

High Level design and technical approach

**Table of Contents**

[Document Purpose 4](#_Toc23373)

[Overview 5](#_Toc23374)

[Section Purpose 6](#_Toc23375)

[Platform Architecture 7](#_Toc23376)

[Production Architecture v1.1 7](#_Toc23377)

## Document Purpose

. This document will include information about the design and technologies used and the proposed architecture

***Overview***

Platform Architecture

The below architecture has been used for the basis of the architecture of the solution and the move to the cloud (**Azure Cloud** is used in this design and approach)

### Diagram Description automatically generated

### Target audience

This document is only for POC and pilot purposes

### Environments setup

* The Solution will be deployed into three environments to isolate resource and responsibilities between different teams:

1. **Dev** Environment: in this environment there is two stages

**IOC build stage** is executed to create the initial infrastructure as code for the hosting of the Docker images and the MySQL database During the dev environment roll out

**compile code stage** : will maven build the source code and upland the artifacts and the docker image\K8s manifest to the build artifacts will be tested for vulnerabilities and issues using Jasco Code coverage tool (\* Junit test and JMeter tests also can be incorporated in this for enhancement)🡪Target Audience Dev \ Dev ops team members

1. **Staging** Environment: the code has passed the quality gates and checked into the Docker images registry (azure container registry in this proposed design) and we will use the image to deploy it on an azure web app for containers. the web app will have a staging slot which will be used for A / B Testing and lead and warm up enhancement. the app will be monitored and audited using application Insights as well the auto Scale functionality of the app is setup using Azure Monitor 🡪 target audience Testing team and early adapters and service evangelists
2. **Production Environment:** after a successful UAT and quality gate success the code will be deployed to a production environment which will be consisting of Azure for MySQL server database and Azure Kubernetes Service Cluster. the AKS cluster will connect to the images registry to obtain the latest images (or whatever tag needed) to have it in a single node but with 3 pods. Monitoring and load balancing is enabled the cluster

### Approach considerations

1. **Cloud based resources** resources to deviate from any costs associated with on prem (infra or software license costs)
2. **Config as Code approach**: all the deployment | environments and stages as well as the infrastructure that will be created and the permission given to these resources are provided as config and near to zero manual work is required
3. **Security and Isolation of environments and responsibilities:** as best the solution provide different layer to adhere to security concepts and allow multiple teams to work on the pipeline with no issues
4. **Automated builds and code tests**: all builds will have to be built with no issue and a condition can be set on the pipeline to gate the build based on the results from the code coverage tool
5. **containerization**: the app is packaged not a docker for redistribution to different environments
6. **Monitoring and auditing:** all resources deployed either code or infrastructure are monitored either using Azure application insights in case of the web app or azure audit logs in case of ARM resourced deployed or azure monitor and container insights with log analytics workspace in case of AKS
7. **Ease of Replication**: this design was made to ease redistributing the app to different teams with different configuration with minimum rework

### Enhancements to be done

1-multi containerize the application to use front end a backend and database layer infrastructure

2-automathe the creation of service connections and service principals rather than asking for manual input

3-define the permission hierarchy and teams within the dev ops project

4-add Unit testing to the app

5-incoroprate automated web testing case (Selenium) and performance testing (LoadRunner, etc)

## 