**Performance Test Plan**



**Performance Test Plan**

Version 0.2

January 17, 2024

**Documentation Policy**

**Document Name:** SG-Performance-Test-plan-v0.2

**Date**: January 17, 2024

This document contains information confidential and proprietary to Scientific Games International, Inc. (SGI), a wholly owned subsidiary of Scientific Games Corporation (SGC) and may be protected by other forms of intellectual property. The information contained herein may not be reproduced, paraphrased, translated into another language, or otherwise disclosed or duplicated by any means without the prior written consent of SGI. Violators may be prosecuted. Due to the confidentiality of the subject matter, revisions or alterations are undertaken only by written agreement with SGI. Distribution of this document or any portion thereof may only be conducted by SGI or through its designated officials.

The software / systems described in this document are protected by copyright law. ***Unauthorized duplication or use of any software described herein is prohibited.*** Software features may be described herein which are not implemented in every system. Similarly, ongoing enhancements to this system may result in the presence of features on a particular system, which are not included in this edition of the document. Authorized users will be notified of software / systems modifications through the release of new editions of this documentation.

SGI MAKES NO WARRANTY, EXPRESSED OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY OF FITNESS FOR ANY PARTICULAR PURPOSE. SGI ACCEPTS NO LIABILITY FOR ERRORS CONTAINED IN THIS DOCUMENT OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES ASSOCIATED WITH ITS PROVISION, CONTENTS, OR USE.

© 2024 by Scientific Games International, Inc.

All rights reserved.

**Revision Record**

| Version | Date | Description | Authored By |
| --- | --- | --- | --- |
| 0.1 | 15-11-2024 | Initial draft | Rajesh Chaudhari |
| 0.2 | 17-01-2024 | Updated Workload & Test Data Sections | Rajesh Chaudhari |

**Document Endorse & Approval**

| Name | Position | Organization | Approval Date |
| --- | --- | --- | --- |
| LINDMAIER Stefan | Director, Customer Program Management | Scientific Games |  |
| BROWN Robin | Senior Solution Architect, PM | Scientific Games |  |

**Distribution List**

| Name | Position | Organization |
| --- | --- | --- |
| Tanya Syme | Principal Quality Assurance Manager | Lotto NZ |
| Gideon Lee | Quality Assurance Manager | Lotto NZ |
| LINDMAIER Stefan | Director, Customer Program Management | Scientific Games |
| PEARCE Nathaniel | Solution Delivery Manager | Scientific Games |
| MACGREGOR Stephen | Infrastructural Delivery Manager | Scientific Games |
| VILLEMAIN Jean-Baptiste | Software Quality Assurance Engineer | Scientific Games |
| BROWN Robin | Senior Solution Architect, PM | Scientific Games |
| DUARTE Miguel | Senior Migration Engineer | Scientific Games |
| FOSSATI Julien | Senior Software Quality Assurance Tester | Scientific Games |
| YARAMASA Murali | Test Manager | Planit |

Table of Contents

[SECTION 1 INTRODUCTION 6](#_Toc189469719)

[1.1 Background 6](#_Toc189469720)

[1.2 Purpose of this document 6](#_Toc189469721)

[SECTION 2 TESTING overview 7](#_Toc189469722)

[2.1 Overall Performance Testing Strategy 7](#_Toc189469723)

[2.2 Performance Testing Approach 9](#_Toc189469724)

[2.3 Performance Testing Objective 9](#_Toc189469725)

[2.4 In Scope (Phase 1) 10](#_Toc189469726)

[2.4.1 For SGI Component Testing: 10](#_Toc189469727)

[2.5 Other In Scope (Phase 1) 12](#_Toc189469728)

[2.6 Excluded from Scope 12](#_Toc189469729)

[2.7 In Scope (Phase 2) 12](#_Toc189469730)

[2.8 Workload Model – SG Components 13](#_Toc189469731)

[2.8.1 Test Scenario Details (Phase 1) 13](#_Toc189469732)

[2.8.2 Test Scenario Details (Phase 2 – Using Mule) (To be confirmed by Project Team) 14](#_Toc189469733)

[2.9 Non-Functional Requirements 15](#_Toc189469734)

[2.10 Test Data 15](#_Toc189469735)

[2.10.1 Data Requirements 15](#_Toc189469736)

[2.11 High Level Test Scenario Execution Approach 15](#_Toc189469737)

[2.12 Test Analysis & Reporting 15](#_Toc189469738)

[2.13 Support 16](#_Toc189469739)

[2.13.1 Database Backup and Restore Support 16](#_Toc189469740)

[SECTION 3 Test environment Details 17](#_Toc189469741)

[3.1 Test Environment 17](#_Toc189469742)

[3.2 Performance Testing Environment 17](#_Toc189469743)

[3.3 Load Test tool & Monitoring 18](#_Toc189469744)

[3.3.1 JMeter Configuration 19](#_Toc189469745)

[SECTION 4 Test Management & Controls 20](#_Toc189469746)

[4.1 Defect Management 20](#_Toc189469747)

[4.2 Entry and Exit Criteria 21](#_Toc189469748)

[4.2.1 Entry Criteria 21](#_Toc189469749)

[4.2.2 Exit Criteria 22](#_Toc189469750)

[4.2.3 Suspension Criteria 22](#_Toc189469751)

[4.2.4 Resumption Criteria 22](#_Toc189469752)

[4.2.5 Test Reporting 22](#_Toc189469753)

[4.3 Deliverables 23](#_Toc189469754)

[4.4 Roles & Responsibilities 23](#_Toc189469755)

[4.5 Dependencies & Assumptions 23](#_Toc189469756)

[4.5.1 Dependencies 23](#_Toc189469757)

[4.5.2 Assumptions 24](#_Toc189469758)

[4.6 Risk and Issue Management 24](#_Toc189469759)

[APPENDIX 26](#_Toc189469760)

[Appendix A - Glossary 26](#_Toc189469761)

[SECTION 5 Customer USEr journeys 28](#_Toc189469762)

[5.1 Digital User Journeys 28](#_Toc189469763)

[5.1.1 Pre-Draw User Journey 28](#_Toc189469764)

[5.1.2 Post-Draw User Journey 31](#_Toc189469765)

# INTRODUCTION

## Background

Lotto NZ has an opportunity to replace its core, online and front-end systems, making its technology flexible and adaptable for new game types and services while aligning with the organization's strategy. Lotto NZ has appointed Scientific Games International, Inc. (SGI) as a partner for this major project to upgrade the systems supporting MyLotto and Lotto NZ in-store technology.

The primary objective of the EDGe Programme (Experience Driven Gaming Engine) is to replace the interactive system supporting MyLotto and the core gaming system, including platforms and retail point-of-sale technology for around 1,200 retail stores in New Zealand.

As a part of go live requirements, Planit has been engaged by SGI to carry out POC followed by Performance Testing for individual SG components.

## Purpose of this document

This document provides a high-level overview of the Performance Testing activities for the SGI Components (UPAM, DGE & SGCH). It outlines the Overall Performance Test Strategy, Approach, and Test Process for assessing SG’s individual component as well as integrated component performance, with the goal of mitigating project performance risks.

The primary purpose of this Performance Test Plan is to detail and communicate a comprehensive test approach that encompasses all aspects of Performance Testing, aligned with the Overall SGI Test Strategy.

This plan defines the scope, coverage, and prioritization of the testing effort in key components of SG system, ensuring effective risk mitigation. Its intent is to provide clear direction, enabling the testing process to be executed in a structured and managed manner, consistent with SGI’s best practices and quality standards.

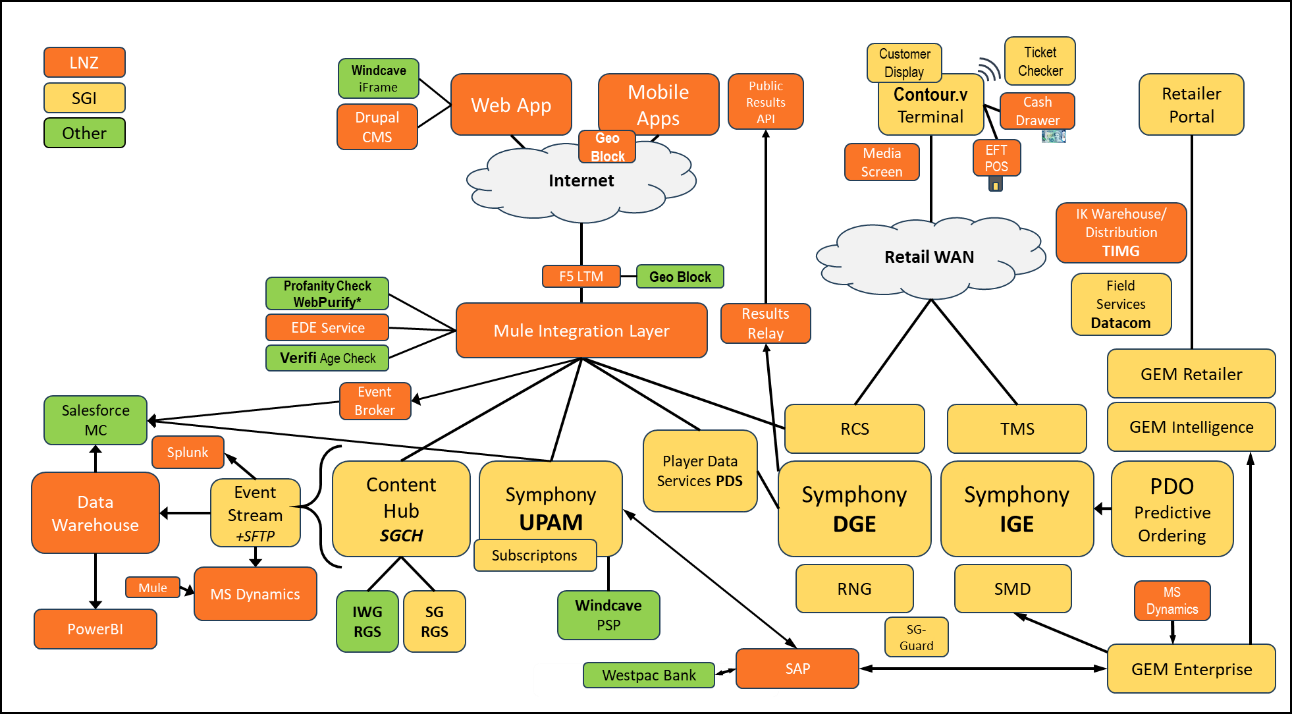
# TESTING overview

## Overall Performance Testing Strategy

The performance assessment of the overall EDGe solution will be conducted in two distinct phases. **Phase 1** will focus on the performance testing of individual SG components. **Phase 2** will encompass similar performance testing exercise as Phase 1 but will be carried out using Mule Layer. This phase will also include co-ordination and providing support to Lotto Performance Team for end-to-end customer user journey testing, integrating both the SG and Lotto systems. Scope of this performance testing exercise to test SG components from performance standpoint (test data creation, scripting, execution, results analysis, reporting) and help (test data creation, script modifications, results analysis, issue resolution) Lotto NZ Performance test team with their performance testing exercise whenever required during Phase 2.

Phase 1 of the exercise will ensure that individual SG components are both robust and performant prior to their integration in Phase 2, where they will be combined with other Lotto components, such as Mule.

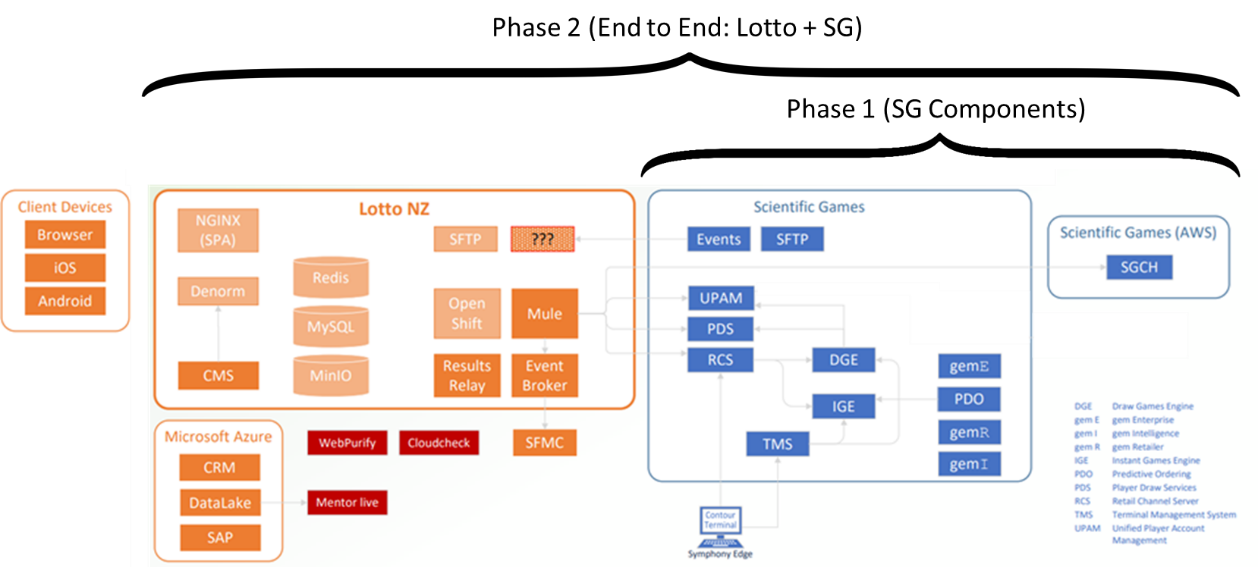
Both phases will ensure that the system not only meets current demands but is also capable of scaling to accommodate future growth.

The deployment diagram below illustrates the high-level architecture of the overall EDGe solution.

The following diagram illustrates the end-to-end digital customer journey for Lotto NZ.A picture containing graphical user interface

Description automatically generated

The end to end customer journey will be divided into multiple segments, with the corresponding SG components identified and tested during Phase 1.



## Performance Testing Approach

The Performance Testing Approach systematically evaluates a system's responsiveness, stability, and scalability under various conditions. Various testing types (peak load, stress/scalability and endurance/soak) will be employed to assess SGI component performance under expected and extreme conditions. Tools like JMeter, Grafana, and InfluxDB will be used to capture and monitor critical metrics, including response time, throughput, and load generator resource utilization. Analyzing this data helps identify bottlenecks and optimization opportunities, ensuring the system meets current demands and can scale for future growth. Ongoing monitoring is essential for maintaining performance standards in production and managing potential issues proactively.

The scope and non-functional requirements (NFRs) will be provided by business teams. POC and Smoke tests will ensure the test environment (T2 HAL) and Performance testing and monitoring framework are functioning correctly before conducting performance tests.

In-scope APIs (User Journeys) for Phase 1 will be subjected to varied load conditions and different workloads will be simulated using Apache JMeter. This approach provides an accurate assessment of API response times, effectively identifying performance bottlenecks.

Proposed test execution scenarios are outlined in section “**Test Scenario Details**”. After each scenario, results will be analyzed, and interim reports will be shared with relevant stakeholders to keep them informed. A final performance test summary report will document findings, resolved issues, and any outstanding concerns from the executions.

## Performance Testing Objective

The high-level objectives of the performance test exercise are:

1. Conduct a POC (scaled down version of load test with selected APIs) ensuring the test environment (T2 HAL) and Performance testing and monitoring framework are functioning correctly before conducting individual component performance tests and present initial results to stakeholders.
2. Baseline (Peak Load Test) the performance characteristics of the in-scope Application APIs from SGI Component perspective and validate if established response time NFRs are met.
3. Verify whether the solution can handle the forecasted demands (Scalability Test) considering future growth.
4. Implement an Endurance test (Soak Test) in the Performance Test Environment to detect potential memory leaks and validate the application's performance over extended usage, to ensure long-term stability and performance consistency across all in-scope modules.
5. Assess SGI Components from performance standpoint and make sure they are robust, performant and scalable which will provide confidence before entering into Phase 2 i.e. End to End testing involving Lotto NZ components (through Mule middleware).
6. Publish report(s) for key findings, recommendations, risk ratings for steering group
7. Build an enduring performance test and monitoring framework complete with documentation and Handover activities to SGI.

**Note:** The test objectives will be adaptive in nature, and the test team will work with the project team to determine the next test to be executed based on the outcome of the test.

## In Scope (Phase 1)

The scope of Phase 1 Performance Testing exercise is limited to SGI components and Integrated SGI component’s performance (API traffic between internal SG components) under load with a focus on ensuring their robustness and performance.

In this section a high-level view of the scope is provided for context.

## For SGI Component Testing:

**User Journey Based: (SG APIs mapped to Mule APIs):**

All front end user actions for user journeys will be simulated using SG APIs (mapped to LNZ-Mule APIs). JMeter tool will be used for simulation and generating virtual load.



1. **Pre-Draw: Create Wager**

|  |  |  |
| --- | --- | --- |
| **Pre-Draw - Create Wager – DIP /**  **Browser Action** | **API Calls (LNZ Domain) to Mule** | **Mapped API Requests (SG Domain) to SG Components** |
| Load URL | GET - /api/time |  |
| Login | POST - /api/sessions GET - /api/players/{playerId}/claims GET - /api/players/{playerId}/profiles GET - /api/players/{playerId}/blocked-games GET - /api/players/{playerId}/limits GET - /api/players/{playerId}/ede | **/api/sessions: (EMR-71: Session Login)** POST - /pfi/login?type=player (SG UPAM) **/wallet: (EMR-65: Fetch Wallet Balance)**  GET - https://{sg.domain}/pfi/wallets/{playerId} (SG UPAM)  **/claims: (EMR-70: List Claims)** GET - /pfi/players/{playerId}/winnings-to-process (SG UPAM) **/profiles: (EMR-87: Retrive Player Profile)** GET - /pfi/profile/players/{playerId}/details (SG UPAM) GET - /pfi/players/{playerId}/payment-methods (SG UPAM) **/blocked-games: (EMR79: Retrieve Blocked Games)** GET - /pfi/exclusions/players/{playerId} (SG UPAM) **/limits: (EMR-88: Retrieve Spend Limits)** GET - /pfi/limits/players/{playerId} (SG UPAM) **/ede: (EMR-61: Fetch EDE Tickets and Results)** GET - /draw-history-api/v1/games/{gameId}/draws/{drawNo} (SG PDS) GET - /pfi/players/{playerId}/winnings-to-process (SG UPAM) PUT - /pfi/players/{playerId}/winnings-to-process/{winningToProcessId} (SG UPAM) GET - /pfi/players/{playerId}/game-history (SG UPAM) POST - /wager-history-api/v1/wagers/get-many (SG PDS) |
| Top Up Wallet | POST - /api/players/{playerId}/wallets/top-ups/credit-cards | **/top-ups: (EMR-95: Top Up Wallet)** GET - /pfi/players/{playerId}/payment-methods (SG UPAM)  POST - /pfi/players/{playerId}/deposits (SG UPAM) |
| Select Lotto Powerball / Keno / Bullseye (Random games will be selected) | GET - /api/players/{playerId}/favourites | **/favourites: (EMR-59: Fetch Favourites)** GET - /pfi/profile/players/{playerId}/favorite-bets (SG UPAM) |
| Click Buy A DIP | GET - /api/draws/keno | **/draws/keno: (EMR-230: Fetch Current Draw)** GET - /draw-history-api/v1/games/{gameId}draws/next?by=SELL\_END\_DATE (SG PDS) |
| Buy Keno DIP  (Random games will be selected) | POST - /api/players/{playerId}/wagers GET - /api/players/{playerId}/limits | **/wagers: (EMR-51: Create Wager)** GET - /pfi/wallets/{playerId} (SG UPAM) POST - /rcs-dge-wager/v1/wagers (SG DGE) **/limits: (EMR-88: Retrieve Spend Limits)** GET - /pfi/limits/players/{playerId} (SG UPAM) |
| Logout | DELETE - /api/sessions | **/api/sessions: (EMR-72: Session Logout)** POST - /pfi/logout (SG UPAM) |

1. **Post-Draw: Check Results**

|  |  |  |
| --- | --- | --- |
| **Post-Draw – Check Results /**  **Browser Action** | **API Calls (LNZ Domain) to Mule** | **Mapped API Requests (SG Domain) to SG Components** |
| Load URL | GET - /api/time |  |
| Login | POST - /api/sessions GET - /api/players/{playerId}/claims GET - /api/players/{playerId}/profiles GET - /api/players/{playerId}/blocked-games GET - /api/players/{playerId}/limits GET - /api/players/{playerId}/ede | **/api/sessions: (EMR-71: Session Login)** POST - /pfi/login?type=player (SG UPAM) **/wallet: (EMR-65: Fetch Wallet Balance)**  GET - https://{sg.domain}/pfi/wallets/{playerId} (SG UPAM)  **/claims: (EMR-70: List Claims)** GET - /pfi/players/{playerId}/winnings-to-process (SG UPAM) **/profiles: (EMR-87: Retrive Player Profile)** GET - /pfi/profile/players/{playerId}/details (SG UPAM) GET - /pfi/players/{playerId}/payment-methods (SG UPAM) **/blocked-games: (EMR79: Retrieve Blocked Games)** GET - /pfi/exclusions/players/{playerId} (SG UPAM) **/limits: (EMR-88: Retrieve Spend Limits)** GET - /pfi/limits/players/{playerId} (SG UPAM) **/ede: (EMR-61: Fetch EDE Tickets and Results)** GET - /draw-history-api/v1/games/{gameId}/draws/{drawNo} (SG PDS) GET - /pfi/players/{playerId}/winnings-to-process (SG UPAM) PUT - /pfi/players/{playerId}/winnings-to-process/{winningToProcessId} (SG UPAM) GET - /pfi/players/{playerId}/game-history (SG UPAM) POST - /wager-history-api/v1/wagers/get-many (SG PDS) |
| Check Lotto / Keno / Bullseye Results | GET - /api/draws/bullseye GET - /api/results/v1/results/bullseye GET - /api/players/{playerId}/tickets?game=bullseye&start=2024-10-31&end=2024-11-14 GET - /api/results/v1/results/bullseye/122 GET - /api/players/{playerId}/ede GET - /api/results/v1/results/bullseye | **/draws: (EMR-230: Fetch Current Draw)** GET - /draw-history-api/v1/games/{gameId}draws/next?by=SELL\_END\_DATE (SG PDS) **/tickets: (EMR-64: Fetch Ticket History)** GET - /pfi/players/{playerId}/winnings-to-process (SG UPAM) GET - /pfi/players/{playerId}/game-history (SG UPAM)  GET - /draw-history-api/v1/games/{gameId}/draws/next (SG PDS) POST - /wager-history-api/v1/wagers/get-many (SG PDS) **/ede: (EMR-61: Fetch EDE Tickets and Results)** GET - /draw-history-api/v1/games/{gameId}/draws/{drawNo} (SG PDS) GET - /pfi/players/{playerId}/winnings-to-process (SG UPAM) PUT - /pfi/players/{playerId}/winnings-to-process/{winningToProcessId} (SG UPAM) GET - /pfi/players/{playerId}/game-history (SG UPAM) POST - /wager-history-api/v1/wagers/get-many (SG PDS) |
| Scan Ticket | POST - /api/tickets/{serialNumber} | **/tickets: (EMR-91: Scan Ticket)** GET - /wager-history-api/v1/wagers?wagerIds={serialNumber} (SG PDS) PATCH - /rcs-ige-prize-payout/v1/tickets/{ticketNo}/prize-inquiry (SG IGE) |
| Logout | DELETE - /api/sessions | **/api/sessions: (EMR-72: Session Logout)**  POST - /pfi/logout (SG UPAM) |

**Component Based: Standalone Component Testing (UPAM, Symphony & GEM) will be conducted by respective product team.**

## Other In Scope (Phase 1)

The table below shows the other In-Scope processes and areas that will be covered during the performance testing.

|  |  |  |
| --- | --- | --- |
| Area | Name | Comment |
| Dashboards | Grafana, Splunk Monitoring Dashboards | Create intuitive dashboards to monitor performance metrics during test executions |

## Excluded from Scope

The table below shows areas/activities that will not be covered during Phase 1 performance testing exercise.

|  |  |  |
| --- | --- | --- |
| Area | Name | Comment |
| Systems | Lotto system (middleware Mule),  Any Other Applications ( Windcave) | The focus of performance testing will be limited to APIs defined in the scope section. End to end Customer journeys via Middleware Mule will be considered in Phase 2. |
| Performance Engineering | All components | Any kind of performance engineering effort is out of scope for this engagement. The performance test team will identify issues and provide recommendations for improvement based on the results shared during analysis. |
| Functional Testing | System, SIT, UAT etc. | Functional verification of any kind is out of scope for performance testing. |
| Non-Functional Testing | Operability, Security, Maintainability, Portability, Usability. | These other non-functional test types are out of scope for performance testing. |

## In Scope (Phase 2)

During Phase 2, performance testing exercise will be carried out using Mule Layer and necessary support will be provided to Lotto NZ Performance Test Team in terms of test data creation, workload design, results analysis during their testing through Mule Layer. The scope and Component diagram for each use case (for Phase 2) involved in Pre-Draw and Post-Draw User Journeys are mentioned in Section 5.

## Workload Model – SG Components

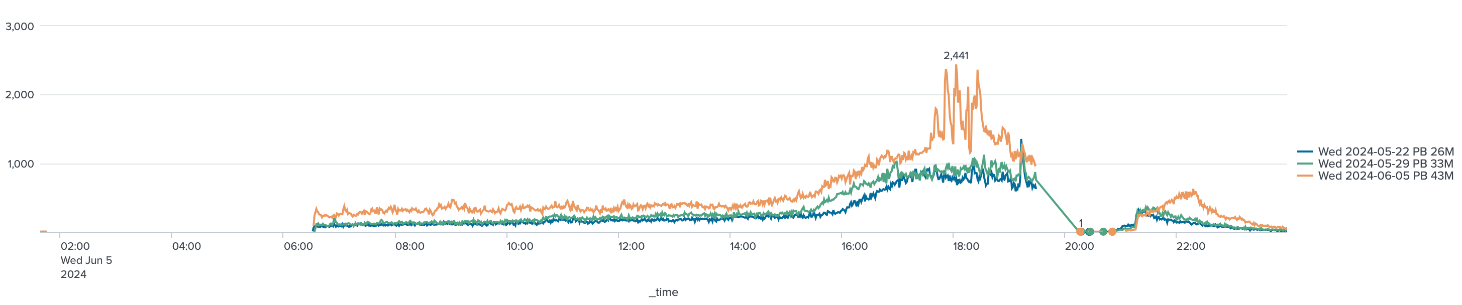
## Test Scenario Details (Phase 1)

All the test scenarios will be executed for APIs (User Journeys) listed in In-Scope section (2.4). This will help us identify performance issues related to individual component, if any. Phase 2 will include end to end workloads. For Phase 1, Peak Load and Scalability tests are of highest priority. All other tests will be targeted based on available timelines. Below throughput data is provided by LNZ, based on latest draw.

Based on raw data available from current draw, below are the details about high throughput APIs (Mule and their corresponding SG APIs).



The following TPM pattern is related to Purchase Wager, and as illustrated in the graph, the 43M category shows the highest peak, with a total of 2,441 wagers per minute.

Based on above API throughput data derived from current draw, below workload is designed for performance testing exercise.



## Test Scenario Details (Phase 2 – Using Mule) (To be confirmed by Project Team)

Below throughput details are extracted from earlier version of LNZ Performance Test Plan. This needs to be updated based on current & expected usage. Project team will confirm the same once all details are available.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Type | Test Case / User Journey | Login / min | Requests per min | Duration |
| Peak Load Test | Pre-Draw | 1900 | 1500 Wagers, 900 Top Ups | 2 hrs |
| Post-Draw | 5000 | 5000 EDE, 450 Wagers, 180 Top Ups |
| Scalability Test | Pre-Draw | Max TPM | Max TPM Wagers, Max TPM Top Ups | 2 hrs |
| Post-Draw | Max TPM | Max TMP EDE, Max TPM Wager, Max TPM Top Up |
| Soak Test | Pre-Draw | 950 | 750 Wagers, 450 Top Ups | 1 Full Business Day (9am – 5pm) |

## Non-Functional Requirements

The following Non-functional Requirements (NFR) will be validated against the performance test objectives.

|  |  |  |  |
| --- | --- | --- | --- |
| Title | Call Type | Current State | Forecast |
| Individual Mule API Response Times | Wagering Calls | < 1 second | < 1 second |
| Non-wagering Calls | < 2 seconds | < 2 seconds |

## Test Data

## Data Requirements

Below test data has been prepared in T2 HAL environment to define the data volumes and characteristics necessary to simulate realistic load scenarios effectively.

|  |  |
| --- | --- |
| Number of Players | 600K |
| Number of Players with Wallet Balance | 600K |
| Number of Player with wager history | 600K |
| Total Wagers in Database | ~26M |
| Wallet Balance for each Player | ~$4500 |
| Number of Players with registered Credit Card | 400K |

## High Level Test Scenario Execution Approach

The high-level approach to the test execution will be:

1. Execute POC to ensure the test environment (T2 HAL) and Performance testing framework are functioning correctly along with data quality and workload rate.
2. Execute Shakeout Test (Smoke Test) with 20% of the volume.
3. Execute Peak Load Test cycles.
4. Identify and report performance issues.
5. Execute second round of tests post fix.
6. Same approach to be followed for Scalability test & Endurance/Soak Tests.

## Test Analysis & Reporting

After completion of the test scenario execution, the response time, throughput, and resource usage statistics will be analysed. Throughput statistics will be assessed against any NFRs laid out in the requirements and defects raised where appropriate.

Jira will be used for defect management. Project team to setup account details for the Planit team.

At a minimum the following details will be included in the report:

1. A summary of what tests were executed.
2. The start and stop times of each test.
3. Any defects and issues that were discovered during the performance tests.
4. Performance metrics of each test:
   * Response times
   * Error rates
   * Error messages

## Support

The test team will need to work together with stakeholders to re-assess the approach based on the test results from each cycle.

The test team will co-ordinate with Project team to monitor SGI system components and capture performance metrics like CPU Consumption, Memory Consumption, application logs, errors etc.

## Database Backup and Restore Support

Over time, junk data generated during performance tests will be accumulated in a database, causing it to bloat. This database bloat can negatively affect subsequent test results, especially if tests are designed to evaluate performance under real-world conditions.

Support needed from SGI Project team to take snapshot or backup of the database before performance test exercise and then roll it back to that state after the performance test completes. This ensures the database is restored to its original state, eliminating accumulated junk data.

# Test environment Details

## Test Environment

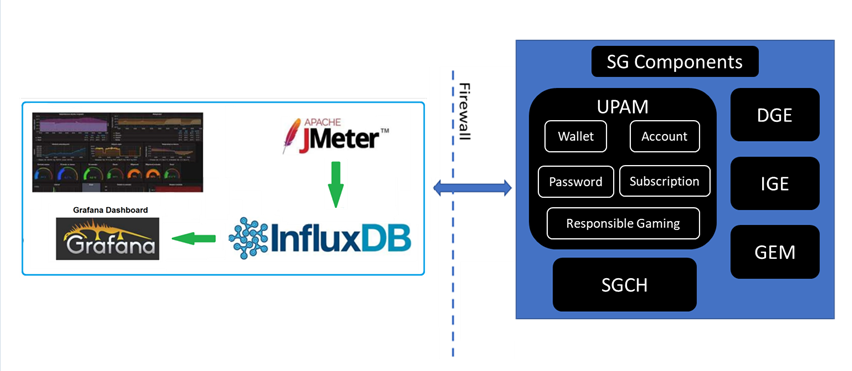
SGI Component Performance Testing will be performed in the T2 HAL environment. It is assumed that this environment will be sized the same or as close as possible to Production. It is assumed that the test environment has the same size of Data as in production if not then it lies with the environment team to uplift the T2 HAL environment DB similar to production by means of production data refresh and Masking if needed. If any discrepancies are found an assessment will be made to understand that whether the differences are likely to have a material impact on the validity of the test results.

Need access details for T2 HAL environment where POC and individual component testing will be carried out.

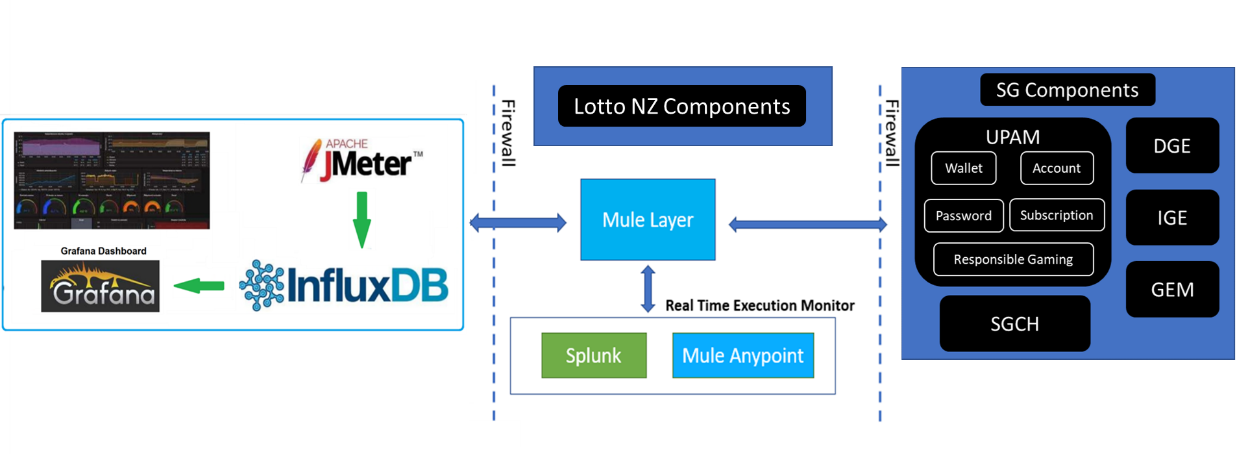
## Performance Testing Environment

The performance testing exercise will utilize the **T2 HAL environment** as the test environment. **JMeter** will serve as the primary tool for performance testing, facilitating load simulation across the system. To ensure accurate monitoring and efficient analysis, **Grafana** and **InfluxDB** will be deployed on separate, dedicated virtual machines (VMs), designated specifically for results collection and data visualization.

**Phase 1: (SGI Component Performance Testing)**

****

**Phase 2: (End to end Solution Performance Testing)**

****

## Load Test tool & Monitoring

The following tools will be used for Performance Testing, Monitoring and Analysis.

|  |  |  |
| --- | --- | --- |
| Tool | Description | Metrics Captured (where applicable) |
| JMeter | Load testing tool used for scripts automation and test executions, simulate concurrent virtual users  Note: JMeter is not a browser. It simulates browser-based HTTP/HTTPs traffic | End user response times  Average, Median, Min, Max, 90 and 99 percentiles  Request errors |
| Grafana | Grafana is an open-source real-time data visualization tool. It provides a webpage where users can create, view, and interact with graphs and dashboards displaying the data from its data source (in this case Influx DB). This will be used primarily for showing real-time test metrics as well as trend analysis | Dashboards |
| InfluxDB | Influx DB is a time series database that is mainly used to store monitoring data, metrics, real-time analytics, and other data that is co-dependent on time. InfluxDB listens to JMeter and stores the test results in a database, and Grafana reads the data from the InfluxDB database to display in its graphs. | Test Raw Data |
| Splunk | Splunk will be used for monitoring, searching, analysing, and visualizing the JMeter results and logs | Dashboards |
| JIRA and Confluence | Defect logging and testing status reports | Defects |

## JMeter Configuration

The performance tests will be conducted within the SGI network, leveraging high-specification virtual machines (VMs) provisioned and configured in a distributed mode by the test team. These VMs will be hosted within the same network and geographical location as the SG components, ensuring a controlled environment. This approach minimizes the impact of external factors such as network fluctuations, bandwidth limitations, or other external variables that could affect test results.

Details will be updated once VMs are available.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | IP Address | Hostname | # CPU Speed | Memory |
| 1 | 10.82.111.113 | LNZT1PERF101 | Intel(R) Xeon(R) Gold 6430, 2100 Mhz, 8 Core(s), 8 Logical Processor(s) | 32 GB |
| 2 | 10.82.111.114 | LNZT1PERF102 | Intel(R) Xeon(R) Gold 6430, 2100 Mhz, 8 Core(s), 8 Logical Processor(s) | 32 GB |

# Test Management & Controls

## Defect Management

* The Performance Testing activities will use SG Jira Defect Management System in line with the Overall SG Test Strategy.
* Any issues/bugs found will be first raised to the project team, who will provide initial triage. If it is a bug, then the defect management process will be followed and be managed by the project team.
* Severity for any incidents observed during the testing stages will be classified as follows:

|  |  |
| --- | --- |
| **Defect Severity** | **Description** |
| **1 – Showstopper** | Such a defect does not allow the application to work properly due to system failure or corruption of data. Critical defects do not allow testing to continue |
| **2 – Critical** | The critical defects are little less severe than showstopper defects. They can cause system to fail, however in case of critical defect there is a workaround or able to test other features or functions. |
| **3 – Major** | These defects do not cause the system to fail but produce wrong or contradictory output. |
| **4 – Minor / Cosmetic** | Minor: Defects that do not cause system failure or affect the usability of the system and can be easily rectified are known as Minor defects.  Cosmetic: Defects related to the outlook or appearance of the system are called cosmetic defect. |

* Priority for any incidents observed during the testing stages will be classified as follows:

|  |  |
| --- | --- |
| **Defect Priority** | **Description** |
| **P1 – Immediate** | **Critical**impact on the**system**  **Major**impact to testing (blocking)  **No**workaround  **Must be fixed immediately** |
| **P2 – High** | High **impact on the** system  High **impact to testing**  **Workaround**exists but must be fixed as soon as possible |
| **P3 – Medium** | **Medium**impact on the**system**  **Workaround exists.**  **Should be fixed but to be agreed when.** |
| **P4 – Low** | Minor **or** no effect **on the** system  To be fixed at a later stage if agreed |

## Entry and Exit Criteria

## Entry Criteria

The following are the pre-requisites for the test preparation and execution to commence.

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Entry Criteria Description** | **Action Owner** | **Due By** |
| Test Preparation | All outstanding information has been received and incorporated into the Performance Test Plan | Planit / SGI | Start of test execution |
| Performance Test Plan has been reviewed and approved | SGI | Start of test execution |
| There are no functional defects blocking scripting/testing from commencing | SGI | Start of test scripting |
| The test environment has been verified against production and any deviations assessed for potential impacts to results validity | SGI | Start of test execution |
| All existing Jmeter scripts have been verified | Planit | Start of test execution |
| New Jmeter VM instances have been created and Jmeter configured | Planit | Start of test execution |
| Test data required to support load set up as specified | Planit / SGI | Start of test execution |
| Preloading of test data generated by Jmeter scripts has been completed | Planit | Start of test execution |
| Database backup has been taken and restore process has been verified | SGI | Start of test execution |
| Test Environment has been uplifted and confirmed to be as per Production | SGI | Start of test execution |
| Test Execution | Test tool infrastructure verified connectivity established, and test tool deployed and configured | Planit | Start of test execution |
| Server monitoring support available to provide Planit with results on test completion | SGI | Start of test execution |
| Test scenarios designed and built in test tool | Planit | Start of test execution |
| Shake-out tests for test environment have been completed successfully | Planit | Start of test execution |

## Exit Criteria

Testing will be deemed complete when these activities have been carried out.

|  |  |
| --- | --- |
| **Entry Criteria Description** | **Action Owner** |
| Test Summary Report (TSR) has been reviewed and signed-off by relevant stakeholders | Planit / SGI |
| Performance Testing has been completed, signed off and results are available | Planit |
| A list of all performance testing cases which have not been executed has been documented showing reasons why execution could not be completed including residual risk statements and this has been reviewed and approved by the Test Manager (or delegate), Project Manager and Development Lead | Planit/ SGI |
| Root-cause analysis has been carried out for all performance issues found during the test phase | Planit / SGI |

## Suspension Criteria

Performance testing will be suspended during the following scenarios.

* If sign off is still pending prior to test execution.
* When performance or functional issues block further testing
* Environment unavailability

## Resumption Criteria

* Performance issues blocking performance testing are fixed and have been validated.
* The existing performance and volume scripts do not need any re-factoring/re-work.

## Test Reporting

Planit will create clear and meaningful charts for reporting on Performance Testing. These charts will convey key insights derived from data analysis, aiding informed decision-making and system performance optimization.

## Deliverables

The key deliverables will be the production of the final Test Summary Report (TSR) as well as interim test summary reports during the test execution phase. The TSR will outline:

1. Type of testing completed (including target component/s and workload model) and any remediation conducted.
2. Objective of the test conducted.
3. Results gathered, observations and recommendations.

At the completion of testing, the following assets will be handed over to SGI as a repeatable test suite.

1. Performance Test Plan (this document)
2. Performance Test scripts
3. Performance Test results
4. Test Summary Report
5. Detailed Handover Document

## Roles & Responsibilities

|  |  |  |
| --- | --- | --- |
| **Roles & Responsibilities** | **SGI** | **Planit Performance Test Lead** |
| Workload Model |  |  |
| Performance Test Plan |  |  |
| Test Data Creation |  |  |
| Create new JMeter Test Scripts |  |  |
| Performance Test Execution |  |  |
| Performance Tuning |  |  |
| Reporting |  |  |

## Dependencies & Assumptions

## Dependencies

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Area** | **Description** | **Impact** | **Due By** | **Owner** |
| PTD\_01 | Scope | The Scope should be signed off prior to test preparation | Potential delay to deliverables, possible impact to costs | Start of test preparation | SGI |
| PTD\_02 | Documentation | Access to SGI infrastructure and configuration documentation | Potential delays to deliverables | Start of test preparation | SGI |
| PTD\_03 | Environment | The performance test environment will be sized and configured as per production | Potential to invalidate test results and delay to deliverables | Start of test execution | SGI |
| PTD\_04 | Support | Access to key stakeholders for review and sign-off documentation | Potential delay to deliverables | Throughout the engagement | SGI |
| PTD\_05 | Support | Technical support will be available from SGI throughout the engagement to assist with requirements gathering, data creation, environment provisioning, monitoring, and issue investigation | Potential delays to deliverables | Throughout the engagement | SGI |
| PTD\_06 | Data | System / Database data support will be provided by SGI to support the test execution | Potential delays to deliverables | Start of test preparation | SGI |

## Assumptions

|  |  |  |
| --- | --- | --- |
| **ID.** | **Area** | **Description** |
| PTA\_01 | Monitoring | It is assumed that server monitoring is available across all in-scope environment and some support would be provided to capture stats, if no direct monitoring is available for the test team |
| PTA\_02 | Environment | It is assumed that the test environment is similar or will be appraised to production environment. |
| PTA\_03 | Test Data | The data used for performance testing will be a subset of the production data. However, it may consist of dummy test data and will not include any actual player information. |

## Risk and Issue Management

This section describes risks related to project delivery that may affect the progress of performance testing and impact the estimates and proposed schedule:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Risk Description** | **Impact** | **Mitigating action** | **Owner** |
| PTR\_01 | The initial test plan has been created prior to all information being received from the project | Delays in running tests and providing results. Possible changes to test scope | Work with the project team to identify any missing information and confirm objectives | Planit / SGI |
| PTR\_02 | There is a risk that the performance test plan is not agreed and signed off by all relevant stakeholders prior to testing commencement | Scope is not agreed, and test commencement or completion may be delayed | Close coordination of review and sign-off processes | Planit / SGI |
| PTR\_03 | There is a risk that while every effort is made for the test environment to match production there will be differences in Network, Firewalls, Load Balancer, Interfaces | This may impact the accuracy of the captured performance results and lead to delays in execution and a possible reduction in scope | Close coordination with the infrastructure team to validate the test environment matches production prior to test execution | Planit / SGI |
| PTR\_04 | There is a risk that a large number of blocking performance defects are raised during testing | If serious defects are discovered this may block further testing activity leading to delays in execution and possible reduced testing scope | Ensure all defects are fully understood. Potentially move ahead with testing not affected by defects if possible | Planit / SGI |
| PTR\_05 | There is a risk that the Performance team do not have access to appropriate systems to enable monitoring or investigation of issues found during testing | Delays in gathering data for issue resolution, leading to progress slippage | Liaise with project team and ensure relevant access is provided to the Performance team | Planit / SGI |
| PTR\_06 | There is a risk that the lack of clearly defined NFRs / SLAs impact exit criteria | This would cause delays in the test cycle and potentially impact the sign-off of tests | Communicate the response times to key stakeholders at an early stage and confirm they are acceptable. | Planit / SGI |
| PTR\_07 | There is a risk that suitable test data cannot be extracted or setup before the test execution | Delay to test execution and potential impact to overall timescales if additional data needs to be created | Identify the data requirements early | Planit / SGI |
| PTR\_08 | There is a risk that database snapshot is not available for rollback after every performance test. | Over time, junk data generated during performance tests will be accumulated in the database, causing it to bloat. This database bloat can negatively affect subsequent test results, especially if tests are designed to evaluate performance under real-world conditions. | Before each performance test, take a snapshot or backup of the database, then roll it back to that state after the test completes. This ensures the database is restored to its original state, eliminating accumulated junk data. | SGI |

# APPENDIX

## Appendix A - Glossary

| **Term** | **Description** |
| --- | --- |
| SLA | Service Level Agreement |
| Response Times | See Transaction Times |
| Throughput | Throughput is the rate at which data is sent through the network, usually expressed in bits per second (bps), bytes per second (Bps) or packets per second (pps). Throughput most commonly refers to the total data transfer rate for all traffic being carried Throughput is measured by counting bytes transported during a specified time interval. |
| Transaction Times | Transactions times are used to measure the performance of the server. Each transaction measures the time it takes for the server to respond to specified requests. These requests can be simple tasks such as waiting for a response for a single query, or complex tasks, such as submitting several queries and generating a report. A number of Transactions make up a Business Scenario. |
| 90th Percentile | The maximum response time during performance testing can sometimes be exceptionally long and not indicative of real-life experiences. By using the average of all transactions and including these “outliers” times are often skewed. Therefore, using the 90th percentile allows for the spurious results to be removed. |
| User Ramp-up | User ramp-up involves gradually increasing the number of users accessing the system over a specified period to evaluate its scalability and performance under increasing loads. |
| Concurrent Users | Concurrent Users refers to the number of users accessing the system simultaneously. This metric is crucial for assessing system performance under real-world usage conditions, as it measures the system's ability to handle multiple user interactions concurrently. |
| Non Functional Requirements (NFR) | These specifications define how a system should behave rather than what it should do. NFRs typically encompass various attributes, including Performance, Scalability, Reliability, Security, Usability, Maintainability and Compliance |
| Baseline/Sub-Peak Tests | Simulates a Wednesday or Saturday Lotto and Powerball draw > 10m. The measurements will be used as benchmarks for comparison. |
| Peak Load Tests | Represents high traffic days, i.e. MBW draw, simulates a stable high traffic profile for the test duration with frequent short spikes up to much higher numbers to monitor if the system recovers within the anticipated timeframe. |
| Scalability Tests | Run under controlled environment often against a single node in the cluster, e.g. single Mule instance with default workers & other configurations. These tests are run to gauge additional compute requirements by scaling vertically or horizontally under Optimisation/Tuning work stream |
| Soak or Endurance Test | Run with Baseline load for an extended period with sudden short spikes up to peak in between, i.e. 24+ hours. This test is run to check if there are any memory leaks in the code and whether the system can maintain a healthy state under high traffic. |
| NFRs | Non-Functional Requirements |
| Successful Test | A performance test that runs, completes successfully and has met all the Acceptance Criteria set |
| RCA | Root-cause Analysis |
| SLAs | Service level agreements |
| RAID Log | Risks Assumptions Issues Dependencies |
| Defect | A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g. an incorrect statement or data definition. A defect, if encountered during execution, may cause a failure of the component or system. |
| Entry Criteria | The set of specific conditions for permitting a process to go forward with a defined task e.g. test phase. The purpose of entry criteria is to prevent a task from starting which would entail more (wasted) effort compared to the effort needed to remove the failed entry criteria. |
| Exit Criteria | The set of specific conditions agreed upon with stakeholders, for permitting a process to be officially completed. |
| MBW | ‘Must Be Won’ Draws |
| TPM | Transactions per minute |
| LTM | F5 Big-IP Local Traffic Manager |
| WLM | Workload Model |

# Customer USEr journeys

## Digital User Journeys

Below User Journeys will be considered during **Phase 2** Performance Testing:

## Pre-Draw User Journey

|  |  |
| --- | --- |
| **#** | **Details** |
| 1 | Login > Wager PYO |
| 2 | Login > Top Up > Wager PYO |
| 3 | Login > Wager Triple Dip |
| 4 | Login > Top Up > Wager Triple Dip > Wager IKO |
| 5 | Login > Wager IKO |
| 6 | Login > Top Up > Wager IKO |
| 7 | Login > Wager PYO > Wager IKO |
| 8 | Login > Top Up > Wager PYO > Wager IKO |
| 9 | Login > Wager Keno |
| 10 | Login > Top Up > Wager Keno |
| 11 | Login > Wager Bullseye |
| 12 | Login > Top Up > Bullseye |
| 13 | Login > Wager Lotto PB Favourite |
| 14 | Login > Top Up > Wager Lotto PB Favourite |

|  |  |
| --- | --- |
| **Use Case** | **Components** |
| **EMR-71: Session Login** |  |
| **EMR-65: Fetch Wallet Balance** |  |
| **EMR-70: List Claims** |  |
| **EMR-87: Retrieve Player Profile** |  |
| **EMR-95: Top Up Wallet** |  |
| **EMR-79: Retrieve Blocked Games** |  |
| **EMR-88: Retrieve Spend Limits** |  |
| **EMR-61: Fetch EDE Tickets and Results** |  |
| **EMR-51: Create Wager** |  |
| **EMR-59: Fetch Favourites** |  |
| **EMR-230: Fetch Current Draw** |  |
| **EMR-84: Retrieve IKO List of Games** |  |
| **EMR-96: Try IKO Games (Not Logged in)** |  |
| **EMR-46: Try/Buy IKO Games (Logged in)** |  |
| **EMR-72: Session Logout** |  |

## Post-Draw User Journey

|  |  |
| --- | --- |
| **#** | **Details** |
| 1 | Login > EDE |
| 2 | Login > EDE > Wager Lotto PB > Wager IKO |
| 3 | Login > EDE > Wager IKO |
| 4 | Ticket Scan > Result |
| 5 | Ticket Scan > EDE > Ticket Scan |

|  |  |
| --- | --- |
| **Use Case** | **Components** |
| **EMR-71: Session Login** |  |
| **EMR-61: Fetch EDE Tickets and Results** |  |
| **EMR-51: Create Wager** |  |
| **EMR-64: Fetch Ticket History** |  |
| **EMR-91: Scan Ticket** |  |
| **MR-230: Fetch Current Draw** |  |
| **EMR-88: Retrieve Spend Limits** |  |
| **EMR-59: Fetch Favourites** |  |
| **EMR-72: Session Logout** |  |