Topic 2.2: The API Economy and Banking-as-a-Service

How Non-Banks Offer Financial Services

Joerg Osterrieder

Digital Finance

Digital Finance

Learning Objectives

By the end of this topic, you will be able to:

1. **Define** what an API is and explain why it matters for modern finance
2. **Explain** how open banking regulation enables third-party access to banking data
3. **Describe** the Banking-as-a-Service model and how it separates licenses from experiences
4. **Analyze** how embedded finance allows non-banks to offer financial products
5. **Compare** token-based security with traditional password sharing
6. **Apply** open banking concepts through API simulation exercises (NB03)

Key Competency: Explain how APIs, open banking, and BaaS work together to let any company offer financial services — and identify the risks involved.

Prerequisites: What You Need to Know

No Prior Technical Knowledge Required
This topic is self-contained. We will explain:

- How software systems communicate with each other
- What it means to authenticate and authorize
- How data flows between a bank and a third party
- Why regulation matters for innovation

Key Background Concepts

- **Client-Server**: One system requests, another responds
- **Internet Protocol**: Rules for data transmission
- **Authentication**: Proving *who* you are
- **Authorization**: Defining *what* you are allowed to do

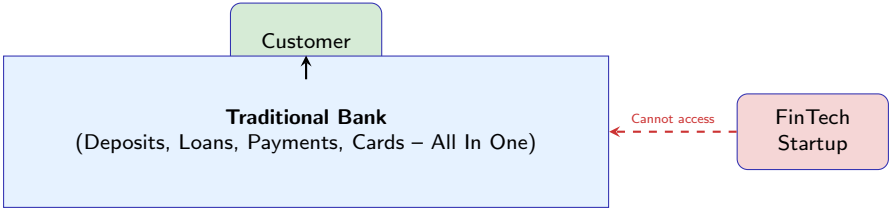
From Topic 2.1

Recall: Banks have traditionally been "bundled" — offering all services under one roof. This topic explains *how* APIs enable "unbundling."

The Traditional Banking Problem

The Problem

Banks built closed, proprietary systems for decades. How could a startup innovate on top of banking infrastructure it could not access?



The Traditional Banking Problem (cont.)

The Concept:

- **Closed Systems**: Each bank built its own proprietary technology
- **No Standard Communication**: No common language between institutions
- **Barrier to Innovation**: Startups could not build on banking infrastructure
- **Customer Lock-in**: Switching providers meant starting from scratch

The Insight

The solution is Application Programming Interfaces (APIs) — standardized contracts that let any authorized software talk to bank systems.

What is an API?

The Problem

How do different software systems talk to each other without knowing each other's internal workings?

Non-Technical Definition:

- An API is a **contract** between software systems
- It defines what you can **request** and what you will **receive**
- Think of it like a restaurant menu: a fixed set of options, in a standardized format

Jargon explained: *API* stands for Application Programming Interface. *Endpoint* = a specific address where you send requests (like a phone number). *Request* = asking for data. *Response* = the data you get back.

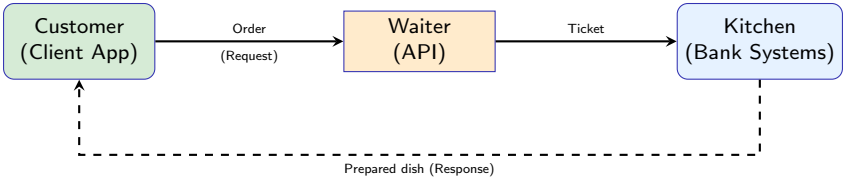
Why APIs Matter for Finance

- **Unbundling**: Break a monolithic bank into separate, accessible components
- **Speed**: Integrate with banking infrastructure quickly instead of building from scratch
- **Innovation**: Any authorized developer can access banking capabilities
- **Competition**: A more level playing field between incumbents and startups

The Insight

APIs let any developer access banking capabilities without building them — just as a restaurant menu lets you order food without knowing how to cook.

The Restaurant Analogy



The Menu (API Spec)

- Lists available options
- Standardized formats
- Clear descriptions of what you can order

The Waiter (API)

- Takes your request
- Validates the order
- Returns the result

The Kitchen (Bank)

- Does the actual work
- Hidden from the customer
- Can change internally without affecting the menu

Key takeaway: You do not need to know how the kitchen works. You just need to know the menu.

How APIs Communicate

The Problem

What does an API request actually look like? How does a FinTech app "talk" to a bank?

Four Core Operations:

- **GET**: Read information
  ("Show me my account balance")
- **POST**: Create or send something
  ("Send a payment to Alice")
- **PUT**: Update existing information
  ("Change my address")
- **DELETE**: Remove something
  ("Cancel a pending transfer")

Simplified Example:

```
# Read account balance
GET /accounts/balance

# Send a payment
POST /payments
{
  "recipient": "Alice",
  "amount": "lunch money"
}
```

The Insight

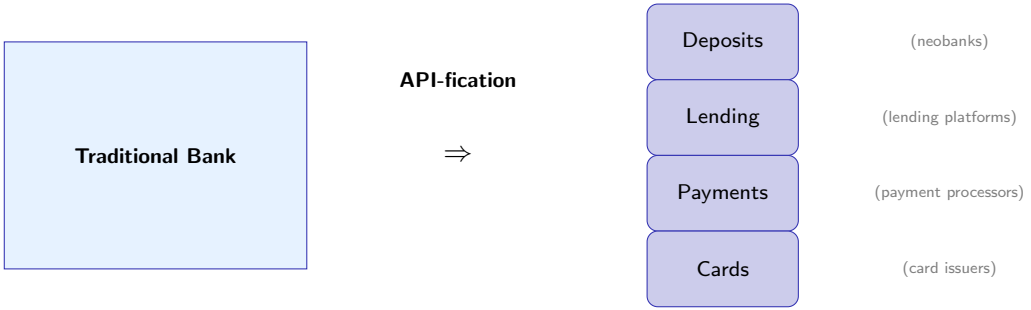
These four operations — read, create, update, delete — cover nearly everything a financial application needs to do. The simplicity of this contract is what makes APIs so powerful.

Jargon explained: *HTTP* is the language computers use to communicate on the web. Every time you load a webpage, your browser sends HTTP requests.

The Unbundling of Banking

The Problem

What happens when you break a monolithic bank into separate, independently accessible services?



The Unbundling of Banking (cont.)

The Concept:

- Each banking function becomes a standalone service accessible via API
- Different companies can specialize in one service and do it exceptionally well
- A single startup can compete with a bank on one specific dimension

The Insight

Any service can now be offered independently by a different company. The "bank" is no longer a single institution — it is a collection of components that can be mixed and matched.

Open Banking: The Concept

The Problem

Why would banks voluntarily share your data with competitors? Most would not — unless regulators require it.

Definition

Open Banking is a regulatory and technical framework in which banks provide third-party financial service providers access to consumer banking data and payment functionality through secure APIs — *with customer consent*.

What Banks Must Share (with consent):

- Account balances
- Transaction history
- Payment initiation capability
- Account holder information

Key Principles:

- **Customer Control**: You decide who sees your data
- **Standardized**: Same format across all banks
- **Secure**: Strong authentication required
- **Regulated**: Government oversight ensures compliance

The Insight

Open banking shifts power from banks to consumers. Your financial data belongs to *you*, not to your bank.

Open Banking: Regulatory Approaches

The Problem

How do different regions approach opening up banking data — and why does the approach matter?

Regulated Approach:

- Governments **mandate** that banks provide APIs
- Standardized specifications ensure interoperability
- Third parties must be licensed and supervised
- Adopted by several European and Asia-Pacific jurisdictions

"Banks must open up — whether they want to or not."

Market-Driven Approach:

- No government mandate for standardized APIs
- Data aggregators negotiate access with banks individually
- Screen-scraping (sharing passwords) remains common
- Innovation happens, but without standardization

"Let the market figure it out."

The Insight

The regulatory approach shapes how fast innovation happens. Mandated standards create a level playing field quickly; market-driven approaches allow flexibility but create fragmentation.

AISP and PISP: Read Access vs. Write Access

The Problem

What can third parties actually *do* with your banking data — and how is this regulated?

AISP — Account Information

Account Information Service Provider

- **What**: Reads your account data
- **Use cases**:
  - Budgeting apps that show all your accounts in one place
  - Credit assessment based on transaction history
  - Financial planning tools
- **Permission**: Read-only access

PISP — Payment Initiation

Payment Initiation Service Provider

- **What**: Initiates payments from your account
- **Use cases**:
  - E-commerce checkout directly from your bank
  - Bill payment services
  - Money transfer apps
- **Permission**: Write access (with explicit approval)

AISP and PISP: Read Access vs. Write Access (cont.)

The Insight

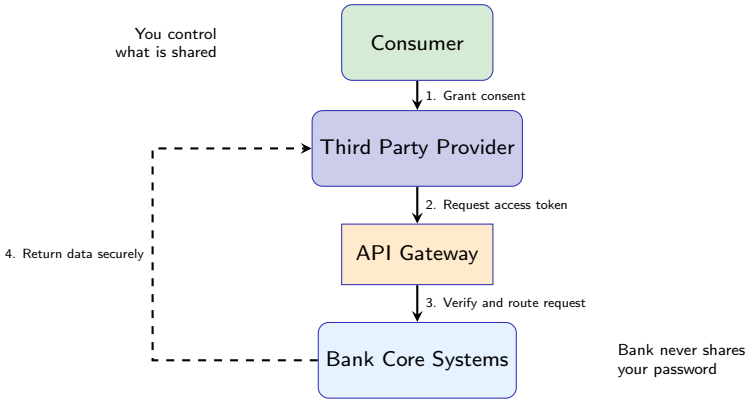
Read access and write access are regulated differently because the risks differ. Seeing your balance is far less risky than moving your money.

Jargon explained: *TPP (Third Party Provider)* = a company (not your bank) that accesses your bank data with your permission. *Account aggregation* = combining data from multiple banks into one view.

Open Banking Architecture: The Consent Flow

The Problem

How does the consent flow actually work — and how is your data kept secure?



Open Banking Architecture: The Consent Flow (cont.)

The Insight

You never share your password with the third party. Instead, the bank issues a temporary *token* — like a concert wristband that grants limited access and can be revoked at any time.

API Types in Financial Services

The Problem

What financial services can be accessed via API — and what does each one do?

API Type	Function	Who Uses It
Account Information	Read balances, transaction history	Budgeting apps, lenders
Payment Initiation	Trigger bank-to-bank transfers	E-commerce, bill pay
Card Issuance	Create virtual or physical cards	Neobanks, expense tools
Lending	Originate and service loans	Lending platforms
Identity / KYC	Verify a customer's identity	Any regulated service
Core Banking	Full account ledger functionality	Companies building banks

The Insight

You can assemble a "bank" from API building blocks without building any of the underlying infrastructure yourself. This is the foundation of Banking-as-a-Service.

Jargon explained: *KYC* = Know Your Customer, the process of verifying someone's identity before providing financial services. Required by law in most countries.

Banking-as-a-Service: The Concept

The Problem

What if you want to offer banking products to your customers — but you do not want to become a bank?

Definition

Banking-as-a-Service (BaaS) is a model where licensed banks provide their banking infrastructure — including their charter, compliance systems, and account ledger — to non-banks via APIs, enabling them to offer financial products under their own brand.

Without BaaS:

- Obtaining a banking license requires enormous capital and years of effort
- Must build compliance infrastructure from scratch
- Need to hire regulatory and legal experts
- Must create core banking systems

With BaaS:

- "Rent" a license from a partner bank
- Use pre-built compliance tools
- Focus entirely on customer experience
- Launch much faster than building a bank

Banking-as-a-Service: The Concept (cont.)

The Insight

BaaS separates “who builds the experience” from “who holds the license.” This is what allows technology companies to offer bank accounts, cards, and loans.

The BaaS Stack: What You Build vs. What You Rent

The Problem

In a BaaS arrangement, what does the FinTech actually build — and what does it rent from others?

Brand Layer: Your Company's Brand and Identity

UX Layer: Mobile App, Web Portal, Customer Support

Middleware: BaaS Platform (connects everything)

Banking Layer: Partner Bank (holds the license)

Regulatory Layer: Deposit Insurance, Compliance, Charter

The BaaS Stack: What You Build vs. What You Rent (cont.)

The Insight

The FinTech only needs to build what the customer sees and touches. Everything underneath — the license, the compliance, the ledger — is rented from specialized providers.

Jargon explained: *Middleware* = software that connects different systems together. *Charter* = a government-issued license to operate as a bank.

BaaS Business Model: How Everyone Gets Paid

The Problem

If a FinTech is not a bank, how does revenue flow in a BaaS arrangement — and who captures the most value?

Revenue Sources:

- **Interchange fees**: When a customer uses a card, a small fee is generated and split among all parties
- **Subscription fees**: Monthly charges for premium features
- **Interest spread**: Difference between deposit and lending rates
- **Per-account fees**: The BaaS platform charges for each account maintained

Who Gets What (Conceptually):

- **FinTech brand**: Captures the largest share because it owns the customer relationship
- **BaaS platform**: Takes a cut for technology and compliance orchestration
- **Partner bank**: Receives fees for providing the license and deposit insurance

The party closest to the customer typically captures the most value.

The Insight

In the API economy, value accrues to whoever owns the customer relationship — not necessarily to whoever holds the banking license.

Jargon explained: *Interchange* = a fee paid between banks when processing card transactions; a major revenue source for card issuers and now also for FinTechs.

Embedded Finance: Financial Services at the Moment of Need

The Problem

Why are non-financial companies — retailers, ride-sharing apps, e-commerce platforms — starting to offer financial products?

What is Embedded Finance?
Financial services integrated directly into non-financial platforms and experiences — offered at the exact moment a customer needs them.

Conceptual Examples:

- An e-commerce platform offering merchant loans based on sales data
- A ride-sharing app providing driver banking and instant payouts
- A retail checkout offering installment payments
- A freelancing platform with built-in invoicing and tax savings

Why Non-Banks Have Advantages

- **Distribution**: They already have the customers
- **Data**: They know customer behavior intimately
- **Context**: They can offer finance at the exact moment of need
- **Trust**: Customers already have a relationship with the brand

Embedded Finance: Financial Services at the Moment of Need (cont.)

The Insight

When finance is embedded where you already are — shopping, working, traveling — the traditional bank becomes invisible. The bank still exists in the background, but the customer never interacts with it directly.

API Security: How Trust Works

The Problem

How do you let a third party access your bank account without giving them your password?

The Old Way (Dangerous):

- Share your username and password with the third party
- They log in *as you*
- Full access — no limits
- If compromised, everything is exposed
- You cannot revoke without changing your password

This is called "screen scraping" and is still used in some regions.

The New Way (Token-Based / OAuth):

1. You grant consent on your bank's website
2. The bank issues a temporary access token
3. The third party uses the token — never your password
4. The token has limited scope (e.g., read-only)
5. You can revoke access at any time

Like giving a valet a special key that only starts the car — not one that opens the trunk.

The Insight

Token-based access is fundamentally more secure than password sharing. It gives you granular control — you decide *what* is shared, *with whom*, and for *how long*.

The Problem

What can go wrong when a FinTech depends on a partner bank for its entire banking infrastructure?

Risks for FinTechs:

- **Partner bank failures**: If the partner bank faces regulatory action, the FinTech's customers may lose access to their accounts
- **Regulatory scrutiny**: Regulators increasingly hold banks responsible for their FinTech partners' behavior
- **Concentration risk**: Only a small number of banks serve as BaaS partners
- **Revenue pressure**: Partner banks may demand a larger share over time

Risks for Partner Banks:

- Reputation damage from FinTech failures
- Compliance responsibility for FinTech actions
- Anti-money-laundering obligations extend to FinTech customers
- Capital requirements may increase with growth

Key Trend

Regulators are increasingly demanding that banks "know their FinTech partners' customers" — adding cost and complexity to BaaS relationships.

Risks and Challenges in BaaS (cont.)

The Insight

Renting infrastructure means sharing someone else's risks. A FinTech's fate is tied to its partner bank's regulatory standing.

Hands-On Exercise: Notebook NB03

Open Banking API Simulation

In Notebook NB03, you will build a simulated open banking environment:

1. Create a mock bank with customer accounts
2. Implement Account Information API endpoints
3. Simulate a token-based authentication flow
4. Build a Payment Initiation service
5. Create an Account Aggregator (multi-bank view)

What You Will Build:

- A simulated bank with realistic account structures
- API endpoints for reading accounts, balances, and transactions
- A payment initiation endpoint
- A multi-bank aggregation dashboard

Learning Goal: Understand the mechanics of API-based banking by building it yourself — even without prior programming experience, the notebook guides you step by step.

Notebook: day_02/notebooks/NB03_Open_Banking_API.ipynb

Discussion: Who Wins in Open Banking?

Winners

- Consumers (more choice, better innovation)
- FinTechs (access to infrastructure they could not build alone)
- Data aggregators (critical middleware position)
- Tech-savvy banks (new revenue from APIs)

Losers

- Traditional banks (losing their data monopoly)
- Legacy technology systems (forced to modernize or die)
- Screen-scraping services (replaced by proper APIs)

Uncertain

- Regulators (struggling to keep pace with innovation)
- Privacy advocates (more data sharing means more risk)
- Small banks (high compliance cost relative to their size)

Discussion Questions

1. If every company can embed financial services, what happens to traditional banks' competitive advantage?
2. Is open banking ultimately good or bad for consumer privacy?

Discussion: Building a FinTech with BaaS

Scenario: You want to launch a neobank for freelancers. Walk through the key decisions.

1. Define product: Accounts, cards, invoicing, tax savings
2. Choose a BaaS partner to provide banking infrastructure
3. Integrate APIs: Accounts, cards, payments, identity verification
4. Build UX: Mobile app, web dashboard, customer support
5. Launch and acquire customers

Discussion Questions

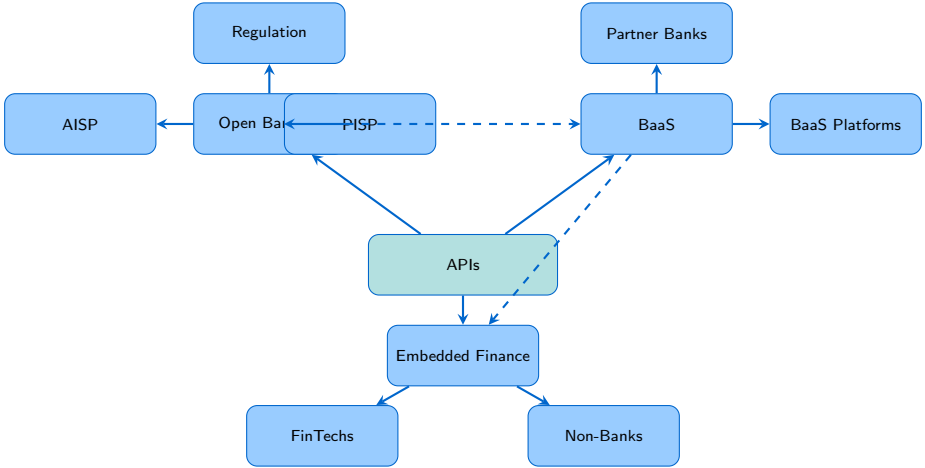
1. What are the biggest risks in this model?
2. What happens if your partner bank faces regulatory problems?
3. How would you differentiate your product from existing neobanks?
4. What data would you need from freelancers that traditional banks do not collect?

Executive Summary

Key Takeaways

1. **APIs unbundled banking**: Any financial service can now be offered separately via standardized interfaces — breaking the bank monopoly on financial products
2. **Regulation drives adoption**: In regions where governments mandate open banking, innovation accelerates; in market-driven regions, adoption is slower and more fragmented
3. **BaaS enables non-banks**: Companies can "rent" banking infrastructure (charter, compliance, ledger) instead of building it from scratch, dramatically lowering the barrier to entry
4. **Embedded finance is the frontier**: Non-financial platforms increasingly offer financial services at the moment of need, making the traditional bank invisible to the end consumer
5. **Value accrues to the customer interface**: Whoever owns the customer relationship captures the most value in the API economy — not necessarily whoever holds the banking license

Concept Map: The API Economy Ecosystem



APIs are the technical foundation enabling **Open Banking** (regulated data access), **BaaS** (infrastructure rental), and **Embedded Finance** (integration into non-financial products). Each builds on the previous.

Key Terms and Definitions

API	**Application Programming Interface** — A standardized contract allowing software systems to communicate, defining available requests and expected responses.
Open Banking	A regulatory and technical framework requiring banks to share customer data with authorized third parties via secure APIs, with customer consent.
AISP	**Account Information Service Provider** — A licensed third party authorized to read and aggregate bank account data (read-only access).
PISP	**Payment Initiation Service Provider** — A licensed third party authorized to initiate payments from a user's bank account (write access).
BaaS	**Banking-as-a-Service** — A model where licensed banks provide banking infrastructure to non-banks via APIs.
Embedded Finance	Financial services integrated directly into non-financial platforms and customer journeys.
OAuth	An authorization framework allowing third parties to access resources using tokens instead of passwords.

Common Misconceptions and Self-Assessment

Common Myths:

Myth 1: "Neobanks are banks"
Reality: Most neobanks are FinTechs partnering with licensed banks. They do not hold a full banking charter themselves.

Myth 2: "Open Banking means anyone can see my data"
Reality: Explicit customer consent is required. You control who accesses what, and can revoke access at any time.

Myth 3: "APIs make banking less secure"
Reality: Proper APIs with token-based authentication are more secure than screen-scraping, which requires sharing passwords.

Self-Assessment Questions:

1. In your own words, explain the difference between an AISP and a PISP.
2. Why does a FinTech using BaaS depend on its partner bank's regulatory standing?
3. What are the advantages of token-based access over password sharing?
4. If you were building a FinTech, would you apply for your own banking license or use BaaS? Justify your reasoning.

Coming Up

Topic 2.3: Data-Driven Finance — Lending, Scoring, and Algorithmic Decisions

Preview:

- How do lenders decide whether to trust you with money they might never see again?
- Traditional credit scoring vs. machine learning approaches
- Alternative data: what if your everyday financial behavior could help you get a loan?
- Algorithmic bias: can a "neutral" algorithm discriminate?
- The right to know why you were turned down

Connection to this topic:
Open Banking APIs provide the **data** that powers data-driven finance. Without API access to transaction history, alternative credit scoring would not be possible.

Questions?

Remember: Complete Notebook NB03 (Open Banking API Simulation) before the next session.

Next: Topic 2.3 — Notebook: NB03 — Open Banking API Simulation