**Application Architecture and benefits**

* + Microservices are **Independently Executable, Upgradable , Replaceable and Scalable**
  + **Domain specific features** – Achieve **Single Responsibility Principle**
  + **Light-Weight , Independent, Loosely Coupled** business unit
  + Owns **Codebase, Managed, and Developed** by a small team.
  + Own their **Database – Decentralized Data Management**
  + Choose **Best Technology Stack - Evolutionary Design**
  + Test, Release, Deploy, Scale, Integrate, and Maintain Independently – **Own Devops Plan**
  + **Smart Endpoints and Dump Pipes -** REST over HTTP or asynchronous messaging
  + **Recovery** and **Fault Tolerance – Auto Healing Capability**
  + **More Effective , Efficient Deliverables** and **Customer Focused**

**Implementation of application**

12 Factor architecture

I. Codebase

II. Dependencies

III. Config

IV. Backing services

V. Build, release, run

VI. Processes

VII. Port binding

VIII. Concurrency

IX. Disposability

X. Dev/prod parity

XI. Logs

XII. Admin processes

* Distributed/versioned configuration
* Service registration and discovery
* Routing
* Service-to-service calls
* Load balancing
* Circuit Breakers
* Global locks
* Leadership election and cluster state
* Distributed messaging
* Two-factor authentication

**PURPOSE OF CONFIGURATION-SERVICE**

-----------------------------------------------------

Eureka server acts as the registry to which other services register themselves.

When one microservice needs to communicate with another,

it can query Eureka to find the active instance of that microservice.

**PURPOSE OF DISCOVERYSERVICE**

--------------------------------------------

Eureka server acts as the registry to which other services register themselves.

When one microservice needs to communicate with another,

it can query Eureka to find the active instance of that microservice.

**PURPOSE OF GATEWAY-SERVICE**

---------------------------------------------

Zuul is (of course) our gatekeeper to the outside world,

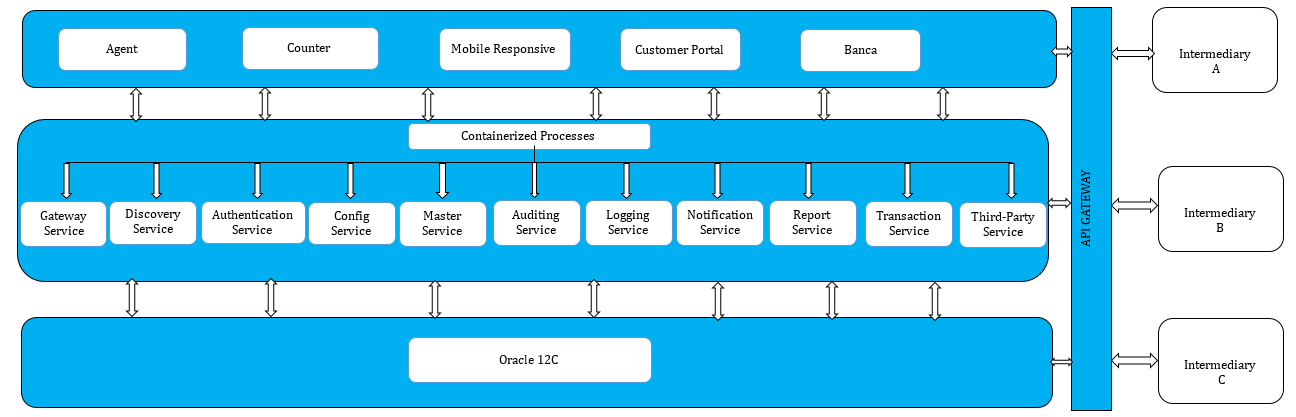
not allowing any unauthorized external requests pass through. Zulu also provides a well-known entry point to the micro services in the system landscape. Using dynamically allocated ports is convenient to avoid port conflicts and to minimize administration but it makes it of course harder for any given service consumer. Zuul uses Ribbon to look up available services and routes the external request to an appropriate service instance.

ADD-ON for FALLBACK

-------------------

* Hystrix is a circuit-breaker. Hystrix lets you define a fallback method that gets invoked if your network calls to another microservice fails.
* It reverts back to normal behavior once the service is available again.
* Ribbon is added for loadbalancing

**HIGHLEVEL ARCHITECTURE**

****

**Cloud Compatible Architecture**

* Application is added with docker for containerization on below pattern

