Vol: 2, Issue: 1, 2024
ISSN: 2980-3152
Pages:48-59

CURRENT TRENDS IN COMPUTING
https://dergipark.org.tr/en/pub/ctc
RESEARCH ARTICLE

THE GATHERING DECK BUILDER WITH REACT.JS AND CUTTING-EDGE WEB DEVELOPMENT

DANIEL MCCLOY[1], KEVIN BRYANT[1] AND YOUSEF FAZEA[1*]

[1] *Department of Computer Sciences and Electrical Engineering, One John Marshall Dr., Huntington, WV 25704, USA*

ABSTRACT. **This paper comes with a new web-based application "MtGDeckBuild". The main challenge addressed by this application is the integration of complex data management and user authentication systems within a responsive and interactive web interface. MtGDeckBuild leverages React.js, Node.js, and SQL Server to tackle these challenges. It uses Scryfall's full card data and Frontegg's secure user authentication APIs. By using the RAD paradigm, the article was able to successfully include several elements, such as user identification, responsive design, interactivity, and dynamic data display. The main achievements include the seamless incorporation of secure user authentication, efficient data management, and a scalable architecture, which significantly enhances the user experience and application performance. The prototype showcases the team's skill in obtaining and deploying new technologies and highlights the need to efficiently handle third-party dependencies. Improving functionality, increasing efficiency and scalability, and exploring greater IoT integration inside Smart Cities utilizing advanced web development frameworks are some of the prospects.**

1. INTRODUCTION

Although web development is a dynamic field, initiatives such as MtGDeckBuild illustrate how user-centered design and cutting-edge tools can simultaneously accommodate niche audiences and the general public. We will examine the intersection of frictionless user experiences and the rapid expansion of digital systems in this study. The foundations of initiatives like MtGDeckBuild can be traced back to previous research that prioritized user identification, dynamic data, and flexible design [1]. The findings of this study hold significant ramifications for application design across various domains, encompassing strategies for leveraging emerging technologies to enhance application performance and user engagement. React.js, SQL Server, and Node.js comprise the foundation of MtGDeckBuild's architecture. This paper demonstrates how React.js can be effectively used to create robust, scalable, and responsive web applications. Although primarily focused on web development, the study also touches upon the potential integration of IoT systems within Smart Cities, showcasing how web technologies can support advanced

E-mail address: yousef.fazea@marshall.edu (*).
Key words and phrases. JavaScript library, Front-End development, Virtual DOM, React JS..

urban infrastructures. MtGDeckBuild utilizes Frontegg and Scryfall, third-party APIs to ensure a seamless user experience and robust data management [2, 3]. By employing expertise and the most efficient web development methodologies, initiatives like MtGDeckBuild expedite the development of digital infrastructures. To eliminate risks, ensure the stability and scalability of the program, and address the difficulties of managing and regulating dependencies, additional R&D is required. Making this update effortless is the well-known server-side web application framework React.js. An improvement in customer happiness, an improvement in display quality, and a simplification of developer work are the three primary benefits of the product.

This paper is organized as follows: Section 2 introduces the literature, Section 3 presents the methodology and development process, followed by the findings and discussion in Section 4. The paper is concluded in Section 5.

2. Related Work

Web applications are made more useful and faster with the help of React.js since it incorporates capabilities like routing 6. As smaller component-based apps bring this framework online, the architecture, structure, and style of the primary program are retained [4,5]. With its crossover application concept, the React.js framework makes it easy to combine server-side and client-side websites [6–9]. This method allows programmers to construct robust JavaScript web applications without being concerned with the intricacies of the backend. While this approach is simple, it has to be fine-tuned for speed to meet the high expectations of e-commerce customers who want perfect online buying experiences. The second iteration of the Digitalization & Sustainability Review measures the amount of time it takes to provide data, styles, and code to clients, with an emphasis on how crucial it is for developers to use tactics that enhance the efficiency of React.js apps [10, 11]. For instance, sports-related web apps have complex tasks, such as collecting and processing many game videos and generating data during page load, which might hinder optimization and performance. Furthermore, conventional web development approaches can produce inefficient monolithic apps; in contrast, React.js's component-based design is quite different. Architectures based on cutting-edge parts also tend to use antiquated methods of web development, which leads to less performant monolithic apps. React.js has revolutionized online application development by introducing substantial improvements, making it the optimal solution for creating web apps that are scalable, easy to maintain, and deliver exceptional performance [12–14]. Finally, utilizing React.js in online shops and apps overcomes obsolete technology and capitalizes on new buying patterns. User-centered design and cutting-edge technology may boost customer satisfaction and help companies compete in the e-commerce industry [15–17]. This literature analysis emphasizes the drawbacks of traditional methods as well as React.js' advantages.

3. METHODOLOGY AND DEVELOPMENT PROCESS

Rapid Application Development (RAD) is used for the design, testing, and monitoring of study prototypes. The process has a few sequential steps: requirements, design, execution, validation, and maintenance. Efficient team communication is facilitated when everyone has a clear understanding of their

designated job. Consistently document and oversee tasks to guarantee excellence. This website seeks to actively involve and cater to those who have a strong interest in Magic via its goals, methodology, and range of coverage. Figure1 displays the website for Magic: The Gathering, which is built on the RAD approach.

- Frontegg's user identification Software-as-a-Service (SaaS) is unparalleled. Users may now access and use Software as a Service (SaaS) via internet platforms.
- Frontegg provides authentication services for website administrators. Users have the option to either login or use Single Sign-On (SSO) to access the site.
- Utilize Frontegg's robust authentication system to limit access to site resources. Implement multi-factor and adaptive risk-based authentication to deter illegal access. API calls include the manipulation of dynamic data and encompass several application technologies. Here are the specific details of what we offer:
  - Frontegg is a sophisticated platform for user authentication. Frontegg may be used to authenticate API requests.
  - The card data API is the recommended method for accessing magic-related information on Scryfall. The app can display Scryfall card information by using its API.
  - Axios is an excellent JavaScript library that is interoperable with both browsers and nodes. JavaScript queries sent to websites. Axios can establish communication with external APIs, applications running on the server side, and databases. Frontegg, Scryfall, and Axios are reliable choices for app developers looking to make API calls. These technologies enable the application to verify the identity of users, conduct tests on API requests, get data from other sources, and establish communication with other apps and services.
  - Sign in to create and distribute presentations. The use of a responsive stylesheet enhances the app's performance and facilitates its usability on mobile devices. The app's layout adapts to the size of the viewport and is optimized for mobile devices, which improves the user experience. Adopting a flexible design ensures that mobile users do not experience delays or have difficulties with navigation. Enabling users to use their increasingly prevalent mobile devices to access the internet might enhance the app's value.

3.1. **List of Requirements:**

Two tables were created from the process's results: Table 1 presented the functional requirements of the study, whereas Table2 delineated the non-functional requirements. The priority and level of relevance of each need are indicated by the same scale in both tables:

- M: Mandatory requirements (essential functions the system must fulfill).
- D: Desirable requirements (functions that are preferred to be included).
- Optional requirements (functions the system may include).

3.2. **Activity and Sequence Diagrams:**

The user proceeds to the login page of the Software-as-a-Service (SaaS) application and enters their credentials. Front-egg verifies the credentials and produces an access token. This token is returned to the SaaS application to verify the user's identity and authorize access to the primary interface. The process

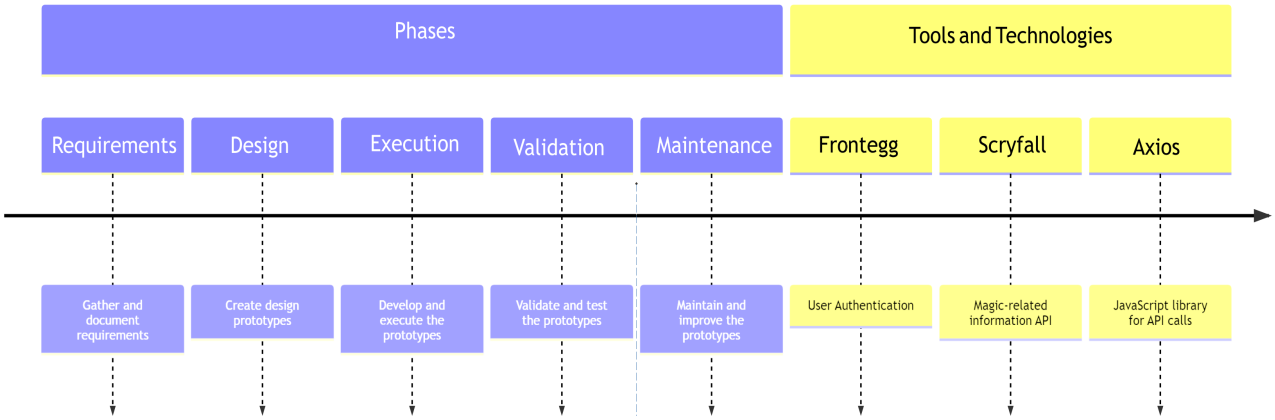

FIGURE 1. **Rapid Application Development Model [18].**

TABLE 1. **Functional Requirements.**

1	MtGDB_01	Login	
	MtGDB_01_02	The User logins in via username and password	M
	MtGDB_01_03	The User logins in via Google Account	M
	MtGDB_01_04	The User logins in via the GitHub Account	M
	MtGDB_01_05	The User logins in via the Microsoft Account	M
	MtGDB_01_06	The User logins in via the Facebook Account	M
2	MtGDB_02	**Deckbuilding**	
	MtGDB_02_01	The User queries a card	M
	MtGDB_02_02	The User adds the queried card to the checklist in a specified quantity	M
	MtGDB_02_03	The User adds the Deck Name and submits the deck	M
3	MtGDB_03	**Deck Viewing**	
	MtGDB_03_01	The User selects a deck from the Recent Decks component	D
	MtGDB_03_02	The User selects "View Decks" from the Navbar component	D
4	MtGDB_04	**Logout**	
	MtGDB_04_01	The User logs out via the "Logout" button from the Navbar component	M

of login is shown in Section (a) and visualized in Figures2 (a) and (b). To conduct a card search, the user enters a string into the designated search field, which initiates an auto-complete feature that produces a compilation of proposed card names. As soon as the user enters a card name, the application loads the

TABLE 2. **Non-Functional Requirements.**

1	MtGDB_05	Security	
	MtGDB_05_01	The system should be secure by specifying the user's tasks and not allowing an unauthorized person to use the system.	M
2	MtGDB_06	**Usability**	
	MtGDB_06_01	The System must be easy to deal with.	D
3	MtGDB_07	**Understandability**	
	MtGDB_07_01	The System must be easy to understand	D
4	MtGDB_08	**Reliability**	
	MtGDB_08_01	The system should present the same selected sequence.	D
5	MtGDB_09	**Performance**	
	MtGDB_09_01	The system must have a reasonable speed according to the technology used to access many Users at the same time.	O
	MtGDB_09_02	The system spends on searching must be less than 2 seconds	D
6	MtGDB_10	**Availability**	
	MtGDB_10_01	The system should be available to all kinds of Users	D
	MtGDB_10_02	The system must not crash and if that happened, it should take less time to be back again.	D

image that corresponds to that card. The process of card searching is shown in Section (b) and visualized in Figures3 (a) and (b).

(a) **Login:**

- The user navigates to the login page of the SaaS application.
- The SaaS application sends a request to Frontegg to initiate the login process.
- Frontegg presents the user with a login page, where they enter their email address and password.
- The user submits their login credentials.
- Frontegg validates the user's credentials and generates an access token for the user.
- Frontegg sends the access token back to the SaaS application.
- The SaaS application uses the access token to authenticate the user and grants access to the application's resources.
- The user is redirected to the SaaS application's main dashboard or home page.
- This activity diagram illustrates how Frontegg's authentication and identity management capabilities can be used to secure access to SaaS applications. By leveraging Frontegg's services, developers can focus on building their core applications, rather than worrying about the complex details of authentication and access control.

(b) **Search Card Activity Diagram:**

- The user navigates to the Search Card component of the application and inputs a string into the search field.
- The application triggers an auto-complete method to generate a list of suggested card names based on the string input by the user.
- The application displays the list of suggested card names in a drop-down menu beneath the search field. The user selects a card name from the list.
- The application loads the corresponding card image associated with the selected card name.
- he application displays the card image on the page.

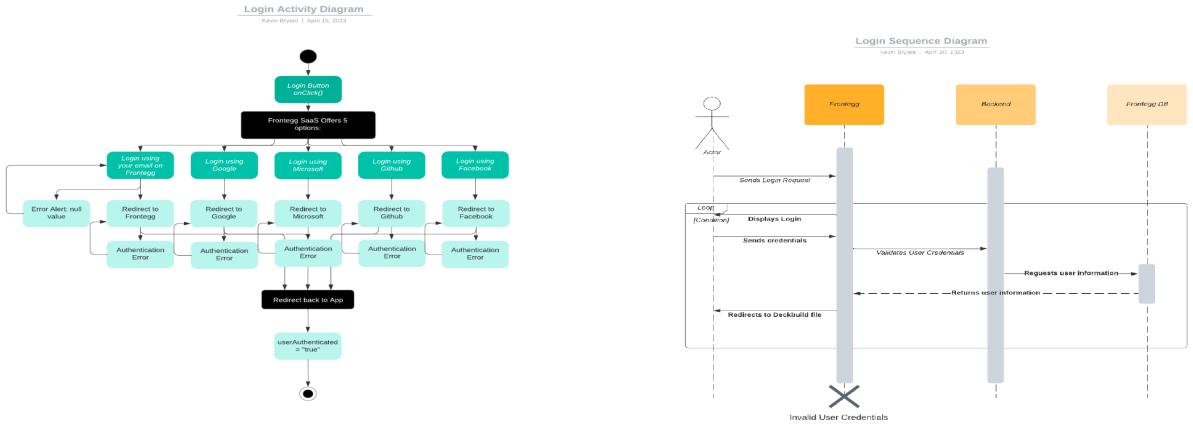

FIGURE 2. **(a) Login activity diagram, (b) Login sequence diagram.**

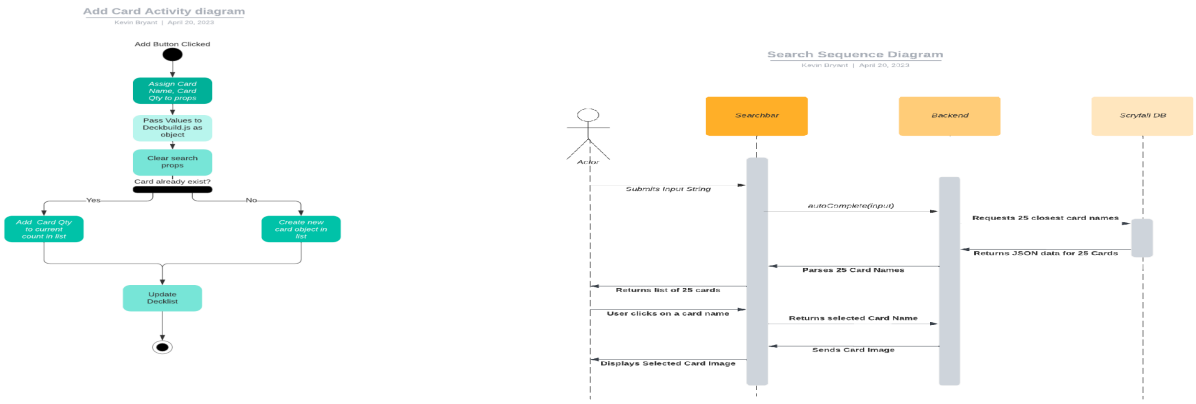

FIGURE 3. **(a) Add and remove card to deck activity diagram, (b) Search card sequence diagram.**

(c) **Case Diagram and Geographical User Interface:**

According to Figure4, a user may do a variety of functions in the system, including logging in, searching for cards, adding cards, removing cards, and disengaging from the system. The fact that these activities are confined inside the system's limits implies that they are system-provided characteristics. The user interacts with each of these skills in line with their needs. Following the completion of the design of each diagram, the next step is to create the website by utilizing React. JS. Figures4,5,6, and 7 illustrate the proof of concept that was found.

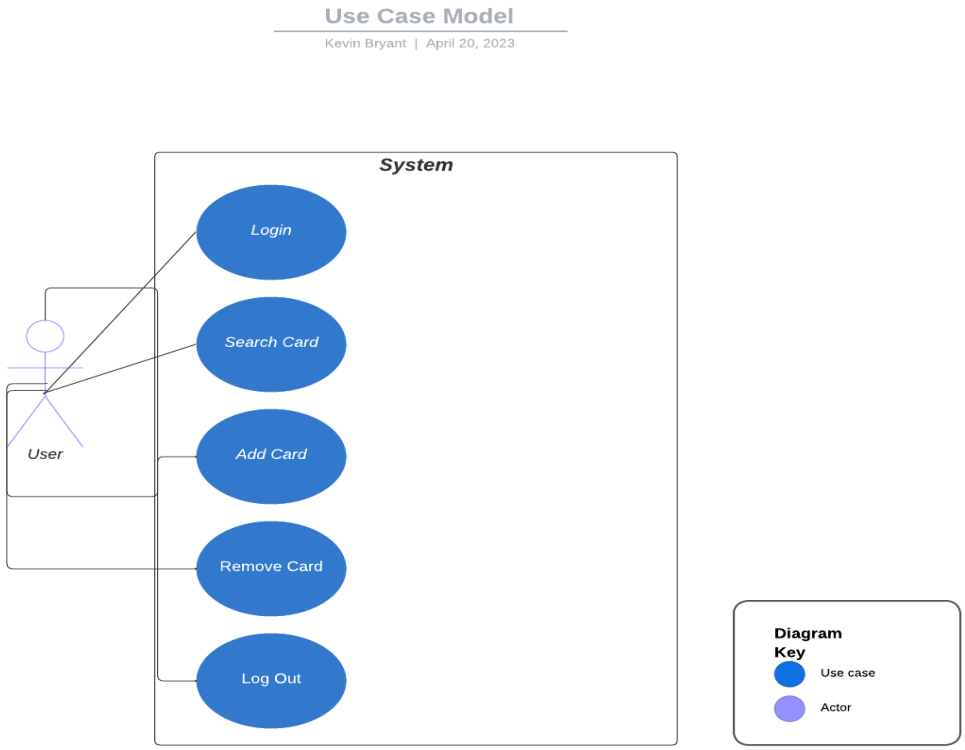

FIGURE 4. **Login page (With Feedback Messages).**

4. RESULTS AND DISCUSSION

To assess the usability of the prototype, data was collected from a sample of thirty distinct participants. We have assessed the Web-based application system utilizing criteria such as Intuitive Design, Easy Navigation, Memorability, and Satisfaction. Figure8 (a) shows the result of a question asked about the

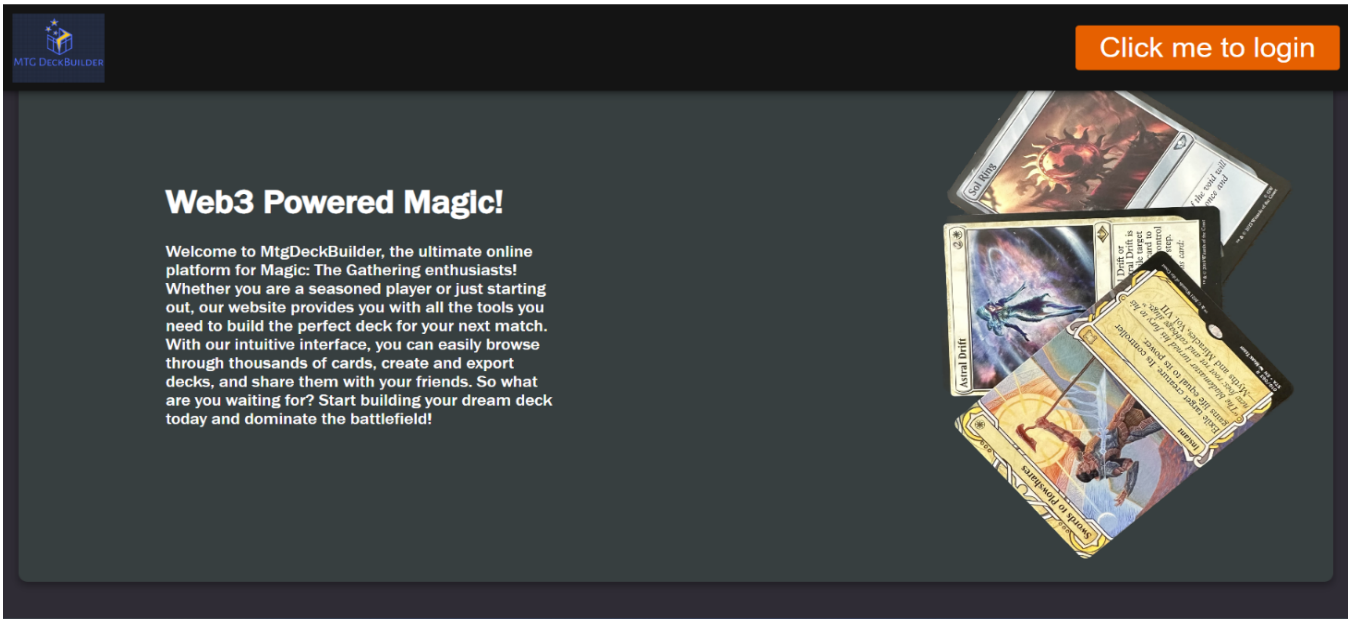

FIGURE 5. **Home page.**

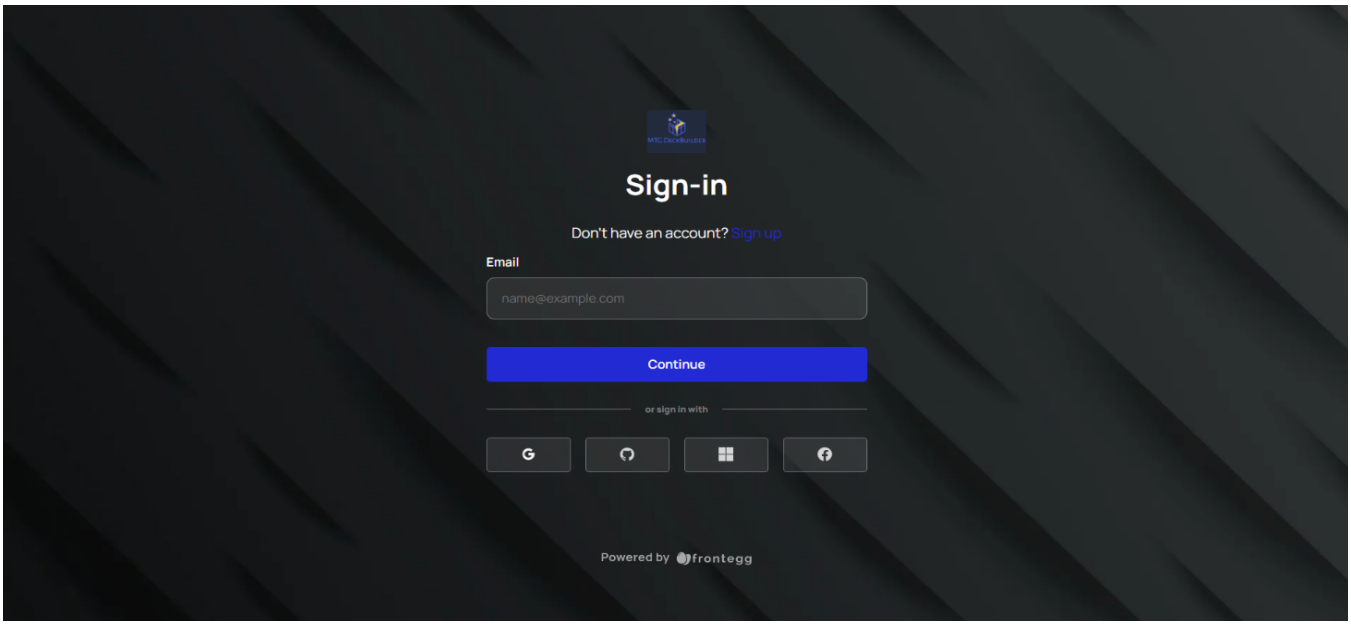

FIGURE 6. **Login page.**

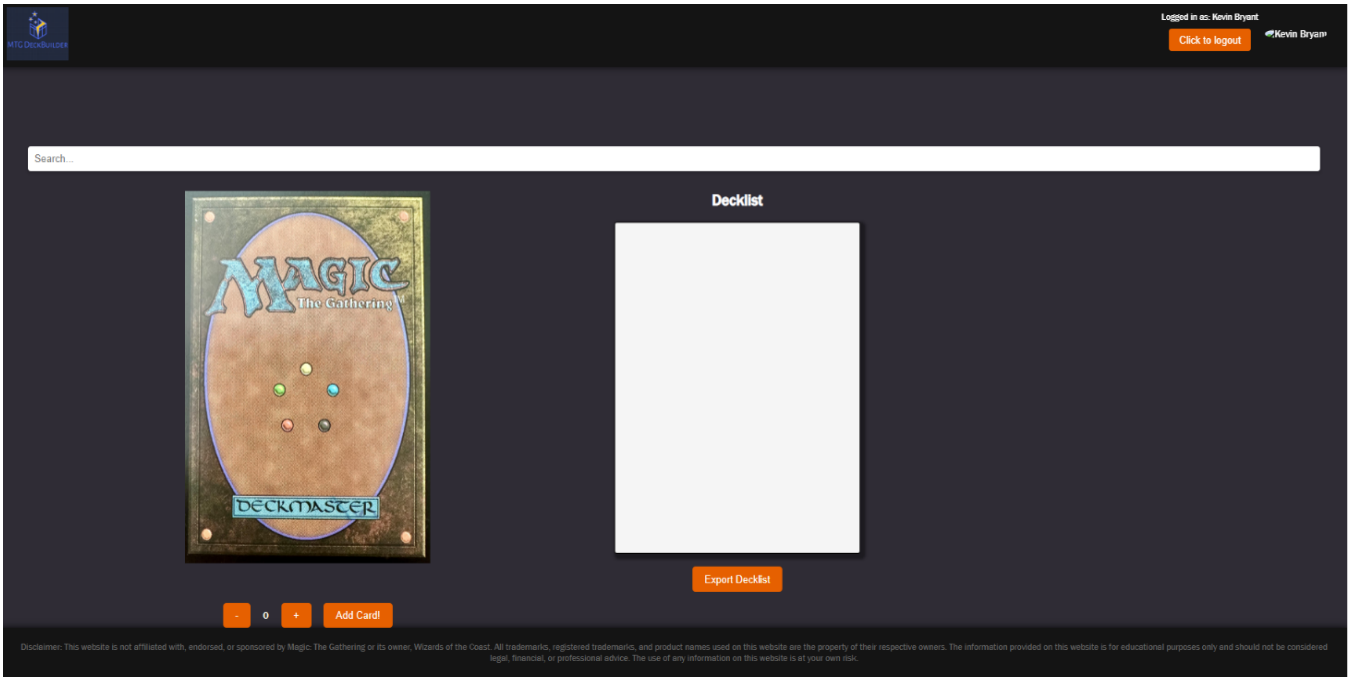

FIGURE 7. **Searching page.**

intuitive design, a nearly effortless understanding of the architecture and navigation of the site. The responses were shown using a Likert scale, where a score of 1 indicated "Strongly Disagree" and a score of 5 indicated "Strongly Agree." The bar chart illustrates the allocation of these scores. The ratings "Strongly Disagree" and "Disagree" both received a score of 0%. Only 5% of the survey respondents selected group 3, indicating their indifference. 20% of the poll respondents agreed with the statement, classifying them into group 4. Category 5 received 75% of the vote, indicating a widespread consensus among most individuals. Figure8 (b) shows the ease of navigation, and how fast users can accomplish tasks. A significant majority of respondents expressed strong agreement with the statement, indicating a largely positive consensus. Figure8 (b) illustrates that no users had any difficulties with navigation. Category 3 represented a minuscule proportion of individuals who had a neutral stance, amounting to just 5%. Unsurprisingly, 20% of the study respondents rated the navigation as straightforward, categorizing it as category 4. The data from Category 5 indicates that most respondents 75% found the navigation to be straightforward. Based on these statistics, most visitors have a positive perception of the site's navigational simplicity.

The bar chart Figure9 (a) measures the site's memorability after visiting the site if a user can remember enough to use it effectively for future visits. Only 5% disagreed, suggesting minimal trouble remembering site characteristics for future visits. 15% of users were unclear if the site was memorable. With a 25% agreement, many consumers remembered the site well enough for future visits. The fact that 55%

of users agreed suggests that most visitors liked the site and could recall things for their future visits. These figures suggest that most users find the site simple to remember and can recall details from past visits. Figure 9 measures user satisfaction where the users were asked if they like using the system and find it friendly. 30% agreed indicating good satisfaction. Most responders 70% strongly agreed, suggesting high site satisfaction. According to these statistics, most users are satisfied with their experience, suggesting a positive user impression.

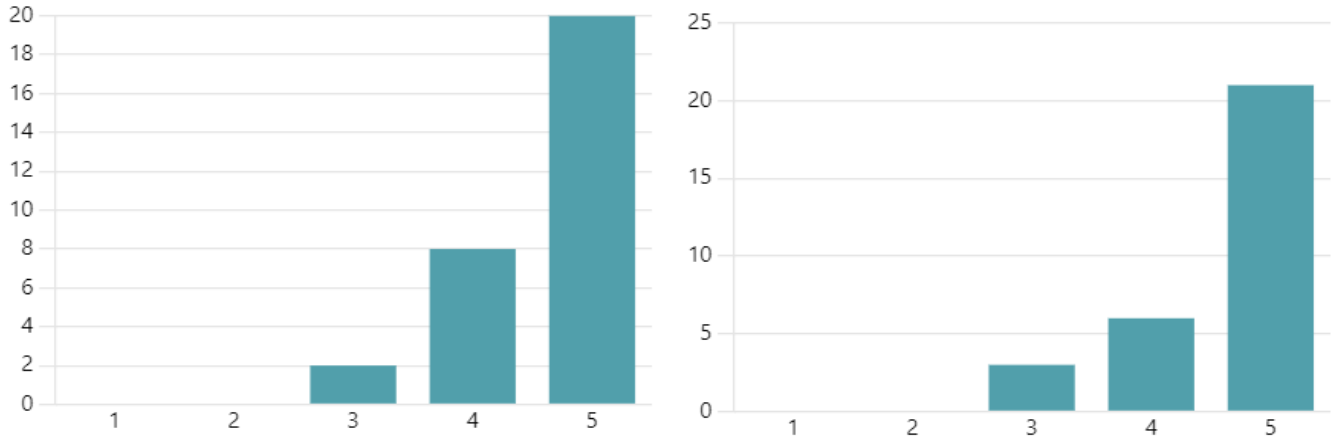

FIGURE 8. **(a) Intuitive design (b) Easy navigation.**

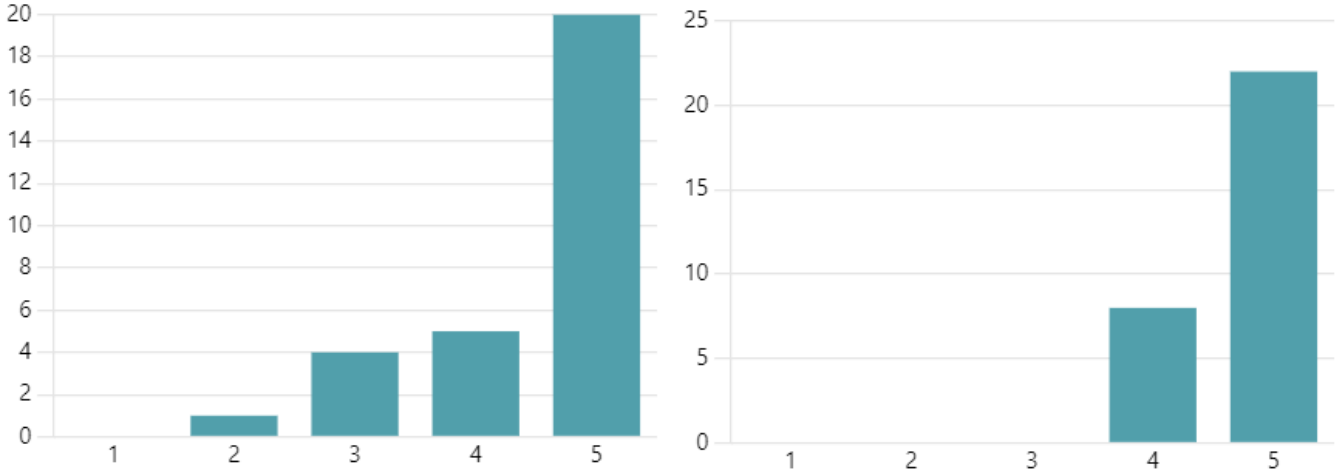

FIGURE 9. **(a)Memorability (b) Satisfaction.**

5. Conclusion

The extensive MtGDeckBuild study investigated contemporary web development methodologies and tools. Frontegg and Scryfall oversaw the administration of user accounts and the enormous Magic: The Gathering card database. Simple access to the API via Axios enhanced the app's usability. Due to the complexity and breadth of the software development process, adaptability and problem-solving skills are essential. Webhooks implemented by Frontegg impeded deck storage and viewing integration. The implementation of these fixes and the maintenance of change logs assisted the group in decreasing reliance on external services and enhancing backend architectural control. Players of Magic: The Gathering derive the greatest benefit from the intuitive and dynamic platform of the study. MtGDeckBuild's innovative technologies and user experience establish it as the industry leader in both casual and competitive play. Enhancing the development, learning, and innovation processes are the objectives of the study team. By enhancing and empowering users, MtGDeckBuild ensures its success in the ever-evolving market for web-based gaming applications.

Declarations

- **Conflict of Interest:** The authors have not disclosed any competing interests.

References

[1] D. Abramov, R. Nabors, Introducing react.dev (2023).
URL https://react.dev/blog/2023/03/16/introducing-react-dev

[2] S. Chen, U. R. Thaduri, V. K. R. Ballamudi, Front-end development in react: an overview, Engineering International 7 (2) (2019) 117–126.

[3] A. Bodepudi, M. Reddy, S. S. Gutlapalli, M. Mandapuram, Voice recognition systems in the cloud networks: Has it reached its full potential, Asian Journal of Applied Science and Engineering 8 (1) (2019) 51–60.

[4] Z. Dinku, React. js vs. next. js (2022).

[5] G.C.Trends, The new experience economy (2022).
URL www.dynata.com/content/GCT_The-New-Experience-Economy.pdf

[6] L. Duy, Web application development (2024).

[7] Q. Odeniran, H. Wimmer, J. Du, Javascript frameworks—a comparative study between react. js and angular. js, in: Interdisciplinary Research in Technology and Management, CRC Press, pp. 319–327.

[8] D. M. Nguyen, Design and implementation of a full stack react and node. js application: simulating driver's license exams (2024).

[9] E. B. Pranata, T. Tony, Utilizing orb algorithm in web-based sales application, Journal of Information Systems and Informatics 6 (1) (2024) 378–398.

[10] O. Lyxell, Server-side rendering in react: When does it become beneficial to your web program? (2023).

[11] C. Minnick, Beginning reactjs foundations building user interfaces with reactjs: an approachable guide, John Wiley & Sons, 2022.

[12] Gaper, How react js is revolutionizing web development (2023).
URL https://gaper.io/how-react-js-is-revolutionizing-web-development/

[13] V. Sahni, A. Chopde, M. Goswami, A. Kumar, Mern (mongodb, express-js, react-js, node-js) stack web-based themefied education platform for placement preparation, Educational Administration: Theory and Practice 30 (5) (2024) 1918–1928.

[14] T. A. Kràlusz, Mobile application development with react native and leveraging third-party libraries (2024).

[15] I. Rizvi, H. Gupta, I. Bharadwaj, et al., Connect easy: Revolutionizing the college experience through innovative web-based solutions (2024).

[16] N. Garg, J. Chopra, V. Kumar, K. Aggarwal, J. Parashar, A. Jain, Applego: React js (web application).

[17] B. Gamage, R. Ranaweera, A. Dilshan, R. Paranagama, D. De Silva, S. Vidhanaarachchi, Centralized platform for managing activities in e-commerce store, International Journal Of Engineering And Management Research 12 (5) (2022) 203–208.

[18] T. Kissflow, What is rapid application development (rad)? an ultimate guide for 2024. (2024).
URL https://kissflow.com/application-development/rad/rapid-application-development/#:~:text=Rapid%20Application%20Development%2C%20or%20RAD,less%20emphasis%20on%20specific%20planning