MULTI-LAYERED QA STRATEGY FOR MOBILE-CLOUD-WEB INTEGRATION: CASE OF EXTERNAL FUND TRANSFERS IN HYBRID BANKING APPS

Udayan Verma

Denver, USA
udayanverma7@gmail.com

Abstract:
Hybrid banking applications cut across various technological levels, such as mobile customers, cloud-based applications, and web administrator portals that generate elaborate systems that require rigorous quality assurance. An in-depth testing plan at all levels underlines the significance of such values that are necessary in the proper fulfilment of the main workflows, such as External Fund Transfers (EFT), which are to be performed accurately and promptly. The report lists a multi-layer quality assurance scheme to ensure mobile submissions, processing of cloud transactions, and web portal monitoring and reconciliation. With mobile automation, API testing, web automation, and backend validation, individual transactions are adequately processed. This unified method would minimise platform errors, increase the speed of the regression test, and improve financial control. Continuous integration and deployment frames manage testing and evidence gathering, execute synchronisation execution, and compile evidence units automatically.

Keywords: Multi-layer QA, Hybrid Banking Applications, External Fund Transfers (EFT), Mobile-Cloud-Web Integration, Automated Testing Framework.

I.Introduction

The interaction of mobile consumers and clouded intermediaries with the web portals through the hybrid banking applications forms a complicated ecosystem that needs proper quality assurance. One of the most important workflows is External Fund Transfers (EFT), which begins with the mobile devices, is processed in a cloud solution and is monitored and reconciled through the web. According to the failures in synchronisation, updates on status, errors in data accuracy can lead to violation of financial integrity, compliance and customer trust.. Such systems cannot be tested manually as they are diverse in platforms, devices, and browsers, and the changes happen rapidly. Automated multi-layer QA frameworks resolve these issues by guaranteeing dependable implementation of EFT and integrity of services by offering quality validation and consistent, as well as repeatable, validation at all levels.

II.Background and Rationale

The hybrid financial systems are upgraded at the same time with the development of new features, overall performance adjustment, and security updates. Represented by each update, there is a threat of regression wearing off when emerging, especially in key processes like EFT, with any mistake potentially operational and costly. These systems are labour-intensive in terms of testing manually, error-prone in outcomes, and remain underpowered to achieve high rates of accuracy. Multi-layered automation provides solutions to such issues by certifying every phase of the transaction life cycle within mobile, cloud, web and back-end systems. Automated tests remove the risks of incorrectly transacting, inconsistency in data, and meet the regulatory standards [1]. The reasons behind the layered approach are motivated by the fact that hybrid systems can be difficult since a platform specific problem could arise when any of its layers is tested individually.

III.System Architecture and QA Layer Design

The QA plan of hybrid banking apps is structured primarily in four layers All layers are interdependent on mobile, cloud, web, and backend apps. The mobile layer provides interaction with the clients making EFT transactions as the first line of operation. Layered automation, which is at this layer, under consideration is the accuracy of the user work process, input data, and error handling in addition to making sure that the validity of transactions made through mobile devices are formatted appropriately and that the identified transactions are sent to the processing layer without interruption. Mobile automation applications run processes on different devices and operating systems (different versions) to ensure a stable output and quality.

Business logic processing and transaction authentication is done in the cloud layer. This level of API testing ensures that EFT requests are made with integrity, as well as ensuring the payload is valid, business rules are enforced and valid responses are generated [2]. Until updates are propagated to the web or the backend systems, automated API validation takes place to guarantee that every operation falls within operational constraints and all regulatory boundaries.

Administrative and customer interfaces will be offered in monitors, supervisors and reconcilers through the web layer. Robotic verification verifies transactional or data correctness and reliability of dashboard. This layer will provide observability to every step of the transaction lifecycle where administrators can reconcile and learn about transactions at any point of variability.

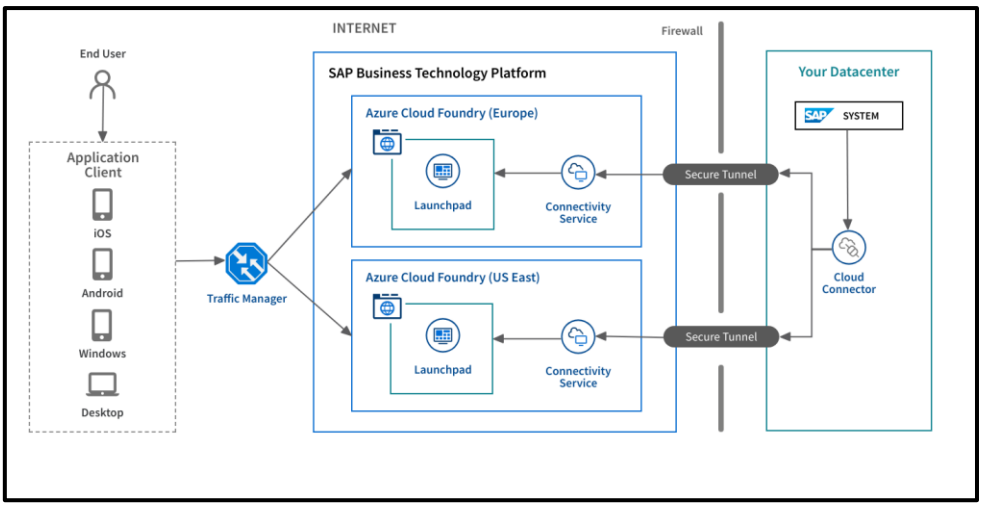

Fig.1: Hybrid Mobile-Cloud-Web Application Architecture Diagram

The data persistence, ledger update, and audit trail maintenance functions fall under the backend layer. Formatted queries are fired to confirm the storage of transaction, balance changes, and compliance of regulation. The system contains risks of hidden inconsistencies that cannot be observed at the user interface or API level but can be eliminated by combining their check with the backend verification as part of the QA.

IV.Integration and Automation Strategy

Continuous integration and deployment pipelines aid in the integration of testing on the mobile, cloud, web, and background layers. The test run can be easily orchestrated by robots to remain constant and the result set can be lumped into a single set of dashboards that can be analyzed. In sync checks ensure that mobile device initiated transactions are bound to be correctly mirrored in cloud services, web portals and back livers. Concurrently running test speeds regression cycles due to the generation of validations of multiple devices, browsers and operating systems at one time [3]. The containerized environments define congruencies in configurations such as the browser versions, drivers, and dependencies among applications whose behavior is guaranteed to be the same when this configuration applies to the test executions in any of the situations.

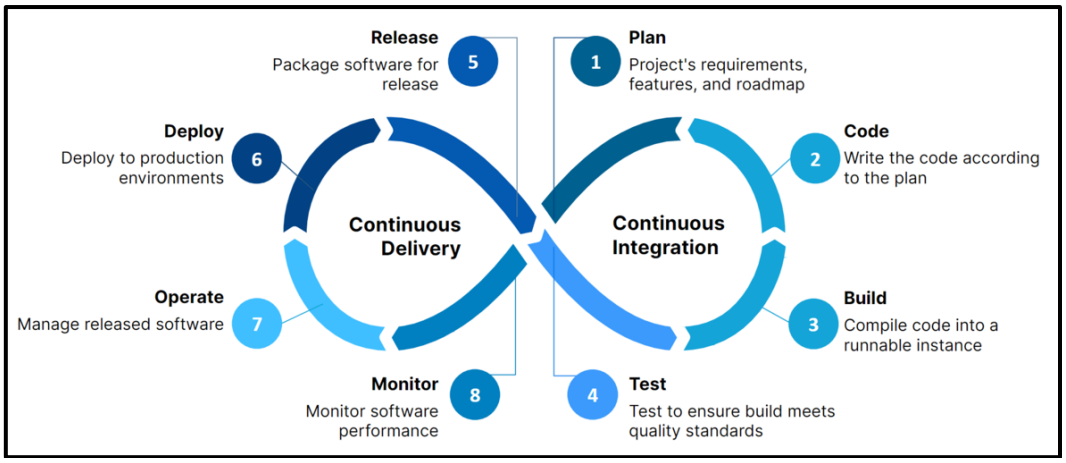

Fig.2: CI/CD Pipeline Flow

V.Scalability and Optimization

The QA framework is streamlined with scalability and efficiency. When learning is executed in parallel on several environments, the entire test execution time is minimized and nightly regression tests can be fastened within reasonable times. Elasticity of the availability provided by clouds can fulfil the demand during peak periods, and the containerisation will offer the same test of the computing machines [4]. Resource assignment too is done dynamically in order to optimize resource usage putting idle environments to free and offering more resources on demand. This design allows the framework to execute hundreds of cases of tests at the same time without obliterating the accuracy of the results.

VI.Reporting and Monitoring

The framework of QA will consist of detailed reporting and monitoring. Results of mobile tests, cloud tests, web tests, and backends tests are delivered to united dashboards that provide real-time metrics of the presence or absence of stability of the systems. All critical metrics, such as rates of successful transactions, the time of executions, and anomalies during a specific layer can be displayed in such a way that it is possible to analyze them in real time. Automated notices signal the failure of a system to the QA engineer and developers to ensure that the routes to raising point defects are mended as early as possible [5]. Detailed reporting provides the opportunity to report the transaction records, excess circumstances, and reconciliation that will help trace the transactions and allow meeting compliance requirements.

VII.Challenges and Best Practices

Implementation of multi-layer QA in hybrid banking system has some challenges which can include, synchronization of the test data between layers, compatibility of browser and device version, modular test design. It is advisable through best practices to use standardized and versioned environments, presence of modular test scripts to aid in reuse and regular maintenance to aid stability in the pipeline [6]. Active maintaining check of the running of the tests as well as settings of the environment environment prevents any potential situations of failure. Continuous sufficiency execution of the framework ensures consistent working of the structure, proper validation as well as scalability with minimal risks associated with composite frameworks.

VIII.Conclusion

The approach to Multi-layered QA strategy of hybrid banking applications provides general guidance of verifying the works of the External Fund Transfer of mobile, being cloud, web and supportive systems. Accuracy, reliability and compliance with regulatory standards are built with the help of the framework automatization pipelines, alignment transaction processes and aggregation of the results to activity dashboard. The methodology minimizes the requirement of manual solutions, is also, less time consuming due to the repetitions allowed by the regression and is also consistent throughout all platforms.

REFERENCES:

1. Capgemini, Sogeti, and Micro Focus, *World Quality Report 2019–20: The State of Quality Assurance*, 11th ed. Paris, France: Capgemini SE, 2019. [Online]. Available: https://www.capgemini.com/insights/research-library/world-quality-report-2019-20/
2. Appium Project, *Appium Documentation*, ver. 2.0. [Online]. Available: https://appium.io/docs/en/2.0/
3. Zampetti, F., Geremia, S., Bavota, G. and Di Penta, M., 2021, September. CI/CD pipelines evolution and restructuring: A qualitative and quantitative study. In *2021 IEEE International Conference on Software Maintenance and Evolution (ICSME)* (pp. 471-482). IEEE.
4. Pan, Z., Shen, W., Wang, X., Yang, Y., Chang, R., Liu, Y., Liu, C., Liu, Y. and Ren, K., 2023. Ambush from all sides: Understanding security threats in open-source software ci/cd pipelines. *IEEE Transactions on Dependable and Secure Computing*, *21*(1), pp.403-418.
5. Pinto, C.M. and Coutinho, C., 2018, September. From native to cross-platform hybrid development. In *2018 international conference on intelligent systems (IS)* (pp. 669-676). IEEE.
6. Kaoudi, Z. and Quiané-Ruiz, J.A., 2018, April. Cross-platform data processing: use cases and challenges. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)* (pp. 1723-1726). IEEE.