BRIDGEAPPS

Customer Case Study

Business + Sales Enabling a Legacy System

A TECHNICAL JOURNEY

SUMMARY

Rearchitected a Monolithic legacy Software into Flexible Business-Driven modular Microservices bringing in benefits around - **Pricing, Flexibility, Stability, Scalability & Delivery Agility**
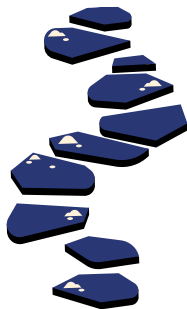
AT A GLANCE

CHALLENGES

- Unflexible Monolithic Architecture
- Poor Agility for Change
- High Installation & Run Cost
- Poor Product Scalability
- Poor Reporting & High Integration cost

BENEFITS

- Better Client Business Fitment
- Improved Agility for Development and release
- Cheaper to Support
- Better Reporting and Audit Support

PROBLEM STATEMENT

The ISV focussed on their product growth sales at an exponential rate resulting in multiple implementations while accruing technical debt, poor and inflexible architecture, complicated development and support processes, and multiple patches.

Onboarding newer customers and enhancing existing implementations on a monolith leads to business downtimes.

The solution should - do minimal changes to Codebase to resolve the problems faced.

OBJECTIVES

ReArchitect Existing Monolithic Solution, introduce Microservices Architecture, embed Agility as part of the product development methodology leading to :

- ability to sell module wise - leading to better sales propositions, economical and standardized
- ability to develop and bug fix in tandem, without impacting the stability of the bigger platform
- ability to separately operate modules while releases with other modules are being fixed.
- better fitment with customer businesses, agile workflows, better data and support management
- better integration capability allowing for stability of data transmission and auditability

> Our Startup Struggled to Scale up as we grew with more customers, with larger user base and more complicated requirements - both mandating functionality and integration into our Enterprise.
>
> ReArchitecting our Product into the new Digital model gave us the very edge we badly needed.

D Jose, eKapital Ltd.

AS-IS ARCHITECTURE

The Application was a Monolith with a System and a Backend Database.

The environment was initially migrated to AWS as a Cloud-hosted solution for QuickWin (branding-wise enabling better marketing). In this form, it was a Java-8 App hosted on an EC2 Instance connected to an RDS.Auth/ Auth was fully built-in as a customized layer.

This architecture had major drawbacks including - difficult to maintain with even simple changes - for e.g. placing a new Christmas offer had a cascading effect on other parallel changes, ongoing & releases.

This needed a complete rework, but with <u>minimal code changes</u> so as to maximize the ROI on investment from past years.
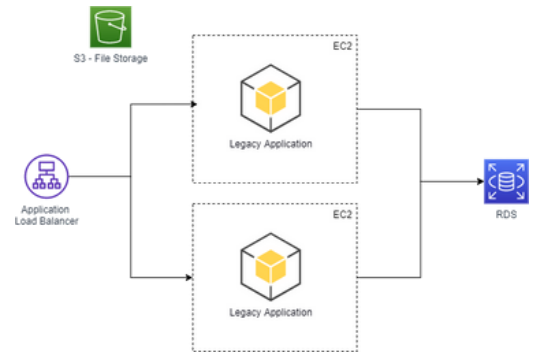

Fig 1. Source Architecture

TARGET ARCHITECTURE

The challenge extended the mandated minimum amount of code and schema changes ensuring the integrity of past implementations and paving way for easier migration to the renewed architecture.

At a high level, the solution involved -

- Splitting the UI layer with a "backend" layer - which included (at a high level) - business, data access, business rules, integration, error handling, and auditing layers.
- Each layer was split as a Microservice and exposed via multiple APIs - some over HTTP/JSON, others via Kafka
- All microservices share the same backend Data layer
- Orchestration of Microservices is applied only when data cannot be directly read from the backend, reducing noise - retaining processing power - and cheaper compute
- Deployment involved Containerizing codebase and separating code repositories per Microservice
- Leveraged ECS clustering services along with Fargate for hosting containers
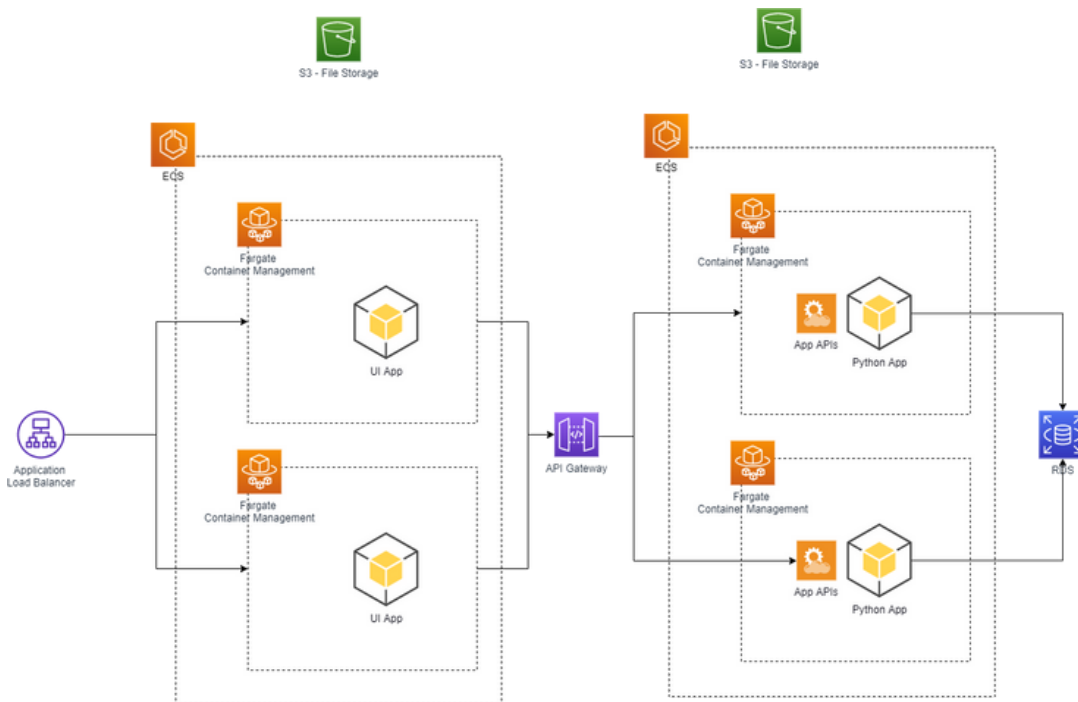- Secured Different layers to meet Strict Infosec Guidelines

Fig 2. Target Architecture

INTROSPECTION

The Technological change on products achieved two major changes -

1. Enabled product sales to separately sell functional modules based on customer demand, making it cheaper and more competitive
2. Enabled Delivery teams and Product Management to focus on the high-value modules for more business benefit and react quickly by embedding Agile methodology in its delivery process.