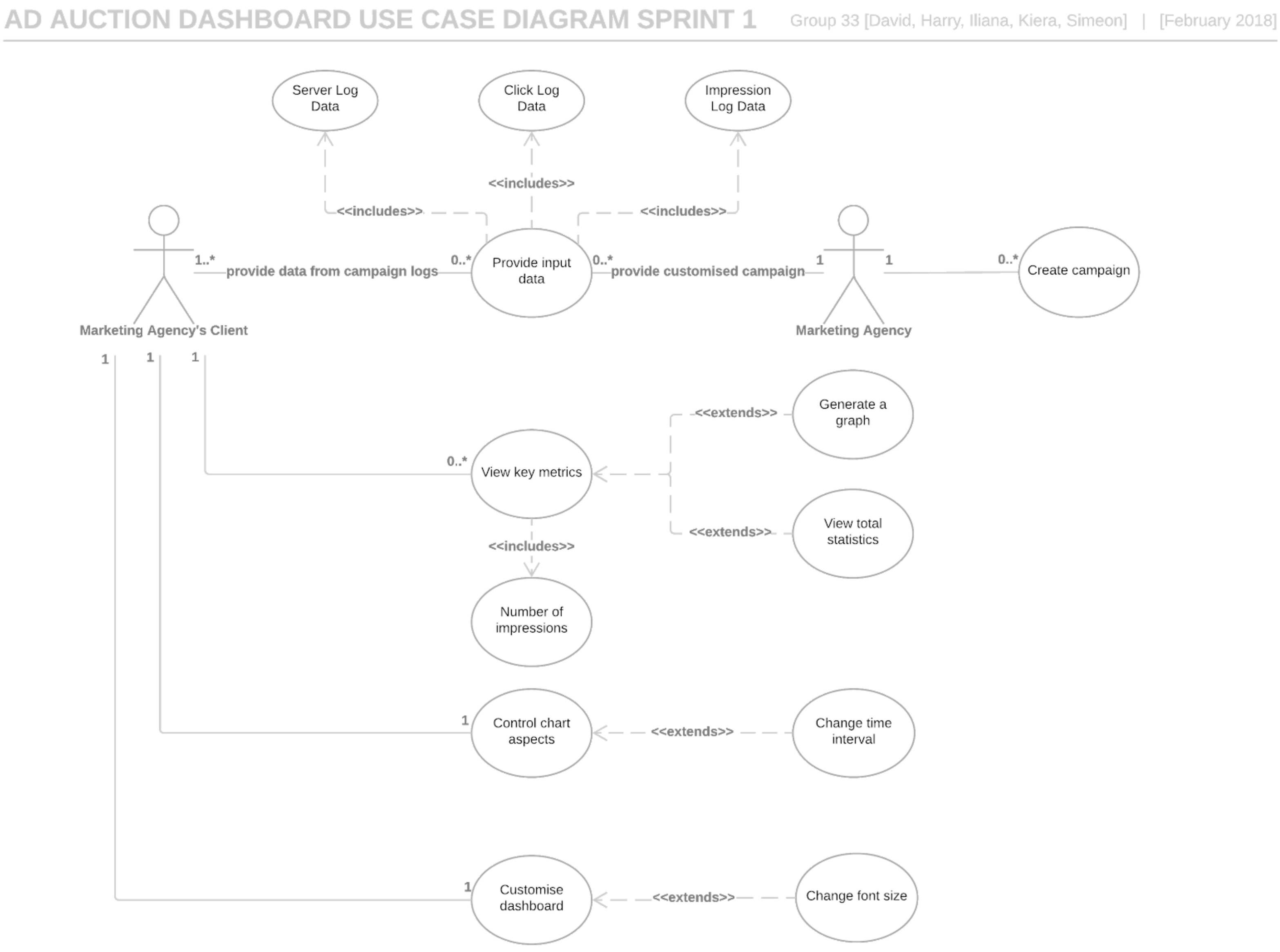


Figure 1: Use Case Diagram

Sequence Diagram

Based on the user stories and the use case diagram, we were able to create a sequence diagram to describe the interactions between the entities in the system. It also allowed us to visualise and validate various runtime scenarios. In the diagram, we have only one actor as the Client will be the only one interacting with the system directly.

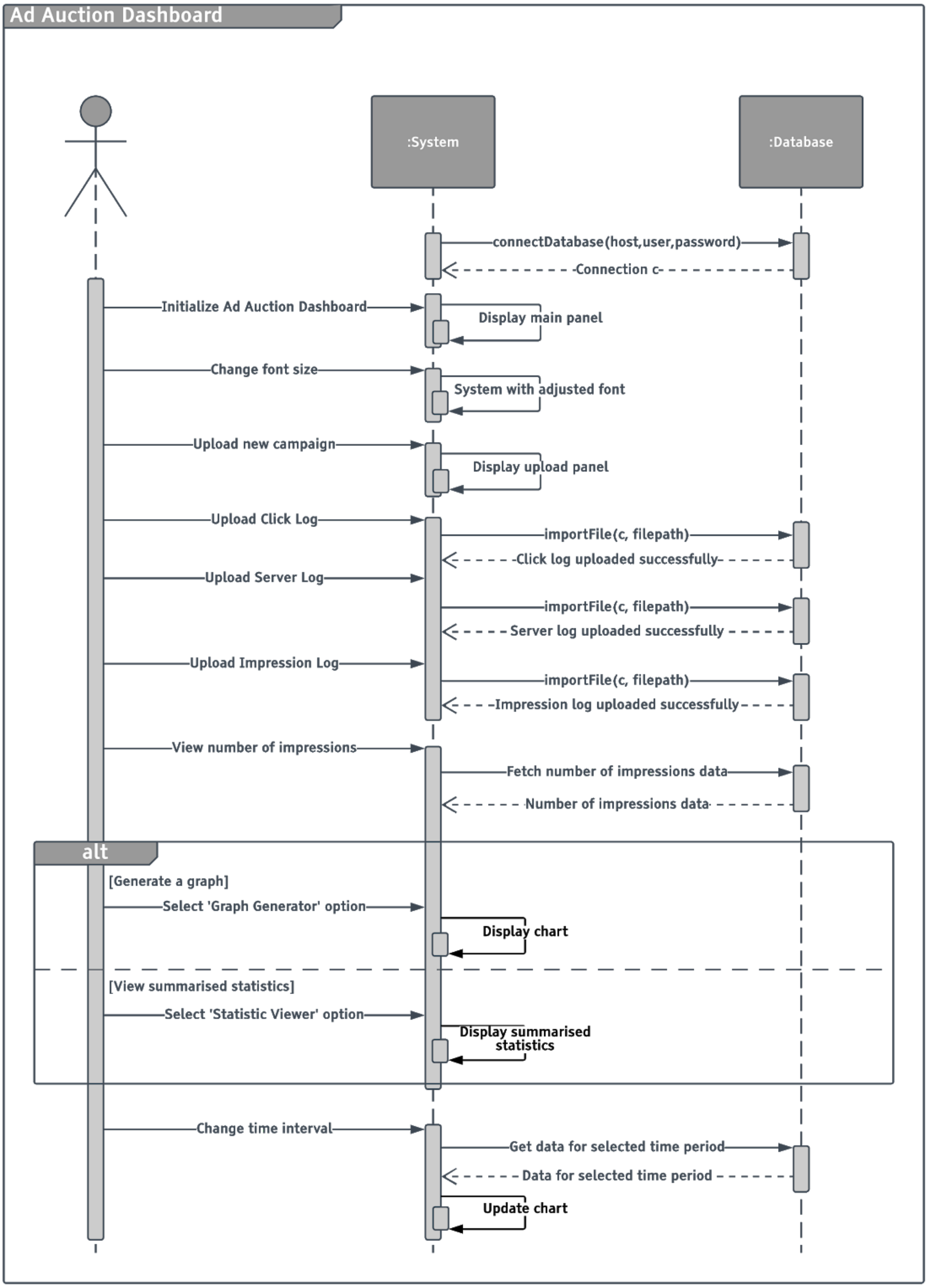
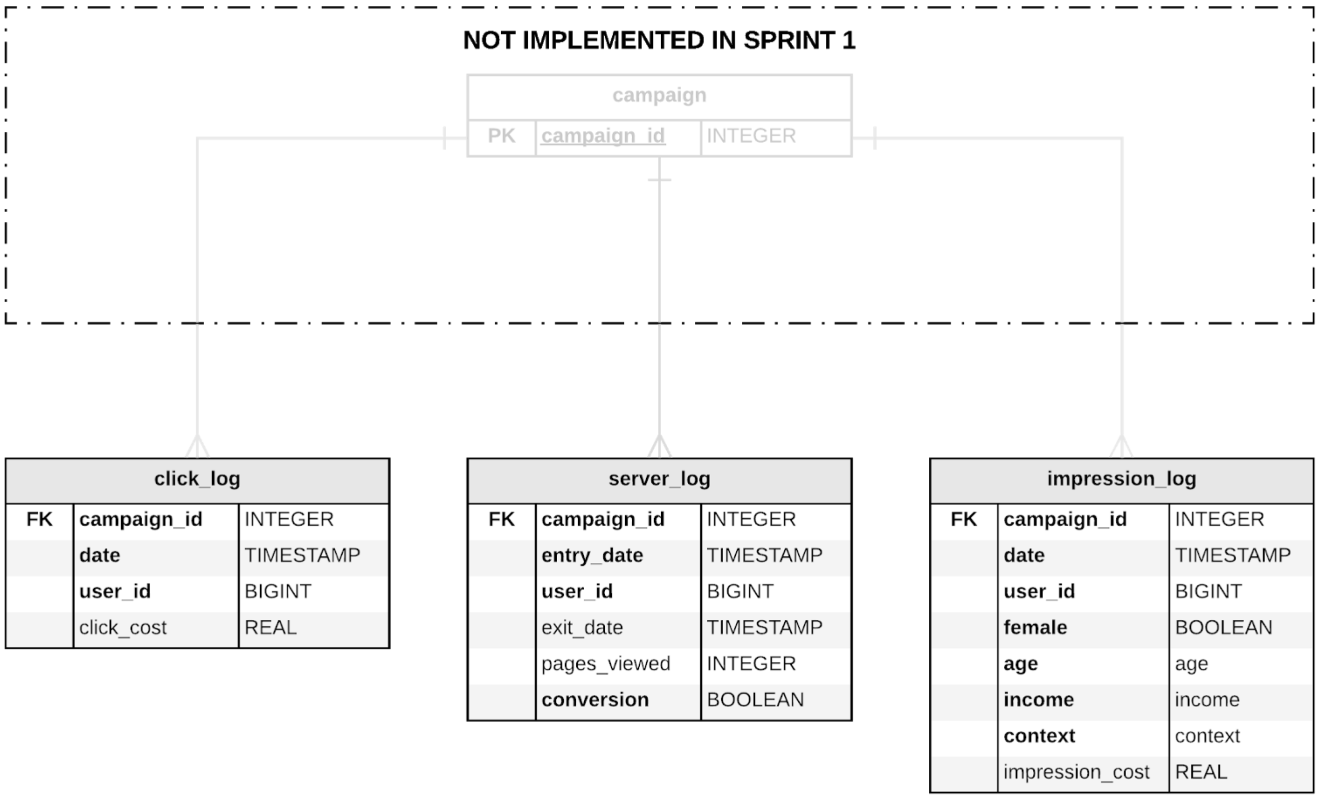

Ad Auction Dashboard
:System
:Database
connectDatabase(host,user,password)
Connection c
Initialize Ad Auction Dashboard
Display main panel
Change font size
System with adjusted font
Upload new campaign
Display upload panel
Upload Click Log
importFile(c, filepath)
Click log uploaded successfully
Upload Server Log
importFile(c, filepath)
Server log uploaded successfully
Upload Impression Log
importFile(c, filepath)
Impression log uploaded successfully
View number of impressions
Fetch number of impressions data
Number of impressions data
alt
[Generate a graph]
Select 'Graph Generator' option
Display chart
[View summarised statistics]
Select 'Statistic Viewer' option
Display summarised statistics
Change time interval
Get data for selected time period
Data for selected time period
Update chart

Figure 2: Sequence Diagram

Entity Relationship Diagram

The ERD below shows the relationships between the tables stored in the database linked to the Ad Auction Dashboard System. In addition to the three tables - ***server_log***, ***impression_log*** and ***click_log***, which are populated with data provided by the Marketing Agency's Client, we will be including a **campaign** table, if dealing with multiple campaigns**.**

This way, we can create multiple tables in one database and avoid creating and connecting to a new one when a new campaign data is uploaded.



campaign		
PK	campaign_id	INTEGER

click_log		
FK	**campaign_id**	INTEGER
	date	TIMESTAMP
	user_id	BIGINT
	click_cost	REAL

server_log		
FK	**campaign_id**	INTEGER
	entry_date	TIMESTAMP
	user_id	BIGINT
	exit_date	TIMESTAMP
	pages_viewed	INTEGER
	conversion	BOOLEAN

impression_log		
FK	**campaign_id**	INTEGER
	date	TIMESTAMP
	user_id	BIGINT
	female	BOOLEAN
	age	age
	income	income
	context	context
	impression_cost	REAL

Figure 3: Entity Relationship Diagram

1.2 - Design Architecture (MVC Approach)

To ensure our application is easy to maintain we have used 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.

In our case the model consists of all data stored by the system, that is campaign data stored in a database and configurations for user items like graphs. The controller is responsible for triggering events such as model updates or view updates. The view decides how the controller updates are shown to the user and as such consists of the graph display and viewable statistics. The view also contains the main structure of the graphical interface.

Because this project is making use of Swing we have 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.

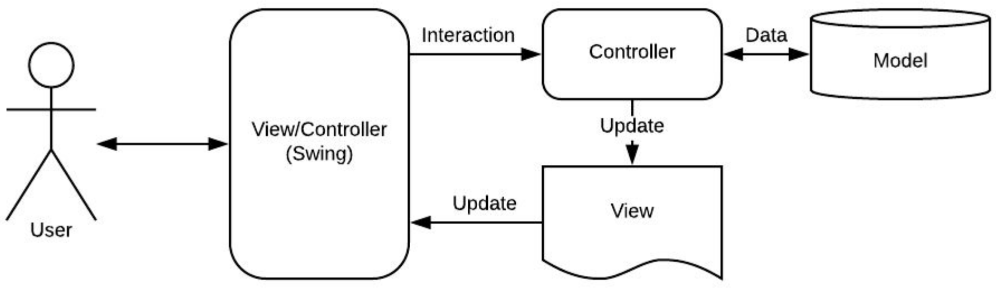


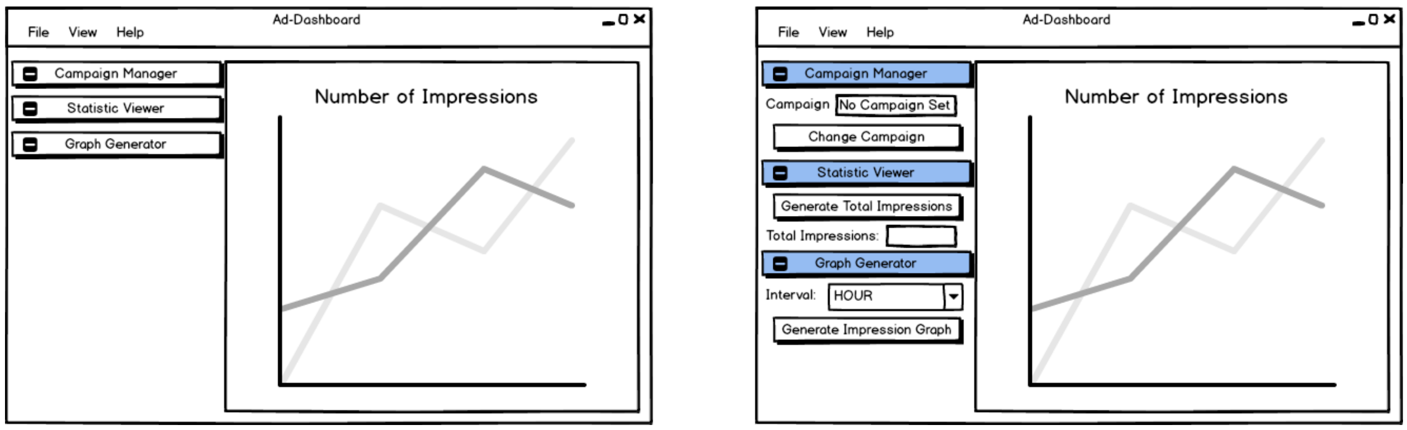
Figure 4: MVC Diagram

1.3 - Key Design Choices (Storyboarding)

As discussed in Section 3, for this deliverable we are only providing an interface that provides functionality for the current sprint. However, we still wanted to give the client a rough understanding of how the final application may be interacted with. For this reason, we made sure that we kept the personas in mind and communicated with the client during the design process. We were however careful not to make design choices that would negatively affect future development of the interface by keeping the design simple and extensible.

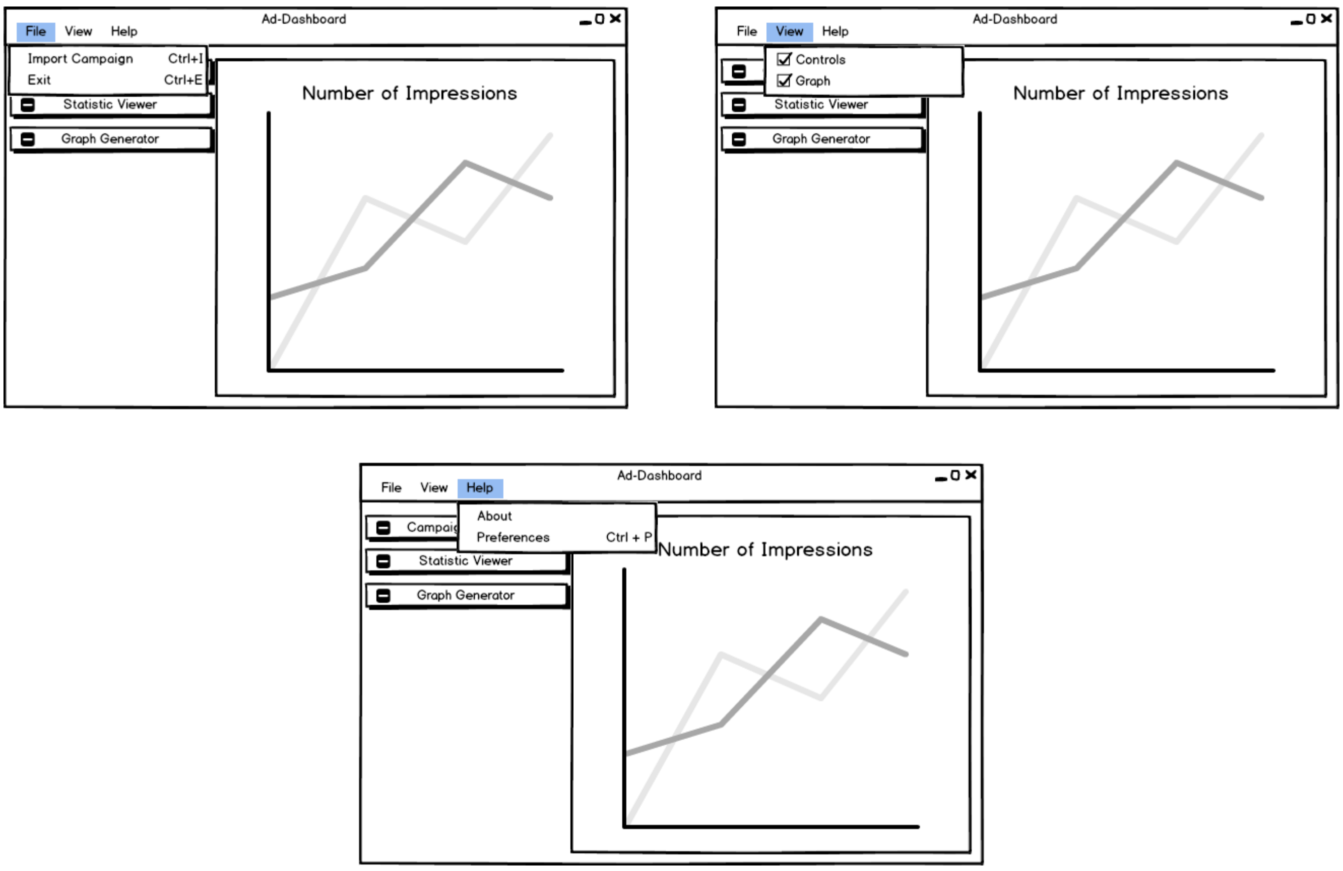
Main Panel (Figure 5)

The Main Panel is displayed after launching the application. It consists of two main components - on the left-hand side is the menu with expandable options whilst on the right-hand side is the graph display. This is extensible as control blocks can be swapped out. Depending on choices made by the user, the graph will be updated. For the current sprint, the user is only able to view impressions data. Therefore the graph can be generated from a single button press using the impression data from the currently loaded database with the selected interval. This generation method will be subject to change, once more functionality is provided.



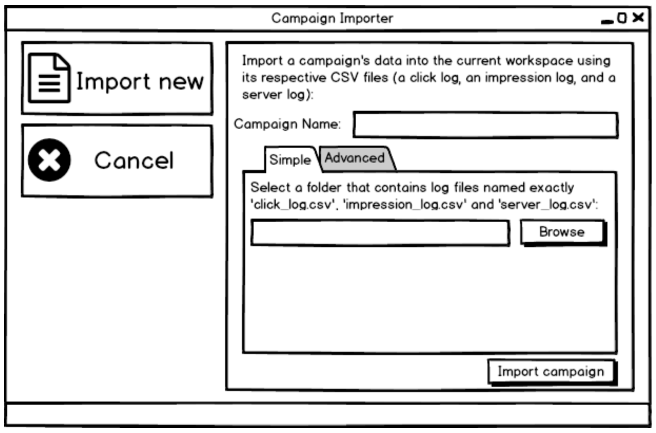
Menu Toolbar (Figure 6)

The Menu Toolbar is available from anywhere within the Main Panel. This gives the user control over the application, and the ability to customise the dashboard. For the first sprint, the only key feature to implement in the menu bar is preferences access as it is the only access method.



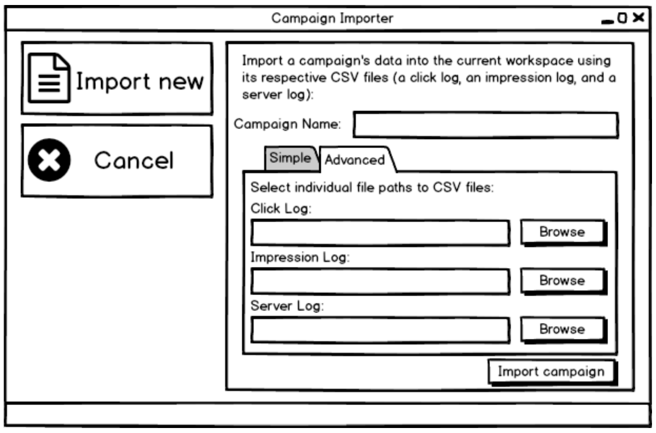
Campaign Importer Dialog (Figure 7)

The Campaign Importer Dialog is used to import log files into the database system. The system will display this panel when the user clicks on the “Change Campaign” button within the “Campaign Manager” expandable option. The campaign importer panel allows the user to select between two methods of import: advanced or simple. Simple allows importing from a single folder, whereas advanced allows importing from three different locations. When importing, progress will be communicated back to the user using a progress bar.



Import a campaign's data into the current workspace using its respective CSV files (a click log, an impression log, and a server log):

Select a folder that contains log files named exactly 'click_log.csv', 'impression_log.csv' and 'server_log.csv':

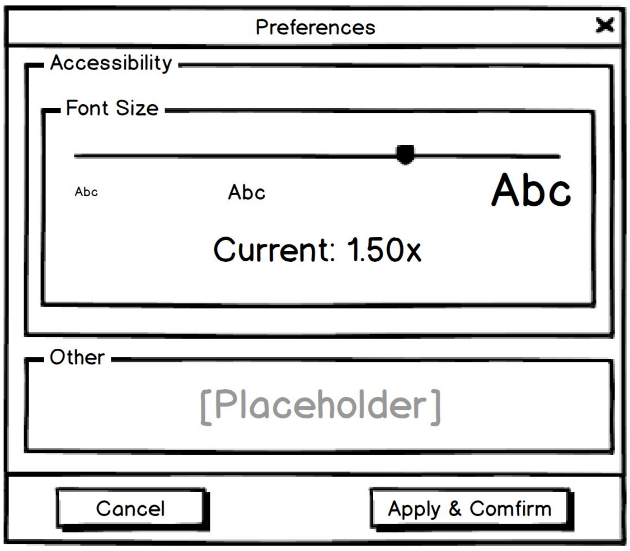


Import a campaign's data into the current workspace using its respective CSV files (a click log, an impression log, and a server log):

Select individual file paths to CSV files:

Preferences Dialog (Figure 8)

The Preferences Dialog is displayed after choosing the “Preferences" option from the “Help" menu toolbar or the keyboard shortcut “Cmd/Ctrl + P”. It will give the user the opportunity to change the font size of the system using a slider. The font sizes demonstrated alongside the font slider will be scaled as if that setting were applied. The “Current: #.##x” label indicates the currently selected font size on the slider to allow finer granularity. An option must be applied using the “Apply & Confirm” button before it affects the rest of the application.



2 - SCENARIOS AND TESTING

2.1 - Scenarios and key test outputs against them

Peter Quinn

Import new campaign
Scenario 1

Peter, who is a CO of a kitchen equipment company, wants to upload data from a campaign recently made for him by the Marketing Agency.

Peter clicks on the Ad Auction Dashboard application on his desktop.

He is now looking at the main application panel.

Peter clicks on the “Change campaign" button.

The System displays the ‘Campaign Importer" panel.

Peter writes “New campaign 2018” in the “Campaign name” textbox.

He selects the “Advanced tab” because his files are in different folders.

Peter clicks on the “Browse" button for the “Click log" file and navigates to the location of the click_log.csv file.

Peter selects the file and clicks the “OK” button.

He does the same for impression and server log.

The System displays the file paths in the text boxes.

Peter now clicks the “Import Campaign” button.

Peter can see a progress bar on his screen.

The System creates the tables in the current database and populates them with data from the files provided by Peter.

The System displays the Ad-Dashboard panel and the loaded campaign on the left side of the screen.

Peter can now use the data he imported to view a summary of key metrics for his campaigns.

Corresponding Epic:

As a <Marketing Agency Owner> I want <data to be gathered from **impression log**/**server log**/**click log**> So that <my clients can have access to accurate data for impressions generated during an advertising campaign> (**PbID 01/02/03**)

Tests performed:

Type of test: Manual test **Preconditions:** ● The configured database has no created tables. **Actions:** ● Navigate to the Campaign Importer. ● Select the “Advanced” tab and the 3 files. ● Confirm choice.	**All buttons worked as expected.** **Tables were created in the database.** **Tables in the database were populated with data from the corresponding files.** ***The test was also performed when tables already existed in the database. In this case they were populated with the new data from the csv files.**
Type of test: Unit test **Preconditions:** No preconditions. **Actions:** ● To test the database connection and configuration we created mock objects, using the Mockito framework in conjunction with JUnit.	**The program had the expected behaviour when given illegal arguments.** **Correct exceptions thrown.**

View campaign metrics statistics **Scenario 2**
Peter now wants to view the number of impressions for the campaign that he recently uploaded data for. Peter launches the Ad Auction Dashboard application. The System displays the main panel. Peter expands the “Statistic Viewer” option. He selects the “Generate Total Impressions” option. The System displays the total number of impressions in the “Total Impressions” textbox. Peter makes a note of the data and closes the application. He is now going to pass on the information to his colleagues.
Corresponding User Story:
As a <Marketing Agency Client> I want to <**view a summary of number of impressions**> So that <I can manage my outgoing finances for every time an ad is shown to a web user>

Type of test: Manual test **Preconditions:** ● Database is connected; tables are created and populated. **Actions:** ● Run a query on the database to fetch data for total number of impressions and output the result.	**Query was executed as expected and gave the correct result in a reasonable time.** **Data was correctly displayed.**
Type of test: Manual test **Preconditions:** ● Database is connected and tables are created and populated. ● GUI exists and is interfaced with the database. **Actions:** ● Expand the “Statistic Viewer” option. ● Click on the “Generate Total Impressions” button.	**All buttons worked as expected.** **The correct result was displayed in the provided textbox within reasonable time.**

Jessica Pierce

View key metrics as a chart
Scenario 3

Jessica, a PR manager in a well-known toy company, wants to view a summary of performance metrics for a campaign made for her company a few months ago and uploaded in the Ad Auction Dashboard system.

She clicks on the Ad Auction Dashboard icon on her desktop.

The System shows the main panel.

Jessica expands the “Graph Generator” option on the left side of the application.

She wants to see the weekly changes in the number of impressions.

Jessica selects the “WEEK” option from the “Interval” dropdown menu.

She clicks on the “Generate Impression Graph” button.

The System now displays a weekly graph for number of impressions on the right side of the main panel.

Jessica makes a note of the data and can now easily analyse the information and make appropriate conclusions.

Corresponding User Story:

As a <Marketing Agency Owner> I want <my clients to have the option to **view key metrics about a campaign as a chart**> So that <they have better visualization of data> (**PbID 05**)

Change time intervals
Scenario 3

Jessica launches the Ad Auction Dashboard application from her desktop.

The System shows the main panel.

Jessica expands the “Graph Generator” option on the left of the panel.

She wants to see the hourly and monthly changes in the number of impressions.

Jessica selects the “HOUR” option from the “Interval” dropdown menu.

She clicks on the “Generate Impression Graph” button.

The System now displays an hourly graph for number of impressions on the right side of the main panel.

Jessica makes a note of the data and now selects the “MONTH” option from the “Interval” dropdown menu on the left of the graph.

She clicks on the “Generate Impression Graph” button.

The System now displays a monthly graph for number of impressions on the right side of the application.

Jessica makes a note of the data.

She can now easily analyse the information and make appropriate conclusions.

Corresponding User Story:

As a <Marketing Agency Client> I want to <view the performance metrics of advertising campaigns **over a time interval**> So that <I can detect common trends amongst web users / sudden changes in the performance of an advertising campaign> (**PbID 06/07**)

Tests performed for Scenarios 2 and 3:	
Type of test: Manual test **Preconditions:** ● Database connection is established. ● GUI exists and is interfaced with the populated database. **Actions:** ● Run the program. ● Select the “Graph Generator” option. ● Select “MONTH” from the dropdown menu. ● Click on the “Generate Graph” button. ● Repeat the same actions for different time interval.	**All buttons work as expected.** **Graph is generated within reasonable time.** **Data on the graph corresponds to data from the database.** **When changing the time interval, the graph is updated quickly.**

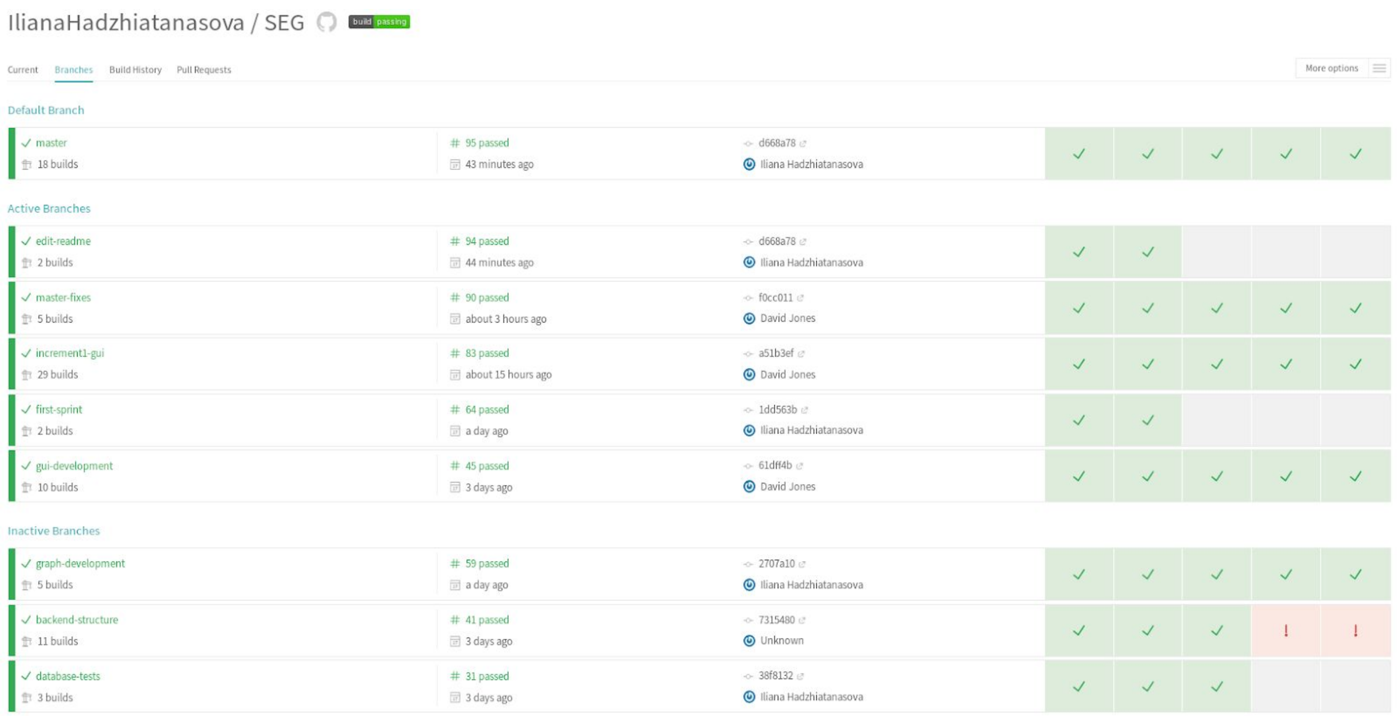
Change font size Scenario 4
Jessica launches the Ad Auction Dashboard application from her desktop. The System now displays the main panel. Jessica wants to adjust the dashboard font size so that she can see the information on the screen better. She selects the “Help” option from the menu toolbar. She clicks on “Preferences”. Jessica adjusts the font by moving the slider. She then clicks the button “Apply & Confirm” to confirm the changes she made. The dialog box automatically closes. The System now displays the application with a bigger font. Jessica can now continue working comfortably with the system.

Corresponding User Story:
As a <Marketing Agency Client> I want to <change the font size of the software for evaluating advertising campaigns> So that <my eyes don’t hurt if the font is too small> (**PbID 32**)

Tests performed:	
Type of test: Manual test **Preconditions:** ● Existing GUI. **Actions:** ● Run the program. ● Click on “Help” option on the toolbar menu. ● Click on “Preferences” ● Select preferred font size using the slider. ● Click on “Apply & Confirm” button. Same test was ran by pressing Ctrl + P.	**All buttons work as expected.** **The Ctrl + P option also has expected behaviour.** **After confirming the changes, the GUI is modified to show bigger font size.** **The change in the font does not cause the application to become unusable.**

2.2 - Continuous Integration with TravisCI

Through the project, the GitHub repo has been connected to TravisCI. This runs all unit tests whenever a user ‘pushes’ to any branch. If any test fails the user is notified. Alongside this, we made the ‘master’ branch protected so that changes can only be ‘pushed’ to it if all tests pass; this ensures the ‘master’ branch is always stable and a good basis for code development.



IlianaHadzhiatanasova / SEG build passing

Default Branch

✓ master 18 builds	# 95 passed 43 minutes ago	d668a78 Iliana Hadzhiatanasova	✓	✓	✓	✓	✓

Active Branches

✓ edit-readme 2 builds	# 94 passed 44 minutes ago	d668a78 Iliana Hadzhiatanasova	✓	✓			
✓ master-fixes 5 builds	# 90 passed about 3 hours ago	f0cc011 David Jones	✓	✓	✓	✓	✓
✓ increment1-gui 29 builds	# 83 passed about 15 hours ago	a51b3ef David Jones	✓	✓	✓	✓	✓
✓ first-sprint 2 builds	# 64 passed a day ago	1dd563b Iliana Hadzhiatanasova	✓	✓			
✓ gui-development 10 builds	# 45 passed 3 days ago	61dff4b David Jones	✓	✓	✓	✓	✓

Inactive Branches

✓ graph-development 5 builds	# 59 passed a day ago	2707a10 Iliana Hadzhiatanasova	✓	✓	✓	✓	✓
✓ backend-structure 11 builds	# 41 passed 3 days ago	7315480 Unknown	✓	✓	✓	!	!
✓ database-tests 3 builds	# 31 passed 3 days ago	38f8132 Iliana Hadzhiatanasova	✓	✓	✓		

Figure 9: Screenshot from TravisCI

3 - Responses to Feedback on Envisioning Deliverable

3.1 - Personalised Feedback

Dyslexic Client

In the marking meeting, we discussed the reasoning behind having a dyslexic persona and how viewing data as charts would be better for them than viewing it as plain text. Our choice to include the persona was provoked by our desire to make the system accessible by people with learning difficulties. Despite being able to give arguments and explain our motives, we decided that further research on people with dyslexia and their graph comprehension skills was needed to be able to bridge the knowledge gap and we found the following information to support our statements:

Dyslexic users find it difficult to keep track of what they are reading and can often lose their place with long paragraphs of text. [1]

To improve the quality of teaching and knowledge obtained by people with specific learning difficulties, a lot of sources recommend combining different methods, including representing information as pictures, graphs, and diagrams when possible. [2]

GUI

Something else that was referenced in the marking meeting was that *TaskID 01C* in the Sprint 1 backlog may be overly ambitious and not required to provide value to the client. This task was creating a fully laid out GUI interface to give an indication of what the final application may look like and the methods of interaction.

Though we agreed that producing this full GUI interface was not strictly necessary to provide value, we believed that some form of GUI was required to give the client a stronger physical interaction with our system so that they can provide suitable feedback. After the meeting, we discussed the task with our client and as a group, making the decision that instead of providing a full skeleton GUI, we would provide a GUI that displays functionality pertaining only to the first sprint. This would likely be of a similar style to the final application but is by no means a guarantee of layout or interaction methods.

Misconstrued Information

In addition, 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 will provide an inbuilt definition viewer alongside help tooltips that appear next to definitions on *mouse over*.

3.2 - General Feedback

Stakeholder Analysis

We still believe that the Web User (Ad Viewer) will be considered a secondary stakeholder because they indirectly provide data in the form of impressions and clicks. This input will influence the campaign success, which might result in the campaign no longer being shown to users.

Risk Analysis

We will be updating our risk analysis every sprint to ensure will we be able to find mitigations for every risk we come across as a team. For example, we added the strike action to our risk analysis document. This is shown below:

Risk	P	S	RE	Mitigation
Lack of personalised feedback due to the strike action	5	3	15	Communicate with supervisor more often to get confirmation for decisions that were made by the team.
Lack of supervisor meetings due to strike action	5	3	15	Have longer supervisor meetings when strikes are not happening.

P = Probability [1 - 5 (high)], S = Severity [1 - 5 (high)], RE = Risk Exposure [E E = P x S]

3 - Planning

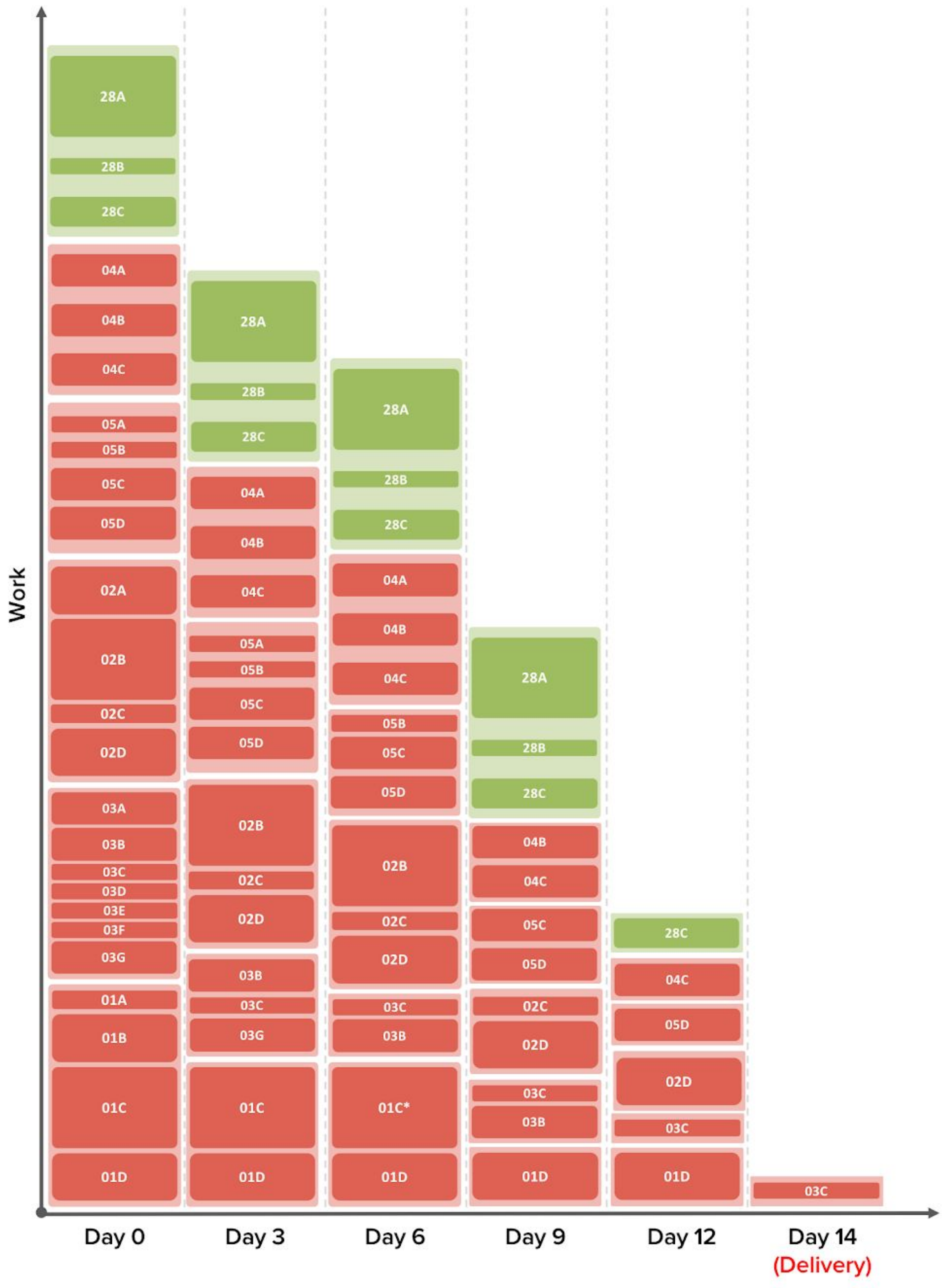
3.1 - Sprint 1 Burndown Chart

The Burndown Chart below shows when the tasks we identified for our first sprint were completed. At the start of the sprint progress was relatively slow as the initial tasks required theoretical work, researching the problem space and making design decisions; these fundamentals were part of tasks but didn't necessarily appear as their own.

Once we had built up knowledge and a back-end infrastructure task completion rate improved.

As mentioned in the *Personalised Feedback* section, we had to modify *TaskID 01C* and instead of making an overall GUI design, we only made wireframes for the current sprint.

After having a discussion, we decided that *TaskID 03C,* related to creating initial database indexes, was not needed for this sprint, because the queries we ran were executed fast enough. We agreed that indexes will be useful once we are querying more data from the database.



3.2 - Sprint 2 Backlog

Sprint Backlog				
PbID	TID	Task	Size	Dependencies
06 [Conversion rate summary]	06A	Create query to get campaign calculation.	S	-
	06B	Create query to get conversion data based on time intervals.	S	-
	06C	Extend GUI to support displaying conversion rate summary.	M	-
	06D	Create graph for conversion rate data over time.	M	06B
	06E	Test the accuracy of implemented queries and compare to what is displayed on the graph.	M	06A/06B/06D
07 [Total cost statistic]	07A	Create query to get total cost statistic.	S	-
	07B	Create query to get time interval cost statistic.	S	-
	07C	Extend GUI to support displaying total cost statistic.	M	-
	07D	Create graph for total cost data.	M	07B
	07E	Test the accuracy of implemented queries and compare to what is displayed on the graphs.	M	07A/07B/07D
08 [CPA statistic]	08A	Create query to get CPA statistic.	S	-
	08B	Create query to get time interval CPA statistic.	S	-
	08C	Extend GUI to support displaying CPA statistic.	M	-
	08D	Create graph for CPA statistic data.	M	08B
	08E	Test the accuracy of implemented queries and compare to what is displayed on the graphs	M	08A/08C/08D
09 [CPC statistic]	09A	Create query to get CPC statistic.	S	-
	09B	Create query to get time interval CPC statistic.	S	-
	09C	Extend GUI to support displaying CPC statistic.	M	-
	09D	Create graph for CPC statistic data.	M	09B
	09E	Test the accuracy of implemented queries and compare to what is displayed on the graphs	M	09A/09D
10 [CPM statistic]	10A	Create query to calculate CPM statistic.	S	-
	10B	Create query to get time interval CPM statistic.	S	-
	10C	Extend GUI to support displaying CPM statistic.	M	-
	10D	Create graph for CPM statistic data.	M	10B
	10E	Test the accuracy of implemented queries and compare to what is displayed on the graphs	M	10A/10B/10D
11 [CTR statistic]	11A	Create query to calculate CTR statistic.	S	-
	11B	Create query to get time interval CTR statistic.	S	-
	11C	Extend GUI to support displaying CTR statistic.	M	-
	11D	Create graph for CTR statistic data.	M	11B
	11E	Test the accuracy of implemented queries and compare to what is displayed on the graphs	M	11A/11B/11D

12 [Bounce Rate Summary]	12A	Create query to calculate Bounce Rate statistic from database	S	-
	12B	Create query to get time interval bounce rate statistic from database	S	-
	12C	Extend GUI to support displaying CPA statistic.	M	-
	12D	Create graph for fetched data	M	12B
	12E	Test the accuracy of implemented queries and compare to what is displayed on the graphs	M	12A/12B/12D
13 [Number of clicks]	13A	Create query to calculate number of clicks.	S	-
	13B	Create query to get number of clicks during time interval.	S	-
	13C	Extend GUI to support displaying Number of clicks statistic.	M	-
	13D	Create graph for Number of clicks data.	M	13B
	13E	Test the accuracy of implemented queries and compare to what is displayed on the graphs	M	13A/13B/13D
14 [Number of uniques]	14A	Create query to calculate number of uniques.	S	-
	14B	Create query to get number of uniques during time interval.	S	-
	08C	Extend GUI to support displaying Number of uniques statistic.	M	-
	14C	Create graph for Number of uniques data.	M	14B
	14D	Test the accuracy of implemented queries and compare to what is displayed on the graphs	M	14A/14C/14D
20, 21 [Change bounce definition]	20A	Extend GUI to support change of bounce definition.	S	-
	20B	Create queries for the two types of bounce definition.	S	-
	20C	Test the correctness of the implementation	M	20A, 20B
22 [Date range filter]	22A	Extend GUI to support filtering by date range.	M	-
	22B	Write queries for selecting date range.	S	-
	22C	Create graph based on the filtered range.	S	22B
	22D	Test the correctness of the implementation.	M	22A/22B/22C
23 [Context filter]	23A	Extend GUI to support filtering by context.	M	-
	23B	Write queries for context filtering.	S	-
	23C	Create graph based on the filtered context.	S	23B
	23D	Test the correctness of the implementation.	M	23A/23B/23C
24 [Gender filter]	24A	Extend GUI to support filtering by gender.	S	-
	24B	Write queries for gender filtering.	S	-
	24C	Create graph based on the filtered gender.	S	24A/24B
	24D	Test the correctness of the implementation.	M	24A/24B/24C

25 [Age filter]	25A	Extend GUI to support filtering by age.	M	-
	25B	Write queries for age filtering.	S	-
	25C	Create graph based on the filtered age	S	25A
	25D	Test the correctness of the implementation.	S	25A/25B/25C
26 [Income filter]	26A	Extend GUI to support filtering by income.	M	-
	26B	Write queries for income filtering.	S	-
	26C	Create graph based on the filtered income	S	26A
	26D	Test the correctness of the implementation.	M	26A/26B/26C
31 [Saving option]	31A	Decide whether the option provided by JFreeChart is suitable for the purposes of the application	S	-
	31B	Modify code to allow saving charts	M	31A
	31C	Test to make sure the files are saved in the correct format and can be reopened	M	31B

References

[1] K. Maceri, "Document Design for Users with Reading Disorders" [Online]. Available: http://www.angelfire.com/tn3/writing/DesignUsersReadDis.pdf. [Accessed March 2018].
[2] C. Queen, “Exploring methods that can be used to improve Dyslexia”, Method 1: Coping Strategies and Learning Approaches, 2014

Notes

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