

Monolithic To Microservices Architecture

HOW MANTRA LABS MADE PROPERTYSHARE HIGHLY SCALABLE

MANTRA LABS



CLIENT

Property Share is a leader in making real estate investment accessible and manageable. They facilitate easy investment in real estate for retail/small investors and also allow them to manage their investments easily.

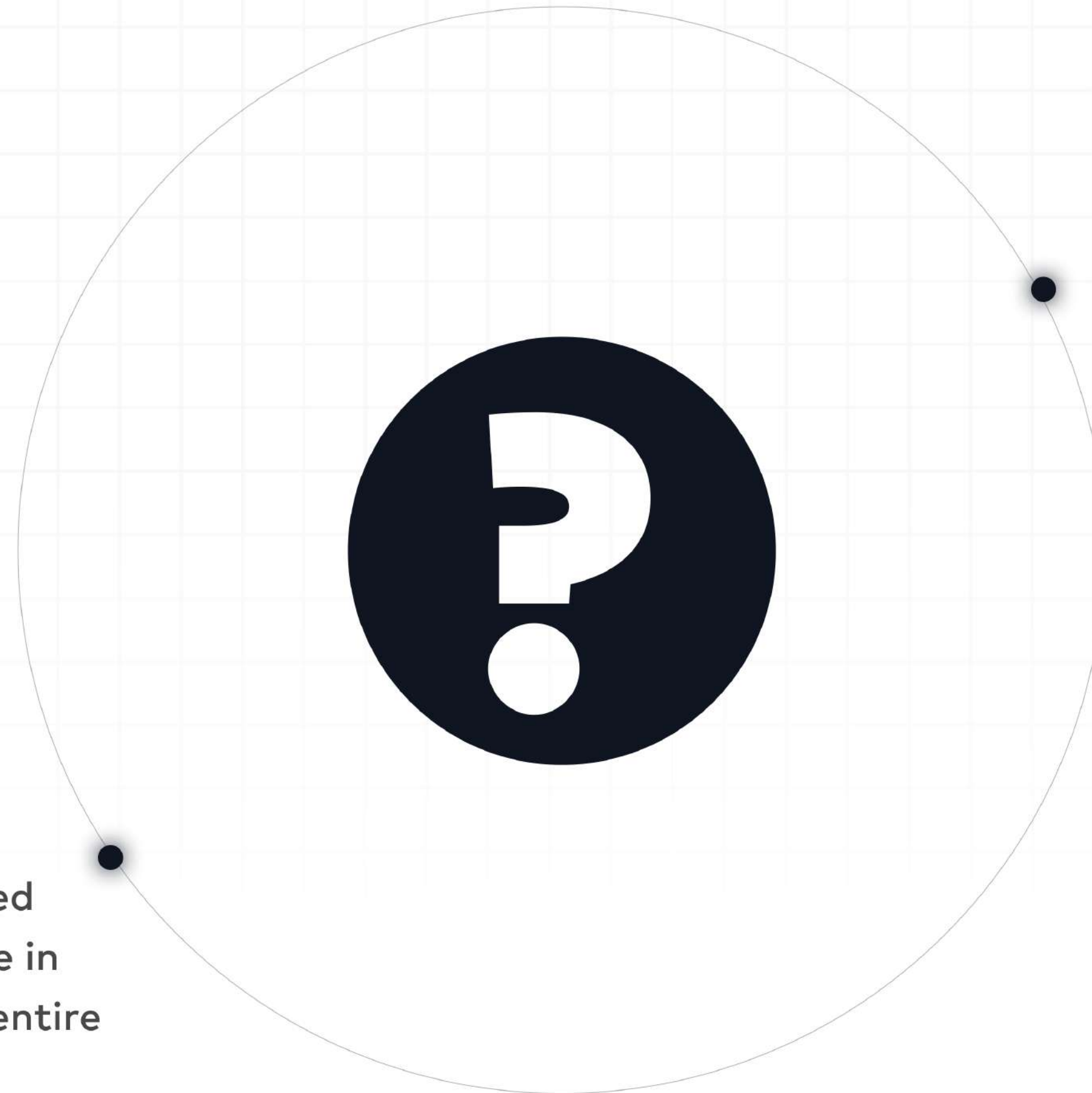
The company initially operated with an investment portal built on monolithic architecture. This setup, while straightforward, began to limit the company's potential as it grew.



CHALLENGES

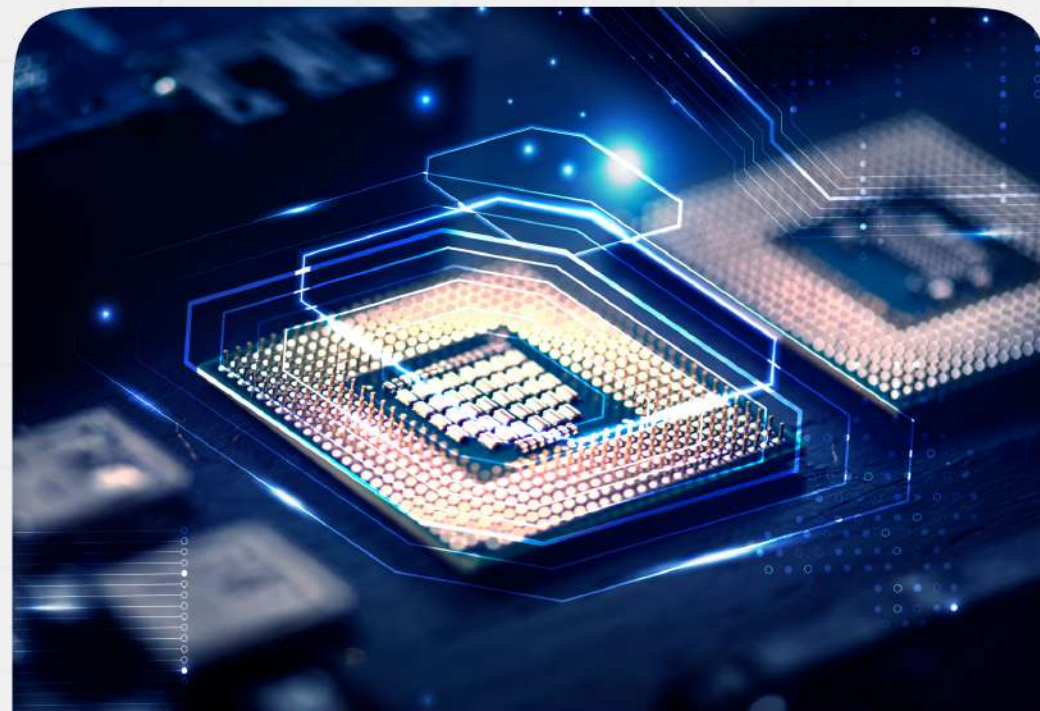


The monolithic architecture hindered scalability as development/upgrade in any module requires access to the entire code base.



To maintain their growth trajectory and improve services, they recognized the need for a significant technological overhaul.

OBJECTIVES OF THE MODERNIZATION



**MIGRATION FROM
MONOLITHIC TO A
MICROSERVICES
ARCHITECTURE**



**IMPROVING THE
SYSTEM'S ABILITY TO
MANAGE INCREASED
USER TRAFFIC**



**OPTIMIZING COST AND
SECURITY**

ORIGINAL SYSTEM ANALYSIS

**Assessing system for its
capability to handle
increasing user traffic**



**Ability to efficiently handle
growing user numbers and
data volumes.**



**Examination of the
codebase for high
complexity and
interdependencies**



**Identifying inefficiencies
in handling large-scale
data**



ORIGINAL TECHNICAL STACKS AND LIMITATIONS

Web Application

(PHP-based Platform)

Challenges in scaling and integrating modern, dynamic features.



Database

(Relational Database like MySQL)

Low flexibility and scalability to accommodate rapidly growing data needs.



Frontend

(HTML/CSS, Basic JavaScript)

Falling short of modern web application standards.



ARCHITECTURAL TRANSFORMATION



BREAKING DOWN THE MONOLITH

- Deconstructed the PHP-based monolithic application.
- This process involved carefully segmenting the existing codebase into more manageable and isolated parts.



DEVELOPING MICROSERVICES

- Over 20 different services, including vital communication services, were identified and reconstructed as independent microservices.



ENSURING SEAMLESS INTEGRATION

- Each microservice was developed to function independently yet seamlessly integrate with others.



TESTING AND DEPLOYMENT

- We also oversaw rigorous testing of each microservice to ensure stability and performance.

TECHNICAL INNOVATIONS INTRODUCED



CONTAINERIZATION AND ORCHESTRATION



Technologies like Docker for containerization and Kubernetes for orchestration are used to ensure that each microservice can be deployed, scaled, and managed efficiently.

CLOUD INTEGRATION



The team leveraged cloud technologies to enhance scalability and reliability. This integration allowed Property Share to manage resource allocation better and handle varying loads.

SYSTEM-WIDE IMPACT

- Upgrading a module like user authentication **module typically requires 1-2 weeks**
- coding, integrating the changes with other parts of the system, and extensive testing.



- Post migration **1-2 days.**
- The isolated nature of the module allows for quicker development and testing

1 week

Updates in monolithic architecture system



1 day

After microservices based architecture was implemented

- **AGILITY AND ROBUSTNESS**

The microservices architecture allowed for updates in individual modules without affecting the entire platform, significantly increasing system agility and robustness.

- **FUTURE-PROOFING**

This approach laid a foundation for easy adaptation to future technological changes and user demands.

NEW ADDITIONS AND UPGRADES



A new dashboard for investors was introduced, both a web and mobile application, developed using Flutter



A new web application for distributors, built using React, was rolled out, featuring API integration for enhanced functionality.



The investor web portal was transitioned from HTML/CSS to React, significantly improving the user interface and overall experience.



LESSONS LEARNED AND FUTURE SCOPE

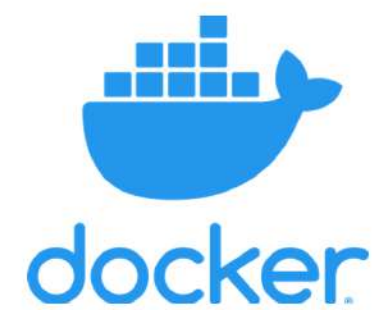
- The project underscored the importance of agility and robustness in software architecture.
- There's potential for further breaking down the application into more micro-services, paving the way for continuous improvement and adaptation.



TECH STACK

New Microservices Oriented Architecture

CONTAINERIZATION



Docker for individual microservices.

ORCHESTRATION



Kubernetes for managing containers.

MICROSERVICES DEVELOPMENT



Combination of PHP, Node.js, or Python.

API MANAGEMENT



REST or GraphQL for efficient data handling

TECH STACK

Cloud Integration and Deployment

- **CLOUD SERVICES**

Leveraged for scalability and resource management.

- **DEPLOYMENT**

Managed by **Mantra Labs**, ensuring smooth transition and minimal operational disruption.

This tech stack was pivotal in transforming Property Share's architecture, enhancing scalability, operational efficiency, and system robustness.

CONCLUSION

The journey of **Property Share** from a monolithic system to a microservices architecture was a pivotal step in their growth. It not only improved their technical capabilities but also positioned them for future innovation and success in the real estate investment sector.

CLIENTS



Thank You



P: +91 987- 033- 3426

E: hello@mantralabsglobal.com

L: India | Bengaluru, Kolkata

P: +1-872-362-0414

E: parag@mantralabsglobal.com

L: USA | North Carolina

