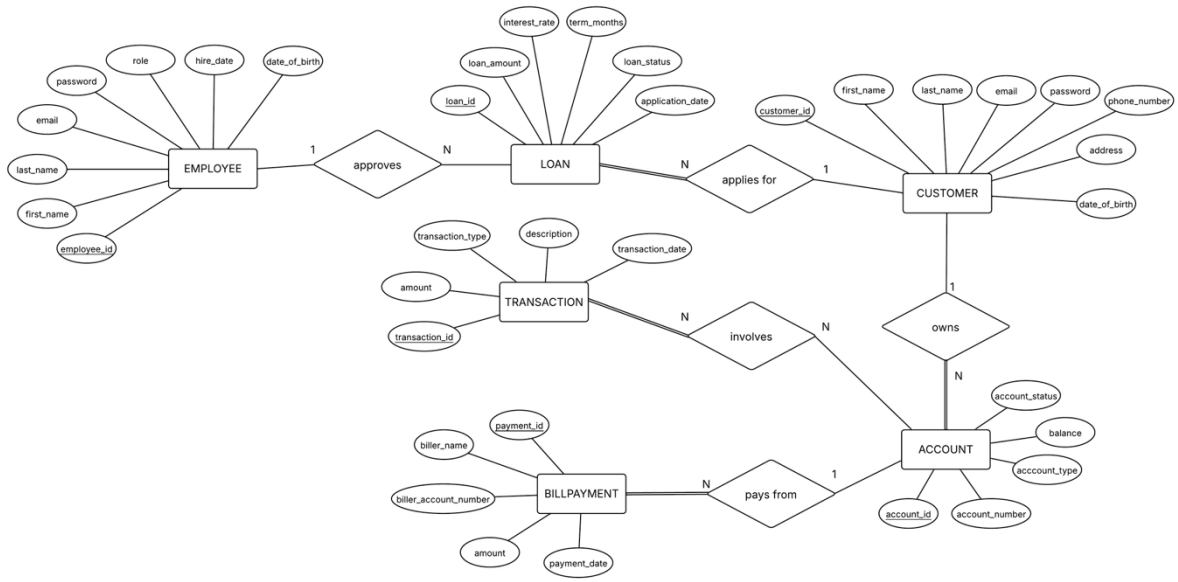
MK BANKING APP

Kaan Altaş, Murat Enes Erdoğan

Project Description

We created a comprehensive banking application that provides a complete digital banking experience for both customers and bank employees. Using our system, customers can register for accounts, view their checking and savings account balances, transfer money between accounts, pay bills to registered billers, apply for loans, and track their transaction history. Bank employees can access an administrative dashboard to manage customer accounts, review and approve/reject loan applications, generate advanced analytical reports, and monitor system-wide financial metrics. The application features real-time data synchronization, role-based access control, and sophisticated financial analytics to support data-driven decision making.

Entity-Relationship Diagram



Relational Database Design

```
-- Enum Types
CREATE TYPE account_type AS ENUM ('Checking', 'Savings');
CREATE TYPE account_status AS ENUM ('Active', 'Inactive', 'Closed');
CREATE TYPE loan_status AS ENUM ('Pending', 'Approved', 'Rejected', 'Paid
Off');
CREATE TYPE transaction_type AS ENUM ('Transfer', 'Deposit', 'Withdrawal',
'Bill Payment');

-- Customers Table
CREATE TABLE Customers (
  customer_id SERIAL PRIMARY KEY,
  first_name VARCHAR(50),
```

```sql
  last_name VARCHAR(50),
  email VARCHAR(50) UNIQUE NOT NULL,
  password VARCHAR(50),
  phone_number VARCHAR(50),
  address VARCHAR(50),
  date_of_birth DATE
);

-- Employees Table
CREATE TABLE Employees (
  employee_id SERIAL PRIMARY KEY,
  first_name VARCHAR(50),
  last_name VARCHAR(50),
  email VARCHAR(50) UNIQUE NOT NULL,
  password VARCHAR(50),
  role VARCHAR(16),
  hire_date DATE,
  date_of_birth DATE
);

-- Accounts Table
CREATE TABLE Accounts (
    account_id SERIAL PRIMARY KEY,
    customer_id INT,
    account_number VARCHAR(20) NOT NULL UNIQUE,
    account_type account_type NOT NULL,
    balance DECIMAL(15, 2) NOT NULL DEFAULT 0.00,
    status account_status NOT NULL DEFAULT 'Active',
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);

-- Bill Payments Table
CREATE TABLE BillPayments (
    payment_id SERIAL PRIMARY KEY,
    account_id INT,
    biller_name VARCHAR(100) NOT NULL,
    biller_account_number VARCHAR(50) NOT NULL,
    amount DECIMAL(15, 2) NOT NULL,
    payment_date DATE,
    FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);

-- Loans Table
CREATE TABLE Loans (
    loan_id SERIAL PRIMARY KEY,
    customer_id INT,
    loan_amount DECIMAL(15, 2) NOT NULL,
    interest_rate DECIMAL(5, 2) NOT NULL,
    term_months INT NOT NULL,
    status loan_status NOT NULL DEFAULT 'Pending',
    application_date DATE,
    approved_by_employee_id INT,
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id),
    FOREIGN KEY (approved_by_employee_id) REFERENCES Employees(employee_id
```

```sql
)
);

-- Transactions Table
CREATE TABLE Transactions (
    transaction_id SERIAL PRIMARY KEY,
    from_account_id INT,
    to_account_id INT,
    amount DECIMAL(15, 2) NOT NULL,
    transaction_type transaction_type NOT NULL,
    description VARCHAR(255),
    transaction_date DATE,
    FOREIGN KEY (from_account_id) REFERENCES Accounts(account_id),
    FOREIGN KEY (to_account_id) REFERENCES Accounts(account_id)
);
```

Data Sources

We populated our database using Python's **Faker** library together with custom scripts to ensure realistic relationships between entities. Our database contains:

- **Customers**: 52 tuples with realistic names, emails, phone numbers, and demographic information
- **Employees**: 10 tuples representing various bank roles (Loan Officers, Customer Service, Admins)
- **Accounts**: 104 tuples (approximately 2 accounts per customer on average, mix of Checking and Savings)
- **Transactions**: 500+ tuples representing various transfer types between accounts
- **Loans**: 50 tuples with different statuses (Pending, Approved, Rejected, Paid Off)
- **BillPayments**: 200+ tuples representing utility and service payments

The data includes edge cases such as high-value customers (>$75 000 balance), inactive customers (no transactions in 6+ months), and customers with multiple loans to test our analytical queries effectively.

Advanced SQL Queries

Query 1: High-Value Customers (Balance > $75,000)

```sql
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    SUM(a.balance) AS total_balance
FROM customers c
INNER JOIN accounts a ON c.customer_id = a.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name
HAVING SUM(a.balance) > 75000
ORDER BY total_balance DESC
LIMIT 10;
```

Purpose: Identifies VIP customers with significant deposits for targeted services and retention programs.

Integration: Displayed on the Admin Dashboard and Reports page to help bank managers focus on high-value relationships.

Query 2: Pending Loan Applications with Risk Assessment

```
WITH customer_balances AS (
    SELECT
        c.customer_id,
        COALESCE(SUM(a.balance), 0) as total_balance
    FROM customers c
    LEFT JOIN accounts a ON c.customer_id = a.customer_id
    GROUP BY c.customer_id
),
customer_transactions AS (
    SELECT
        a.customer_id,
        AVG(t.amount) as avg_transaction_amount,
        COUNT(t.transaction_id) as transaction_count
    FROM accounts a
    LEFT JOIN transactions t ON (a.account_id = t.from_account_id OR a.acc
ount_id = t.to_account_id)
    GROUP BY a.customer_id
),
active_loans AS (
    SELECT
        customer_id,
        COUNT(*) as active_loans_count
    FROM loans
    WHERE status = 'Approved'
    GROUP BY customer_id
)
SELECT
    l.loan_id,
    l.customer_id,
    c.first_name || ' ' || c.last_name as customer_name,
    l.loan_amount,
    l.interest_rate,
    l.term_months,
    cb.total_balance::numeric,
    COALESCE(al.active_loans_count, 0)::int as active_loans_count,
    COALESCE(ct.avg_transaction_amount, 0)::numeric as average_transaction
_amount
FROM loans l
INNER JOIN customers c ON l.customer_id = c.customer_id
LEFT JOIN customer_balances cb ON l.customer_id = cb.customer_id
LEFT JOIN customer_transactions ct ON l.customer_id = ct.customer_id
LEFT JOIN active_loans al ON l.customer_id = al.customer_id
WHERE l.status = 'Pending'
ORDER BY l.application_date ASC;
```

Purpose: Provides loan officers with comprehensive financial metrics for each pending loan application to make informed approval decisions.

Integration: Used in the Loan Approval page where officers can review applications with risk indicators.

Query 3: Inactive Customers (No Transactions in 6+ Months)

```
WITH active_customers AS (
    SELECT DISTINCT a.customer_id
    FROM accounts a
    INNER JOIN transactions t ON (a.account_id = t.from_account_id OR a.ac
count_id = t.to_account_id)
    WHERE t.transaction_date >= CURRENT_DATE - INTERVAL '6 months'
)
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    c.email,
    6 as months_inactive
FROM customers c
WHERE NOT EXISTS (
    SELECT 1 FROM active_customers ac WHERE ac.customer_id = c.customer_id
)
AND EXISTS (
    SELECT 1 FROM accounts a WHERE a.customer_id = c.customer_id
)
LIMIT 10;
```

Purpose: Identifies dormant accounts for re-engagement campaigns and potential account closure procedures.

Integration: Displayed in the Advanced Reports section to help customer service teams reach out to inactive customers.

Query 4: Monthly Transaction Volume Trends

```
WITH months AS (
    SELECT generate_series(
        date_trunc('month', CURRENT_DATE - INTERVAL '11 months'),
        date_trunc('month', CURRENT_DATE),
        '1 month'::interval
    ) AS month
),
monthly_data AS (
    SELECT
        date_trunc('month', transaction_date) AS month,
        SUM(amount) AS total_volume,
        COUNT(*) AS transaction_count
    FROM transactions
    WHERE transaction_date >= date_trunc('month', CURRENT_DATE - INTERVAL
'11 months')
    GROUP BY date_trunc('month', transaction_date)
)
SELECT
    TO_CHAR(m.month, 'Mon YYYY') AS month,
    COALESCE(md.total_volume, 0)::numeric AS total_volume,
    COALESCE(md.transaction_count, 0)::int AS transaction_count
FROM months m
```

```
LEFT JOIN monthly_data md ON m.month = md.month
ORDER BY m.month;
```

Purpose: Tracks transaction volume trends over the past year to identify seasonal patterns and growth metrics.

Integration: Visualized as a line chart on the Admin Dashboard for at-a-glance performance monitoring.

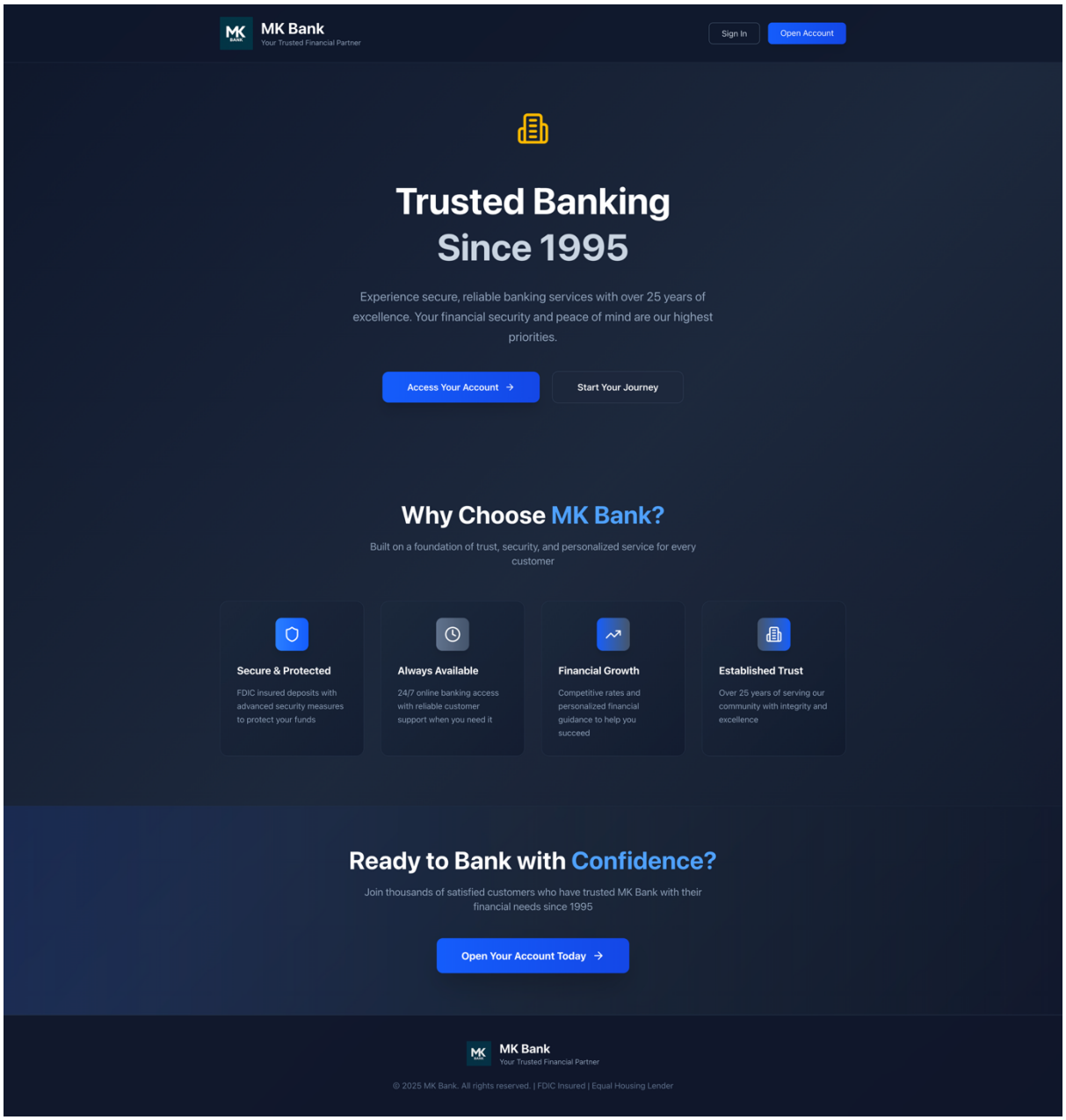
Query 5: Customer Loan Distribution Analysis

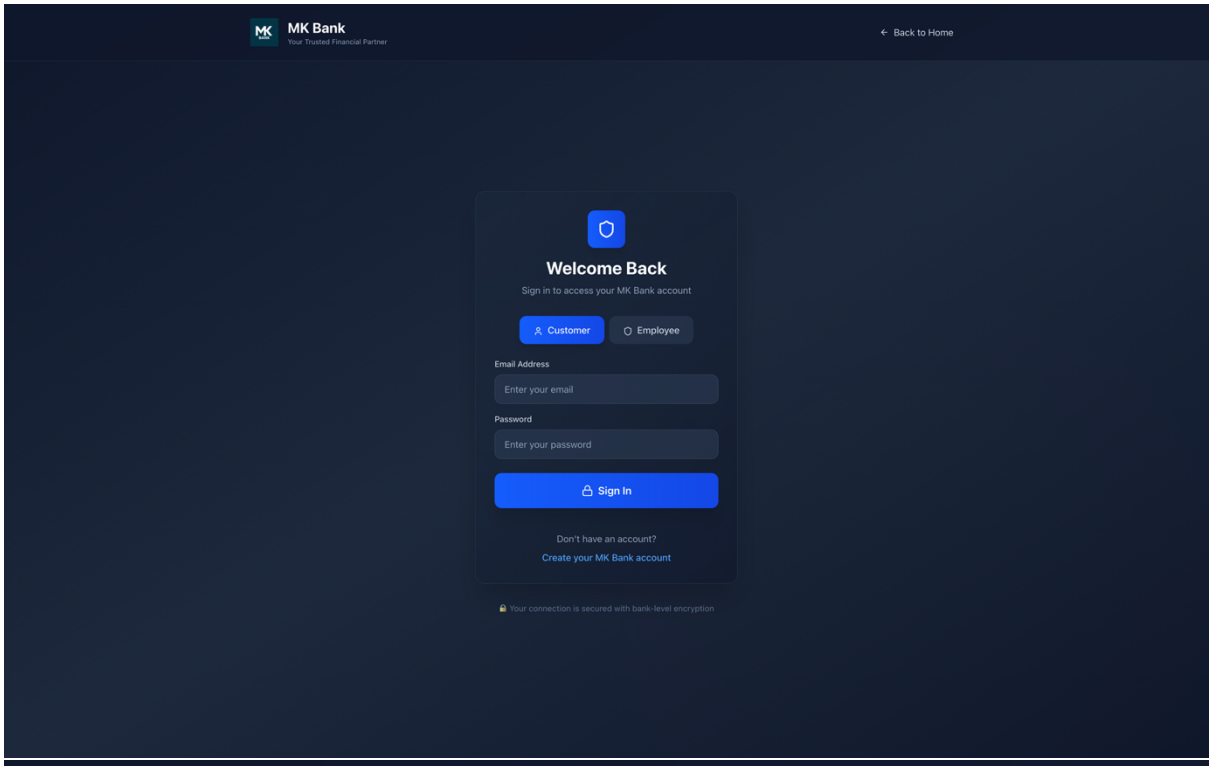
```
WITH customer_loan_stats AS (
    SELECT
        c.customer_id,
        c.first_name,
        c.last_name,
        COUNT(l.loan_id) as total_loans,
        COUNT(CASE WHEN l.status = 'Pending' THEN 1 END) as pending_loans,
        COUNT(CASE WHEN l.status = 'Approved' THEN 1 END) as approved_loan
s,
        COUNT(CASE WHEN l.status = 'Rejected' THEN 1 END) as rejected_loan
s,
        COUNT(CASE WHEN l.status = 'Paid Off' THEN 1 END) as paid_off_loan
s,
        COALESCE(SUM(CASE WHEN l.status = 'Approved' THEN l.loan_amount EN
D), 0) as total_approved_amount,
        COALESCE(AVG(l.interest_rate), 0) as avg_interest_rate
    FROM customers c
    LEFT JOIN loans l ON c.customer_id = l.customer_id
    GROUP BY c.customer_id, c.first_name, c.last_name
    HAVING COUNT(l.loan_id) > 0
)
SELECT
    customer_id,
    first_name,
    last_name,
    total_loans,
    pending_loans,
    approved_loans,
    rejected_loans,
    paid_off_loans,
    total_approved_amount::numeric,
    ROUND(avg_interest_rate::numeric, 2) as avg_interest_rate
FROM customer_loan_stats
ORDER BY total_loans DESC, total_approved_amount DESC
LIMIT 20;
```

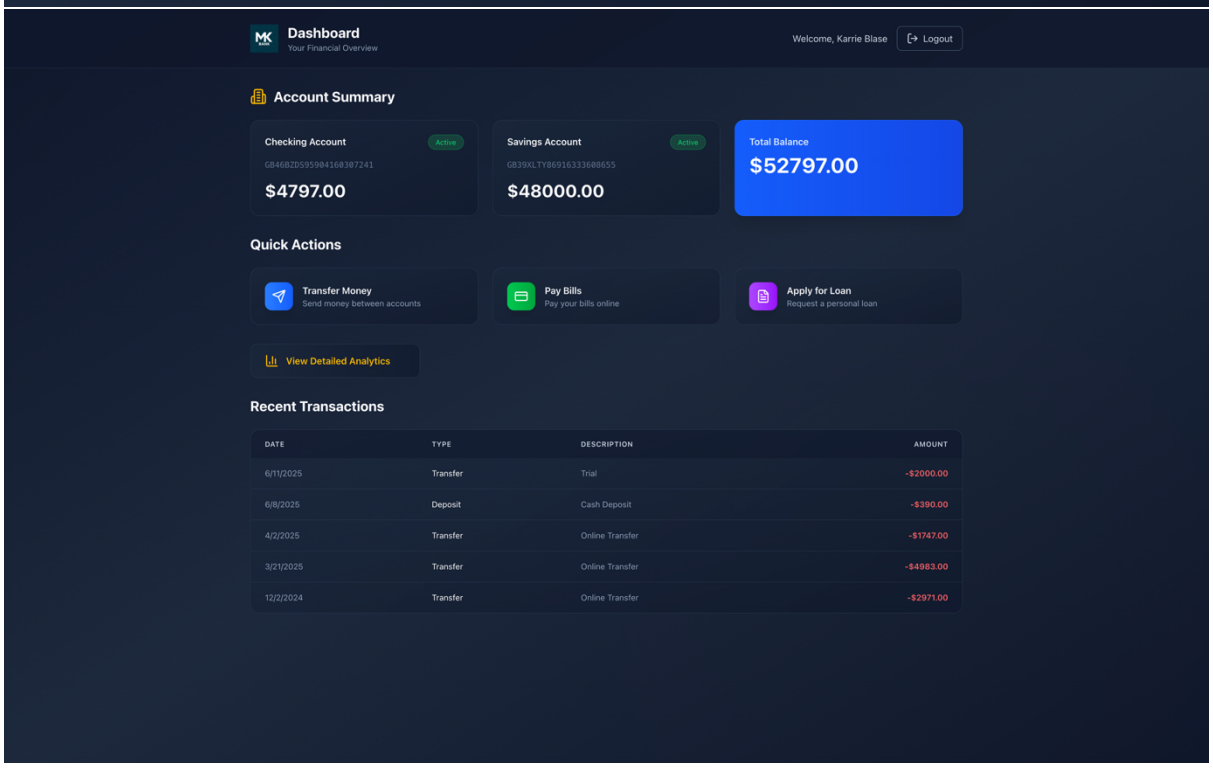
Purpose: Provides comprehensive loan portfolio analysis per customer to identify lending patterns and customer creditworthiness.

Integration: Featured in the Customer Analysis tab of the Reports page for detailed customer loan history review.,

Screenshots




MK Bank
Your Trusted Financial Partner
Back to Home
Welcome Back
Sign in to access your MK Bank account
Customer
Employee
Email Address
Enter your email
Password
Enter your password
Sign In
Don't have an account?
Create your MK Bank account
Your connection is secured with bank-level encryption


Dashboard
Your Financial Overview
Welcome, Karrie Blase
Logout
Account Summary
Checking Account
Active
GB46BZDS95904160307241
$4797.00
Savings Account
Active
GB39XLTY86916333600655
$48000.00
Total Balance
$52797.00
Quick Actions
Transfer Money
Send money between accounts
Pay Bills
Pay your bills online
Apply for Loan
Request a personal loan
View Detailed Analytics
Recent Transactions
DATE
TYPE
DESCRIPTION
AMOUNT
6/11/2025
Transfer
Trial
-$2000.00
6/8/2025
Deposit
Cash Deposit
-$390.00
4/2/2025
Transfer
Online Transfer
-$1747.00
3/21/2025
Transfer
Online Transfer
-$4983.00
12/2/2024
Transfer
Online Transfer
-$2971.00

Back to Dashboard

Checking Account

Account Number: GB46BZDS95904160307241

Status: Active

Current Balance

$4797.00

Transaction History

Filter and export your transactions

Export

From Date: dd.mm.yyyy

To Date: dd.mm.yyyy

Transaction Type: All Types

DATE	TYPE	DESCRIPTION	FROM/TO	AMOUNT	BALANCE
4/2/2025	Transfer	Online Transfer	GB62JWJR86798203316605	-$1747.00	-
3/21/2025	Transfer	Online Transfer	GB39XLTY86916333608655	-$4983.00	-
11/17/2024	Withdrawal	ATM Withdrawal	External	-$718.00	-
5/29/2024	Transfer	Online Transfer	GB73GSLP64250319025663	+$4509.00	-
1/2/2024	Transfer	Online Transfer	GB73GSLP64250319025663	-$843.00	-
12/10/2023	Deposit	Cash Deposit	External	+$1613.00	-
11/1/2023	Bill Payment	Payment to City Gas Co.	External	-$609.00	-
10/29/2023	Transfer	Online Transfer	GB10RHRT27202447956839	-$1854.00	-
9/12/2023	Transfer	Online Transfer	GB97OTRF21953481665368	+$3999.00	-
8/9/2023	Bill Payment	Payment to ConnectMe ISP	External	-$767.00	-

Back to Dashboard

Transfer Money

Send money between your accounts

From Account

Checking - GB46BZDS95904160307241 ($4797.00)

Available Balance: $4797.00

To Account Number

Enter recipient account number

Amount

$ 0.00

Description (Optional)

Add a note for this transaction...

Transfer Money

Back to Dashboard

Pay Bills

Manage your bill payments securely

From Account

Checking - GB46BZDS95904160307241 ($4797.00)

Available Balance: $4797.00

Biller Name

Enter biller name (e.g., Electric Company)

Biller Account Number

Enter biller account number

Amount

$ 0.00

Description (Optional)

Add a note for this transaction...

Pay Bill

Back to Dashboard

+ Apply for Loan

Your Loans

Manage your loan applications and payments

No loans found

You haven't applied for any loans yet.

Apply for Your First Loan

Back to Dashboard

Financial Analytics

Detailed insights into your financial activity

Account Summary

Your active accounts and balances

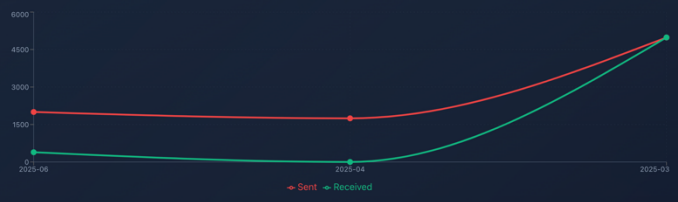
Savings Active
GB39XLTY86916333608655
$48000.00

Checking Active
GB46BZDS95904160307241
$4797.00

Total Balance
$52797.00

Monthly Transaction Volume

Transaction activity over the last 6 months



2025-06	2025-04	2025-03
2 transactions	1 transactions	2 transactions
$2390.00	$1747.00	$9966.00

Bill Payment Summary

Your recurring bill payments

ConnectMe ISP
Account: ACC-7337264378
$767.00
1 payments
Average payment: $767.00
Last payment: 8/9/2023

City Gas Co.
Account: ACC-9187891674
$609.00
1 payments
Average payment: $609.00
Last payment: 11/1/2023

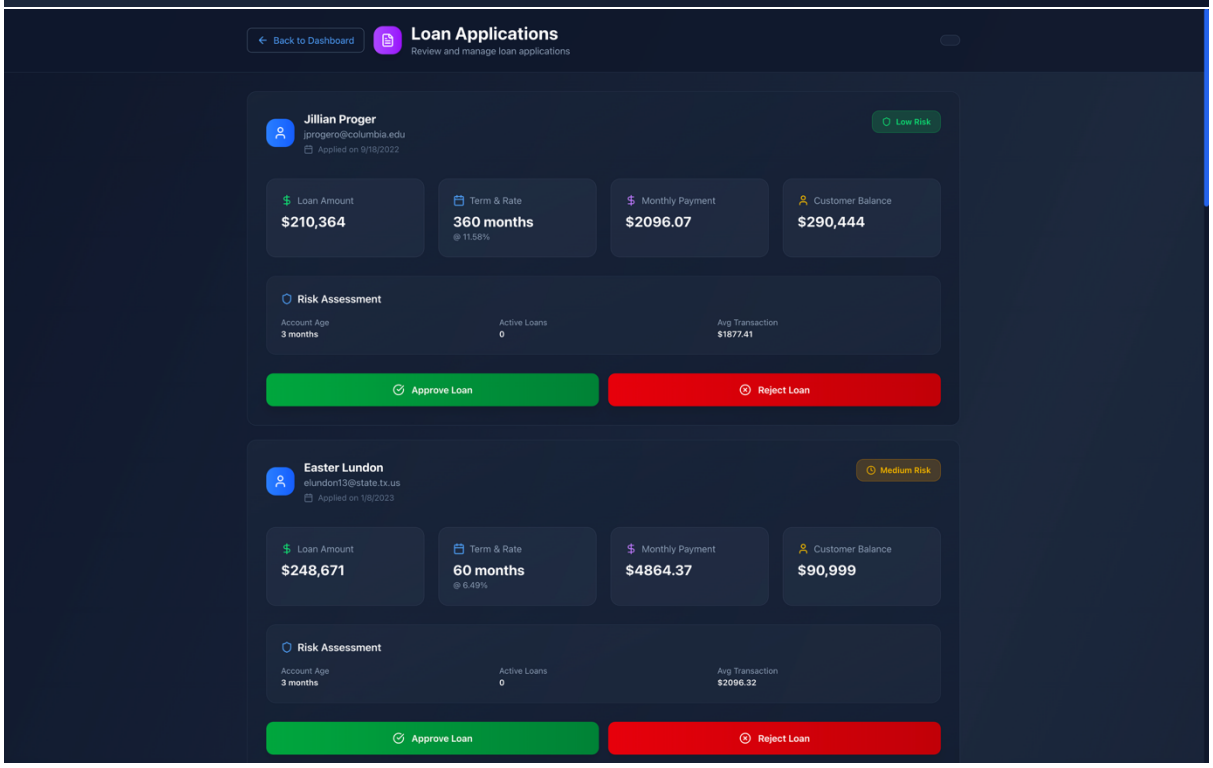
Recent Transactions

Your latest financial activity

Transfer 6/11/2025
Trial
Savings → Checking
-$2000.00

Deposit 6/8/2025
Cash Deposit
-$390.00

Admin Dashboard
Welcome, Coreen Fatkin (Admin)
Logout
View Advanced Analytics
Total Customers
52
Total Accounts
71
Total Deposits
$8,326,679
Pending Loans
8
Loan Approvals
Review and approve loan applications
Customer Management
View and manage customer accounts
Reports
View detailed analytics and reports
Monthly Transaction Volume
$220k
$176k
$132k
$88k
$44k
$0k
Aug 2024
Oct 2024
Dec 2024
Feb 2025
Apr 2025
Jun 2025
volume
High-Value Customers
ID İpek Dükel $699,969
CG Cally Grinston $407,231
MB Margret Brugh $365,428
VS Venus Scotsbrook $363,016
WA Wilbur Arrigo $358,319


Back to Dashboard
Loan Applications
Review and manage loan applications
Jillian Proger
jprogero@columbia.edu
Applied on 9/18/2022
Low Risk
Loan Amount
$210,364
Term & Rate
360 months
@ 11.56%
Monthly Payment
$2096.07
Customer Balance
$290,444
Risk Assessment
Account Age
3 months
Active Loans
0
Avg Transaction
$1877.41
Approve Loan
Reject Loan
Easter Lundon
elundon13@state.tx.us
Applied on 1/8/2023
Medium Risk
Loan Amount
$248,671
Term & Rate
60 months
@ 6.49%
Monthly Payment
$4864.37
Customer Balance
$90,999
Risk Assessment
Account Age
3 months
Active Loans
0
Avg Transaction
$2096.32
Approve Loan
Reject Loan

Back to Dashboard

Customer Management

Monitor and manage customer accounts

Search & Filter

Search customers by name or email...

All Customers

Showing 52 of 52 customers

Customer	Total Balance	Accounts	Loans	Last Activity	Actions
ID İpek Dükel ipek-0234@hotmail.com 05380704595	$699,969	2	0	6/15/2025 2 days ago	View Details
CG Cally Grinston cgrinstone@soundcloud.com 5477034434	$407,231	1	1	6/11/2025 6 days ago	View Details
MB Margret Brugh mbrugh10@foxnews.com 5301392389	$365,428	2	1	6/2/2025 15 days ago	View Details
VS Venus Scotsbrook vscotsbrook1b@unc.edu 5456553006	$363,016	2	0	5/22/2025 26 days ago	View Details
WA Wilbur Arrigo warrigow@tripadvisor.com 5405575141	$358,319	2	0	5/21/2025 27 days ago	View Details
SM Sarge Masdin smasdin1c@drupal.org 5431960014	$298,450	2	1	4/30/2025 48 days ago	View Details
JP Jillian Proger jprogero@columbia.edu 5358243359	$290,444	2	1	6/7/2025 10 days ago	View Details
SS Sandy Stranio sstranioc@behance.net 5413662716	$278,871	2	0	3/21/2025 88 days ago	View Details
AT Allissa Troctor atroctor1@freewebs.com 5497367151	$263,982	2	1	5/3/2025 45 days ago	View Details
Gretna McVicar					

Back to Dashboard

Advanced Reports

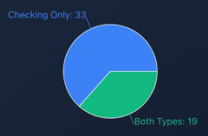
Comprehensive business analytics and insights

Overview | Customer Analysis | Loan Performance | Transaction Volume

High-Value Customers (>$75k Balance)

ID İpek Dükel	$699,969
CG Cally Grinston	$407,231
MB Margret Brugh	$365,428
VS Venus Scotsbrook	$363,016
WA Wilbur Arrigo	$358,319

Account Types Distribution

19

Customers with both account types