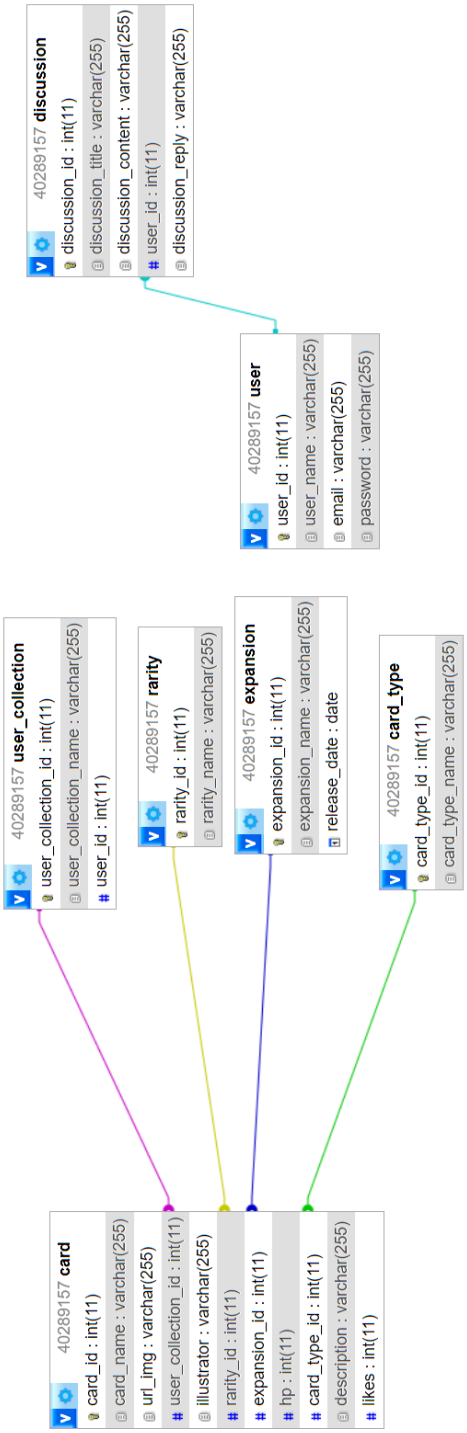
CSC7062 Tradecard Report

Contents



ER Diagram from Database

How to work the website

- Download XAMPP for running Apache and MySQL on ports 80,443 and 3306 respectively.
- Create a database and name it ‘40289157’.
- Import my database schema which can be found under ‘40289157 (3).txt’ in my Github repository, making sure that the data in each table matches the database schema and that the designer tab matches my ER diagram previous.
- No need for user accounts to be created as the username to access the database is ‘root’, the host is ‘localhost’ and there is no password.
- Once the database is created and running on Port 3306 using the information given, open Virtual Studio Code and in a new terminal, enter ‘cd tradecard-api’ and from here, enter ‘node server.js’ to connect the database to the API, I used Postman for this. If done correctly, the terminal should read ‘API started on port 4000, connected to local mysql db using. env properties’.
- To check if the database be accessed on Postman, enter ‘http://localhost:4000/carddata’ into an HTTP GET request and this should display all data from the database on each of the 40 cards.
- Then, once data can be accessed on Postman, it is time to run the web app itself. In another new terminal enter ‘node app.js’ which will run the website. If done correctly, the terminal should read ‘Server is running on http://localhost:3000’.
- From here, enter ‘http://localhost:3000/’ into a web browser and this should take you to the route of the website, which is my index page.
- Using the navbar, the ‘Sign In’ and ‘Search Cards’ buttons as well as the popular cards carousel, you can now navigate freely around my web app.

Database Construction

In the development of my card database, I implemented several features to ensure efficient data management and scalability. Initially, the primary table, the "card" table, was designed to store a substantial amount of data, accommodating the expansive nature of the dataset. I decided to source the data from the TCG Dex GitHub repository but limited the scope to only the first four expansions: Base Set, Fossil, Jungle, and Base Set 2. To maintain a manageable dataset while still ensuring that a range of cards were stored, I used a random number generator to select approximately 10 cards from each expansion. Analysing the dataset provided on GitHub, I identified key attributes to include in the card main table. These included the card name, URL of the image, illustrator, rarity, expansion, hit points (HP), card type, and a description. To enhance the functionality of my database, I also added fields for user collection and likes. This would help to assign an individual card to a user. To adhere to database normalization principles, as suggested by Codd's (1970) rules on normal forms[1], I created separate tables for attributes such as rarity. Each of these smaller tables was assigned a primary key which also functioned as a foreign key in the card table. This approach to normalisation facilitated easier data entry and helped maintain the integrity of the data by reducing redundancy and dependency. Through this, the database was structured to be both robust and efficient, supporting a wide range of queries and data interactions.

Outside of the card table, the expansion table was designed to store their release data and their expansion name. Upon reflection, this could be further enhanced through storing a URL to an image of each expansion which could be stored locally in the database. This would allow for easy retrieval directly at the website level, akin to how individual card data is handled in the card table. Additionally, the user table in the database is structured to include essential information such as username, email, and password and to further refine organisation, the user table is normalised with the user collection table. This normalisation ensures that each user is not only registered with their personal information but is also automatically assigned a collection of cards upon sign up at website level just like existing users.

[1] Codd, E.F. (1970) 'A relational model of data for large shared data banks', *Communications of the ACM*, vol. 13 (6), pp. 377-387.

HTML – Front End

After completing the initial database design, I transitioned to structuring the website by delineating the types of web pages required for a comprehensive user experience. This stage involved creating a diverse set of pages: a homepage (index), a specific page for displaying cards, individual pages for different expansions, sections for user collections, and a sign-in page. Although plans were made to design a discussion page to facilitate user interaction, this component was not implemented in the backend during the initial phase.

Each page was meticulously designed to maintain a consistent aesthetic, featuring a uniform navigation bar (navbar), footer, and a dynamic jumbotron whose appearance is tailored to the specific content of each page. For instance, the cards page is distinguished by a jumbotron with a deck of cards background, visually reinforcing the theme of the page. The navigation strategy was particularly focused on enhancing usability and ease of movement across the website. The navbar on the website includes an 'active' feature that highlights the page you're currently on. This makes it easier to see where you are on the site, which helps improve how user-friendly the website is. This idea follows a well-known design guideline that recommends making things clear and easy to navigate for users, coming Nielsen's (1994)[2] principles for designing easy-to-use interfaces.

Moreover, a dropdown menu in the navbar labelled 'expansions' was used, offering users the ability to select from four expansions, enhancing the interactive experience. This consistent design elements across all pages promotes a sense of familiarity and also facilitate continuity from me, the developer. Overall, this template layout of each web page allowed me to scale the size of the website.

Below each page’s jumbotron, each page presents its main content, which is tailored to the specific context of the page. For example, the index page features a carousel showcasing popular cards, which were planned to be linked to each card’s card details page. This design choice not only leverages the carousel’s ability to highlight featured content effectively but also utilises the link to card details pages to provide more information on our popular cards. This well-planned approach to designing the website, based on well-known principles of good design and usability, has been key in making the website easy to use and pleasant to navigate. It not only meets the basic needs of the website but also makes visitors more engaged and

[2] Nielsen, J. (1993) *Usability Engineering,* Morgan Kaufmann Publishers.

satisfied. This method highlights how important it is to carefully think about design in digital spaces, considering how users expect to interact with the site to make sure they have a good experience.

CSS – Front End

Throughout, I used Bootstrap along with my custom stylesheet, `mystyle.css`, to enhance the visual and structural elements of the website. Bootstrap provided a foundation for creating a responsive design, ensuring that the website looks good on devices of all sizes, from smartphones to desktops. This is crucial as it supports user engagement across various platforms, a key aspect of modern web design principles (Marcotte, 2017)[3]. My own CSS file, `mystyle.css`, was instrumental in fine-tuning specific aesthetic details, such as the size of forms and the implementation of adequate spacing and margins. While being rather repetitive, these adjustments significantly impacted the overall look and usability of the site, making it more user-friendly and appealing. Again, I followed a consistent aesthetic, using a consistent, light blue background while trying to balance colours that harmonise to create an appealing and effective user interface. The strategic use of CSS in this project highlights its indispensable role in web development. CSS not only aids in styling but also plays a pivotal role in functional aspects like responsiveness and user experience enhancement. This highlights how crucial CSS is for creating websites that are not only easy to use and access but also attractive and consistent.

JavaScript – Back End

Initially, the application successfully utilized the `.env` file to manage environmental variables. However, due to complications with database connectivity, the decision was made to hardcode these variables directly into the codebase. For instance, the username 'root' was explicitly specified within the `app.js` file, replacing the flexible approach that relied on environment-specific configurations. This modification in `app.js` established a direct interaction with the `server.ejs` file, which was responsible for defining the routes utilized by the application. These routes facilitated the retrieval of data from Postman, effectively integrating this external data into the website's functionality.

Despite the successful integration, certain data presentation issues arose. For example, the `rarity_id` field, initially displayed merely as a raw number like '1', was not very user-

[3] Marcotte, E. (2017). Responsive web design: A book apart.

friendly or informative. To enhance the user experience and provide clearer insights into the data, a mapping strategy was implemented. This strategy translated these abstract numeric identifiers into more descriptive labels, such as 'Rare Holo.' This adjustment not only improved the aesthetics and readability of the website but also made the data more accessible and understandable to users.

References

External sources:

https://csc7062-2023-24.gitbook.io/lab-02 - Intro to CSS Frameworks.

https://csc7062-2023-24.gitbook.io/lab-04/express.js/express-web-app - Intro to Express.

https://csc7062-2023-24.gitbook.io/lab09 - API and Node.

https://csc7062-2023-24.gitbook.io/lab10/set-up - Rest API and Postman.

https://dev.to/victrexx2002/introduction-to-ejs-a-guide-to-building-dynamic-web-applications-2737 - Integrating data into a template based off card_id.

https://developer.mozilla.org/en-US/docs/Learn/Forms/Basic_native_form_controls - Form controls with buttons.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Map - Mapping data from database to display meaningful data rather than raw numbers.

https://github.com/tcgdex/cards-database/tree/master/data/Base - Card data and urls

https://www.w3schools.com/nodejs/nodejs_mysql_insert.asp - Adding to a database from JS.

Dependencies:

- nodemon 3.1.0
- body-parser 1.20.2
- express 4.19.2
- dotenv 16.4.5
- mysql2 3.9.7
- axios 1.6.8
- ejs 3.1.10
- https://getbootstrap.com/docs/4.5/getting-started/introduction/

Github Repository URL:

- https://github.com/jacksonmckee/tradecard