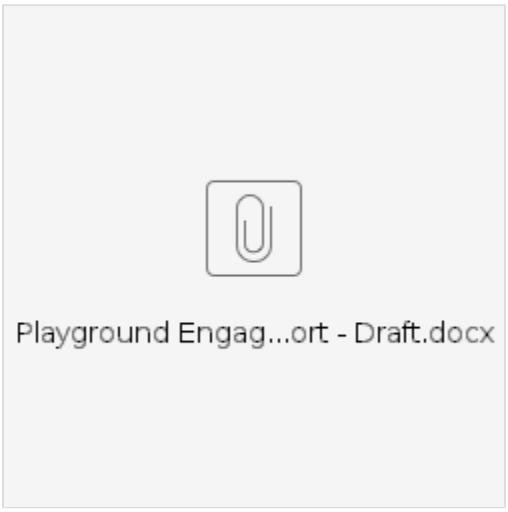
PlayGround Engagement

Assessment Report -



Major Callouts or Concerns -

1. Infrastructure strategy for Non Prod and Prod has scalability and availability risks due to access limitations and gaps on historical understanding of the configuration/implementation insights
2. Delivery Velocity is minimal due to constant change in the requirements and un-planned requests from customers or stakeholders

Delivery Framework -

In the past, there is no standard delivery framework and practices were followed; while the team has self realised and defined basic standards to take the first step towards the transformation to accelerate the deliverables while they structure multi-location delivery model between India and Uruguay.

Team has spent most of the effort in delivering to the random requirements then culturally adopting to delivery principles. In the past there is lack of discipline in keeping the books for the deliverables in JIRA system and there is effort to address this is been taken.

Planning - Portfolio management

- Roadmap and Release Planning are managed through local document; to managed in Confluence.
- Multiple Role Sandhya, Biplab - A
- Roadmap should include the Technical Requirements
- Jira Story ALM needs to mature to support multiple scrum team model and roles - currently lean due to multiple roles assigned to single owner due to lean size
- Product Backlog - Roadmap (Every Month)
- 2 Quarterly Planning and Roadmap is revisited every month
- Steering Committee is planned and to be started from next release onwards
- No Product Direction yet; with Steering committee to start may give the driving factor; product owner is not expertise of the domain, currently it is more of transfer of customer requirements to features list

Business Stakeholders & Steering Committee - Raj S , Raj V, Abhi, Rob, Ryan, Anil Somani

Consumer -

- TA Community
- EL
- Engineers
- Clientele - BNYM & JPMC - Pre-Sales requirements and not active implementations

Major release train is monthly and follows 2 - 2 week sprints model - with built in slots for refinement, development, testing phases, acceptance sign-off achieving within the Sprint activities or ceremonies.

Release Hardening sprints for 1 week; to address Certification, BugFix activities for every user stories.

DoD and Acceptance Criteria are defined and to be followed from upcoming sprints onwards. In the past there is no evidence of documentation.

Requirements are organised as Epic and User Stories. Work Items for these Stories are managed as Task.

Stories are organised and sized using INVEST principles. It is evident that the majority of the stories planned for Sprint 8 and Sprint 9 are not more 5 pointers except one technical debt story is 13 pointer which is due to heavy testing task and a 8 pointer.

UAT include within the sprint is also covered; UAT Test cases are not available; PO does the UAT.

Team velocity is measured at 25 story points for sprint 7 - 63 SP Targeted for Sprint 9 (5 BE, 4 FE, 5.5 QE, 1.5 DevOps, .5 DB, 1 Arch, 1 UXD)

Three Scrum Team distributed as 2 Backend Teams in Bangalore India and 1 UI/Frontend team in Uruguay. Scrum teams are distributed as per the capabilities due to availability of talent in Uruguay.

Scrum of Scrum is conducted for 3 30 minutes ceremonies for a sprint - Th, Tu, Th. Story Time meetings to collaborate on refinement activities are done twice a sprint for 1 hour each - Thu, Wed

Scrum Ceremonies Planned - (India team is practising the ceremonies from Sprint 8)

Daily Scrum Meeting

Runahead team meeting - PM, SM, EL, & TA - Roadmap Review, Release Plan, Bug Prioritisation

Story Time with India Team (Backlog grooming) Story Refinement

Final Integration testing in UAT environment

Sprint Retrospection (India)

India Team Next Sprint planning

Scrum of Scrum meeting with Uruguay team -

Story time with Uruguay team (backlog grooming) - 2 hours - 1st Thu and 2nd Wed

Combined Sprint Demo - 1.5 hours - 2nd Monday

Uruguay Team Next Sprint planning - 6 hours - 2nd Tuesday

REST API Contract Finalization - 1 hour - 1st Wednesday

Branching and Environment Strategy -

Branching	Environment
Feature	3 Dash - Dev
Develop	2 Dash - QA
Master	1 Dash - UAT
Release	Production

There is no Integration environment which will lead to quality issues slipped to UAT and Prod environment. UAT is an integration environment to demo the user stories to the PO.

There is no Sustenance/DR Environment to manage HotFix Validation maturely and handle Production roll out seamlessly; delay of production push is limited to downtime availability from product usage by the user. Production Push is likely to take 2 hours maximum time becoz only few components out of 29 components are candidates for a release due to minimal content delivery for each releases.

Release Management -

There is no Release Manager function to define, and drive the release management activities. Product Owner, Engineering Leader and Developers drive the release management activities. No Release Playbook evident. No mature release approval process exist.

Incremental push, on demand component push is done. So roughly around 2 hours activities are expected.

Release Cadence is immature and the team is planning to re-defined the Release Cadence.

Backend Tech Stack -

Orchestration Server - Primarily supports Orchestration and light on API Gateway - Zulu - Api Gateway

User Management Server - Supports Federated Identities and Social Authorisations - key cloak migration

Redis - Session Caching and SQL Result Caching

Java 8 - Microservices

GIT Lab Server -

Eclipse IDE - SHA Cloud Version

Harper - Container

Kubernetes -

HackRanker

Mettl API

Udemy -

Salesforce - SFDX

MySQL Database

MicroService Template for ELK stack, Zulu API Gateway, RabiitMQ, etc

Swagger - API Doc Management

Jasper and Python

UI Tech Stack -

NodeJS

AngularJS7

HTML5

Tool Chain Framework -

Slack For Communication

JIRA - Issue Management and Planning

Docker Containers

JUnit & Mockito

Selenium Automation

RestAssured and Cucumber

TestLink

JIRA - Agile Management

Bitbucket -

Jenkins -

JFrog - Used for Docker Image Repository; not effectively used for artifact repository

AWS EC2 Instances - Non Prod and Production Environment Infrastructure Cloud Solution

Photoshop and Envision - Used for UI Prototype and Wireframe designs and sharing with community

Eclipse IDE - Used by Developers for coding locally in their desktop or workspace

Consul - Used for service mesh solution to control the service discovery, configuration and segmentation aspects of different services being hosted

No Configuration Management Tool

No Deployment Orchestrator

Deployment Process -

3 dash - Environment - Feature Branch (On-demand)

2 dash - Environment - Develop Branch (once in 2/3 days)

1 dash - Environment - 2 or 1 time in a Sprint for Planned Release and for hot fix on demand

Hotfix -

Branch out from Master for Validation - Porting the fix and build & deploy for certification

Branch out from Release for HotFix Implementation and Delivery to Prod

Build is generated while PR is approved on Release Branch rather baseline build is generated to validate before merging the changes from PR of the Hotfix branch

Porting is done for pushing the changes from Release branch to lower branches

Single Branch Strategy - Sideline Proposal from team

Development Practices -

27 Backend + 2 FrontEnd - Services without common/unified templates or framework. These services are not truly isolated, they have dependencies New framework/template is in progress to bring in standard development practice and migrate to true Microservice model

To setup a CI pipeline job for New Service takes few minutes to configure

No Redundancy and High Availability is not configured

Sonarcube - No CodeSmell Check; but standard rules are configured

Run code coverage using Jococo - <15% Unit Test but aiming to 80% coverage; pass rate are not tracked now

Bugs found by QE is measured Dev Capability

Gates - Not Available now

Code Review - As part of PR request and not religiously followed

TDD - Requires Investment and not ready now to take it up

Peer Programming - Can be tried - Still now ready with maturity

Infrastructure is not allowing to scale

Security Analysis - No

20 to 25 Story Points lands to Release with a 12 member team which is low. Reason is the UI completed stories are pushed to production and backend completed stories are kept aside in the feature branch till the UI task are completed.

Programming Process involves following practices -

- Just Enough Design
- Validate Unknowns
- Execute Design
- Other Modern through review of dependency packages
- API Definition and analysis
- Coding
- Test

Tool Chain -

BitBucket -

Manual Code Review - No FishEye or Crucible System to track the Code Review

No Code Configuration Repository Workflow with JIRA, BitBucket

No ticketing system - For DevOps, Infrastructure, CloudOps

No Database Schema Configuration system - Managed with GIT - No Version, No Rollback - Flyway can be used

DB is managed with container

Sematext - Log Management Tool

Name Convention for Server Name are not followed - Requires tribal knowledge

JFrog - Used for iMage repository - No credentials to validate

All Environments - Maven Repositories are not mapped to Centralized Repository - JFrog

Don’t have Component wise - Dependency Management

Jenkins -

CI Jobs are configured with different stages but no unit test and other quality related stages and gates available

Pipeline as code - Can spawn the CI with any Jenkins instance without major work

High Risk - Credentials and Access for multiple system is not available; to mitigate the risk, downtime of the systems are not available.

No Topology diagram for the environment and infrastructure

High end configurations with tiny utilisations - No automated configuration management and infrastructure/Orchestrator management

Infrastructure - AWS EC2 Cloud

Role Setup in AWS Users are incomplete

Lack of consolidated and formalized AWS Account and Environment

No Monitoring and Watching capabilities configured for the effective utilisation of the environment and failures

Current infrastructure doesnt have implementation design and topology, no standard sizing of the application is available

Metrics and Measurements of effective utilization are required

Side Notes -

- Strengthening the Backlog artefacts through Steering Committee Approval process - Part of PG2.0
- Rudimentary Bug Management Configuration - Requires Standardisation
- Bug Classification is missing - Legacy or InSprint
- User Storie Life cycle was not enforced
- Angular Upgrade and KeyClock Implementation are done in silo’s.
- No Evidence till Feb 10 to analyse the implementation of new process
- Integration and Certification is planned within the sprint; experimenting it with current release for Feb 10