TECHQUITY

Web Application Development Log

Design Section

Project link on Github:
https://github.com/thesoakedfox/12DTP-Balatro

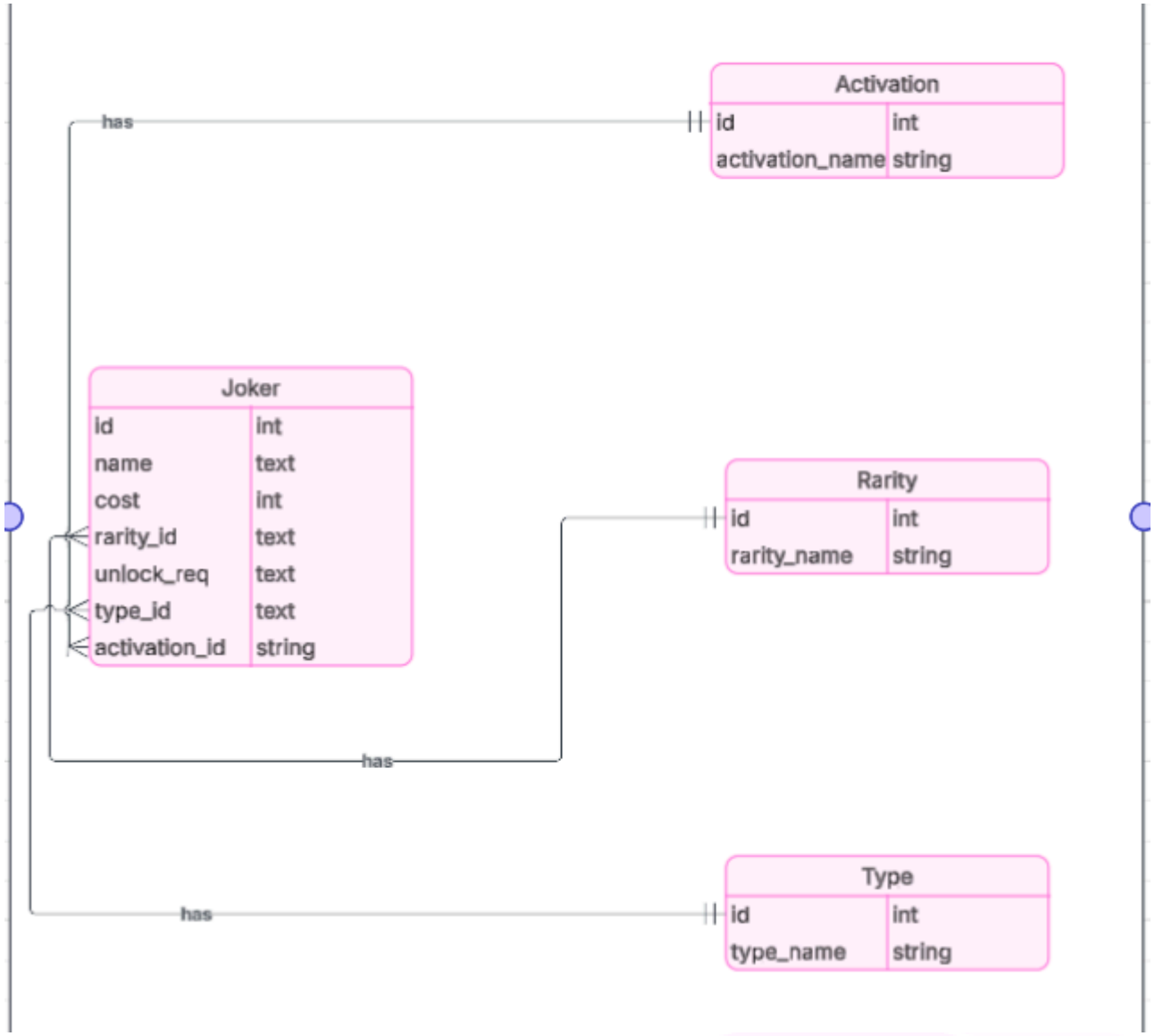
Project name:
12DTP-Balatro

Project Summary: Include purpose and target audience

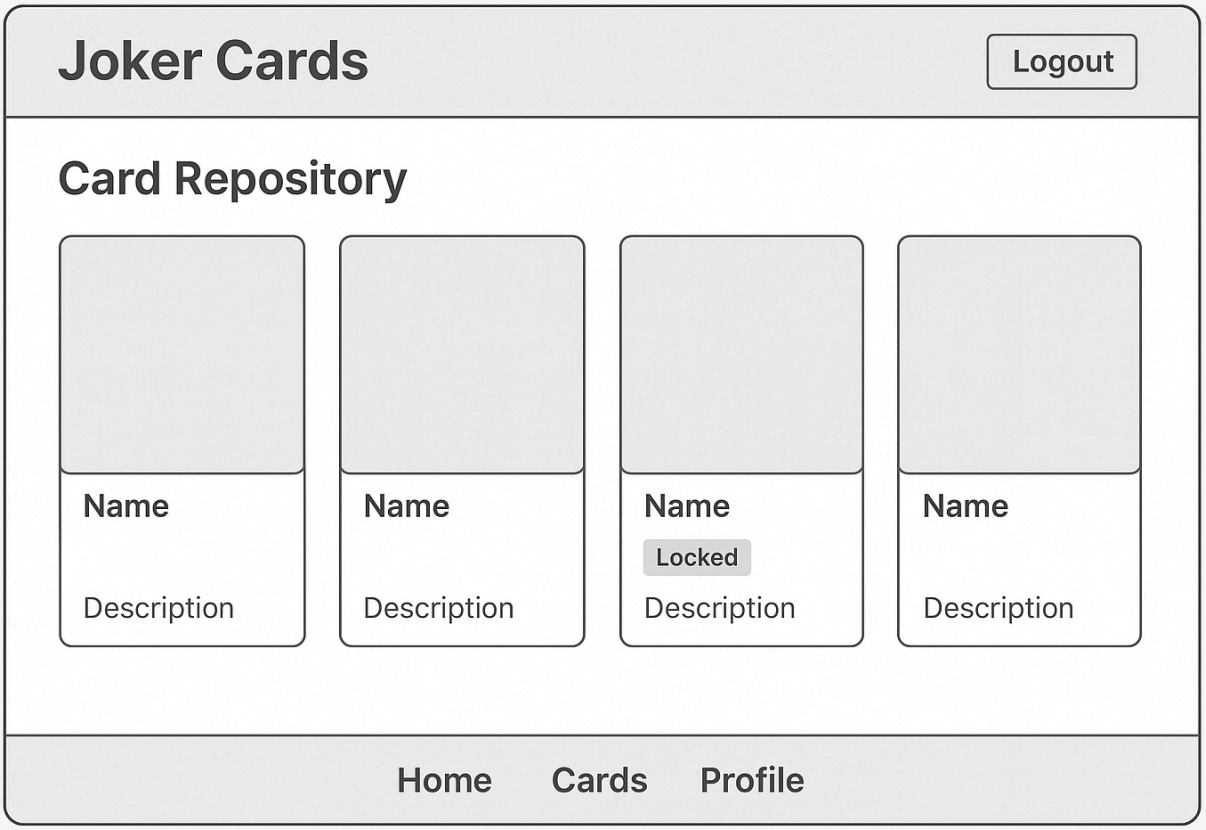
Purpose
The purpose of this project is to develop a web application that allows users to collect and track virtual Joker-themed cards. The system includes a card repository, where users can see which cards they have unlocked and which remain locked. The project demonstrates the integration of programming, database design, and web development using Python (Flask), SQLite, and HTML/CSS. It also explores user authentication, data integrity, and engaging interface design.

Target Audience
The target audience is secondary school students and game enthusiasts who enjoy collectible card systems and gamified progression. The application is designed to be simple and intuitive to use, while also visually appealing with a Balatro-themed style. It is intended for users who want to log in securely, track their progress, and enjoy unlocking cards in a playful digital environment.

Initial Database Design ERD:



WebPage Design- Wireframe and/or Palette




```
--raisin-black: #16141Cff;
    --blue-ncs: #2289C0ff;
    --rich-black: #0F2130ff;
    --prussian-blue: #062B3Dff;
    --fire-engine-red: #CD2C19ff;
```

Relevant Implications (explain at least 3)- Before Development.

Legal
I will ensure that all information contained in the database is used appropriately and complies with legal requirements. This means not storing or sharing any unnecessary personal data, and making sure usernames and passwords are handled securely. Media included will also comply with legal reqs

Privacy
I will ensure that all user information is kept private. Passwords will be stored in a hashed form rather than plain text, and only the logged-in user will be able to see their own progress and unlocked cards.

Aesthetics and Minimalist Design
I will follow a minimalist design approach so the website looks clean and uncluttered. A focus on Joker-themed

colours and simple layouts will make the application visually appealing without overwhelming the user. Unnecessary information will be excluded

Recognition Rather Than Recall
The interface will be designed so that users can recognise what actions are possible, instead of having to remember complicated steps. For example, unlocked and locked cards will be visually distinct, so users can easily understand their progress at a glance. A search bar will be included to reduce users' cognitive load

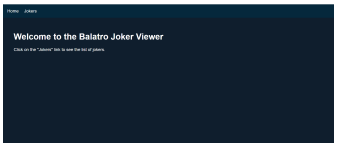
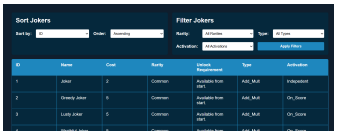
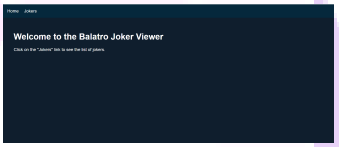
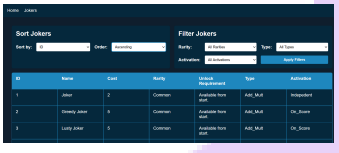
TECHQUITY

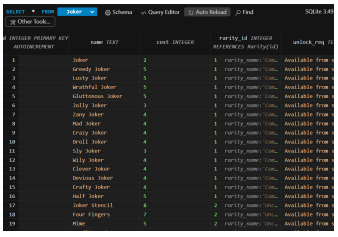
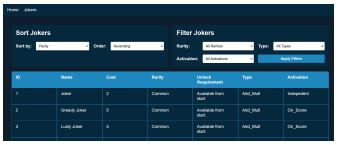
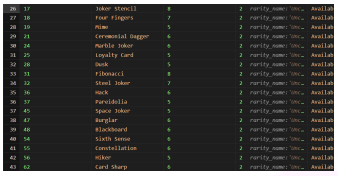
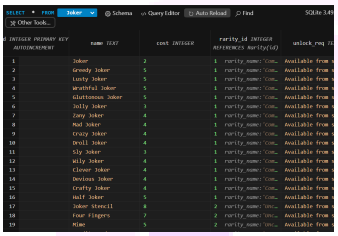
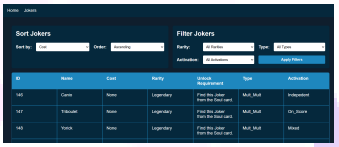
Sprint #1

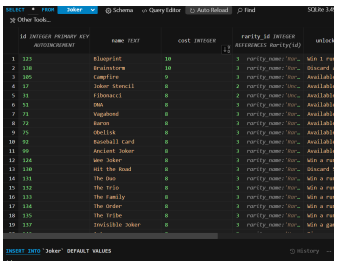
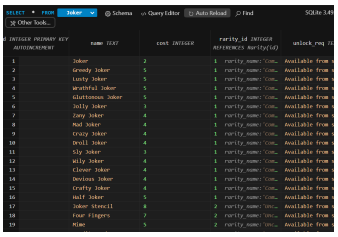
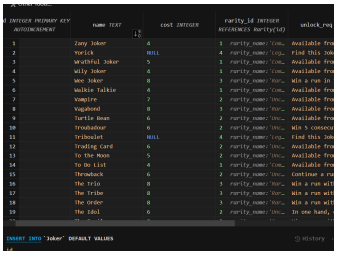
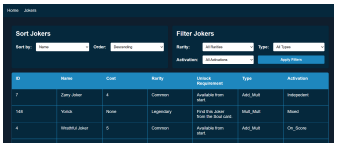
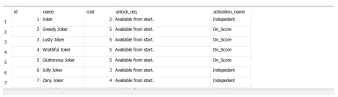
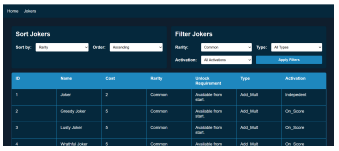
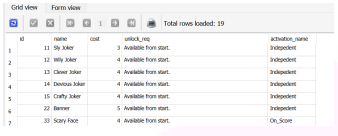
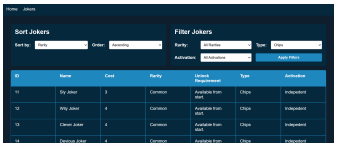
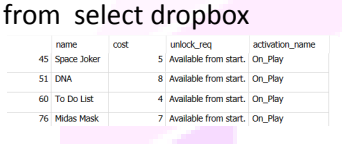
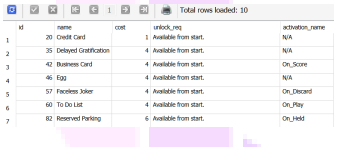
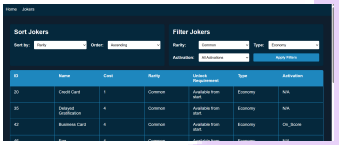
Sprint #1 Goals **(before starting sprint 1):**
-finish the page where it presents all jokers -initialise flask app, database, templates and stylesheet -incorporate sort function for jokers -incorporate select function for jokers

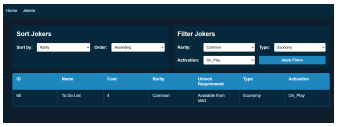
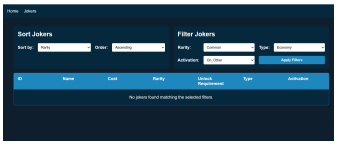
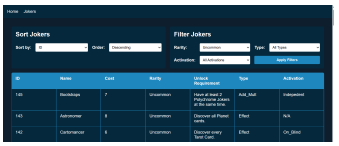
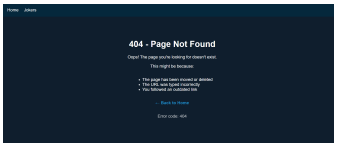
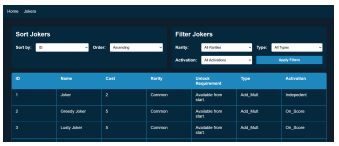
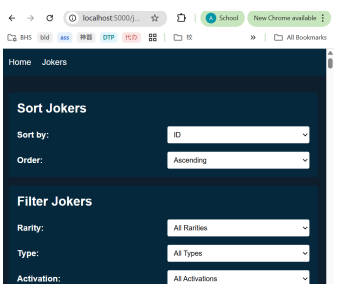
Sprint #1 Testing

Include tests for everything in your application including all links, input forms, correctness of data and correctness of queries. Add more columns as required.

Testing Table				
What are you testing	**How are you testing it**	**Expected Result**	**Result**	**Pass/ Fail**
`Test that the home page loads correctly`	Testing id = http://127.0.0.1:5000/	Homepage shows		pass
`"""Test that the jokers page loads correctly"""`	Testing http://127.0.0.1:5000/jokers	Joker page shows		pass
Navigation between joker and home pages	Clicking on jokers link on home page	Joker page shows		pass
-cont	Clicking on home link on joker page	Home page shows		pass
Sorting function	Id asc from dropbox	Shows id asc		pass

				
	Id desc	Shows id desc 		pass
	Rarity asc	Shows rarity asc 		pass
	Rarity desc	Shws rarity desc 		pass
	Cost asc	Shows cst asc 		pass
	Cost desc	Shows cost desc		pass

				
	Name asc	Shows name asc 		pass
	Name desc	Shows name desc 		pass
Functionality filter for data integrity	Filter rarity	Selecting certain rarities from select dropbox 		pass
	Filter type	Selecting certain type from select dropbox 		pass
	Filter activation	Selecting certain activation from select dropbox 		pass
	Filter combination of two			pass

	Filter combination of three			pass
	No possible jokers	No jokers shown		pass
Sort and filter together	Uncommon with id desc			pass
Nonexistent page	http://127.0.0.1:5000/jokers/nonexistent	404 page		pass
Invalid sort param	http://localhost:5000/jokers?sort_by=invalid&order=asc	Default to valid sorting		pass
Resizable ui	Shrinking window size	UI adjust accordingly		pass

Feedback and comments from user with minimal computer knowledge
Ben(player of balatro): I really like the dark theme as it looks cool and reminds me of the actual game. The joker cards are displayed nicely in rows, which makes it easy to scan through them. I can sort them by different things like cost or name, which is helpful when I'm looking for a specific joker. Some possible improvements suggested: a variety of colour scheme for aesthetics, sprite for jokers for better visualisation – both will be incorporated in sprint 2 Titus: The website is pretty straightforward to use. I like how I can click on the dropdown menus to sort the jokers by different things.The colors are nice and the text is easy to read. I was a bit confused at first

because there were so many jokers and options, but the layout is clean.

Suggestions:
Pixel style font to suit balatro: will be done later
Sprites for cards: sprint 2

Feedback and comments from developers

David Dai:
The application demonstrates solid understanding of Flask framework and web development principles. The code structure is well-organised with proper separation of concerns - using templates for HTML, separate CSS file for styling, and clean Python code for the backend. The database integration with SQLite is implemented correctly using parameterised queries which shows good security awareness. The CSS Grid implementation for the layout is modern and responsive. One area for improvement would be adding more comprehensive error handling and logging. Also, consider adding input validation for the query parameters to prevent potential issues. Overall, it's a well-executed first iteration with a solid foundation.

Suggestions:
Login system: later sprints
Error handling: sprint 2

Jerry Liu:This is a good implementation of a data-driven web application. The use of JOIN queries to combine data from multiple tables shows understanding of relational database concept .The sorting and filtering functionality is well-implemented with proper parameter validation to prevent SQL injection attacks.

Suggestions:
unit test.py for error prevention: no enough time
Login system:sprint 3

Sprint #1 Review

Progress Review (100+ words)

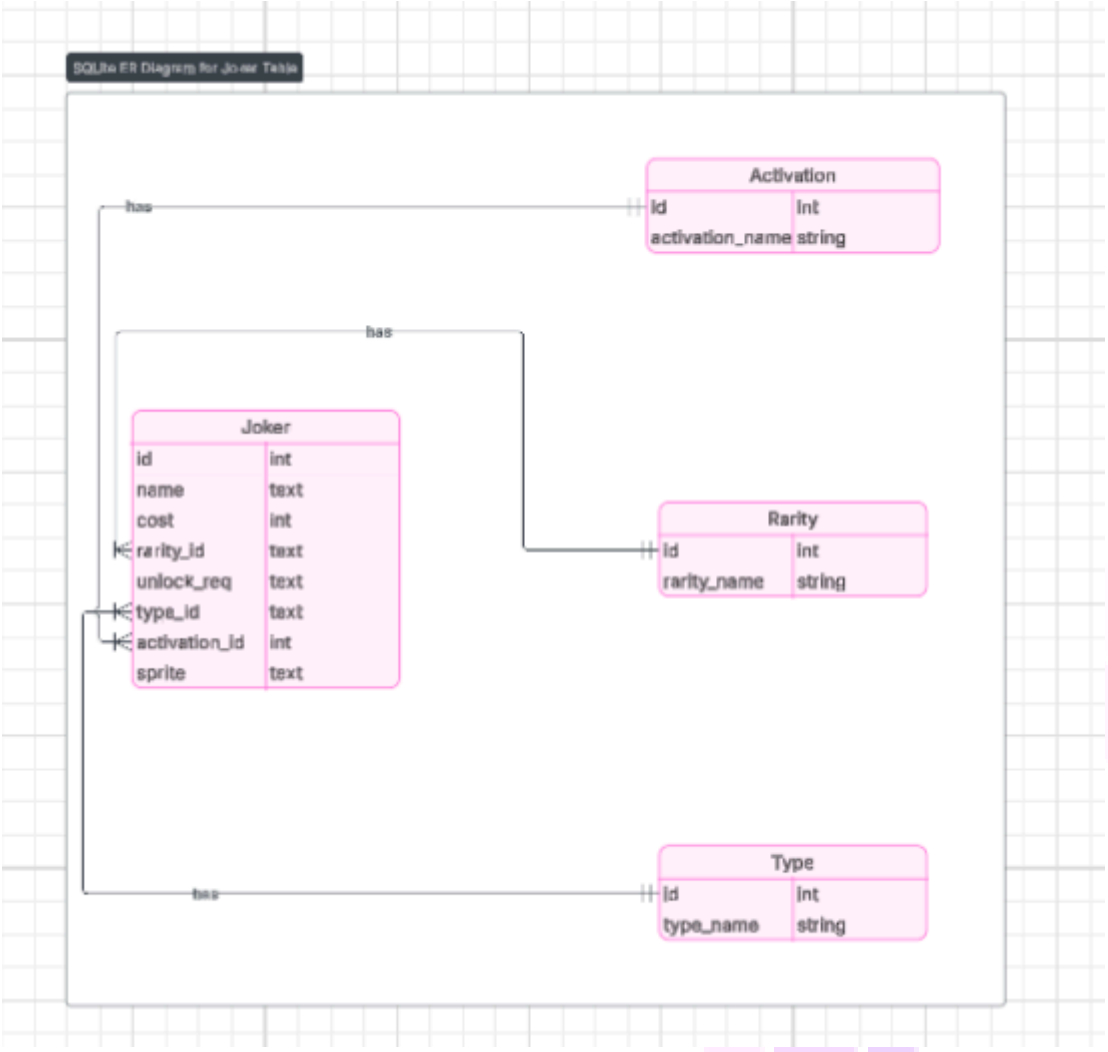
I am satisfied with sprint 1. I have achieved all my goals before starting sprint 1. I have tested cases, and ensured my code is robust. The selection and sorting functions ensured functionality of my app.
In terms of web structure, I moved the nav bar to the top of the page.
In terms of implications, I ensured accessibility across devices by switching to grid html from flex. I ensured aesthetics and minimalist design through appropriate styling and palette. In terms of improvements I could add a search bar to address recognition rather than recall, and add a login system to ensure users' privacy while using the app.

Sprint #2

Sprint #2 Goals **(before starting sprint):**
Add a search bar to the jokers page Include sprites for jokers Add balatro font

Database improvements (if any)

Entity Relationship Diagram

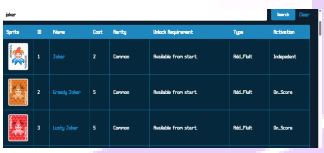


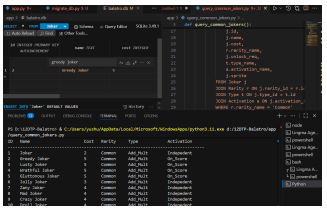
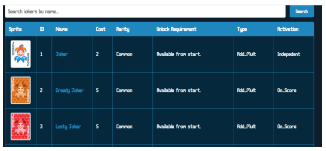
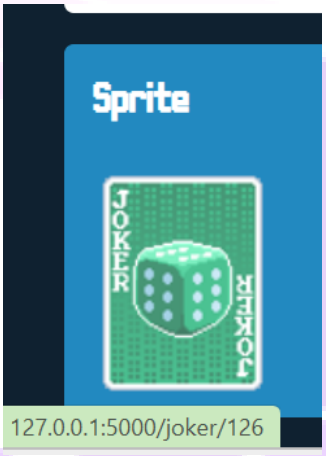
WebPage Design Improvements (if any)- Wireframe/Sketches/Pallette/Font etc

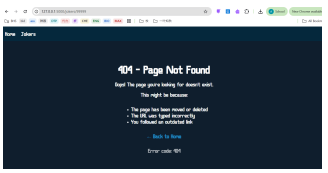
Font used:

Sprint #2 Testing

Include tests for everything in your application including all links, input forms, correctness of data and correctness of queries. Add more columns as required.

Testing Table				
What are you testing	**How are you testing it**	**Expected Result**	**result**	**Pass/Fail**
Search bar- normal general query	inputting”joker’			

Specific query	Input”greedy joker			
In combination with filters	Joker with asc id, rarity=”Common			
No query	blank	No search queries applied		
Special characters	apostrophe	No result		
Max input length/input with no supposing output	100xAs	No result,doesn’t go off the screen		
numbers	Input 6	Outputs joker with 6 in name		
Sql injection test	'; DROP TABLE Joker; --	Nothing should happen		
Joker sprites- redirect	Clicking on sprits	Directs you to individual page		
Links correct	Bottom left preview			

Joker non existent id/out of range	/99999	404 page		

From Casual users:

Titus: The search function is super helpful when I'm looking for a specific joker. Clicking on the joker pictures to see more details is a nice touch. The website is easy to navigate and looks great on both my computer and phone.

ERic: I'm impressed by how the search feature works so well! I tried typing in weird characters and even some code-like stuff, but it didn't break anything. The website looks really nice with all the joker images, and clicking on them takes you to a special page with more information.

Feedback and comments from developers
David Dai: This implementation demonstrates a solid understanding of SQL injection prevention through parameterized queries. The Flask application correctly separates SQL code from user input by using placeholders (?) in the database queries. This approach is significantly more secure than string concatenation methods. Additionally, the input length limitation (100 characters) provides a reasonable safeguard against potential abuse while still allowing meaningful searches Suggestions: Login system: sprint 3 comments : sprint 3   Jerry Liu: The Flask routes handle business logic appropriately, while templates manage presentation concerns. The CSS Grid implementation for the control panels is well-executed, though I would recommend considering semantic HTML tables for the main data display as noted in the project specifications.

Sprint #2 Review

Progress Review (100+ words)
I am satisfied with sprint 2. I have achieved all my goals before starting sprint 2. I have tested cases, and ensured my code is robust.

During the sprint I fixed two major issues in my code. One being sprite ruining the layout of the app, and fontclipping which made documentation hard to read.
No major issues were present in this sprint.
I have added the search bar for better recognition over recall. A specialised font and card sprites is in place for better recognition and aesthetic and minimalist design.. The next step would be to try and incorporate recognition abilities of the search bar in sprint 3 and use a login form to protect user privacy.
In terms of database structure a sprite(TEXT) is added to maintain data relevancy

TECHQUITY

Sprint #3

Sprint #3 Goals **(before starting sprint):**
Add login form to record unlock card states

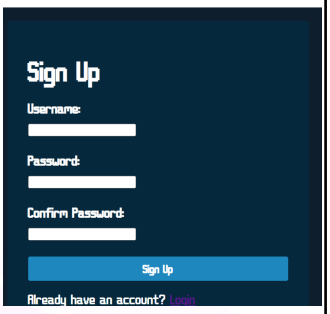
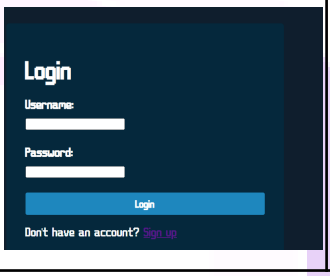
Database improvements (if any)

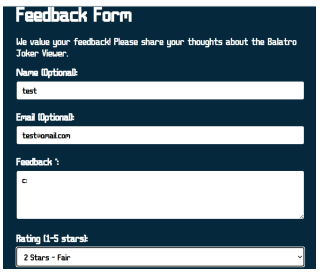
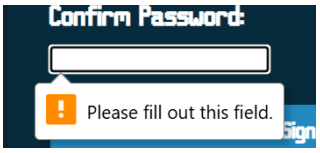
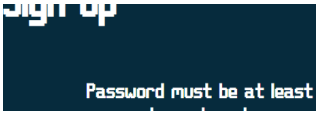
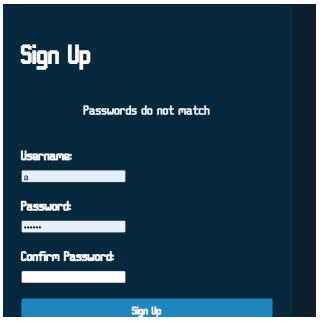
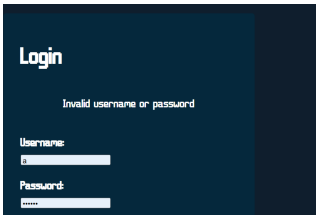
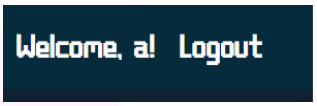
Entity Relationship Diagram

WebPage Design Improvements (if any)- Wireframe/Sketches/Pallette/Font etc

Sprint #3 Testing

Include tests for everything in your application including all links, input forms, correctness of data and correctness of queries. Add more columns as required.

Testing Table				
What are you testing	**How are you testing it**	**Expected Result**	**result**	**Pass/Fail**
Links for signup	Clicking on the link	sign up page		
For login	Click on link	Login page		

For feedback	Click on feedback	Feedback page		
Empty fields	Leaving empty fields	documentation		
Password length	Short password	documentation		
Password matching	Not matching pw	documentation		
Checking pw	Incorrect pw	documentation		
Successful logins	login	Login prompt		

Final Review

What were the results of your testing? Did you get through all that you expected to? What is causing problems? What went well? What was a challenge? What would you do differently if you had a chance?

Discuss how planning, testing and trialling led to the development of a high-quality digital technologies outcome.
Throughout the development process, I tested the core features of the application to ensure they worked as

intended. User registration and login were tested with both valid and invalid inputs to confirm that error messages appeared correctly and that security measures such as password hashing worked. The card repository was tested to check that unlocked cards displayed in colour while locked cards remained greyed out, which made it clear for users to see their progress. Grid layout caused the most trouble but was successfully fixed during testing. Same applied for the login system, which is solved at the end Database queries were tested with different users to confirm that each user only saw their own data. These tests showed that the system is functional, reliable, and meets the original purpose of allowing users to track their unlocked Joker cards. If I had a chance i would try and produce a calculator based on the database, experimenting with testing

How I addressed the Relevant Implications

As discussed before the project, what relevant implications had you identified at the beginning and how did you manage to address them as you went. Were there any others that you came across while making this game? How did you manage to address them? 100+ Words

Final Review of Relevant Implications

During development, I considered several important implications:

- **Legal**: User data is stored securely, with passwords hashed, so no sensitive information is directly exposed. All media used complies to laws
- **Privacy**: Each user's progress is private and can only be viewed by the logged-in account, protecting individual data, implemented through hash

- **Aesthetics and Minimalism**: The interface was kept clean and consistent, using a Joker theme but avoiding unnecessary clutter. Unnecessary information excluded

- **Recognition Rather Than Recall**: The design makes it easy for users to recognise locked versus unlocked cards visually, without needing to remember instructions. A search bar is also included allowing users to search up parts of the name
- Help and documentation: documentation is present when error occurs during login, 404 page etc
- User control and freedom: back to last page link is included in header

TECHQUITY

Marking Schedules: For Teacher use only

AS91892- Use advanced techniques to develop a database

Credits: 4 (internal)
NZQA: https://www.nzqa.govt.nz/nqfdocs/ncea-resource/achievements/2019/as91892.pdf

Achieved- Develop a digital outcome to manage data	**Evidence**	✔
designing the structure of the data	Made an ERD	
using appropriate tools and techniques to structure, organise, query and present data for a purpose and end user	Competently normalized data structure Multiple queries across linked tables Basic Python CRUD functionality with text input/output	
applying appropriate data integrity and testing procedures	Testing Logs. Do the queries work and return what is expected?	
Explaining relevant implications.	Relevant Implications Questions answered appropriately	
Merit- Develop an informed digital outcome to manage data		
using information from testing procedures to improve the quality of the database	Suggested improvements implemented- improve database design, add or refine entities or tables and relationships Made improved ERD(s)	
structuring, organising and querying the data logically	No problems with structure and datatypes in the table. No wildcard queries, JOIN's used efficiently. 2NF at least.	
addressing relevant implications.	The outcome addresses the relevant implications mentioned	
Excellence- Develop a refined digital outcome to manage data		
iterative improvement throughout the design, development and testing process	Improved functionality of database and refinements to structure of the data.	
presenting the data effectively for the purpose and to meet end-user requirements.	Very refined outcome providing all needed functionality	

TECHQUITY

AS91893 - Use advanced techniques to develop a digital media outcome

Credits: 4 (Internal)
NZQA: https://www.nzqa.govt.nz/nqfdocs/ncea-resource/achievements/2019/as91893.pdf

Achieved	**Evidence**	✔
using appropriate tools and techniques for the purpose and end users	Coded a website in HTML and CSS. Looks and functions mostly like intended in the design.	
applying appropriate data integrity and testing procedures	It functions as intended and the right pages show with the correct data when the links are clicked	
using relevant conventions for the media type	Site has basic usability (HCI)	
explaining relevant implications.	Relevant conventions section completed adequately	
Merit		
using information from testing procedures to improve the quality of the outcome	Testing and sprint reviews show evidence of specific improvement made based on the testing. No lorem ipsum anymore!! Accurate Data	
applying relevant conventions to improve the quality of the outcome	Decent HCI. Eg. generally shows “consistency and standards”, and/or “help and documentation” by having an about page.	
addressing relevant implications.	Has met relevant implications like NOT used copyright material, addresses privacy by keeping data safe (eg hashed passwords) etc.	
Excellence		
iterative improvement throughout the design, development and testing process to produce a high-quality outcome	Must be high quality. No obvious errors or inaccurate data.	
using efficient tools and techniques in the outcome’s production.	Kept a good log, used github to keep backups and record of work, lots of commits on github.	

TECHQUITY

AS91896 - Use advanced programming techniques to develop a computer program

Credits: 6 (Internal)
NZQA: https://www.nzqa.govt.nz/nqfdocs/ncea-resource/achievements/2019/as91896.pdf

Achieved	**Evidence**	✔
writing code for a program that performs a specified task	Applications works as intended.	
using advanced techniques in a suitable programming language	“creating methods, functions, or procedures that use parameters and/or return values” flask requires that “responding to events generated by a graphical user interface (GUI) ”- links in webpage run routes that do queries “using additional non-core libraries”- using Flask.	
setting out the program code clearly and documenting the program with comments	Some code comments, code layed out relatively cleanly with constants/variable at the top, routes and functions after.	
testing and debugging the program to ensure that it works on a sample of expected cases.	Testing table shows the basic web page functionality was tested.	
Merit		
documenting the program with appropriate names and comments that describe code function and behaviour	Good variable and function names. Good and plentiful code comments.	
following common conventions for the chosen programming language	PEP8 Followed. Use Pylint or similar and check linting.	
testing and debugging the program effectively to ensure that it works on a sample of both expected cases and relevant boundary cases.	Lots of testing including request for non-existent id returning an error 404 page.	
Excellence		
ensuring that the program is a well-structured, logical response to the specified task	Enhanced program, e.g. functions that aren’t routes to make the code cleaner (eg query_db). Neat and logical code.	
making the program flexible and robust	Great structure making code easy to extend. Unbreakable code. Eg. Error-404 and error 505 handlers	
comprehensively testing and debugging the program.	Unbreakable code with a lot of tests done for every testable element of the program including unexpected input like a non existent route or page returning custom error 404 handler	

TECHQUITY

AS91897- Use advanced processes to develop a digital technologies outcome

Credits: 6 (internal)
NZQA: https://www.nzqa.govt.nz/nqfdocs/ncea-resource/achievements/2019/as91897.pdf

Achieved	Evidence	✔
using appropriate project management tools and techniques to plan the development of a digital technologies outcome	Github used. Goals defined. Good commit messages.	
decomposing the outcome into smaller components	Evidence by regular commits with good messages describing the completed task	
trialing the components of the digital technologies outcome	Completed tasks committed should have been tested	
testing that the digital technologies outcome functions as intended	The whole application should be tested between sprints	
explaining relevant implications.	Relevant implications section complete	
Merit		
effectively using project management and version control tools and techniques to manage the development of a digital technologies outcome	Plenty well named commits done regularly (eg not a rush at the end or large period of inactivity seen in git log)	
trialing multiple components and/or techniques and selecting those which are most suitable	As above with larger number of small components tested and committed	
using information appropriately from testing and trialling to improve the functionality of the digital technologies outcome	Specific examples given in end of sprint reviews showing how they found something out in testing and improved their outcome as a result. Can be observed too.	
addressing relevant implications.	Application meets all relevant implications. Watch for non-copyright material, offensive material, poor functionality or aesthetics despite these issues being mentioned in testing.	
Excellence		
discussing how the information from planning, testing and trialling of components assisted in the development of a high-quality outcome.	There is a section at the end specifically asking this question.	