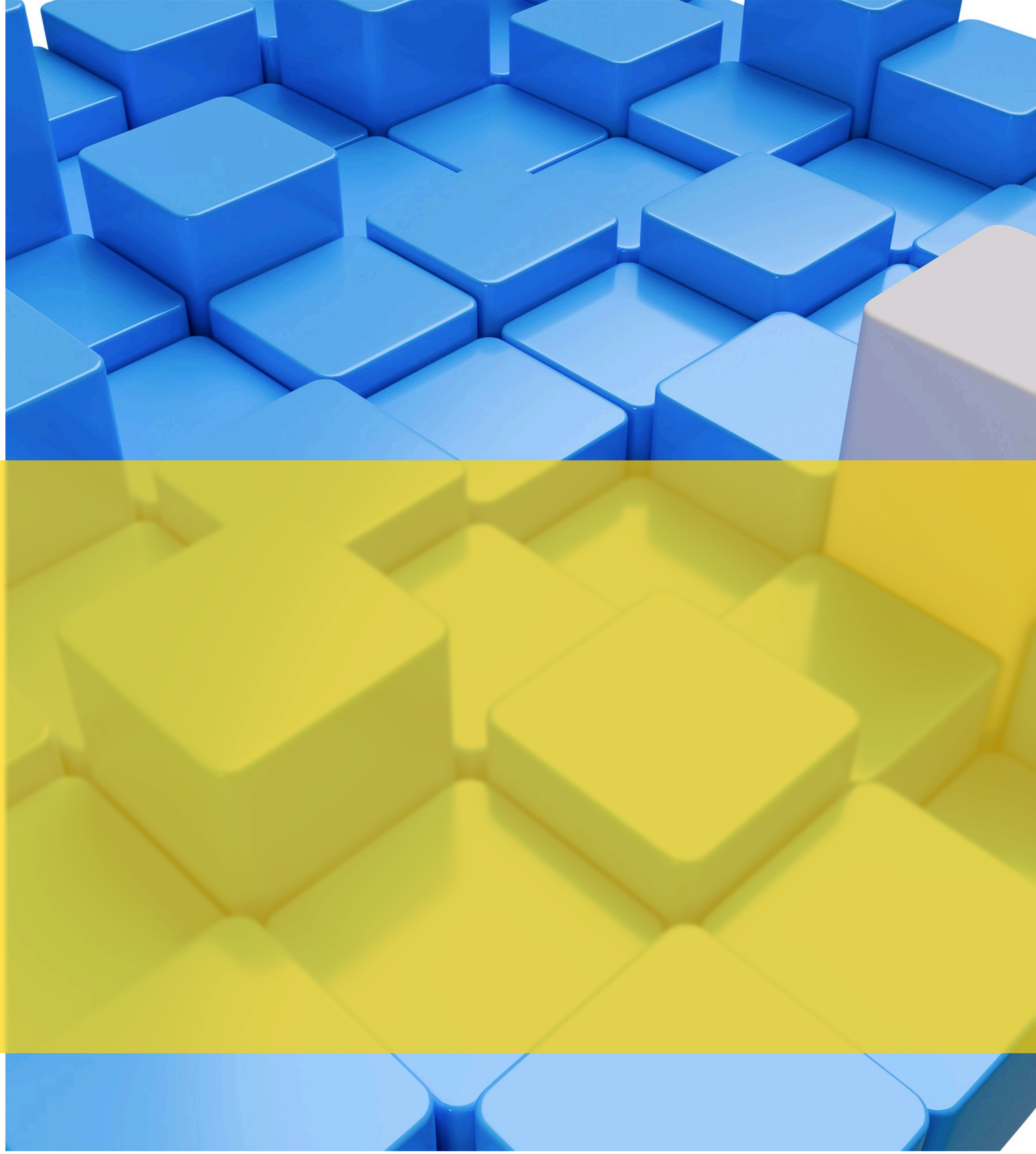
ANDERSEN



Bridging The Gap: Unlocking Value From Legacy Banking Platforms Through Integration And Automation

White Paper

Profile of Beneficiary Companies

At Andersen, we empower financial institutions, digital marketplaces, and payment service providers to modernize critical operations while preserving the stability of proven legacy systems.
Our integration and automation solutions help organizations bridge the gap between outdated infrastructure and modern customer expectations, delivering operational excellence without the risk and disruption of full system replacement.

We focus on organizations that:

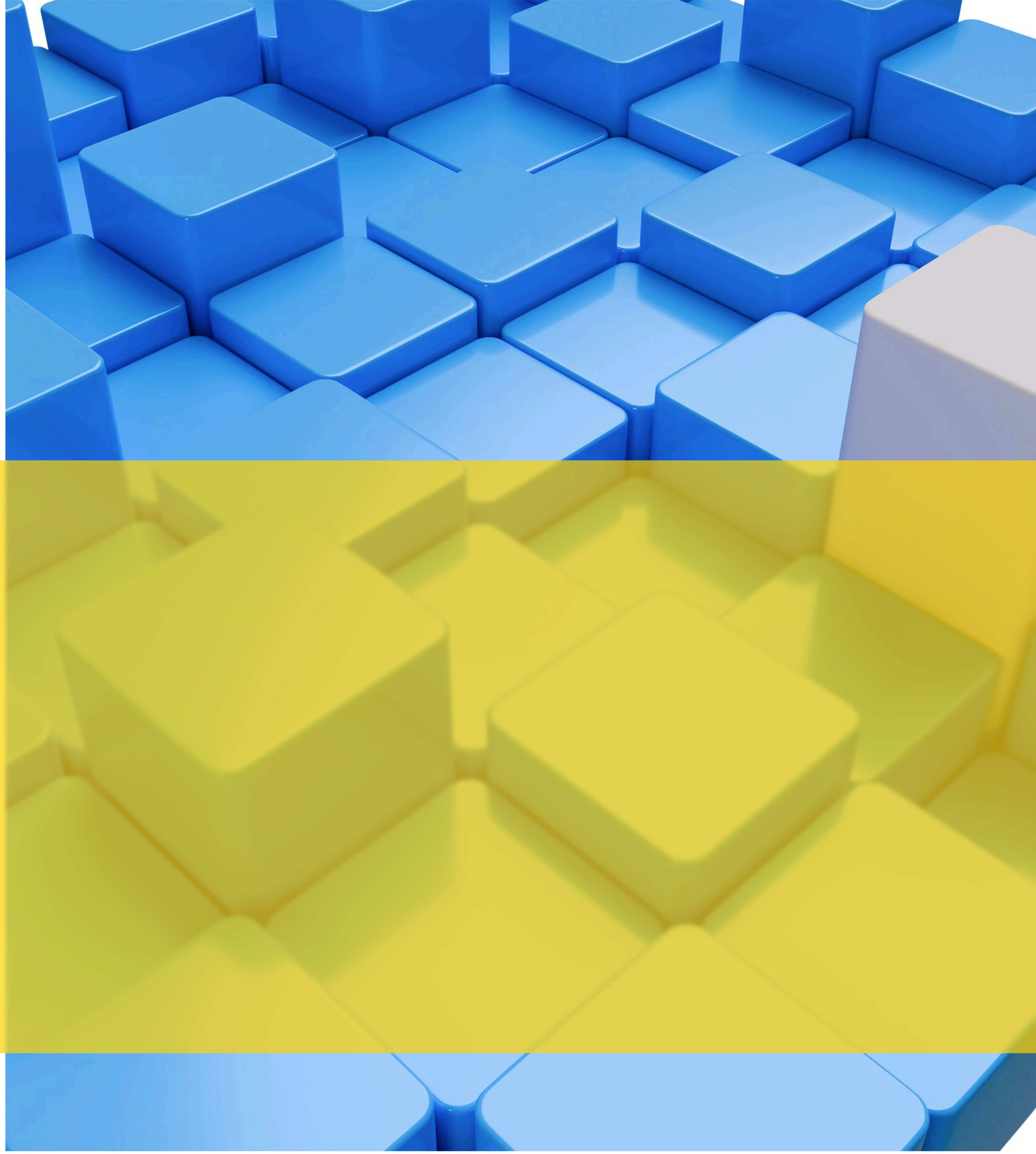
- Rely on legacy core banking or payment processing platforms (such as Temenos, TSYS, FIS, WorldPay) for essential business functions including transaction processing, settlements, and regulatory compliance.
- Struggle with manual, error-prone data reconciliation across high transaction volumes, resulting in delays, operational risk, and compliance challenges.
- Require flexible, scalable integration with modern digital channels (mobile apps, open banking, fintech partners) to remain competitive.
- Operate in highly regulated environments and need transparent, auditable workflows to meet evolving compliance mandates.
- Seek to launch new products or business models quickly, but face barriers due to fragmented data, technical debt, or inflexible system architectures.
- Aim to improve operational efficiency and customer experience without the cost and disruption of full-scale core migration.

By combining robust integration layers, real-time event processing, and intelligent automation, our solutions enable organizations to unlock value from legacy systems, accelerate digital innovation, and deliver faster, more reliable financial services.

ANDERSEN

Bridging The Gap: Unlocking Value From Legacy Banking Platforms Through Integration And Automation

White Paper



Core Problem Statement

Financial institutions and digital marketplaces face significant obstacles when modernizing operations built on legacy platforms or rigid payment systems. Core challenges stem from outdated infrastructure, **slow, batch-based processing,** and **limited real-time capabilities.** Operational teams struggle to deliver the speed, transparency, and reliability demanded by today's customers and partners.

Common pain points include:

- Manual and error-prone reconciliation across high transaction volumes, leading to delays in settlements, compliance, and customer support.
- Fragmented data and limited API capabilities, making integration and automation slow, costly, and inflexible.
- Inability to launch new digital products, payment flows, or reporting features quickly due to technical debt and rigid system architecture.

Typical Indicators That Trigger Change

Companies seeking transformation often report:

- Persistent delays in transaction processing and merchant settlements.
- High operational overhead for data reconciliation and reporting.
- Siloed teams duplicating data integration and exception handling efforts.
- Difficulty scaling to support new products, business models, or partner onboarding.
- Increased financial and regulatory risk due to fragmented, non-orchestrated workflows.

Profile of Beneficiary Companies

At Andersen, we empower financial institutions, digital marketplaces, and payment service providers to modernize critical operations while preserving the stability of proven legacy systems.
Our integration and automation solutions help organizations bridge the gap between outdated infrastructure and modern customer expectations, delivering operational excellence without the risk and disruption of full system replacement.

We focus on organizations that:

- Rely on legacy core banking or payment processing platforms (such as Temenos, TSYS, FIS, WorldPay) for essential business functions including transaction processing, settlements, and regulatory compliance.
- Struggle with manual, error-prone data reconciliation across high transaction volumes, resulting in delays, operational risk, and compliance challenges.
- Require flexible, scalable integration with modern digital channels (mobile apps, open banking, fintech partners) to remain competitive.
- Operate in highly regulated environments and need transparent, auditable workflows to meet evolving compliance mandates.
- Seek to launch new products or business models quickly, but face barriers due to fragmented data, technical debt, or inflexible system architectures.
- Aim to improve operational efficiency and customer experience without the cost and disruption of full-scale core migration.

By combining robust integration layers, real-time event processing, and intelligent automation, our solutions enable organizations to unlock value from legacy systems, accelerate digital innovation, and deliver faster, more reliable financial services.

High-Level Solution Approach

While many organizations aim to modernize their technology stacks, the complete replacement of existing core banking and payment gateway systems — such as legacy versions of Temenos, TSYS, FIS, or Worldpay- can be quite complex, risky, and costly.

For most institutions, these legacy platforms have been deeply embedded into operational workflows over decades. They support critical functions such as customer onboarding, account management, transaction processing, payment clearing, and regulatory reporting. Replacing these systems would often involve high costs of licenses, massive data migration efforts, long project timelines and business disruption.

A more pragmatic and value-driven approach is to extend and modernize existing legacy systems through an integration layer, enabling the banks and other companies to unlock new capabilities without a full core migration. By building modular integration layers (through microservices that expose clear APIs and event streams), banks can expose legacy functionality to modern channels (e.g. mobile apps, open banking interfaces, fintech partners) while maintaining the stability and reliability of proven core systems.

Core Problem Statement

Financial institutions and digital marketplaces face significant obstacles when modernizing operations built on legacy platforms or rigid payment systems. Core challenges stem from outdated infrastructure, **slow, batch-based processing,** and **limited real-time capabilities.** Operational teams struggle to deliver the speed, transparency, and reliability demanded by today's customers and partners.

Common pain points include:

- Manual and error-prone reconciliation across high transaction volumes, leading to delays in settlements, compliance, and customer support.
- Fragmented data and limited API capabilities, making integration and automation slow, costly, and inflexible.
- Inability to launch new digital products, payment flows, or reporting features quickly due to technical debt and rigid system architecture.

Typical Indicators That Trigger Change

Companies seeking transformation often report:

- Persistent delays in transaction processing and merchant settlements.
- High operational overhead for data reconciliation and reporting.
- Siloed teams duplicating data integration and exception handling efforts.
- Difficulty scaling to support new products, business models, or partner onboarding.
- Increased financial and regulatory risk due to fragmented, non-orchestrated workflows.

Typical Solution Description

Channels Layer

- Mobile and web apps for customers, CRM and backoffice apps for operators
- Integrates with API Gateway

Integration Layer

- Translates between modern APIs and legacy interfaces
- Exposes a stable internal interface, while adapting to:
  - **SOAP services (e.g., Temenos)**
  - **FIS proprietary APIs**

API Gateway Layer

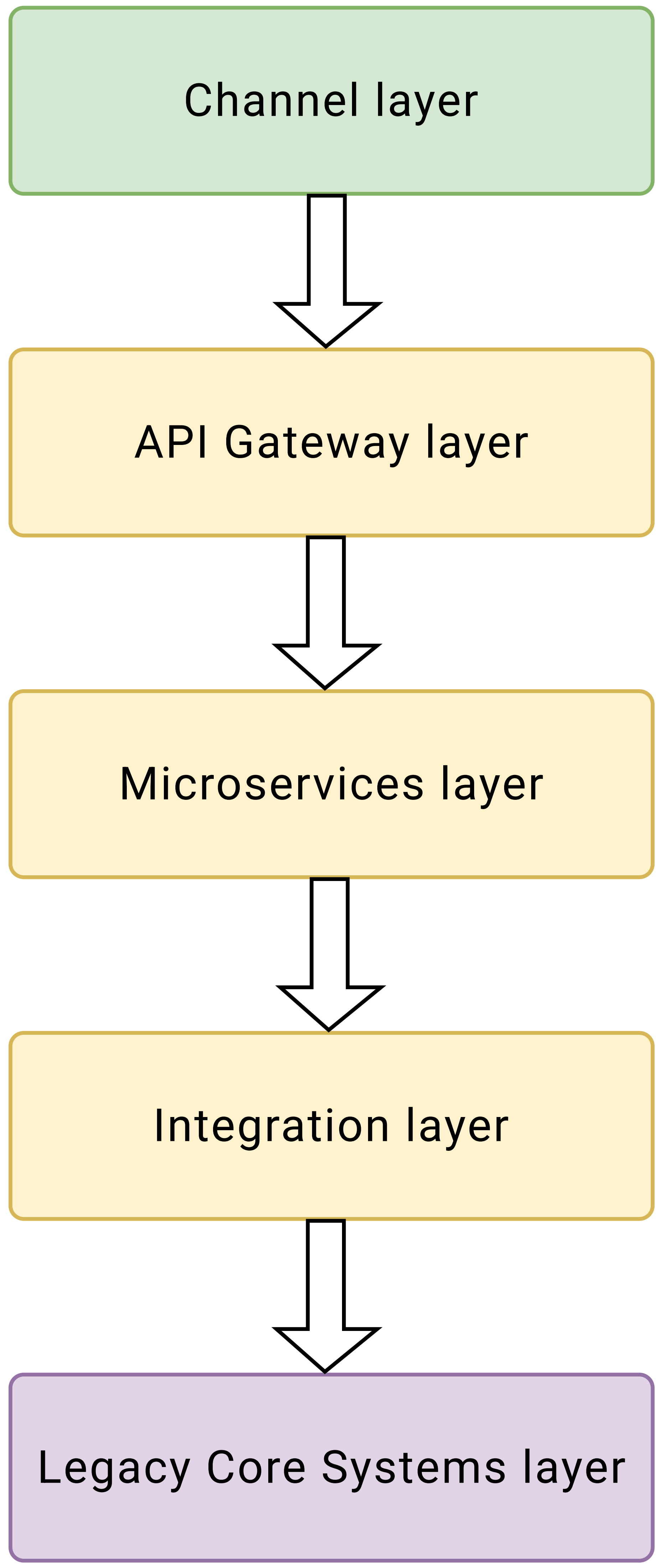
- Central access point for all external/internal consumers
- Handles:Routing, authentication / authorization, rate control

Legacy Core Systems

- FIS/TSYS/Temenos or other core banking and payment systems
- Maintain customer data, accounts, loans, and/or cards, payments, retail banking
- Integrated via REST/SOAP APIs or MQ/SFTP/flat files (in legacy setups)

Microservices Layer

- Domain-driven services, independently deployable
- Each service exposes REST/gRPC API for channels and partners
- Services can communicate via event bus when needed



High-Level Solution Approach

While many organizations aim to modernize their technology stacks, the complete replacement of existing core banking and payment gateway systems — such as legacy versions of Temenos, TSYS, FIS, or Worldpay- can be quite complex, risky, and costly.

For most institutions, these legacy platforms have been deeply embedded into operational workflows over decades. They support critical functions such as customer onboarding, account management, transaction processing, payment clearing, and regulatory reporting. Replacing these systems would often involve high costs of licenses, massive data migration efforts, long project timelines and business disruption.

A more pragmatic and value-driven approach is to extend and modernize existing legacy systems through an integration layer, enabling the banks and other companies to unlock new capabilities without a full core migration. By building modular integration layers (through microservices that expose clear APIs and event streams), banks can expose legacy functionality to modern channels (e.g. mobile apps, open banking interfaces, fintech partners) while maintaining the stability and reliability of proven core systems.

Applied Patterns and Practices

Legacy core banking systems were often designed as monolithic, tightly coupled platforms with limited extensibility and no native support for modern integration patterns. To enable agility, scalability, and innovation without disrupting critical operations, various architectural approaches can be applied to bridge legacy systems with modern technology stacks.

1 API Layer

Expose legacy functionality through a well-defined API layer that abstracts the complexity of underlying systems. RESTful or GraphQL APIs with clear specification should replace downstream SOAP, XML based, file based or other integration methods.

- Promotes reuse and modularity
- Enables partner and mobile integrations

2 API Gateway

All new APIs are exposed via API gateway, allowing external applications to interact with core systems in a secure and controlled manner.

- Supports centralized policy enforcement via the API Gateway — including authentication, authorization, throttling, rate limiting, logging, request transformation, and monitoring

3 Real time events

Implement an event streaming or messaging layer (e.g. Kafka) that captures and propagates changes from legacy systems to downstream services in real time. Either adapters to legacy events systems (IBM MQ) or polling solutions are designed for this.

- Decouples systems for greater scalability
- Enables real-time notifications and analytics

4 Data Virtualization and Caching

Create a real-time or near-real-time data layer using caching (e.g. Redis) and cache invalidation scenario based on events to reduce direct load on legacy systems while returning actual data without delay to end customer.

- Reduces response time for high load read operations
- Minimizes strain on legacy systems

Typical Solution Description

Channels Layer

- Mobile and web apps for customers, CRM and backoffice apps for operators
- Integrates with API Gateway

Integration Layer

- Translates between modern APIs and legacy interfaces
- Exposes a stable internal interface, while adapting to:
  - **SOAP services (e.g., Temenos)**
  - **FIS proprietary APIs**

API Gateway Layer

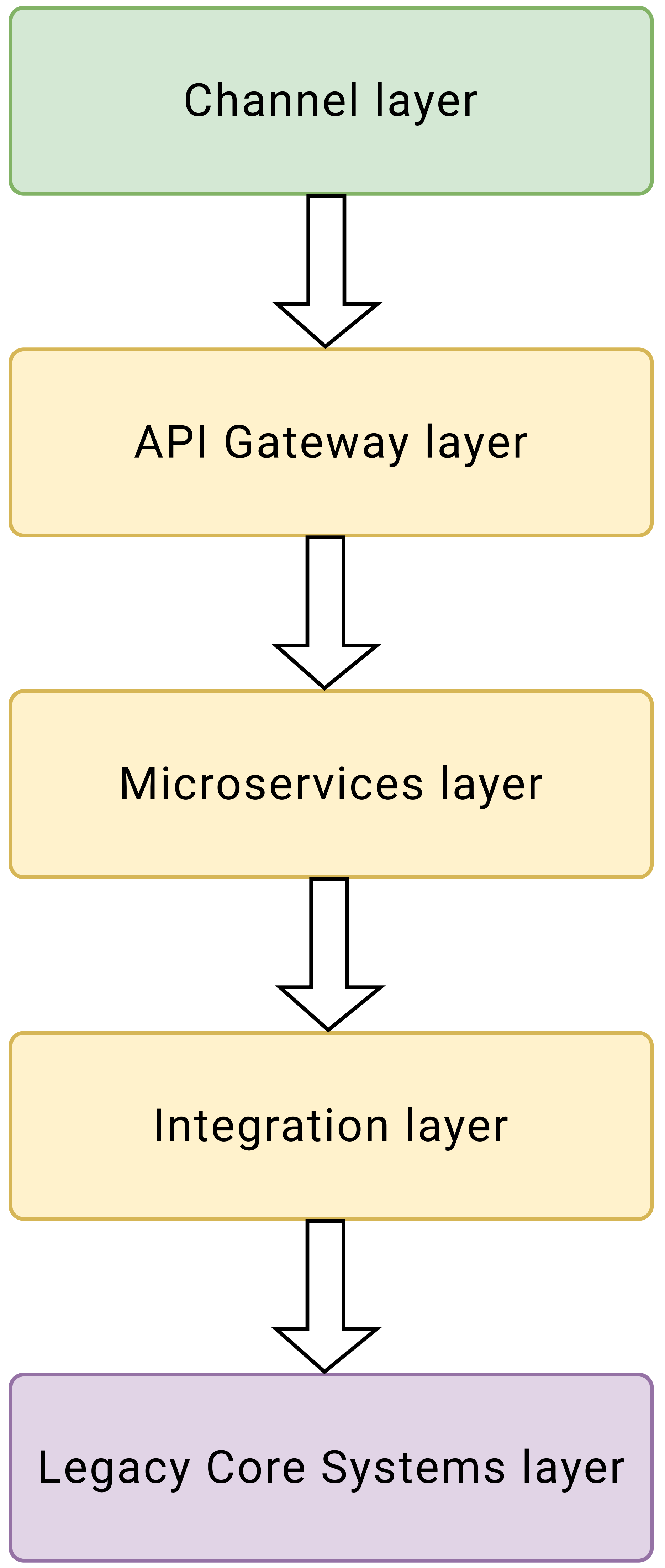
- Central access point for all external/internal consumers
- Handles:Routing, authentication / authorization, rate control

Legacy Core Systems

- FIS/TSYS/Temenos or other core banking and payment systems
- Maintain customer data, accounts, loans, and/or cards, payments, retail banking
- Integrated via REST/SOAP APIs or MQ/SFTP/flat files (in legacy setups)

Microservices Layer

- Domain-driven services, independently deployable
- Each service exposes REST/gRPC API for channels and partners
- Services can communicate via event bus when needed



Applied Patterns and Practices

5 PCI DSS and Regulatory Compliance Architecture

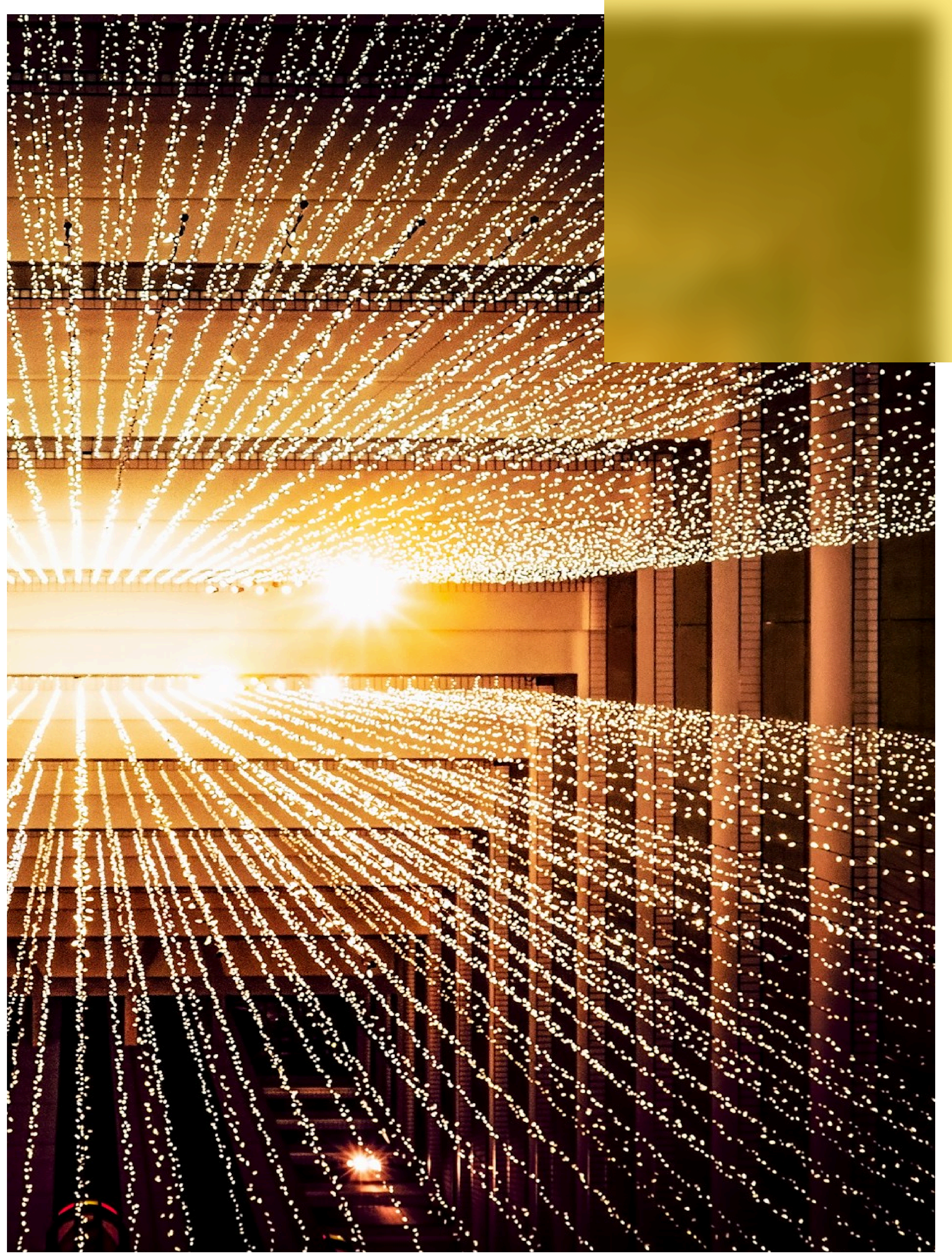
Expose secure APIs on top of legacy systems to meet regulatory mandates such as PSD2 in the EU, allowing third-party providers (TPPs) to access account information and initiate payments. Process only tokenized / encrypted card data to limit PCI DSS impact.

- Reduces regulatory requirements on microservices layer

6 Data Analytics and Reporting

Many legacy systems are built for transactional processing, not real-time analytics or business intelligence. That is why it is not always possible to provide analytics and dashboards required by business.

1. Stream real-time and batch data from legacy systems into modern data lakes or analytics platforms for better risk management, compliance, and customer insights.
2. Use modern BI tools to provide better customer experience to business users.



Applied Patterns and Practices

Legacy core banking systems were often designed as monolithic, tightly coupled platforms with limited extensibility and no native support for modern integration patterns. To enable agility, scalability, and innovation without disrupting critical operations, various architectural approaches can be applied to bridge legacy systems with modern technology stacks.

1 API Layer

Expose legacy functionality through a well-defined API layer that abstracts the complexity of underlying systems. RESTful or GraphQL APIs with clear specification should replace downstream SOAP, XML based, file based or other integration methods.

- Promotes reuse and modularity
- Enables partner and mobile integrations

2 API Gateway

All new APIs are exposed via API gateway, allowing external applications to interact with core systems in a secure and controlled manner.

- Supports centralized policy enforcement via the API Gateway — including authentication, authorization, throttling, rate limiting, logging, request transformation, and monitoring

3 Real time events

Implement an event streaming or messaging layer (e.g. Kafka) that captures and propagates changes from legacy systems to downstream services in real time. Either adapters to legacy events systems (IBM MQ) or polling solutions are designed for this.

- Decouples systems for greater scalability
- Enables real-time notifications and analytics

4 Data Virtualization and Caching

Create a real-time or near-real-time data layer using caching (e.g. Redis) and cache invalidation scenario based on events to reduce direct load on legacy systems while returning actual data without delay to end customer.

- Reduces response time for high load read operations
- Minimizes strain on legacy systems

Customer Success Story

Legacy Transformation for a Retail Bank

Duration
12 months

Location
EU

Technologies
Java / Apache Kafka / PostgreSQL / IBM MQ / React / ReactNative

Challenge

A large retail bank operating across North America and Western Europe faced mounting integration challenges as it sought to modernize its digital channels while relying on legacy mainframe platforms. The bank encountered:

- **Slow, batch-based transaction processing** leading to delays in customer service and regulatory reporting.
- **High operational costs** due to manual reconciliation and frequent data errors caused by outdated file formats and limited automation.
- **Difficulty achieving compliance** with evolving data security (e.g., PCI-DSS) and privacy standards, as legacy systems lacked built-in data masking or tokenization features.
- **Inflexibility in launching new digital products and channels** due to a lack of modern APIs and real-time data access.
- **Rising technical debt** and a shrinking pool of mainframe specialists, increasing the risk and cost of ongoing system support.

These issues led to lost business opportunities, frustrated customers and staff, and increased regulatory and reputational risks.

Applied Patterns and Practices

5 PCI DSS and Regulatory Compliance Architecture

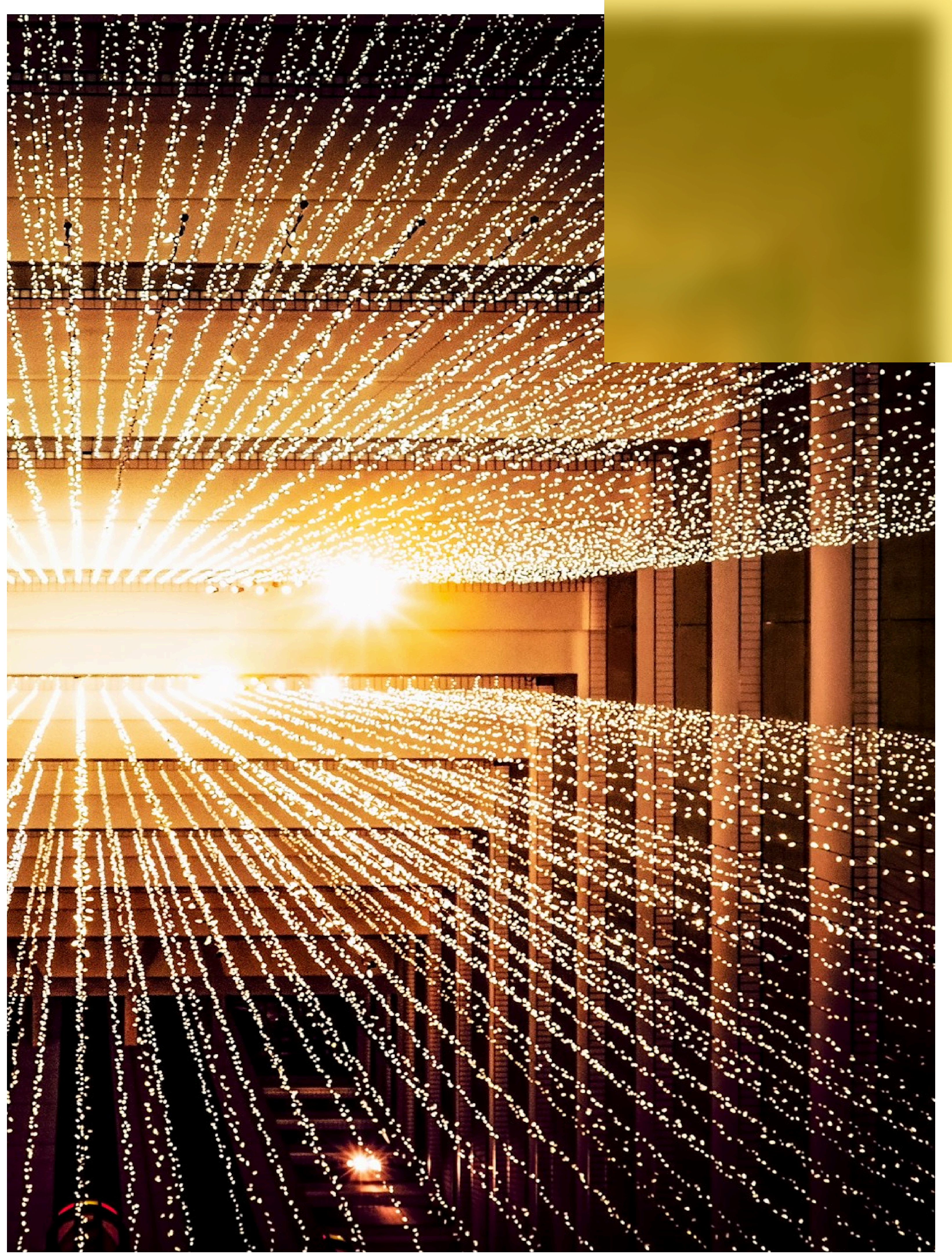
Expose secure APIs on top of legacy systems to meet regulatory mandates such as PSD2 in the EU, allowing third-party providers (TPPs) to access account information and initiate payments. Process only tokenized / encrypted card data to limit PCI DSS impact.

- Reduces regulatory requirements on microservices layer

6 Data Analytics and Reporting

Many legacy systems are built for transactional processing, not real-time analytics or business intelligence. That is why it is not always possible to provide analytics and dashboards required by business.

1. Stream real-time and batch data from legacy systems into modern data lakes or analytics platforms for better risk management, compliance, and customer insights.
2. Use modern BI tools to provide better customer experience to business users.



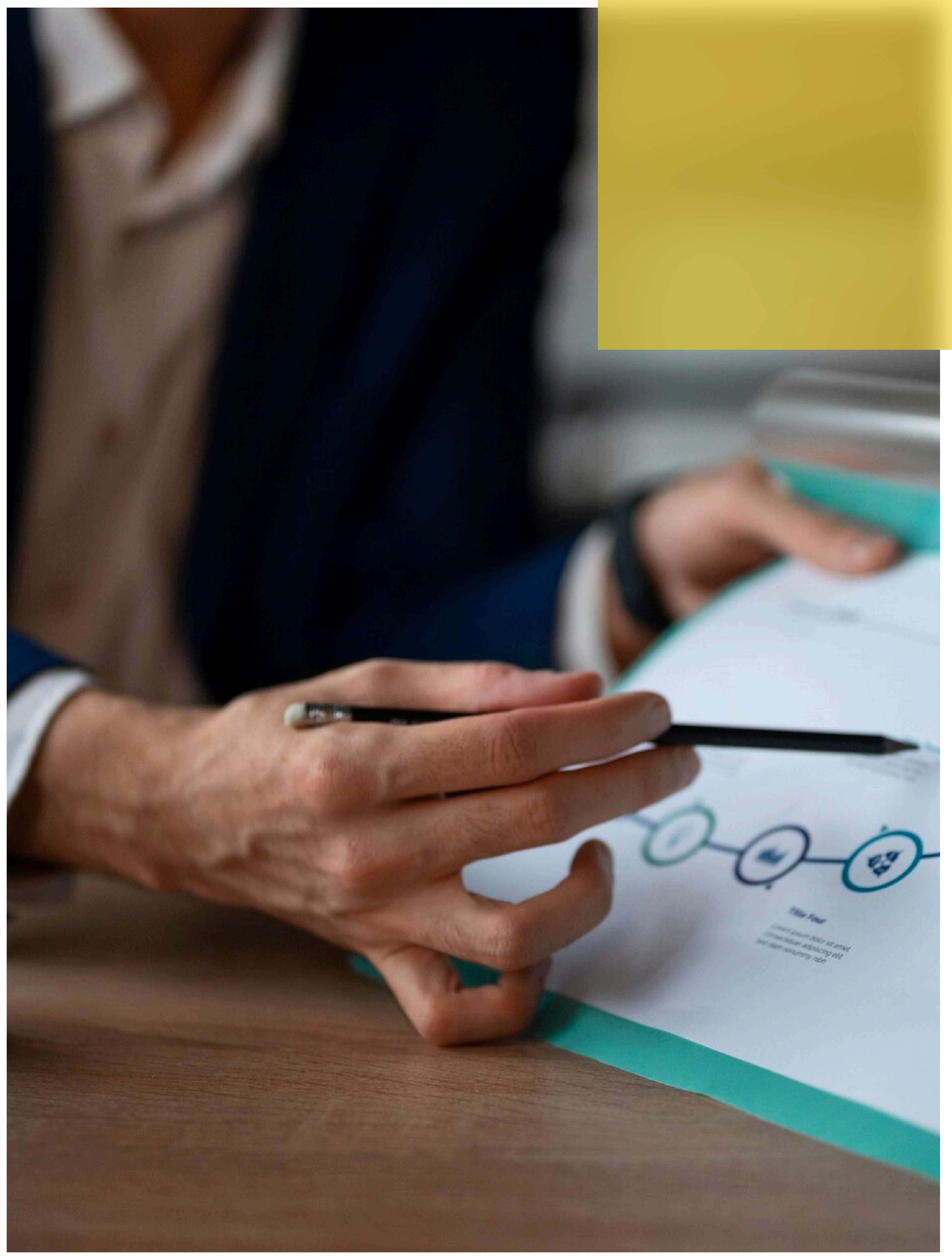
Legacy Transformation for a Retail Bank

Business Value

To address these problems, our integration team delivered a comprehensive modernization solution that:

- **Introduced a caching and middleware integration layer.** → Reduced response time for online and mobile banking, enabling near real-time customer experiences.
- **Automated pre-processing and error correction of legacy data files.** → Decreased manual reconciliation effort and reduced transaction fallout incidents.
- **Implemented robust data masking and tokenization at the integration layer.** → Enabled compliance with PCI-DSS and GDPR requirements, supporting the launch of new card products without costly system-wide certification.
- **Provided modern RESTful APIs on top of mainframe systems.** → Accelerated time-to-market for new digital banking services, allowing business teams to deploy new features up to several times faster.
- **Enabled intelligent orchestration and disaster recovery for core processes.** → Improved system reliability and ensured business continuity during outages or system upgrades.

As a result, the bank significantly enhanced customer satisfaction, reduced operational risks and costs, and created a future-proof foundation for digital innovation.



Customer Success Story

Legacy Transformation for a Retail Bank

Duration
12 months

Location
EU

Technologies
Java / Apache Kafka / PostgreSQL / IBM MQ / React / ReactNative

Challenge

A large retail bank operating across North America and Western Europe faced mounting integration challenges as it sought to modernize its digital channels while relying on legacy mainframe platforms. The bank encountered:

- **Slow, batch-based transaction processing** leading to delays in customer service and regulatory reporting.
- **High operational costs** due to manual reconciliation and frequent data errors caused by outdated file formats and limited automation.
- **Difficulty achieving compliance** with evolving data security (e.g., PCI-DSS) and privacy standards, as legacy systems lacked built-in data masking or tokenization features.
- **Inflexibility in launching new digital products and channels** due to a lack of modern APIs and real-time data access.
- **Rising technical debt** and a shrinking pool of mainframe specialists, increasing the risk and cost of ongoing system support.

These issues led to lost business opportunities, frustrated customers and staff, and increased regulatory and reputational risks.

Legacy Transformation for a Retail Bank

Solution Overview

Bank uses legacy Temenos version R13 or below for Core banking and accounts management. It integrates with TSYS for bank cards management.

Anderson developed an intermediate HTTP Rest API layer that exposes set of APIs to external systems and for customer facing front ends incl. mobile and web applications.

Technical Challenges and Solutions

Bank uses legacy Temenos version R13 or below for Core banking and accounts management. It integrates with TSYS for bank cards management.

Andersen developed an intermediate HTTP Rest API layer that exposes set of APIs to external systems and for customer facing front ends incl. mobile and web applications.

1. **Legacy banking applications like Temenos do not expose HTTP REST APIs and events are available only for IBM MQ that is hard to integrate for external system.**

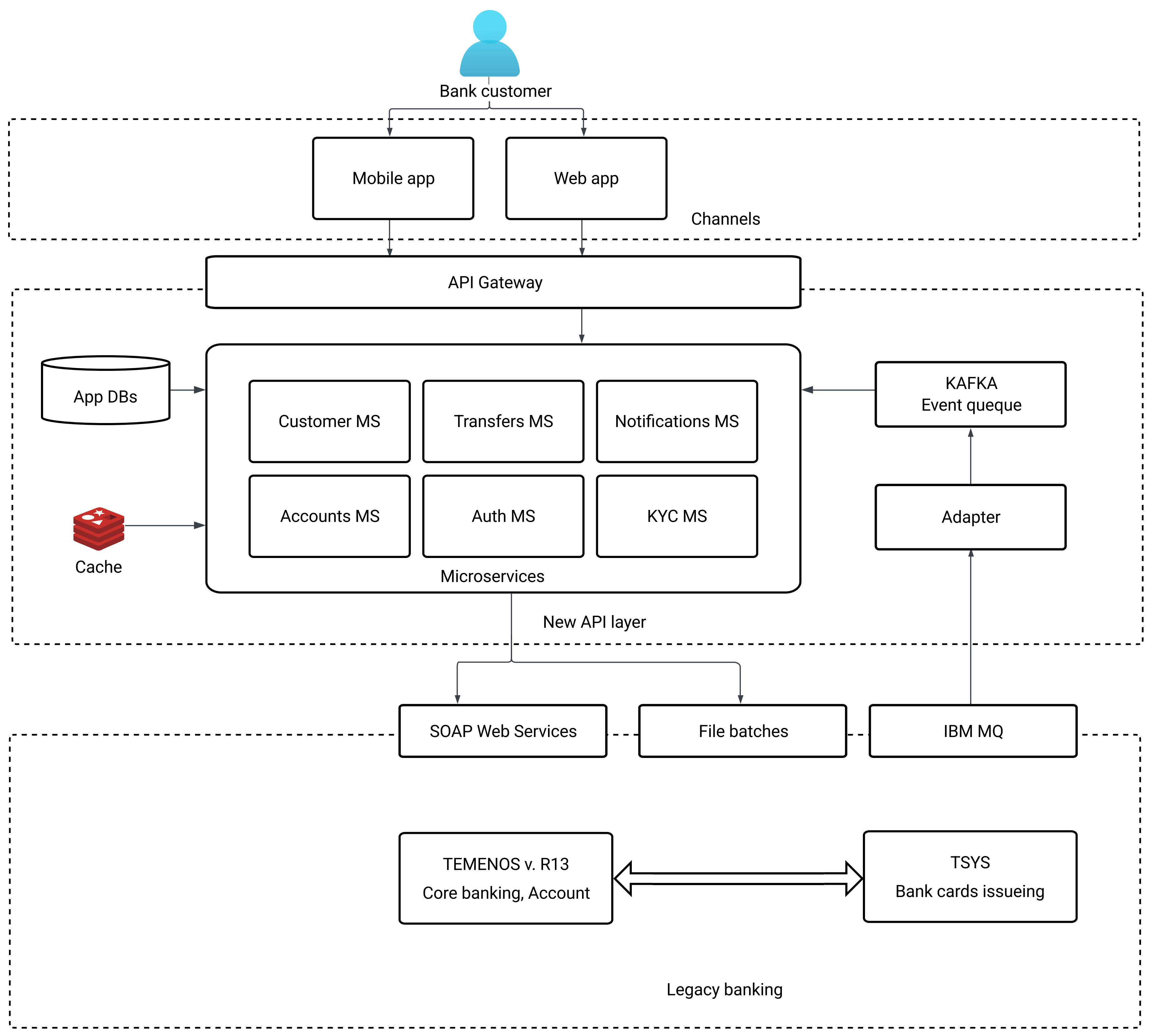
   Some operations are available via Soap API that lacks documentation, some via file batches that require specific format.

   Solution
   Andersen built an intermediate API layer that exposes HTTP REST APIs to outside systems that is well documented, effectively structured and easy to integrate with.

2. **Lack of real time integration**

   Existing SOAP API and File batches do not allow for real time integration.

   Solution
   With new adapter, system exposes events via Kafka that can be consumed by other systems and front ends, that allow customers to immediately see the result of their operations.



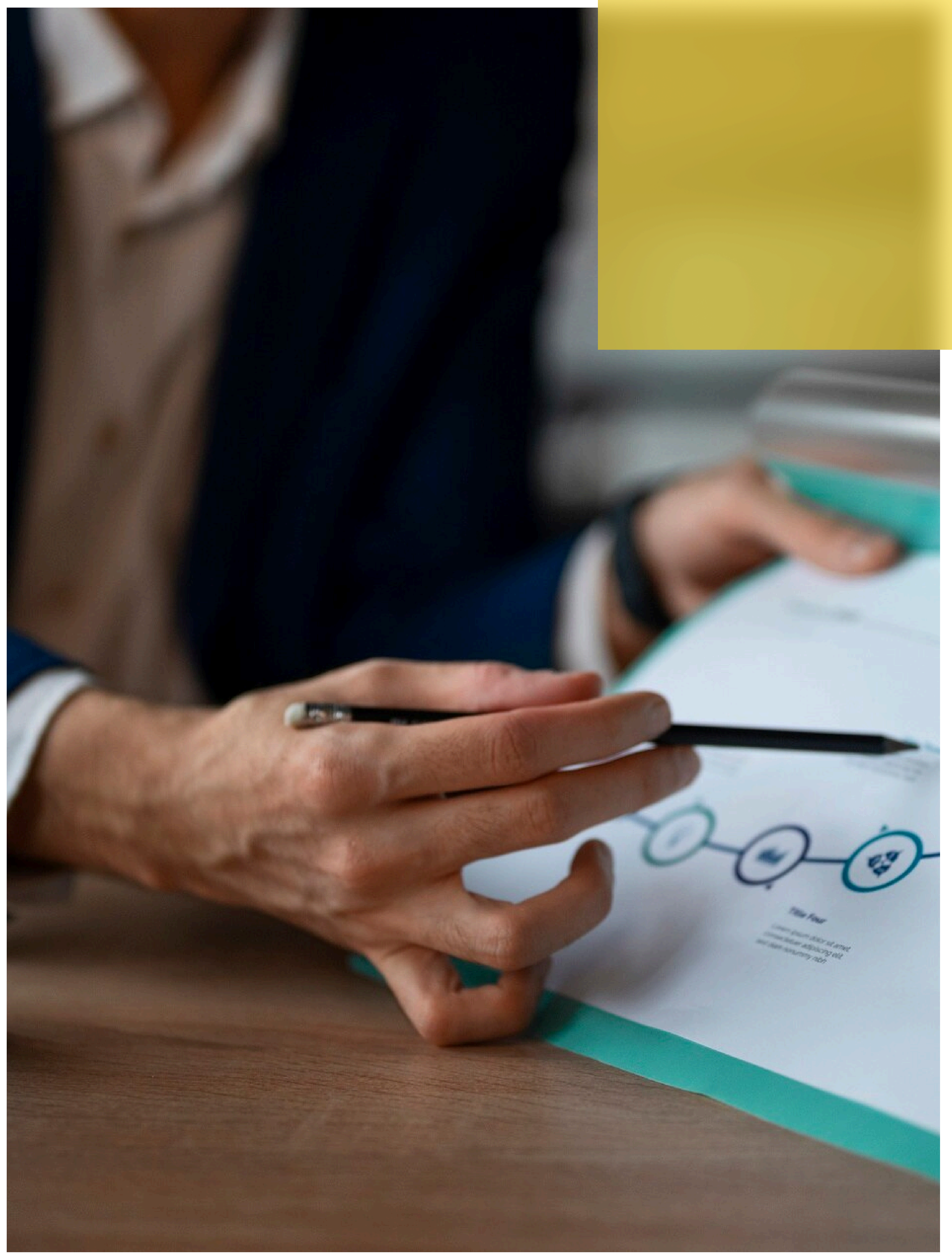
Legacy Transformation for a Retail Bank

Business Value

To address these problems, our integration team delivered a comprehensive modernization solution that:

- **Introduced a caching and middleware integration layer.** → Reduced response time for online and mobile banking, enabling near real-time customer experiences.
- **Automated pre-processing and error correction of legacy data files.** → Decreased manual reconciliation effort and reduced transaction fallout incidents.
- **Implemented robust data masking and tokenization at the integration layer.** → Enabled compliance with PCI-DSS and GDPR requirements, supporting the launch of new card products without costly system-wide certification.
- **Provided modern RESTful APIs on top of mainframe systems.** → Accelerated time-to-market for new digital banking services, allowing business teams to deploy new features up to several times faster.
- **Enabled intelligent orchestration and disaster recovery for core processes.** → Improved system reliability and ensured business continuity during outages or system upgrades.

As a result, the bank significantly enhanced customer satisfaction, reduced operational risks and costs, and created a future-proof foundation for digital innovation.



Legacy Transformation for a Retail Bank

Technical Challenges and Solutions

3. **Performance**

   Temenos and similar legacy systems have limited API throughput and large response times. Modern front end systems with customers checking their balance regularly experience slow response times because of it.

   Solution
   New set of microservices has advanced caching system based on Redis that significantly improves API performance. Microservices are easy to scale horizontally and keep their performance optimal.

4. **Error handling**

   Failures in Temenos and other downstream systems completely block the flow with no options for retries.

   Solution
   On microservices level it is possible to store commands if southbound systems are down and send them for execution once everything is up again.

5. **Complicated transaction handlings**

   Advanced transactions, e.g. charges with tips from Temenos look like multiple independent transactions that complicate the output and decrease customer experience.

   Solution
   On microservices layer it is possible to organize transactions incl merging a few entries into 1. This way on the UI customer can see correct and clean transaction list.

6. **Real time balance**

   Temenos cannot return real time balance that should decrease immediately after transaction. Additionally due transaction exchange rates / settlement exchange rate the balance if often incorrect.

   Solution
   On microservices layer it is possible to get transaction events via IBM MQ / Kafka and maintain proper current charges until settlement file is processed. This allows end customer to see charged amount near real time.

7. **PCI DSS**

   Adding extra components on top of Temenos / TSYS require PCI DSS compliance for those. This is usually complicated and time consuming.

   Solution
   Due to client-side tokenization of card data it is possible to avoid PCI DSS requirements for microservice layer, saving significant effort.

8. **Limited extendibility**

   Temenos has limited customization capabilities and extending existing models is complicated.

   Solution
   Additional model data is stored in DB on new API layer, that allows to easily customize existing data models and use them in microservices.

Legacy Transformation for a Retail Bank

Solution Overview

Bank uses legacy Temenos version R13 or below for Core banking and accounts management. It integrates with TSYS for bank cards management.

Anderson developed an intermediate HTTP Rest API layer that exposes set of APIs to external systems and for customer facing front ends incl. mobile and web applications.

Technical Challenges and Solutions

Bank uses legacy Temenos version R13 or below for Core banking and accounts management. It integrates with TSYS for bank cards management.

Andersen developed an intermediate HTTP Rest API layer that exposes set of APIs to external systems and for customer facing front ends incl. mobile and web applications.

1. **Legacy banking applications like Temenos do not expose HTTP REST APIs and events are available only for IBM MQ that is hard to integrate for external system.**

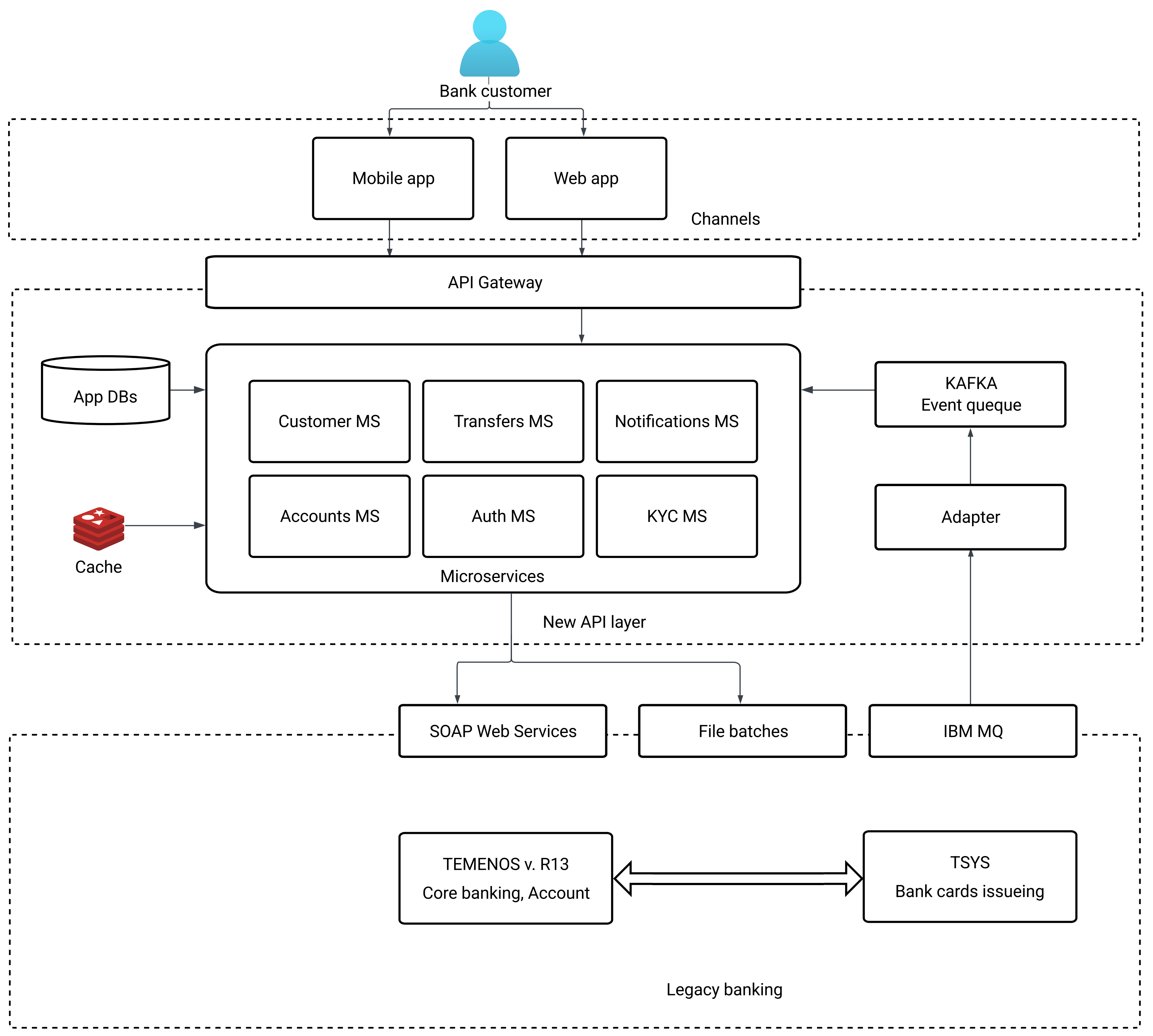
   Some operations are available via Soap API that lacks documentation, some via file batches that require specific format.

   Solution
   Andersen built an intermediate API layer that exposes HTTP REST APIs to outside systems that is well documented, effectively structured and easy to integrate with.

2. **Lack of real time integration**

   Existing SOAP API and File batches do not allow for real time integration.

   Solution
   With new adapter, system exposes events via Kafka that can be consumed by other systems and front ends, that allow customers to immediately see the result of their operations.



Customer Success Story

Optimizing Transaction Reconciliation in a Marketplace Platform

Duration
15 months

Location
USA

Technologies
Java / Apache Kafka / PostgreSQL / Redis / Docker / AWS / PowerBI / Angular / Flutter

Challenge

A rapidly growing marketplace leveraging WorldPay as its acquiring platform faced significant operational and technical challenges as it scaled its merchant payout and transaction reconciliation processes:

- **Limited API capabilities and delayed event processing** in WorldPay's standard integration made it difficult to support real-time payouts, monitor transaction status, or trigger timely notifications for merchants.
- **Manual and error-prone reconciliation** of thousands of daily transactions, including chargebacks and refunds, resulted in increased operational overhead, settlement delays, and financial discrepancies.
- **Inefficient return and refund handling** led to poor merchant and customer experiences, with complex exception flows and limited visibility for support teams.
- **Data consistency issues** due to batch-based processing and disparate data sources created frequent mismatches between transaction records, merchant statements, and actual payouts.
- **Limited flexibility for launching new payment flows or analytics features** due to the rigid architecture and lack of event-driven integration with WorldPay.
- **Difficulty scaling to support new merchants or payment models** without introducing risk to transaction accuracy, data integrity, or system performance.

These obstacles resulted in delayed merchant settlements, increased financial risk, and barriers to rolling out differentiated marketplace services.

Legacy Transformation for a Retail Bank

Technical Challenges and Solutions

3. **Performance**

   Temenos and similar legacy systems have limited API throughput and large response times. Modern front end systems with customers checking their balance regularly experience slow response times because of it.

   Solution
   New set of microservices has advanced caching system based on Redis that significantly improves API performance. Microservices are easy to scale horizontally and keep their performance optimal.

4. **Error handling**

   Failures in Temenos and other downstream systems completely block the flow with no options for retries.

   Solution
   On microservices level it is possible to store commands if southbound systems are down and send them for execution once everything is up again.

5. **Complicated transaction handlings**

   Advanced transactions, e.g. charges with tips from Temenos look like multiple independent transactions that complicate the output and decrease customer experience.

   Solution
   On microservices layer it is possible to organize transactions incl merging a few entries into 1. This way on the UI customer can see correct and clean transaction list.

6. **Real time balance**

   Temenos cannot return real time balance that should decrease immediately after transaction. Additionally due transaction exchange rates / settlement exchange rate the balance if often incorrect.

   Solution
   On microservices layer it is possible to get transaction events via IBM MQ / Kafka and maintain proper current charges until settlement file is processed. This allows end customer to see charged amount near real time.

7. **PCI DSS**

   Adding extra components on top of Temenos / TSYS require PCI DSS compliance for those. This is usually complicated and time consuming.

   Solution
   Due to client-side tokenization of card data it is possible to avoid PCI DSS requirements for microservice layer, saving significant effort.

8. **Limited extendibility**

   Temenos has limited customization capabilities and extending existing models is complicated.

   Solution
   Additional model data is stored in DB on new API layer, that allows to easily customize existing data models and use them in microservices.

Optimizing Transaction Reconciliation in a Marketplace Platform

Business Value

Our solution delivered a modernized, event-driven integration layer on top of WorldPay, providing substantial business value to the marketplace:

- **Near real-time event processing and orchestration engine.** → Enabled instant payout notifications and accelerated settlement cycles, improving merchant cash flow and satisfaction.
- **Automated reconciliation and error resolution.** → Reduced manual effort and reconciliation error, ensuring accurate, timely payouts and dramatically lowering financial risk.
- **Advanced return and refund management.** → Streamlined processing of returns and chargebacks, delivering transparent, user-friendly resolution workflows for merchants and end-customers.
- **Unified API layer and flexible data pipeline.** → Simplified the launch of new payment models, dashboards, and analytics features without changes to the core WorldPay platform.
- **Comprehensive data caching and real-time reporting.** → Provided actionable insights and up-to-date payout status for both operations teams and merchants, reducing support requests and improving transparency.
- **Resilient, scalable architecture.** → Supported onboarding of additional merchants and payment flows with minimal risk and overhead.

As a result, the marketplace reduced operational costs, improved merchant satisfaction and retention, and gained a competitive advantage by offering faster, more reliable financial services than those achievable through WorldPay alone.

Customer Success Story

Optimizing Transaction Reconciliation in a Marketplace Platform

Duration
15 months

Location
USA

Technologies
Java / Apache Kafka / PostgreSQL / Redis / Docker / AWS / PowerBI / Angular / Flutter

Challenge

A rapidly growing marketplace leveraging WorldPay as its acquiring platform faced significant operational and technical challenges as it scaled its merchant payout and transaction reconciliation processes:

- **Limited API capabilities and delayed event processing** in WorldPay's standard integration made it difficult to support real-time payouts, monitor transaction status, or trigger timely notifications for merchants.
- **Manual and error-prone reconciliation** of thousands of daily transactions, including chargebacks and refunds, resulted in increased operational overhead, settlement delays, and financial discrepancies.
- **Inefficient return and refund handling** led to poor merchant and customer experiences, with complex exception flows and limited visibility for support teams.
- **Data consistency issues** due to batch-based processing and disparate data sources created frequent mismatches between transaction records, merchant statements, and actual payouts.
- **Limited flexibility for launching new payment flows or analytics features** due to the rigid architecture and lack of event-driven integration with WorldPay.
- **Difficulty scaling to support new merchants or payment models** without introducing risk to transaction accuracy, data integrity, or system performance.

These obstacles resulted in delayed merchant settlements, increased financial risk, and barriers to rolling out differentiated marketplace services.

Optimizing Transaction Reconciliation in a Marketplace Platform

Technical Challenges and Solutions

1. **WorldPay API inconsistency**

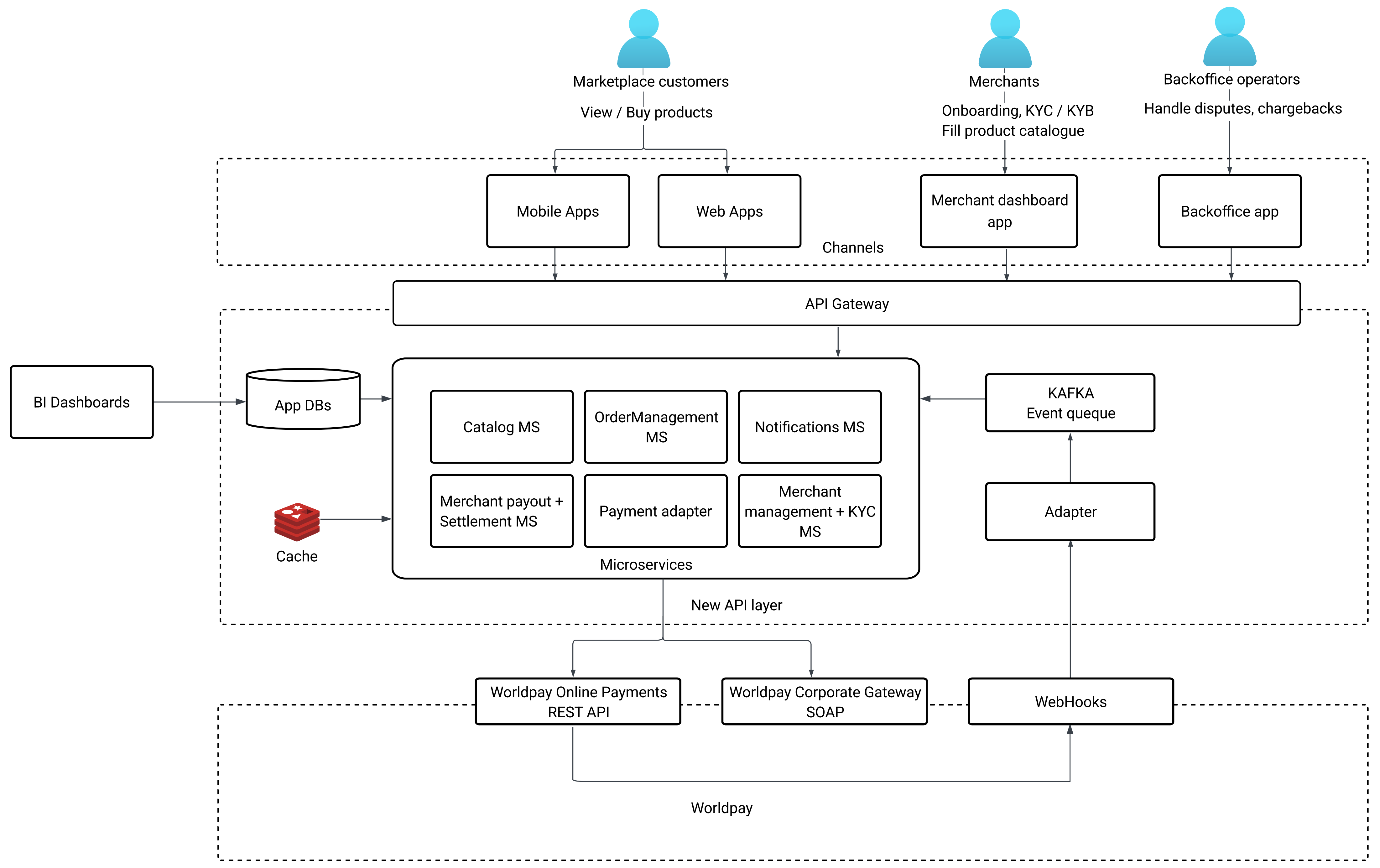
   WorldPay exposes various versions of APIs. Worldpay Online Payments is REST based and covers online payments. Worldpay Corporate Gateway is a SOAP API that handles batch file processing and reconciliations / settlements. All those APIs have various formats, lack clear documentation and are hard to use.

   Solution
   A new set of microservices is developed that exposes uniform set of APIs to channels applications and other 3rd party systems. Microservices transform those requests and trigger downstream WorldPay APIs.

2. **Lack of real time events**

   Worldpay's webhook system can be delayed or out-of-order and requires manual polling for settlement files or reports. Additionally, there is no guaranteed delivery (no retries, no signatures in some versions).

   Solution
   A new adapter is introduced that processes WebHook request and pushes them into Kafka. Kafka events are used by microservices (e.g. notifications MS to notify customers and merchants of the payments success/failures) or by other systems.



Optimizing Transaction Reconciliation in a Marketplace Platform

Business Value

Our solution delivered a modernized, event-driven integration layer on top of WorldPay, providing substantial business value to the marketplace:

- **Near real-time event processing and orchestration engine.** → Enabled instant payout notifications and accelerated settlement cycles, improving merchant cash flow and satisfaction.
- **Automated reconciliation and error resolution.** → Reduced manual effort and reconciliation error, ensuring accurate, timely payouts and dramatically lowering financial risk.
- **Advanced return and refund management.** → Streamlined processing of returns and chargebacks, delivering transparent, user-friendly resolution workflows for merchants and end-customers.
- **Unified API layer and flexible data pipeline.** → Simplified the launch of new payment models, dashboards, and analytics features without changes to the core WorldPay platform.
- **Comprehensive data caching and real-time reporting.** → Provided actionable insights and up-to-date payout status for both operations teams and merchants, reducing support requests and improving transparency.
- **Resilient, scalable architecture.** → Supported onboarding of additional merchants and payment flows with minimal risk and overhead.

As a result, the marketplace reduced operational costs, improved merchant satisfaction and retention, and gained a competitive advantage by offering faster, more reliable financial services than those achievable through WorldPay alone.

Optimizing Transaction Reconciliation in a Marketplace Platform

Technical Challenges and Solutions

3. Manual Settlement Reconciliation

WorldPay requires complicated process for Settlement reconciliation that requires manual file uploads.

Solution

Automated settlement service is implemented that pulls reports via SFTP/API, parses files and matches them to ledger entries. This way it is possible to automate handling for missing payment entries.

4. Chargebacks and Dispute Management

Worldpay requires manual handles of chargebacks in Worldpay Backoffice.

Solution

A separate backoffice application I available for internal operators that can review and automatically process refunds and handle disputes. System will automatically initiate refunds to customers based on dispute resolution result.

5. PCI DSS compliance

Expansions of Worldpay usually require PCI DS certification that is complicated and takes a lot of time and effort. Additionally, it significantly complicates implementation of new features due to need of certification activities for new releases.

Solution

Customer payments can be processed with the use of hosted payment page or client side tokenization. This way client applications and microservices do not handle card data directly and should not comply to PCI DSS.

6. Reporting limitations

WorldPay exposes very limited amount of reports and it is hard to build proper business-oriented reports.

Solution

On new API layer data to be duplicated in the database. That allows BI solution to use this data to build any dashboards required by business team.

7. Manual merchant onboarding

Worldpay OOTB supports only manual process of KYC/KYB to onboard new merchants. That requires a lot of manual work and decreases the customer experience for merchants

Solution

A new service is built to handle KYC/KYB process that uses WorldPay APIs to add new merchants. 3rd party KYC / KYB service that automatically validates uploaded documents is used.

8. Performance / Error handling

Worldpay has limited API performance and response times may be large, especially in case of large load.

Solution

New set of microservices use distributed cache system and advanced cache invalidation scenarios. This way microservices can serve up to data information much later than downstream, systems. Additionally it is possible to handle downstream system errors, e,g, by pushing requests when downstream systems come online.

Optimizing Transaction Reconciliation in a Marketplace Platform

Technical Challenges and Solutions

1. WorldPay API inconsistency

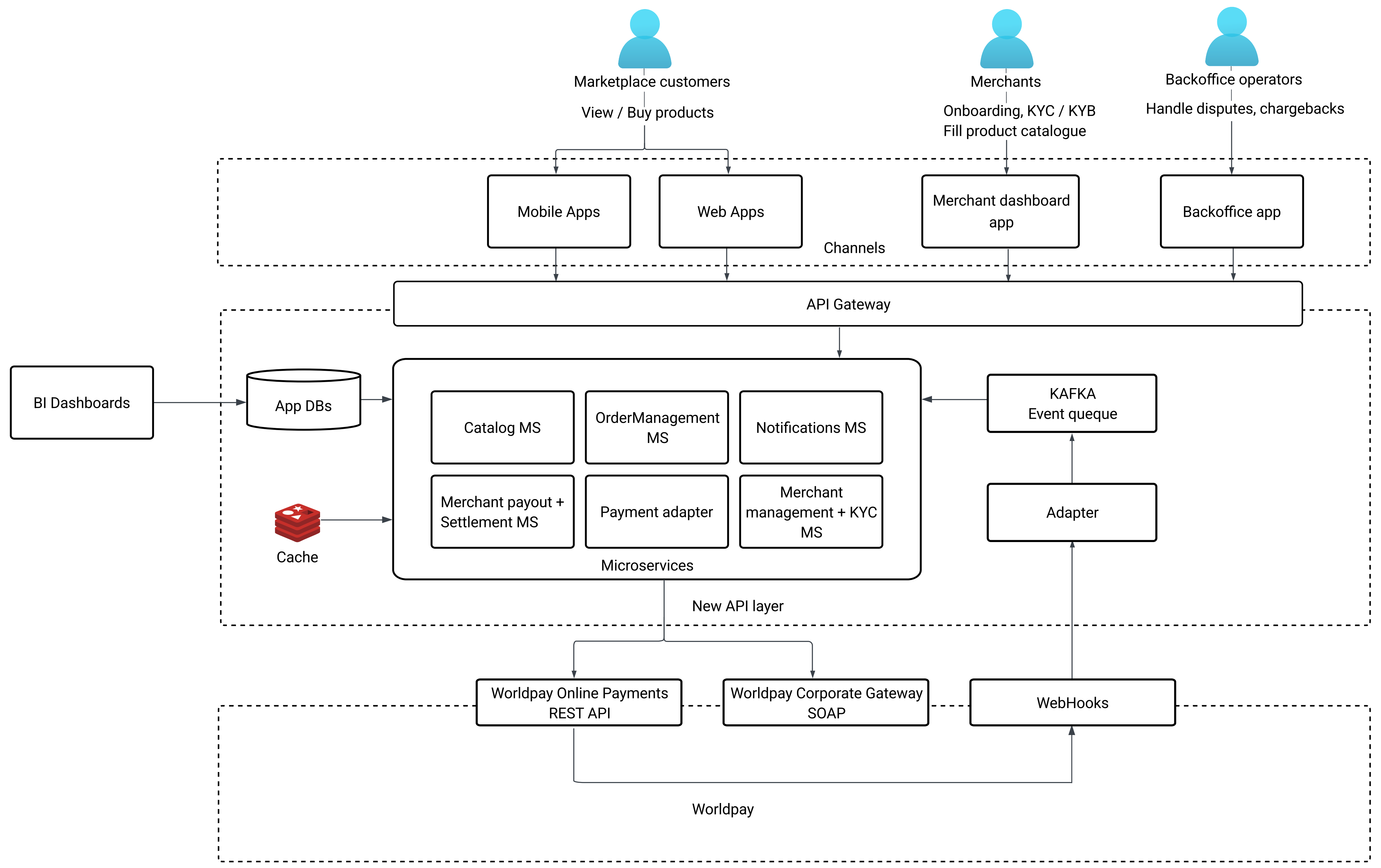
WorldPay exposes various versions of APIs. Worldpay Online Payments is REST based and covers online payments. Worldpay Corporate Gateway is a SOAP API that handles batch file processing and reconciliations / settlements. All those APIs have various formats, lack clear documentation and are hard to use.

Solution
A new set of microservices is developed that exposes uniform set of APIs to channels applications and other 3rd party systems. Microservices transform those requests and trigger downstream WorldPay APIs.

2. Lack of real time events

Worldpay's webhook system can be delayed or out-of-order and requires manual polling for settlement files or reports. Additionally, there is no guaranteed delivery (no retries, no signatures in some versions).

Solution
A new adapter is introduced that processes WebHook request and pushes them into Kafka. Kafka events are used by microservices (e.g. notifications MS to notify customers and merchants of the payments success/failures) or by other systems.



Expected Outcomes

As organizations implement integration and modernization solutions, they can expect a new set of business outcomes:

How to achieve operational efficiency and business agility without disrupting critical legacy systems?

The anticipated result is a flexible, scalable IT environment where:

- Digital products and services can be launched rapidly, meeting evolving customer needs.
- Manual reconciliation and processing delays are minimized, increasing transaction speed and accuracy.
- Compliance with data privacy and regulatory requirements is streamlined through automated, auditable workflows.
- Integration with modern digital channels and third-party services is seamless, enabling new revenue streams.
- Legacy infrastructure continues to deliver value while supporting the organization's digital transformation journey.

The next phase is the adoption of proven integration patterns and automation frameworks—allowing companies to unlock value from their legacy systems while accelerating innovation, reducing operational risk, and maintaining control over sensitive business data.

Optimizing Transaction Reconciliation in a Marketplace Platform

Technical Challenges and Solutions

3. Manual Settlement Reconciliation

WorldPay requires complicated process for Settlement reconciliation that requires manual file uploads.

Solution
Automated settlement service is implemented that pulls reports via SFTP/API, parses files and matches them to ledger entries. This way it is possible to automate handling for missing payment entries.

4. Chargebacks and Dispute Management

Worldpay requires manual handles of chargebacks in Worldpay Backoffice.

Solution
A separate backoffice application I available for internal operators that can review and automatically process refunds and handle disputes. System will automatically initiate refunds to customers based on dispute resolution result.

5. PCI DSS compliance

Expansions of Worldpay usually require PCI DS certification that is complicated and takes a lot of time and effort. Additionally, it significantly complicates implementation of new features due to need of certification activities for new releases.

Solution
Customer payments can be processed with the use of hosted payment page or client side tokenization. This way client applications and microservices do not handle card data directly and should not comply to PCI DSS.

6. Reporting limitations

WorldPay exposes very limited amount of reports and it is hard to build proper business-oriented reports.

Solution
On new API layer data to be duplicated in the database. That allows BI solution to use this data to build any dashboards required by business team.

7. Manual merchant onboarding

Worldpay OOTB supports only manual process of KYC/KYB to onboard new merchants. That requires a lot of manual work and decreases the customer experience for merchants

Solution
A new service is built to handle KYC/KYB process that uses WorldPay APIs to add new merchants. 3rd party KYC / KYB service that automatically validates uploaded documents is used.

8. Performance / Error handling

Worldpay has limited API performance and response times may be large, especially in case of large load.

Solution
New set of microservices use distributed cache system and advanced cache invalidation scenarios. This way microservices can serve up to data information much later than downstream, systems. Additionally it is possible to handle downstream system errors, e,g, by pushing requests when downstream systems come online.

Thank you and...

Let's build seamless solutions together!

Expected Outcomes

As organizations implement integration and modernization solutions, they can expect a new set of business outcomes:

How to achieve operational efficiency and business agility without disrupting critical legacy systems?

The anticipated result is a flexible, scalable IT environment where:

- Digital products and services can be launched rapidly, meeting evolving customer needs.
- Manual reconciliation and processing delays are minimized, increasing transaction speed and accuracy.
- Compliance with data privacy and regulatory requirements is streamlined through automated, auditable workflows.
- Integration with modern digital channels and third-party services is seamless, enabling new revenue streams.
- Legacy infrastructure continues to deliver value while supporting the organization's digital transformation journey.

The next phase is the adoption of proven integration patterns and automation frameworks—allowing companies to unlock value from their legacy systems while accelerating innovation, reducing operational risk, and maintaining control over sensitive business data.

Thank you and...

Let's build seamless solutions together!