Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2018, 5(5):668-670**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Front-End Development Challenges in Enterprise Software

Aparna Manda

mandaaparna95@gmail.com,
Software Analyst (Hexagon Capability Center India)

Abstract: Large client bases, complicated business needs, and legacy systems provide special hurdles for enterprise software development, especially in front-end development. The difficulties faced when redeveloping the user interface (UI) of the product at Hexagon Capability Center India (HCCI) using Oracle Apex while keeping the current backend are examined in this case study. The research looks at the main obstacles encountered, the methods used to get beyond them, and the lessons discovered.

Keywords: Enterprise Software, Front-End Development, Oracle Apex, Lift-and-Shift, Legacy Systems

1. Introduction

Robust and intuitive user interfaces are essential for enterprise software systems, which are frequently indispensable to corporate operations. Front-end development may be quite difficult when modernizing such systems, particularly if the backend is kept intact and just the user interface is changed—a popular strategy called "lift-and-shift."

Such a lift-and-shift change project took place on a comprehensive procurement management platform. Oracle Apex was used to create a new user interface while maintaining the backend. This case study explores the front-end development issues that arose during the project and the strategies that were put in place to guarantee a smooth transition [1].

2. Challenges in Front-End Development

Retaining Backend Compatibility

A major obstacle in a lift-and-shift project is making sure the new front-end and the current back-end systems are completely interoperable. The backend of the product was intricate, requiring several connectors and data processing layers to continue operating while the user interface was totally redesigned [1].

Solution: The new user interface (UI) was built by the development team using Oracle Apex, enabling a seamless interaction with the current backend. Oracle Apex was the best option for this project because of its ease of connection to Oracle databases and compatibility for classic SQL queries. The move was seamless because of the meticulous mapping of UI elements to the current backend procedures.

Handling Legacy Business Logic

A large portion of the business logic in corporate setups is frequently embedded in the backend, making it difficult to change or replace. With intricate regulatory procedures and procurement systems that had been in place for years, this product was no different.

Solution: The team concentrated on developing a front-end that could successfully interface with the current system rather than trying to change the backend functionality. In order to construct dynamic forms and reports

that interacted with the backend seamlessly and preserved the business logic while providing a contemporary and user-friendly interface, Oracle Apex's PL/SQL capabilities were utilized [2].

User Interface Modernization

It may be challenging to modernize a legacy system's user interface (UI) while preserving its functionality, especially when it comes to making sure the new design meets modern user expectations. The prior user interface (UI) for the product was antiquated and not designed with modern user experience standards in mind.

Solution: After conducting a thorough study of the present user processes, the team decided to use a user-centered design approach. They were able to rapidly prototype new UI elements and iterate in response to user input by utilizing Oracle Apex. This strategy made sure that the finished design satisfied the unique requirements of the product's consumers while still being contemporary and user-friendly.

Performance Optimization

Enterprise software performance is always an issue, especially when modernizing the front-end while keeping the historical backend in place. Even though it was constructed on an outdated backend architecture, the new user interface had to offer a fluid and responsive interface to users [3].

Solution: The development team reduced the amount of queries to the database needed for each UI action and adopted effective querying strategies to maximize performance. Lightweight HTML and JavaScript generated by Oracle Apex contributed to the new front-end's speed and responsiveness. Furthermore, asynchronous data loading made possible by AJAX decreased consumers' perceived load times [4]. This pie chart or bar graph could display user satisfaction levels before and after the UI modernization.

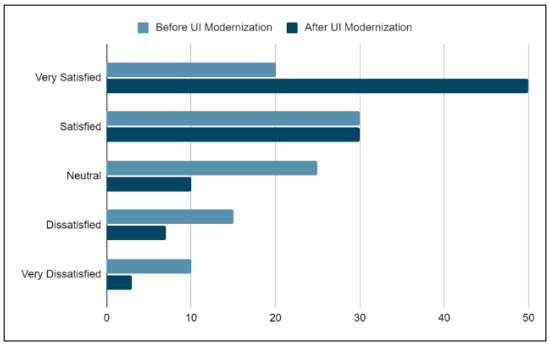

Figure 1: *User Satisfaction Before and After UI Modernization*

Minimizing Disruption During Transition

Reducing the amount of disturbance to current users is a crucial task for any lift-and-shift project. Numerous business clients actively utilized the product. Thus, it was crucial that the switch to the new UI go as smoothly as possible [5].

Solution: The group decided to put out the new user interface (UI) gradually. This made it possible for users to acclimate to the new system gradually and collected input that might be used in later stages. To guarantee a seamless transition, in-depth user training sessions and thorough documentation were also offered. Gradual rollout progress of the new UI was done. A bar graph showing the percentage of users transitioned to the new UI over time.

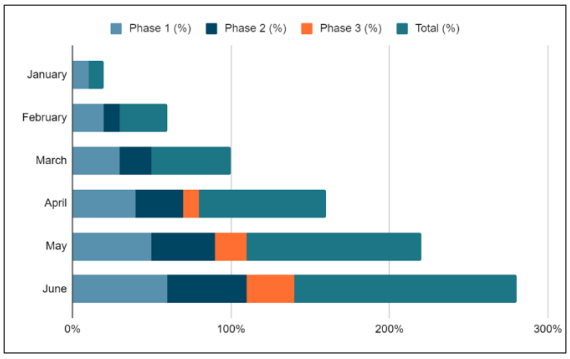

Figure 2: *Gradual Rollout Progress of New UI*

3. Conclusion

The difficulties encountered in front-end development during the lift-and-shift project underscore the intricacies of updating business software while preserving pre-existing backend systems. The development team overcame these obstacles by utilizing Oracle Apex, and as a result, they were able to produce a cutting-edge, user-friendly interface that satisfied user demands without interfering with ongoing business procedures.

The significance of strategic planning, user-centered design, and performance optimization in front-end enterprise software development is emphasized by this case study. Future lift-and-shift initiatives can benefit from the insights learned from this one, especially in situations where preserving backend stability is essential.

References

[1]. B. George and L. Williams, "A structured experiment of test-driven development," *Information and Software Technology*, vol. 46, no. 5, pp. 337-342, Apr. 2004.

[2]. C. R. Larman and V. R. Basili, "Iterative and incremental developments. A brief history," *Computer*, vol. 36, no. 6, pp. 47-56, Jun. 2003.

[3]. G. Meszaros and J. Doble, "A pattern language for pattern writing," in *Proc. 3rd Conf. Pattern Languages of Programs*, Monticello, IL, USA, 1996.

[4]. K. Czarnecki and U. W. Eisenecker, *Generative Programming: Methods, Tools, and Applications*. Boston, MA: Addison-Wesley, 2000.

[5]. T. S. E. Ng, I. Stoica, and H. Zhang, "A case for end system multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1456-1471, Oct. 2002.