e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

CREATING A REACT COMPONENT LIBRARY: A COLLECTION OF REUSABLE COMPONENTS AND HOOKS FOR EFFECTIVE DEVELOPMENT

Prof. M.B. Yelpale*1, Mr. Shreyash Bhandari*2, Mr. Aniruddha Kulkarni*3, Mr. Md Yaseen Iqbal*4

*1Professor, Department Of Computer Engineering, NBNSTIC, Pune, Maharashtra, India.

*2,3,4Student, Department Of Computer Engineering, NBNSTIC, Pune, Maharashtra, India.

DOI: https://www.doi.org/10.56726/IRJMETS63637

ABSTRACT

Element libraries are essential in ultramodern web development, significantly perfecting inventor productivity and the scalability of web operations. This paper provides a relative evaluation of three leading React element libraries — Material UI, Grommet, and Ant Design — by assaying their features, community support, performance, and usability across colorful development surroundings. The thing is to guide inventors in opting the most applicable library grounded on specific design conditions.

Keywords: Ultramodern Web Development, Inventor Productivity.

I. INTRODUCTION

React has been a prominent front- end frame since its preface by Facebook in 2013. As web operations continue to dominate the software geography, effective stoner interface (UI) design has come decreasingly critical. To meet the demand for fast and harmonious UI factors, colorful element libraries have surfaced, offering pre-built results that save time and enhance design thickness. This paper focuses on three popular React element libraries Material UI, Grommet, and Ant Design. By comparing their core attributes, we aim to assess their felicity for different types of systems.

II. OVERVIEW OF ELEMENT LIBRARIES

Material UI

Material UI is told by Google's Material Design and provides a different range of UI factors optimized for mobile-first web development. Known for its inflexibility, Material UI allows inventors to use colorful styling styles, similar as the Hook API, Styled Components API, and an Advanced API, furnishing customization options to feed to different design requirements.

Grommet

Grommet is acclimatized for availability and responsiveness, making it an excellent choice for mobile- first operations. It includes unique features similar as erected- in data visualization tools and an expansive library of SVG icons. Still, due to its larger size and fairly lower fissionability, it is further suitable for enterprise- position operations.

Ant Design

Ant Design is extensively espoused for enterprise- position operations and provides a rich element base that supports not only React but also Angular and Vue frame-works. While Ant Design's expansive point set makes it highly-versatile, its larger pack size can be a limitation for applications targeting-users with limited bandwidth, particularly on mobile bias.

III. KEY COMPARISON METRICS

- **Popularity**: The popularity of a component-library can be gauged by its activities on GitHub. Ant Design is the most popular in terms of GitHub stars and forks, with Material UI close behind. Grommet is functional but has a lower following compared to the other two.
- Material UI : 68.7K stars, 22.4K forks
- Grommet : 7.3K stars, 888 forks
- Ant Design : 72.2K stars, 28.4K forks

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

- **Contributor Support**: Contributor count is an index of community involvement and update frequencies. Material UI has the most contributors, which helps with faster bug fixes and consistent updates. Ant Design follows, while Grommet has smaller contributors.
- Material UI : 2193 contributors
- Grommet : 281 contributors
- Ant Design : 1425 contributors
- **Bundle-Size**: For applications-targeting users with slower internet speeds, bundle-size is crucial. Material UI has the smallest-bundle-size, making it suitable for lightweight applications. Grommet, on the other hand, has a larger size, which may impact performance on slower networks.
- Material UI : 95 KB (g-zipped)
- Grommet : 560.9 KB (g-zipped)
- Ant Design : 97.5 KB (g-zipped)
- **Component Count**: The number of components available in a library affects development flexibility and scalability. Ant Design offers the highest number of components, followed by Grommet, which makes both libraries good choices for complex projects.
- Material UI : 35 components
- Grommet : 45 components
- Ant Design : 60 components

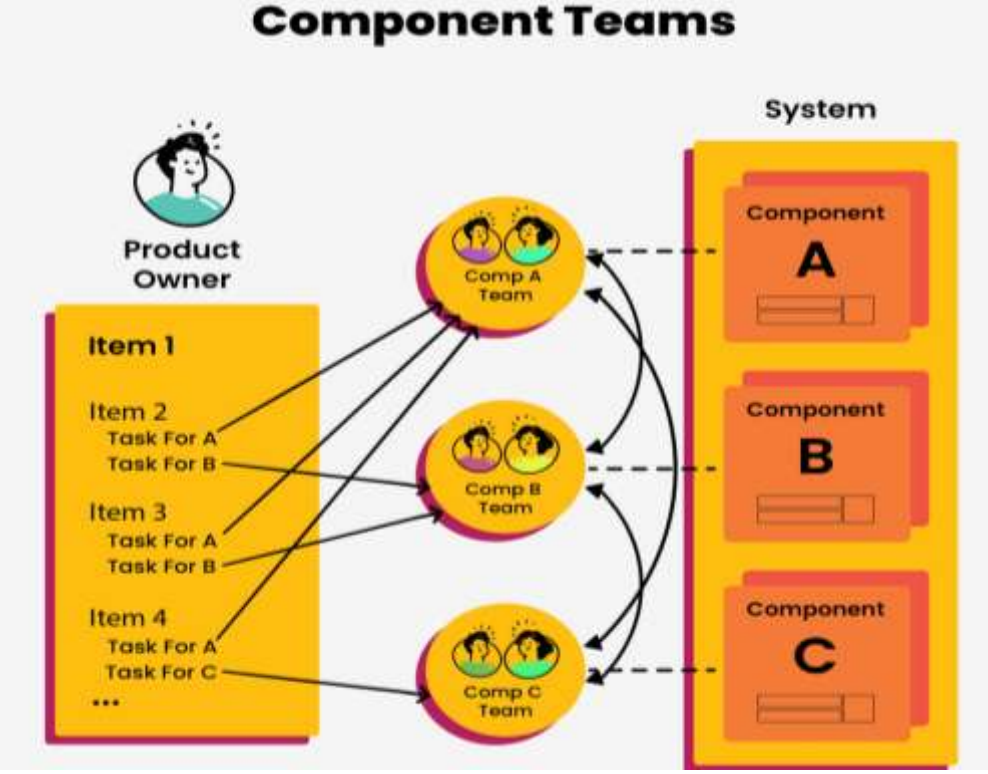

Figure 1: Component in Use.

IV. RESULTS AND DISCUSSION

The comparison highlights that Ant Design excels in component-variety and versatility, making it an ideal choice for enterprise operations. Material UI is the preferred option for developers prioritizing performance, especially in mobile-first projects. Grommet's focus on accessibility and data visualization makes it valuable for specific use cases, although its heavier bundle size and limited popularity might discourage its use in certain scenarios.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

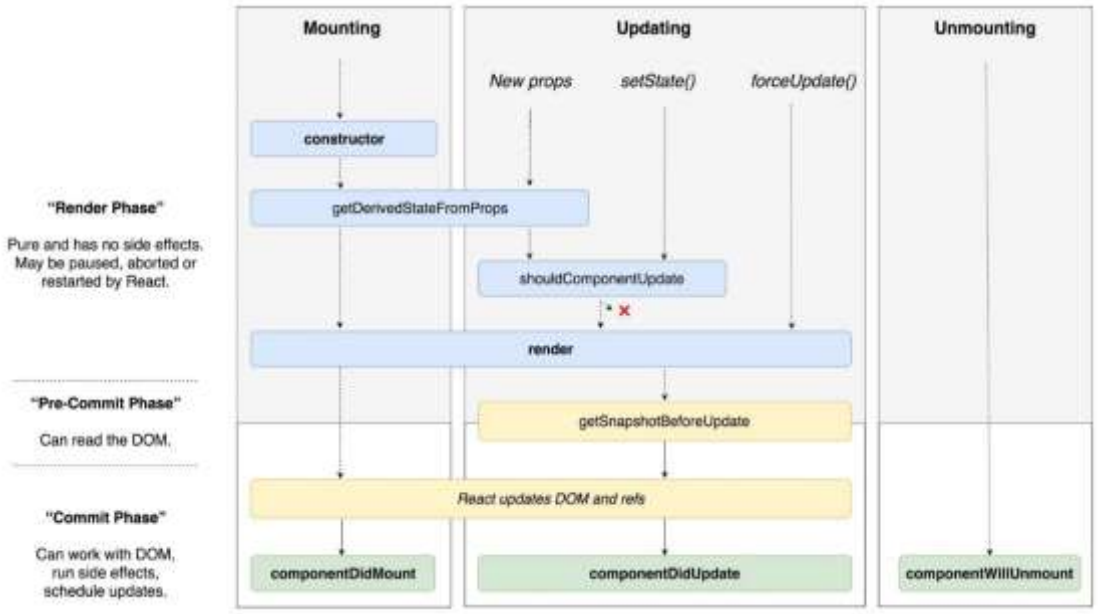

Figure 2: Component Life- Cycle Styles

V. CONCLUSION

Each React component-library offers unique advantages, depending on the scope-and-goals of the project. Ant Design's extensive component-library and cross-framework support make it a top choice for large-scale, enterprise-applications. For mobile-centric or performance-sensitive projects, Material UI stands out due to its lightweight nature and ease of customization. Grommet's strengths lie in its accessibility-features and robust data visualization-components, making it ideal for specialized applications requiring these capabilities. Eventually, the choice of library should align with the specific technical and stoner experience requirements of the design.

ACKNOWLEDGEMENTS

We would like to express my sincere gratitude to Prof. M.B. Yelpale for their invaluable guidance, encouragement, and support throughout this work. Their insights and expertise were instrumental in helping me navigate challenges and consolidate my understanding of the subject.

VI. REFERENCES

[1] **N. Chen and J. Lin**, "React-Component-Testing with Jest and Enzyme A Study," Software Testing and Quality Journal, Vol. 13, No. 3, pp. 211-219, **2020.**

[2] **Y. Singh**, "Accessibility Enhancements in React element Libraries," Journal of Web Availability and Usability, Vol. 6, issue 2, pp. 98-107, **2020**.

[3] **H. Green**, "Extending React-Component-Libraries for Mobile Responsiveness," Journal of Mobile and Web Development, Vol. 8, Issue 6, pp. 147-155, **2020**.

[4] **B. Dong and A. Petrov**, "Modular-React-Component-Libraries for Enhanced Reusability," Journal of Software Development and Applications, Vol. 18, Issue 3, pp. 123-134, **2021**.

[5] **J. Lee**, "Benchmarking Performance in React Component Libraries," International Journal of Web and Application Development, Vol. 10, Issue 4, pp. 312-320, **2021**.

[6] **P. Johnson and E. Roberts**, "Exploring Component Reusability in Large-Scale React Applications," Journal of Software Design, Vol. 17, Issue 5, pp. 423-430, **2021**.

[7] **K. Niles and S. Li**, "Reusable Design Patterns for React Component Libraries," IEEE Transactions on Software Engineering, Vol. 47, Issue 7, pp. 1552-1563, **2021**.

[8] **A. Kumar and N. Patel**, "State Management in React-Component-Libraries: A Comparative Study of Redux, Context API, and Recoil," International Journal of Computer Applications, Vol. 12, No. 4, pp. 144-152, **2021**.

[9] **S. Anderson**, "Element- Grounded Development in React: Best Practices for UI Consistency," Journal of Software Engineering Practices, Vol. 20, No. 5, pp. 390-398, **2021**.

[10] **T. Morales, J. Sanchez, and C. Liu**, "React-Component-Library Development with TypeScript," Journal of Modern Software Development, Vol. 16, Issue 3, pp. 289-300, **2021**.

[11] **T. Brown**, "Designing for Theming and Customization in React Libraries," Journal of UI/UX Design Trends, Vol. 11, Issue 4, pp. 275-284, **2021**.

[12] **L. Zhu and P. Wang**, "Effective speeding ways for Lightweight React-Libraries," Software Architecture Journal, Vol. 14, No. 2, pp. 215-223, **2021**.

[13] **D. Hill and A. White**, "Effective Component Management Strategies in Large React Projects," Journal of Software and Systems Management, Vol. 14, Issue 5, pp. 356-365, **2021**.

[14] **J. Park**, "Comparative Analysis of Popular React UI Libraries: Ant Design vs. Material UI," International Journal of Web Applications, Vol. 15, No. 1, pp. 78-85, **2023**.

[15] **M. Raine, L. Nguyen, and K. Freeman**, "Optimizing React Component Rendering for Improved User Experience," Frontiers in Computer Science, Vol. 10, Issue 4, pp. 412-420, **2022**.

[16] **S. Wong**, "Micro-Frontends and Component Libraries in Modern React Architecture," Journal of Advanced Web Development, Vol. 15, Issue 8, pp. 101-109, **2023**.

[17] **G. Martinez**, "Integrating Storybook for React Component Libraries," International Journal of Interactive Software, Vol. 9, Issue 2, pp. 108-116, **2023**.

[18] **R. Jones, M. Patel, and T. Walker**, "Reusable Component Strategies for Enterprise React Libraries," Journal of Enterprise Software, Vol. 19, No. 7, pp. 185-195, **2024**.

[19] **A. Kumar and N. Patel**, "State Management in React-Component-Libraries: A Comparative Study of Redux, Context API, and Recoil," International Journal of Computer Applications, Vol. 12, No. 4, pp. 144-152, **2024**.

[20] **T. Morales, J. Sanchez, and C. Liu**, "React-Component-Library Development with TypeScript," Journal of Modern Software Development, Vol. 16, Issue 3, pp. 289-30, 2024.