

● *Software 1*
Group 3 - OOPsies

● CFG Final Project

*Art HerStory – A quiz game of
female art using MET API*



MARIE JOSÉPHINE
CHARLOTTE DU VAL
D'OGNES (1786–1868)

MARIE DENISE
VILLERS FRENCH

●
Andrea Hricikova
Amirat Salawu
Dalia Bun
Kristina Novackova
Sefat Nur

Introduction

Our aim is to create an interactive program which will help raise the profile of female artists throughout history. The objective of the game is to display two artworks side by side giving the user the option to select an artwork, by deciding which was created first and clicking on the image of that artwork.

The program will require simple instructions and must be developed in a way that allows it to be easily played in short bursts. The program has been planned to include 3 predominant pages: 1) Homepage, 2) Main Game Play, and 3) Results Page. The game will interact with the users by asking them to click on the image to decide the correct answer for the question: "Which was made first?". The border of the image changes the colour according to the correct/incorrect answer to this question. The lives and score are also amended as the player continues playing.

Alongside the main gameplay, users will have the opportunity to learn more about female artists and artworks. The game will display the artist's name and will include an implementation of a pop-up window for each picture's name tag, displaying additional information. It will also include a sound jingle to emphasise the correct or the incorrect answer.

"Only 13.7% of living artists represented by galleries in Europe and North America are women."

"At the Art Basel fairs (Basel, Miami, and Hong Kong), women made up less than a quarter of the artists on view over the past four years."

"In the U.K., 64% of undergraduates and 65% of postgraduates in creative arts and design are women, but 68% of the artists represented at top London commercial galleries are men."

"A recent survey of the permanent collections of 18 prominent U.S. art museums found that the represented artists are 87% male and 85% white."

Background

The Metropolitan Museum of Art (MET) has released the MET Collection API in 2018, where ‘...users can connect to a live feed of all Creative Commons Zero (CCO) data and 406,000 images from the The MET collection, all available for use without copyright or restriction.’ (Tallon, 2018). The main purpose of this was to allow art to be integrated with Google applications as well as to be accessible for developers to promote art in technology. Female artists have been actively underrepresented in the course of art history which is reflected in their representation in our public spaces. Only around 14.5% of the MET’s collections data are identified as female.

MET is not singular in the trend, museums and galleries around the world have faced challenges in displaying works not by male artists. In the course of art history, it is usually women we are told to observe in the spaces created by men. Recent publications such as *The Story of Art without Men* by Katy Hessel, *Muse* by Ruth Millington or *Why Have There Been No Great Women Artists?: 50th Anniversary Edition* by Linda Nochlin, have highlighted this as a still persistent issue and fought to close the gap in gendered art history.

Finally, for the format of the program we have taken inspiration from the Higher vs Lower game and Google Arts & Culture game called ‘What came first?’. We are using a similar format where the user will need to click on the image of an art to guess ‘Which was made first?’.

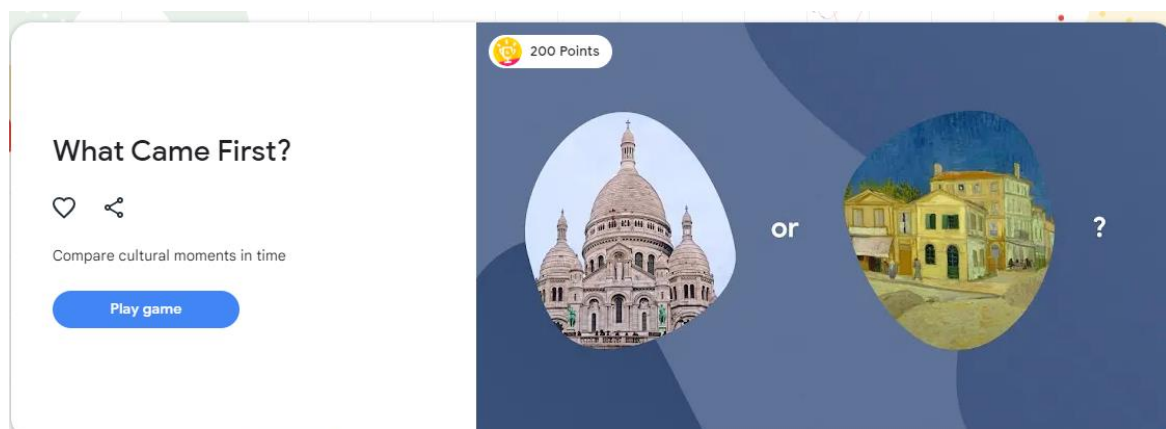


Figure SEQ Figure 1* ARABIC 1 - Google Art game

Design

Front end:

Game

1. Score and Lives words – Label text widget
2. Title of the game – Title text widget
3. 2 images that represent art pieces – Buttons widgets (images - different pair using Random module)
4. Image Name Tags – Buttons which lead to TopLevel pop-up widget - information about the art piece



Welcome!



Welcome to the Art HerStory Game
Explore and play with artworks made by female artists:
1. Choose which was made earlier by clicking on an image.
2. Click on a title to see more info about the artwork.
3. You get a point for each correct answer.
4. You have 3 lives to guess before the game ends.
5. Can you beat your score?
6. Click on the 'i' button if you get stuck.
Ready, set, GO!

OK

Instructions

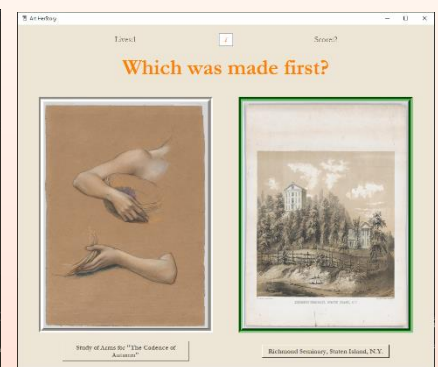
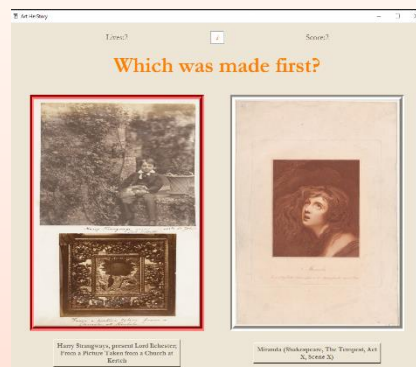
Show info pop-up (messagebox module from tkinter)

Instructions pop-up is the first window to be seen when the game is launched.

Game feedback

The user selects the correct answer the border of the image will display as green, and the score will be added

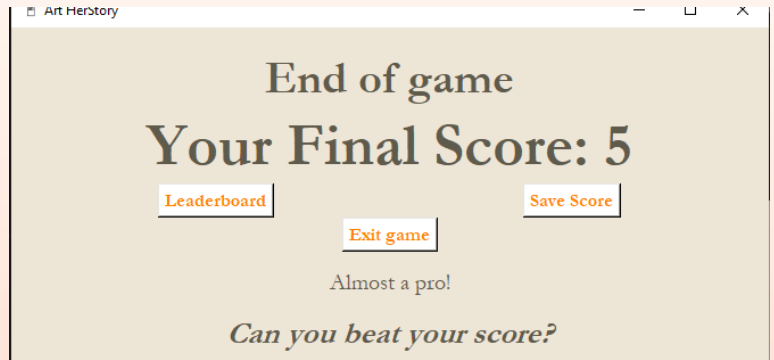
If the user selects the incorrect answer the border of image will display as red, and the life will be lost



Save Score

The game ends once the user loses all lives. The final score is added up and window changes to 'Page 2'

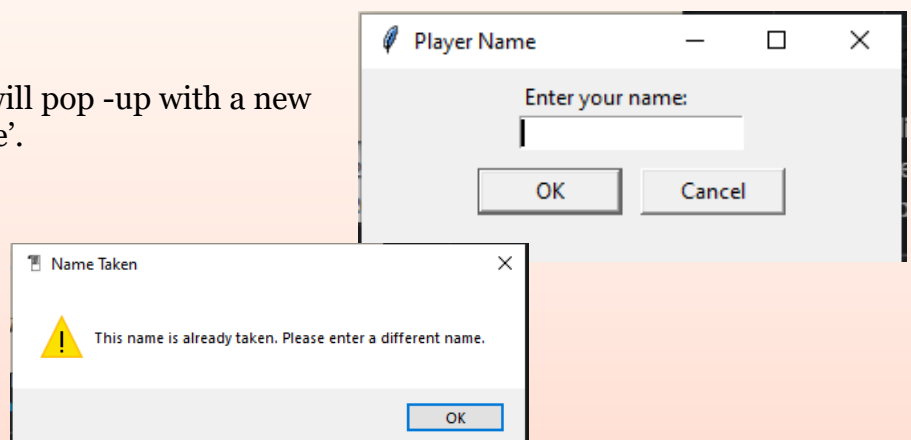
1. Results of the game - 'End of game' text and 'Final score: n' - Label widget
2. 2 buttons – Leaderboard and Save score
3. Exit game button - it ends the program



Save score

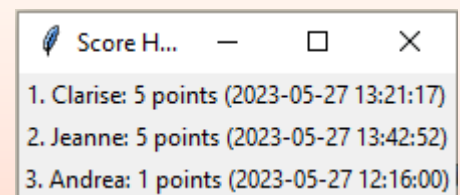
On the click of a button, it will pop -up with a new window to 'Enter your name'.

If the name is already taken a warning message will be prompted.



Leaderboard

On the click of a button a pop-up window, using the pickle module in Python and TopLevel module from tkinter, will show the scores saved ordered by highest at the top to the lowest.



NOTE on planned items which were taken out:

Page 0: (Menu/Welcome Page) – this was initially working but had to be removed in the definitive version of the project as it created challenge which was time consuming, and it was not essential to the game functionality

Play again button in the Page 2 that would lead the user back to the game which was taken out due to time restriction.

Backend Processes:

We decided to take this opportunity to extract data of female artists from MET's Collection API using the ETL (extract, transform and load) approach to represent their artworks. Through engaging with our program users have an opportunity to interact with works created by female artists that are usually overlooked or may not be on display. The program gives an opportunity to learn and raise the profile of female artists in a fun engaging way.

Game:

- Images of the art – code to insert random images from a file, function to swap images, function to highlight border with distinct colour if win/lose, function to play sound linked to 'correct' or 'incorrect' answers
- Image tags – code to link to image, function to show more information about the art piece/artist as a pop-up window
- Score – function that will update score if user wins
- Lives – function which will create restriction on how many times a user can fail (this can be opted out)
- Images – code to transfer data from API to a file

Save Score:

- Save score – function to save score to file using the user name this will require Datetime module to use 'timestamp'
- Leadership Board – update the board with the new scores and new player information
- Using Pickle Python module to serialise and deserialise data, using principles of Serde framework
- Play again button – code to bring you back to page 2
- Exit game – code which will terminate the game

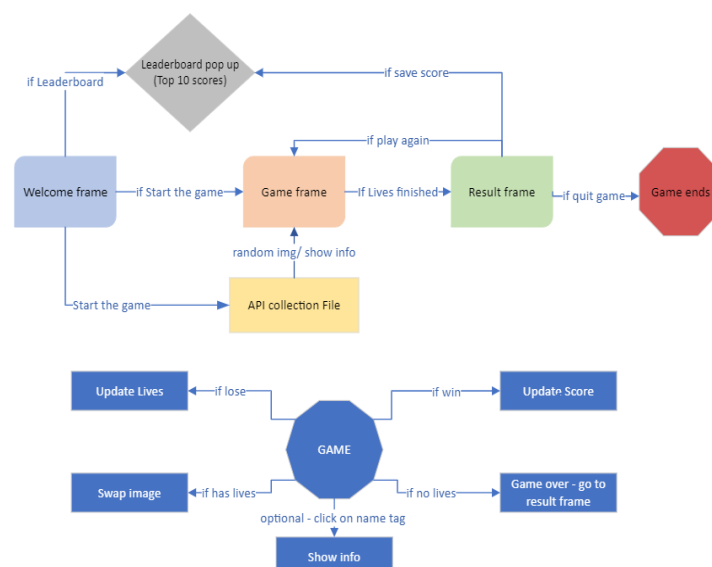
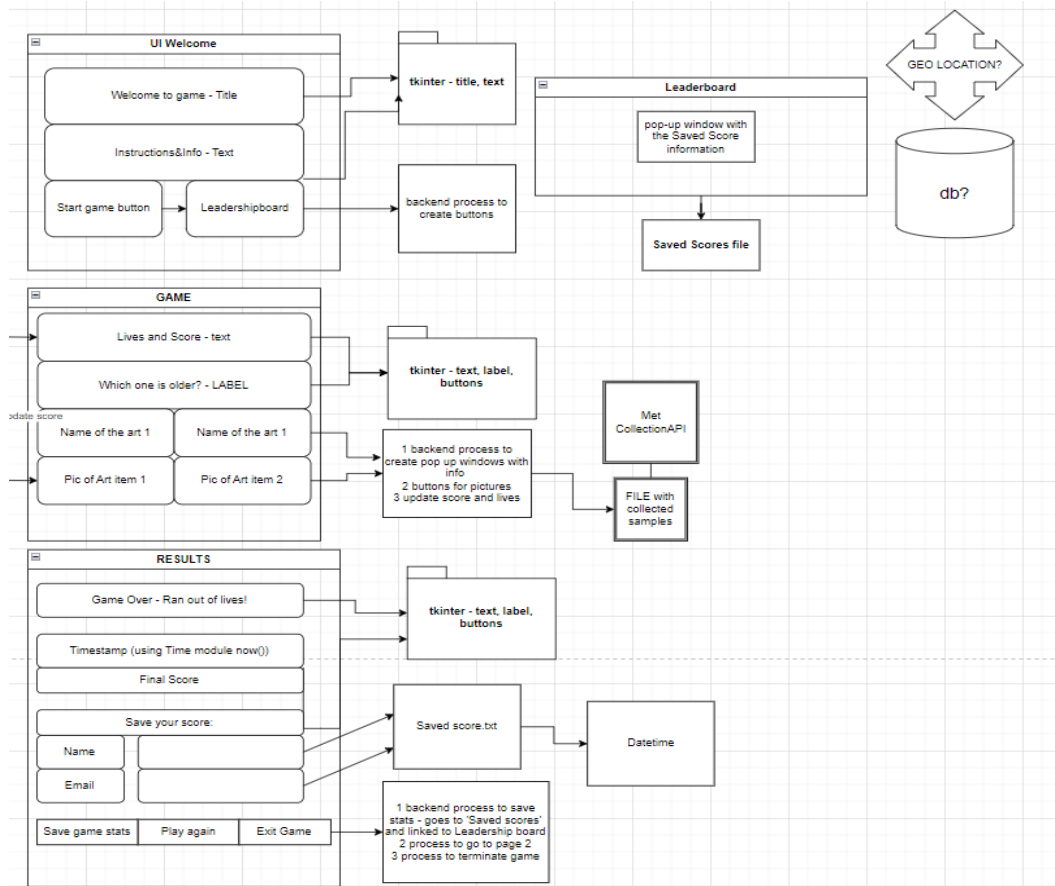


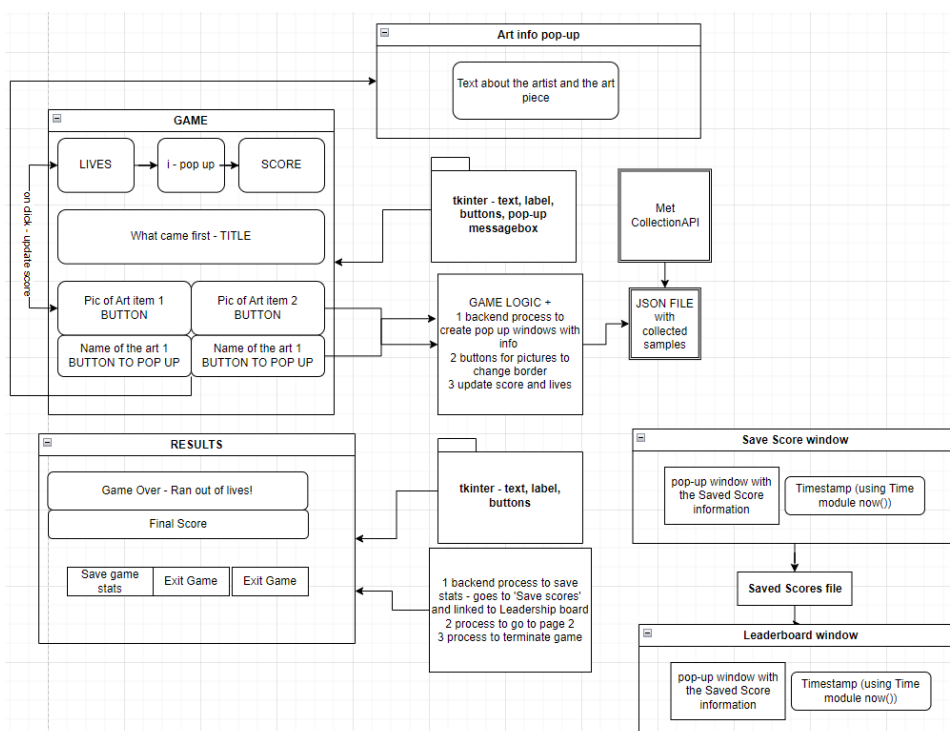
Figure SEQ Figure * ARABIC 2 Simplified diagram of the complete version of the game(including Welcome page)

First version – Brainstorming ideas

The first version of the game had a Welcome page as well as an idea to use DB for the game Leaderboard and GEO location for the local museums. The diagram shows our brainstorming process in creating a workable program.



Last version – the skeleton diagram of the GUI and some backend processes.



Project analysis

All of the members of our group have been working with OOP practices when coming up with the code for the game. As a starting point Dalia and Sefat worked on functions and their definitions which were then set into classes with the help of the rest of the group. Later on, any code added had been already done using the OOP approach.

We have also decided to use MET API as our data store and use ETL approach to create JSON files with exported data into several files, which were then used for the main game function. This was a strong starting point as it enabled us to begin with the main game coding very quickly.

During the project we have used a set of libraries required for each file to be functioning properly. Libraries used: Tkinter, Requests, PIL, IO, Pygame, Pickle, Datetime, logging and Random.

Our exception handling processes are mostly linked to the Saving Score file which manipulates the Pickle file. With this in mind, it has given Sefat the opportunity to use exception handling in her code. Moreover, exception handling is also used in saving names in our Scoreboard as well as part of our main.py when initialising our JSON file.

GAME FUNCTIONALITY CHALLENGES

Whilst each team member faced challenges specific to their individual code the main challenge was integrating the code created by each individual team member and refactoring it appropriately to work as a whole. Moreover, while using different operating systems has proved slightly challenging on occasions, it has also provided the team with the opportunity to test the program further.

FINAL DECISIONS – as a group we have decided to opt out the Welcome page as it was not feasible and would require further research. This function was not essential to the game and we have troubleshooted to replace it with an information Welcome window instead. We have also decided to add on several improvements in the last week such as sound buzzer and pop-up window for instructions.

SAVING SCORE FUNCTIONALITY CHALLENGES

Saving score function needed to have a separate button to save score from the previous page as well as a button with the pop-up window to show Leaderboard. Initially, we had a scoreboard which would save data using timestamps. Sefat and Dalia managed to come up with the solution to show scores from the highest to lowest and only show top ten scores.

Testing process:

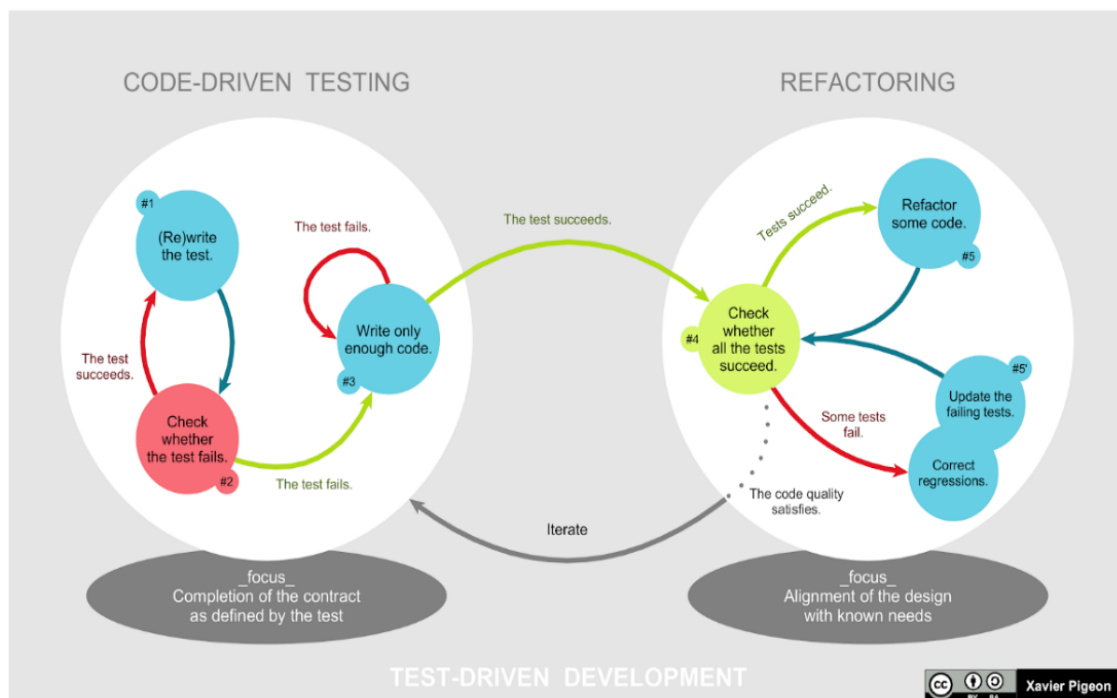
Testing strategy

Our testing strategy was implemented after studying the system architecture. We quickly realised that it would be necessary to split our code for our game away from our code for our GUI. This would make our program more maintainable and open-sourced, be able to apply our logic with other systems at a later time, as well as make it easier to test.

We planned to test our main program separately from the GUI, creating unit tests for the logic part of our program and using automated tools such as Selenium or Cucumber for our GUI.

We implemented a Test Driven Development (TDD) approach with functional testing, specifically unit tests:

1. Write test for what you are building
2. Run the test and it should fail for the expected reason
3. Then write your code in the simplest way possible with the aim of passing your test.
4. Then test again, it should pass. If it fails, you need to go back to work on your code.
5. Refactor your code, get rid of any duplications, make it more readable etc. and make sure your tests are still passing



Functional and user testing

As mentioned in testing strategy, we realised that implementing testing will be complex for our program. We used unit tests to test parts of the code that were not affected by the GUI. These were the `game_logic.py` file and the `main.py` file.

We implemented the `pytest` module in writing tests for our units of our `game_logic.py` file. The reasoning for using `pytest` module is that it is widely used in the sector and there are a lot of explanations. It also does not require creating classes and the code is more readable compared to `unittest` module.

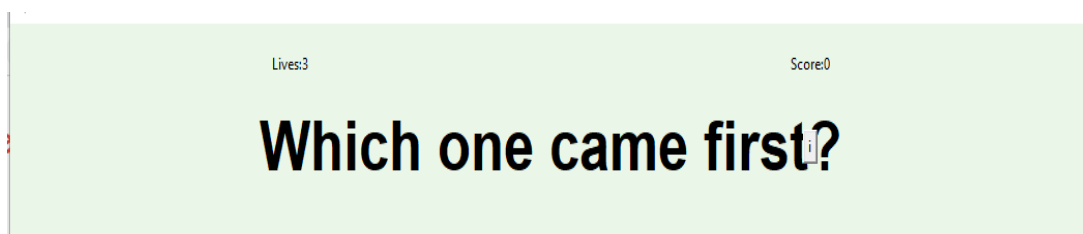
User testing

We tested our application by user testing taking feedback on what can be improved. The user testing was completed by the members of our team as well as our family members who have no affiliation to our project and could give us non-bias input. Following our user testing a few design choices were implemented. We found that the application works and looks differently on various operating systems. For example, the 'i' button icon was firstly next to the title as on Mac OS it looked better there, however, on Windows it clashed with the text. Therefore, the final design choice was implementing it in the middle between Score and Lives titles for better user experience.

Finally, our user testing concluded that the game serves its purpose and engages users with works of art by female artists in a fun and engaging way.

Testing limitations

Due to the short timescale of the project the tests for GUI could not be implemented. We recognise that testing of GUIs is a complex task that would require further knowledge and more time to implement. Since the majority of our program is based on the Tkinter GUI library, we were able to test our `game_logic` file and our `main.py` file as it was separate from the GUI elements. In future we would like to experiment with using Selenium or Cucumber frameworks for GUI testing





TEAM APPROACH

Team roles and responsibilities:

Andrea – Project lead, overall responsibility for the successful delivery of the project. Management and support of the team, delegating and organising work, creating workflows, setting up tools for project management and initial project ideas. Enhancement of extracted data and UI, responsible for testing, writing unit tests and creating logging functionality. Was able to utilise and further expand skills and understanding in project management, team support and delivery.

Amira – refactoring of whole project code and successfully making it run with various files, writing functionality for pop up windows, homework and creating presentation for the project. Amirat has been tasked with the functionality of showing extra information linked to art name tags. Amirat also worked on the final presentation for the project and worked out the logic behind separating large block of codes into several files.

Dalia – Explored the MET API in the first instance to ensure that the data returned in the payload will meet our requirements. Taken the lead on building the main GUI, focusing on both the visual and functional acceptance criteria - new areas which she had the chance of exploring as part of this project were providing the player with audible and visual feedback.

Helped debugging the codebase throughout the project and assisted with the modularisation of our application to make the code more readable, maintainable, and performant.

Kristina – Throughout the process Kristina has been tasked with responsibility to adhere to the documentation for the project. She has drafted first drafts for each document. Moreover, She has worked on the ‘swapping’ between pages functionality as well as refactoring and problem-solving approaches to the game interface functionality. Finally, She has approached each member of the group to support them and help them with their roles and to scaffold their process.

Sefat – For this project, Sefat was mainly tasked with creating the saving score functionality and leadership board to allow the users to save game data using their name and show it in the leadership board with timestamp . The leadership board was sorted with the score. She also generated the idea for adding sound feedback to the game and gathering necessary resources to make it work. Reviewed and edited documentation of the project. Wrote the installation and game play instructions. Wrote unit tests.

Everyone worked closely in refactoring the code, debugging and simplification of the OOP processes. All of the group meetings offered feedback and self-evaluation of everyone’s work in progress. If anyone encountered any issues they would share it with the rest of the group and work closely on finding solutions.

Team organisation

Over the course of the project everyone worked in an Agile framework (Table 1) in which tasks were broken down into small weekly iterations and progresses are reviewed regularly. A designated team lead created the [GitHub project](#), which allowed the group members to implement the agile framework to the best standard as well as helping them to visualise progress on the project.

Another beneficial experience was using GitHub as the version control tool. Everyone practised working on multiple different branches and fixing merge conflicts on a few occasions. Trunk-based development was used in the project. Due to the size of the team and the scope of work, it felt better suited compared to gitflow.

Sprint review and Sprint planning were scheduled for a longer meeting on a Saturday and short meetings happened twice during the working week for stand-up, updating each other on the progress and identifying potential blockers.

Team members worked with assigned individual tasks. During stand-ups and sprint reviews, discussions and reviews of the progress of each task and the difficulty of the tasks were done to ensure each member can handle their task and extra support was given if they required it.

Slack is used as a main driver for continuous communication and a medium to ask questions. Meetings were done via the Zoom platform and sent alerts to all members of the group by Slack. Quick chats were done using slack huddles.

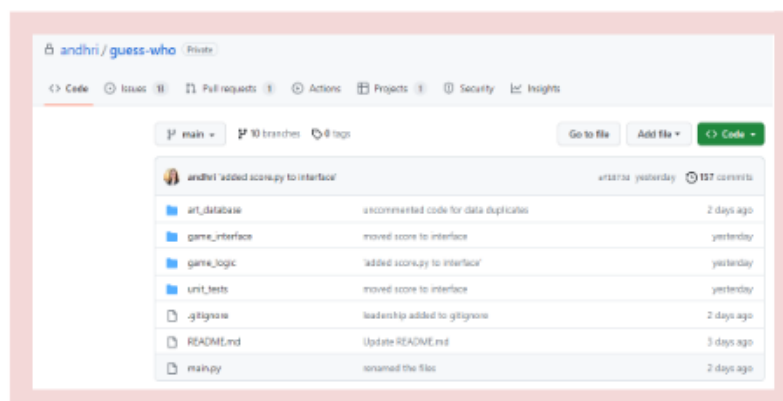
A GitHub repository through which we will all work collaboratively has been created as a way to store our program's code and to track changes. An accompanying GitHub project has been created as a way for us to assign tasks and to track progress in an Agile framework.

Using our GitHub project, communication is unified as members of the group have an opportunity to comment on their specific task, alert to any blockers and pass on any knowledge to the rest of the team. Moreover, we have a dedicated Google Drive shared folder to add any extra files such as evaluation, diagrams, criteria for the project and more.

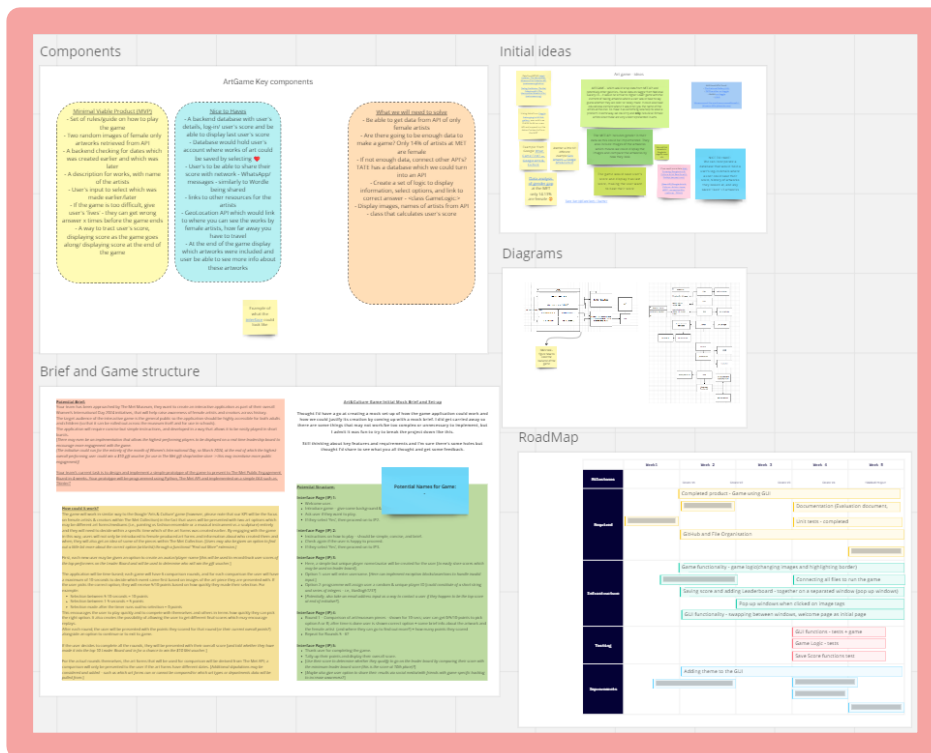
Conclusion

Evaluating our project, we approached each task with appropriate planning which in turn was successfully implemented in the time scale given. The success of our project was rooted in working to our strengths, using SWOT analysis we chose a project that was deliverable in the given time scale. There were few functionalities we were challenged by, as described above, some which we have overcome and some which we troubleshooted to overcome. We have learnt a lot from working together on this project and plan to publish it as an open-source tool for anyone to access, be able to use our codebase and learn about female artists at the MET.

GitHub →



CFG Software 1 – Group 3



← Miro Board

Shared Google Drive →

Shared with me > CFG - Project Group 3

File type People Last modified






Name	Owner	Last modified	File size
HW Week 2	Andrea Hricikova	12 May 2023	Andrea Hricikova...
Meeting recordings	Dalia Bun	6 May 2023	Dalia Bun
Project requirements from CFG	Dalia Bun	30 Apr 2023	Dalia Bun
CFG Group Presentation PowerPoint.pptx	Mira Sal	03:20	Mira Sal
CFG_Group_3_Evaluation_firstDraft.docx	me	25 May 2023	me
CFG_Group3_Eval_secondDraft.docx	me	26 May 2023	Andrea Hricikova...
CFG_Group3_Eval_thirdDraft.docx	me	10:41	me
CFG-Group3-gameGUI	Dalia Bun	3 May 2023	me
Group 3 - Meeting notes 30/04/2023	Dalia Bun	1 May 2023	Andrea Hricikova
Group3_Project Activity Log.xlsx	Dalia Bun	13:26	Sefat Nur
project_diagram.drawio	me	8 May 2023	me
roadmap.docx	me	23 May 2023	me

Appendix 1

Table 1. Breakdown example of our Agile working methodology by weekly sprints across May 2023.

Sprint	Week (May 2023)	Core Objective(s)
1	1 st – 7 th	<ol style="list-style-type: none"> 1) Brainstorm ideas for the project using an interactive Miro board. After choosing our initial idea, brainstorming and planning the components in another Miro board. 2) Decide how the project is going to be organised and how progress will be tracked (i.e., assign a team lead and a working schedule, establish working framework, create GitHub repo, analyse MET API, create shared Google Drive). 3) Break down key functionalities required and delegate tasks to team members.
2	8 th – 14 th	<ol style="list-style-type: none"> 1) Implement a working schedule - when and how often the team will meet (brief check-ins of progress will occur twice during the week). 2) Each team member works through their delegated tasks for sprint 1, with hopes of completion and a prototype in sight by end of week. 3) Get familiar with 'tkinter' as the chosen tool for developing a GUI which will host the game. 4) Hand-in group homework on Fri 12th May. 5) Bigger team meeting at weekend to further discuss progress, obstacles, and the objectives for the next sprint.
3	15 th – 21 st	<ol style="list-style-type: none"> 1) Have a working prototype of the program and be working towards the end-product. 2) Discuss areas for improvement and refinement and further delegate objectives to each member for ideal completion by end of the week. 3) Potentially organise a team meeting with our lead instructor
4	22 nd – 28 th	<ol style="list-style-type: none"> 1) Finalisation of working program and documentation (including evaluation doc and presentation). 2) Test completeness and functionality of entire program – test cases, edge cases 3) Hand-in finished project via GitHub deadline (Sun 28th May). 4) Prepare for the project presentation on Fri 2nd June and rehearse as a team.

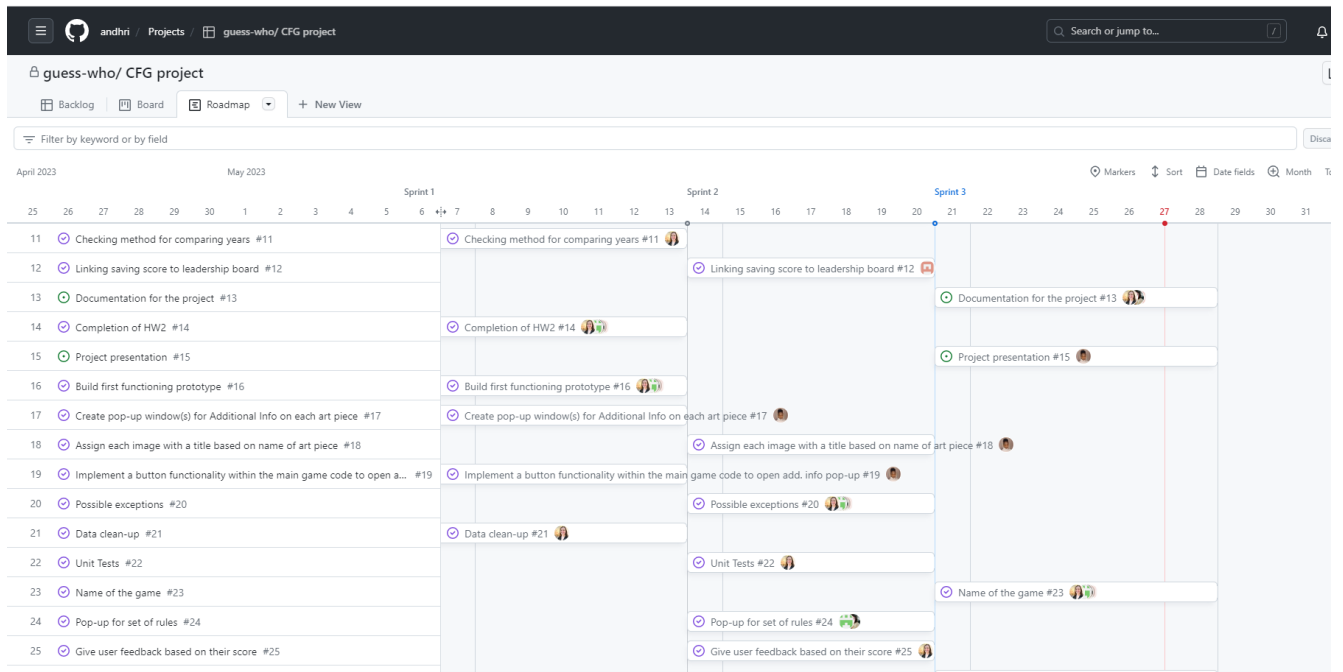
Table 2 - Project Roadmap

	Week1	Week 2	Week 3	Week 4	Week 5		
Milestones		 Create V1	 Create V2	 Create V3	 Create V4	 Finished Project	
Required	Brainstorming - Project	Completed product - Game using GUI					
		HW2 - project related		Documentation (Evaluation document, Code files, Project Log,			
				Unit tests - completed			
		GitHub and File Organisation					
					Presentation		
Infrastructure		Game functionality - game logic(changing images and highlighting border)					
		Game functionality - uploadin data from API		Connecting all files to run the game			
		Saving score and adding Leaderboard - together on a separated window (pop up windows)					
			Pop up windows when clicked on image tags				
		GUI functionality - swapping between windows, welcome page as initial page					
Testing				GUI functions - tests + game launcher tests			
				Game Logic - tests			
				Save Score functions test			
Improvements		Adding theme to the GUI					
		Using a sample of art from MET api		Instructions pop-up			
				Adding sound			
					Group name, logo		

CFG Software 1 – Group 3

Appendix 3

GitHub Roadmap of completed tasks



guess-who/ CFG project							
Filter by keyword or by field							
Title	Assignees	Status	Sprints	Labels			
1 Familiarize yourself with how Tkinter library works. #1	andhri, daliabun, K...	Done	Sprint 1	good first issue			
2 Building the first window view #3	KikaNova	Done	Sprint 1				
3 Pull necessary data from MET's API of female artists only #5	daliabun	Done	Sprint 1				
4 Check if Tkinter allows for swapping of multiple canvases #6	andhri	Done	Sprint 1				
5 Adding images as buttons and border to images for correct/incorrect #7	andhri and daliabun	Done	Sprint 1				
6 Check how the images from API will be displayed #4	daliabun	Done	Sprint 1				
7 Ability to click on image and images are swapped for next if correct... #2	daliabun	Done	Sprint 1				
8 Create class GameLogic with score, lives as attribute and methods f... #8	daliabun	Done	Sprint 1				
9 Save user's score in text file #9	SefatN	Done	Sprint 1				
10 Create pop-up window leadership board #10	KikaNova and SefatN	Done	Sprint 2				

