Project Report

By: Danielle Parker-Jones, Adaeze Eke, Emma Holden, Heather Wright, Laura Jimenez Hernandez and Xinyun Zhuo

Introduction

Aims and Objectives

The group aimed to create an entertaining and informative game called 'Female Icons Top Trumps'; a deviation from the popular game 'Top Trumps'. Through research, we found that the options for playing top trumps online were limited and the ones available were for niche topics. We aimed to address this problem by developing an educational, feminist themed game hosted on an online platform.

The game aims to educate the user on the strengths and achievements of influential women across fields such as politics, art, science and sport. The user overcomes challenges using a card deck of female icons, making strategic choices based on their knowledge.

Roadmap

To streamline our efforts and ensure a focused approach, we divided the project tasks among team members. We then developed a comprehensive roadmap that outlined the chronological sequence of tasks and milestones, grouped in phases. This roadmap served as a guide to the project's progression, enabling us to monitor our progress effectively.

	AREAS OF WORK	RESOURCE ALLOCATION	CRITERIA	ADDITIONAL NOTES
PHASE 1	• Pygame Research • Research how to create games using python • Top Trumps game understanding (online or physical)	• All members to conduct initial research • Heather is scrum master	• Research useful pre-existing games and resources. • Free flow/brainstorm. • Planning	The aim of this phase is to ensure that everyone is aligned and has a good understanding of the game we are trying to build and come up with ideas/ways we could build this.
PHASE 2	• API file • Initial database draft • Creating a card system in python • Creating a comparison function to compare attributes of 2 selected cards • All team members to pick 4 female icons for card creation • Front end work (HTML, CSS)	• Danielle - API • Laura & Adaeze - Main Python file and GitHub • Emma - Front End • Heather - Project Report & SQL Database • Xinyun - Initial database draft	First main file needs to be simple but functional Front end needs to be developed using Flask Database .	This phase allowed us to visualise what the game will look like and was a great starting point. Because we agreed this would be a first draft, it took the pressure off trying to get it '*perfect*'.
PHASE 3	• Class, methods and variables • Use of key OOP principles and SOLID principles • Libraries • API endpoints • Regression tests • Unit and Flask Testing • Connection between Python and Database	• Heather and Danielle to start report • Xinyun, Laura and Heather- Database connection • Emma & Adaeze- Testing	• The second draft needs to be split into different files • Testing • API connection working by the end of phase • Database connection needs to work for both databases • Working through the checklist to ensure each criteria is met.	The aim of this phase was to build upon the first draft of the programme and ensure that it all comes together and works by running and testing. We will work through the checklist to ensure that each criteria is met.
PHASE 4	• Decomposition of algorithms • Code layout consistency • Source tree directory structure • File organisation • Exception handling • Unit and Flask Test cases • API • OOP • Libraries	• Laura to create player_deck sql connection • Danielle to create challenge deck sql connection • Emma & Adaeze- Testing • Heather & Adaeze - Project Report • Xinyun- Re-play game Function • Heather to create preesentation	• The final product needs to connect API, database and Python file • Quality Control • Game playability needs improvement • All members need to review report	The whole group will continuously review each other's code, test that it works and make any necessary improvements. We will also finalise the presentation and decide who presents which parts.

Background

Existing Solution and Research

During the first phase of our project, all team members conducted individual research about online game ideas. From the very beginning we had a card game in mind and also the idea that we wanted to give our game a feminist spin. We came up with a card battle game, inspired by the well-known 'Top Trumps'.

The game features a diverse set of female heroes from various fields, promoting the idea that female heroes and role models come from all backgrounds and areas of expertise. This can contribute to breaking stereotypes and promoting inclusivity. The content of the game can initiate discussions about feminism, women's contributions, and societal challenges. Players might be inspired to research further about the featured heroes, their accomplishments, and the challenges they faced.

By blending educational content with gameplay, the game makes learning about history and important figures more appealing, especially for younger audiences. It encourages players to explore and appreciate the achievements of women who have made significant contributions to society.

Game Rules and Logic

The game involves two types of cards: female hero cards and challenge cards. These cards are characterised by attributes, names, and quotes displaying information about the heroes. The attributes determine the card's abilities in various domains.

Users/ players have the opportunity to interact with the game by drawing female hero cards to build their deck. These hero cards are selected at random from different categories, each containing specific hero cards with specific associated attributes (e.g. artists, scientists, activists)

The game also establishes a challenge deck, which consists of challenge cards. These cards represent various challenges that female heroes must overcome. The challenge deck is shuffled and then challenge cards are drawn at random.

The game follows a loop structure, where rounds are played until the player runs out of cards. In each round, the player is presented with a challenge card from the challenge deck to battle. The player needs to use their own knowledge to figure out which of their cards has more chances to defeat the challenge card. After the player selects a hero card , they pick an attribute from the challenge card they want to challenge. The player cannot see the attributes in their card at this moment. They need to guess (based on their knowledge about the person in the card) , which attribute the female hero card in their hand is most likely to have. After the player has picked a card and an opposing attribute, the game compares the attribute's value of the player's hero card with that of the challenge card. The player wins the round if their card's attribute value is equal to or greater than the challenge card's attribute value.

Once the round is completed, the player can finally see the attributes of the card they have played, as well as a quote explaining who the female hero they have chosen is. This can confirm if they were right or wrong when they guessed if they picked the correct card.

The game also introduces a level of strategy via female hero multidisciplinary cards. These cards can challenge many different attributes. For example, the card Frida Kahlo has attributes of the artist category but does also have attributes of political influence and rebelliousness (attributes that are normally allocated to the activist category). If the player knows they may have a multidisciplinary card in their hand, they should keep this card for the last rounds, because the multidisciplinary card may be their only chance to win the battle.

Once all rounds are completed, the game ends, and the player's score is compared to the computer's score (representing the challenges). The game announces the winner based on the score comparison. If the player has more points, they are congratulated for defeating the challenges. If the computer has more points, the player is encouraged to keep striving for progress. A tie is also possible. After this, the player has the option to play the game again.

SWOT Analysis

During the planning stage of the project we created a SWOT analysis:

Strengths	Weaknesses
• **Large team with diverse backgrounds:** A team of six people means we can delegate tasks based on individual strengths.. We have a broad range of previous work experience and coding knowledge. • **Use of an already-existing game:** Top trumps is a well-known and popular game with well-established logic and game mechanics. We can use this simple system as a starting point for our game. • **Use of beginner-friendly languages:** Python and SQL are high-level languages, which are easy to use and learn. They come with various libraries and thorough documentation. Python can easily handle database connection. Flask offers a useful alternative to Javascript so we can code in Python while using HTML/CSS for our user interface.	• **Team skill level:** We are beginners without extensive programming experience and knowledge. We are also new to using version control systems such as Git to work on a codebase as a team. • **Web development:** We do not feel confident with web application development and lack much prior web development experience. Designing a front end for a game may require a lot of extra learning. • **Creation of new game mechanics:** Although based on Top Trumps, our game has different rules and game logic. It is not always easy to pitch the game difficulty correctly as well as ensure enjoyability. • **Limited time:** With only four weeks to complete the project, it may prove difficult to balance completing tasks, prioritising improvements and troubleshooting issues.
Opportunities	**Threats**
• **Learning and development:** We can put into practice the skills we have learnt, as well as acquire new knowledge to overcome problems. We will gain experience of working in a team using version control and agile practices which will be beneficial in our future careers. • **Portfolio Project:** This project will effectively showcase our coding and teamwork skills and would be valuable to use as part of our portfolios. • **Educate others on female achievements:** As women interested in breaking into a male-dominated industry, we feel passionate about educating others and celebrating female achievements.	• **Mission creep:** It is easy to become carried away with new ideas, which can result in running out of time or making something that is too complex. We must initially focus on the minimum viable product. • **Difficulty in integrating various parts of the program:** Our game uses Python, Flask, an external API and a SQL database. Connecting all of these parts together can result in bugs and errors if not done carefully. • **Merge conflicts:** Human error while using version control tools such as Git can result in merge conflicts and bugs introduced into the code.

Specifications and Design

Design and Architecture

Employing the principles of object-oriented programming, our code incorporates custom classes, functions, unit tests, exceptions and overall a comprehensive system architecture that enables the game to be executed.

Backend Components

The backend of the system is implemented using Python and the Flask web framework. It includes the following components:

- FeministHeroesVsChallenges Class: This class represents the core game logic, managing the players, cards, and game rounds.

- PlayerDeck Class: Handles interactions with the player's card deck, including loading cards from the player_cards database and generating player cards for each round.
- ChallengeDeck Class: Manages the challenge cards used in the game, loading cards from the challenge_cards database.
- Database (MySQL): The system interacts with a MySQL database to store card and category information.
- API Integration: The backend connects to the Google Knowledge Graph Search API to fetch quotes or descriptions for player cards.

Frontend Components

The frontend of the system is implemented using HTML and CSS for the user interface. It includes the following components:

- HTML Templates: These templates are used to render different pages of the game, including the main page, card selection page, battle page, result page, and end-game page.
- CSS Stylesheets: These stylesheets define the visual appearance of the web pages, providing a consistent and appealing user interface.

Communication Flow

The communication flow in the system is as follows:

- The user interacts with the frontend, clicking buttons and making selections.
- The frontend sends HTTP requests to the Flask application running on the backend.
- The Flask application processes these requests, invoking the appropriate methods in the FeministHeroesVsChallenges class and other backend components.
- The backend may interact with the database to fetch card data, and it communicates with the Google Knowledge Graph Search API to obtain quotes or descriptions.
- The backend returns responses to the frontend, which are used to dynamically update the user interface.

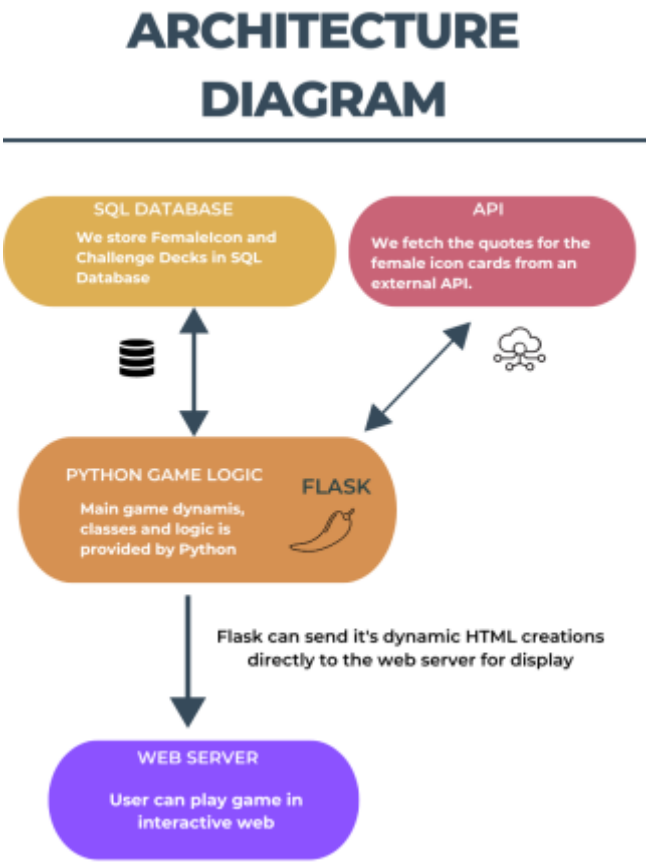
Data Flow

Data flows within the system as follows:

- Card and category data is retrieved from the MySQL database during initialisation of the PlayerDeck and ChallengeDeck classes.
- Player cards are generated by the backend by selecting random cards from different categories.
- User selections and game results are communicated between the frontend and backend via HTTP requests and responses.
- Quotes or descriptions for cards are fetched from the Google Knowledge Graph Search API based on card names.

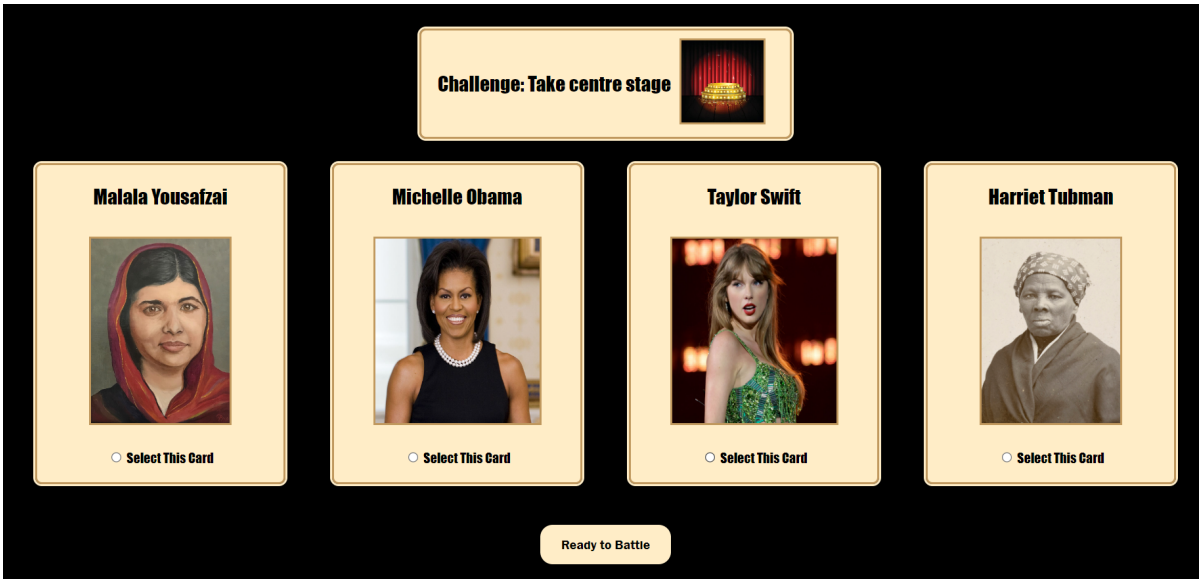
Deployment

The system is deployed on a web server where the Flask application is hosted. The MySQL database stores card and category information. Proper configuration files, including API keys, are used to ensure secure communication with external services.



To the left, the architecture diagram of the system is shown.

The user interface view when the game is being played is shown below:



Implementation and Execution

Development and Approach

This summary below highlights the team's journey from concept exploration to project planning, Agile adoption, collaborative work, and effective communication strategies throughout the project development cycle.

Project Concept Exploration: In the initial session, the team brainstormed various ideas, with a strong inclination towards creating a game. The team decided to refine their ideas over the weekend, reconvening on Monday to share and discuss their thoughts.

Selection of Top Trumps Concept: The team reached a consensus on developing a set of Female Icons Top Trumps, given its enthusiastic reception. This choice was fitting as the team consisted entirely of female-identifying participants.

Project Planning: The team categorised project elements based on the established project requirements. Responsibilities were assigned, focusing on essential elements, and a plan was established for regular progress reviews.

Collaborative Work: Team members initially owned specific parts of the project, allowing them to leverage their expertise. Code sharing and group testing promoted collaboration without creating a rigid working environment.

Regular Team Meetings: Weekly team meetings were held to realign tasks based on expertise, availability, and priority. Meeting notes were shared on Slack for reference, including individual action items.

Adoption of Trello: Halfway through the project, the team recognized the need for improved note-taking and task management. They decided to adopt Trello, which offered a Kanban board for tracking progress. https://trello.com/b/41q6Wpty/project-management

Agile Methodology: Agile was embraced for effective project management, enabling adaptability and responsive planning. One-week sprints, weekly meetings, and task reallocation ensured constructive feedback and flexibility.

Roles and Collaboration: Specific roles were designated within the Scrum framework, but team members often collaborated across multiple categories as needed. Peer reviews and pull requests facilitated collaboration and quality assurance.

Retrospectives: At the end of each sprint, retrospective meetings were conducted to evaluate completed work and provide feedback, allowing team members to address challenges and seek assistance.

Daily Stand-Up Meetings: As the project deadline approached, daily stand-up meetings were initiated to manage frequent changes and workload, ensuring the project met high standards of completion.

Testing and Evaluation

Testing Strategy

We pursued an Implementation-Driven Development approach to testing our programme. We considered the initial learning curve of adopting Test-Driven Development (as covered in lectures) and due to the time frame given, we decided to adopt the more familiar approach. We adopted quality assurance processes through continuous peer code reviews via GitHub, regression tests and adopting Agile methodologies. We also adopted quality control processes through detailed test case design and execution, functional testing and defect reporting and tracking, throughout the development cycle.

Functional and User Testing

We continuously undertook different forms of testing including unit testing, flask testing, smoke testing, regression testing, exception handling, and user acceptance testing.

We implemented unit tests to ensure that each testable unit of code was executed and operated correctly. E.g. We created the test file '*test_player_deck*' to validate the functionality and behaviour of the '*PlayerDeck*' class, with specific focus on its setup, category loading and player card generation mechanisms. The files executed the tests using the in-built Python 'unittest' library. We also conducted flask tests and used the '*Testcase*' class from the Flask-Testing library to establish a robust testing environment for the '*test_main*' test file. We created this file to validate and maintain the correctness and expected behaviour of the Flask application's routes and functionalities, with specific focus on the home, choose_cards, battle, results, end and restart routes.

Additionally, we consistently implemented smoke testing during the development process to ensure the proper behaviour of basic functionalities of each new build e.g after the reset function was added to help restart the game, we manually checked to see that the basic functionality of the *FeministHeroesVsChallenges* game was operational after a reset. We also included a test case ('*test_function_initializes_game*') to check this.

Furthermore, we conducted regression testing, with each pull request undergoing review from team members. Any identified bugs or discrepancies prompted necessary adjustments, which were subsequently committed. The code was merged into the main branch only after we were satisfied with its performance, and this iterative process persisted throughout the project.

Exception handling was also employed to address potential errors during program execution. We integrated assertions to account for all exceptions. In sections of the code that fetched data (either from the API or the user), we used 'try' and 'except' blocks to deal with exceptions raised and handle them appropriately. For example, if the user failed to pick an attribute before clicking the submit button, it would select one randomly from the available options.

Finally, team members asked family and friends to play the game and provide feedback. So, we performed user acceptance tests as well.

System Limitations

Speed: At first, the unit tests took a long time to execute because every single test case was initialising the game, which involved drawing the cards and fetching quotes from the API, even where this was not necessary. We resolved this by using mocks of relevant classes for tests where full game initialisation was unnecessary. Additionally, it takes several seconds for the game to load each time, due to the API calls required for the cards and the loading of the Flask client. Thus, if we were going to increase the number of cards in our decks, the game would presumably be even slower to load.

Dependency on external services: As we used Google API for our HTTP requests, if it fails for any reason, our system will not be able to fetch the quotes for our player deck characters. Also, as not all characters were able to fetch a quote, we had to modify our database accordingly so all characters would have quotes.

Last game round: In the last round of the game, the user has only one card left and therefore has no choice but to select the card, regardless of the challenge.

User setup: Before running the game, the user would need to open MySQL and run the 'player_deck and challenge_deck schemas so that the database exists within their machine. This may be a bit cumbersome to users.

Conclusion

In conclusion, our game development project has been rewarding and we believe we set out what we hoped to achieve. We started this project with a mission to fill a gap in the online gaming market, provide entertainment, and raise awareness about influential women in history. Through research and a well-thought-out roadmap, we have achieved these goals and more.

Our team's approach to project management, adopting the Kanban process and Agile methodologies, ensured that we worked efficiently, collaboratively, and flexibly. It allowed us to adapt to changing priorities and challenges, ultimately leading to a successful outcome.

The technical implementation of the game, with its object-oriented programming principles, API integration, and database management, reflects our commitment to delivering a reliable product. We encountered challenges along the way, but our dedication to continuous testing and improvement ensured that the final product met high standards of quality.

We envision our game sparking discussions around feminism, women's contributions, and societal issues. We hope it serves as a source of inspiration for players of all genders and encourages them to explore the stories of the heroes in the game. Additionally, our commitment to inclusivity through diverse representation has the potential to challenge stereotypes and promote a more inclusive society.

In conclusion, the finished product was only achievable due to teamwork, innovation, and a shared commitment to creating something meaningful yet fun and interactive.