

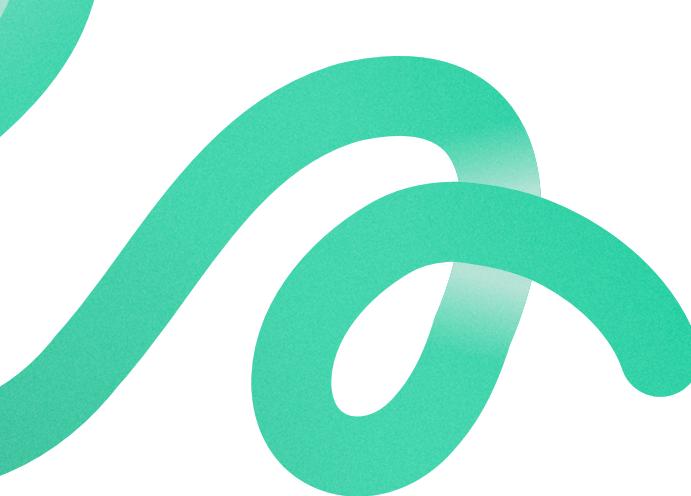
# PROJECT I

Ironhack Payments

Presentado por:

Alvaro Reyes  
Angel Vergara  
David Arenas  
Wilmer Acosta





# INTRODUCCIÓN

IronHack Payments, una empresa de servicios financieros con visión de futuro, ha estado ofreciendo soluciones innovadoras de adelantos de efectivo desde su creación en 2020. Comprometida con brindar adelantos de dinero de forma gratuita y con precios transparentes, IronHack Payments ha logrado una base de usuarios considerable.

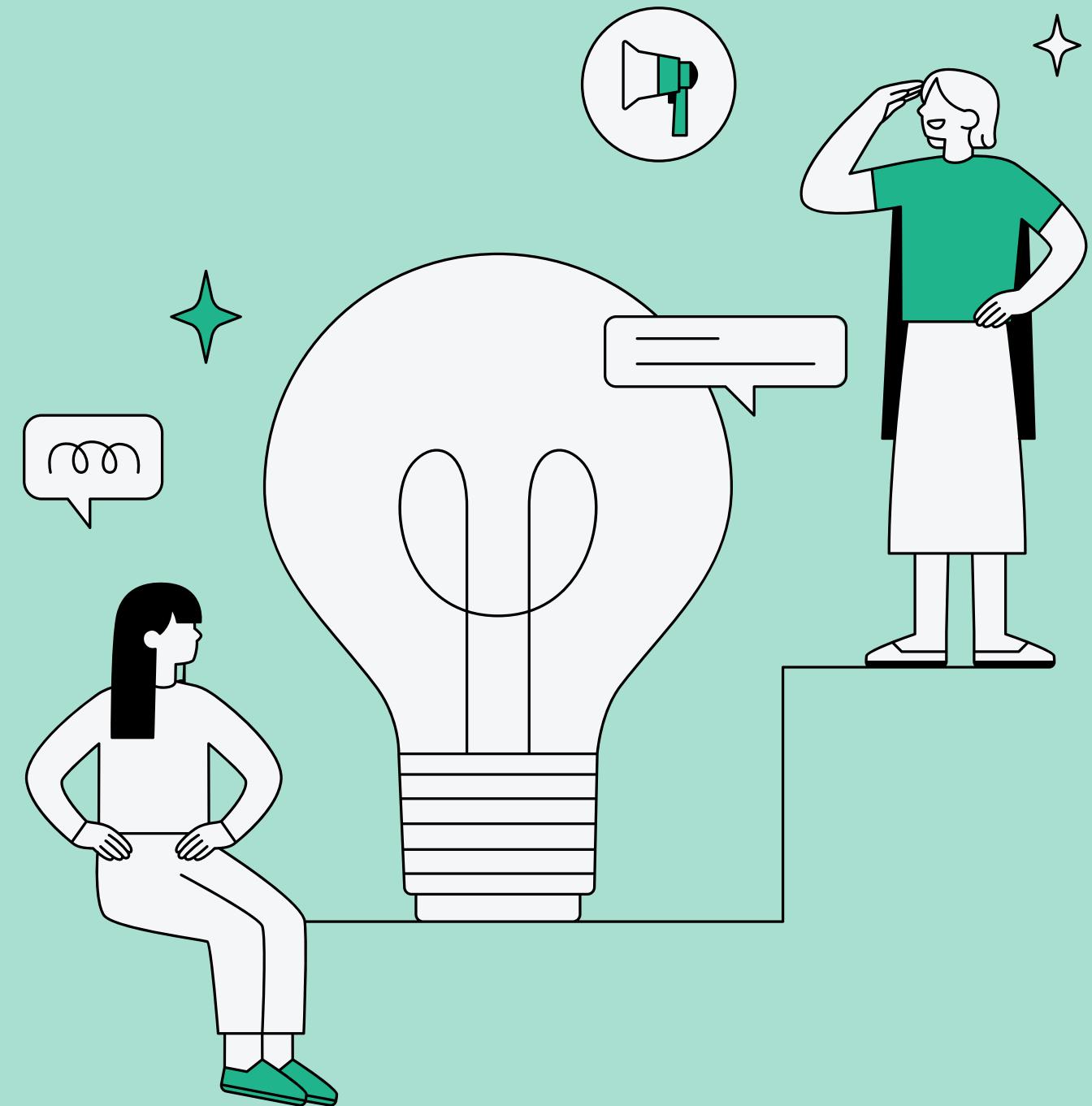
Como parte de su esfuerzo continuo por mejorar sus servicios y comprender el comportamiento de sus usuarios, IronHack Payments ha encargado un proyecto de análisis de cohortes.



# OBJECTIVOS

El objetivo principal es analizar el comportamiento de los usuarios y definirlos por el mes en que realizaron su primer adelanto de efectivo.

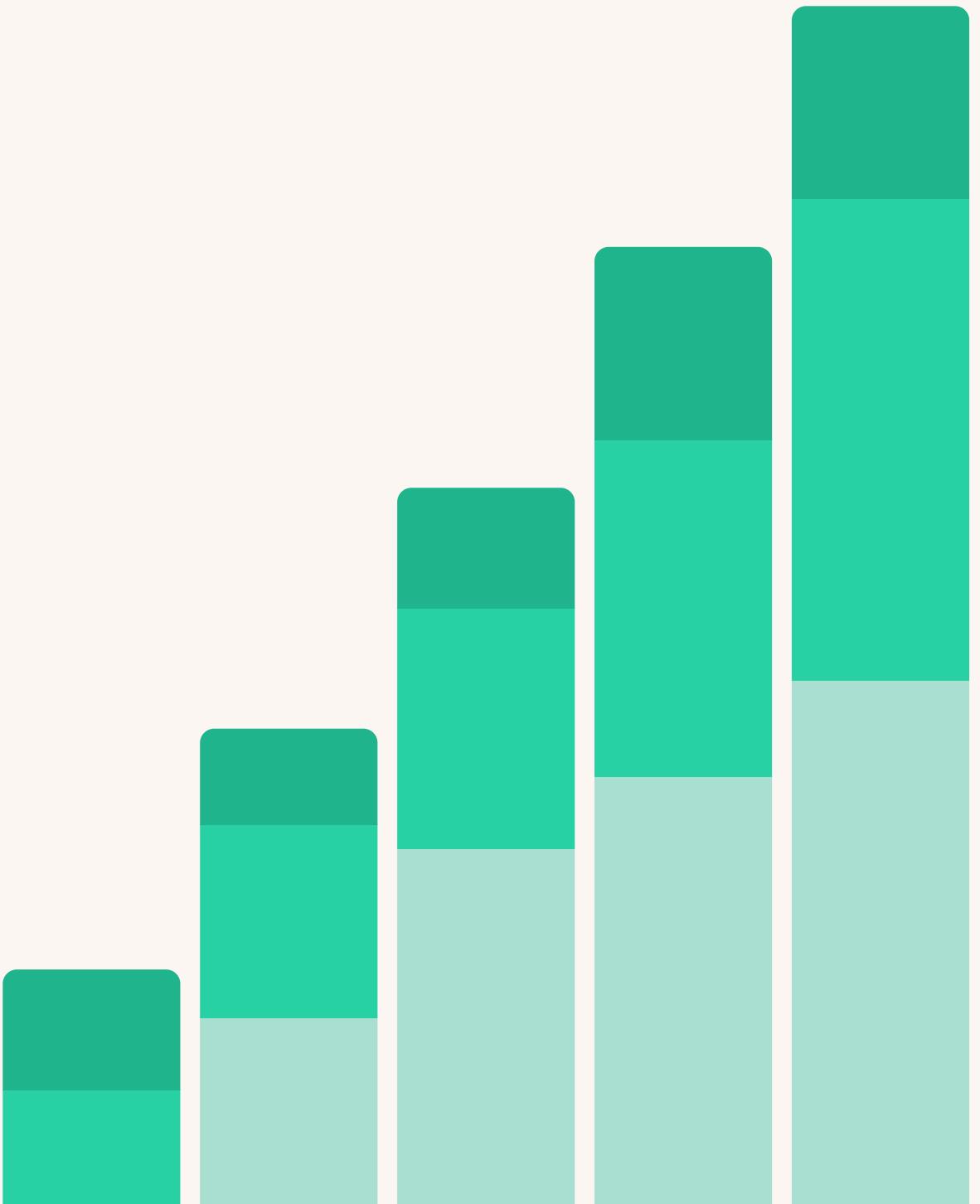
Harás un seguimiento de la evolución mensual de métricas clave, lo que permitirá a IronHack Payments obtener información valiosa sobre el comportamiento de los usuarios y el rendimiento de sus servicios financieros.



# METODOLOGÍA

Realizamos el análisis de cohortes utilizando Python, principalmente con la librería Pandas para la manipulación y el análisis de datos, seguido de la utilización de Matplot, Seaborn y tableau para mostrar gráficos.

Antes de realizar el análisis de cohortes, llevamos a cabo un EDA (análisis exploratorio de datos) para obtener una comprensión completa del conjunto de datos.



# DESAFIOS & Soluciones propuestas

1.

COMPRENDER EL  
PROYECTO EN EQUIPO

2.

COMENZAR A TRABAJAR  
JUNTOS LOS 4 PARA LOS  
PRIMEROS PASOS

EXPLORAR Y LIMPIAR

3.

DISTRIBUIR TAREAS

4.

UNIFICAR EL TRABAJO  
FINAL INDIVIDUAL PARA  
QUE SIGA UNA LINEA  
COHERENTE Y LÓGICA

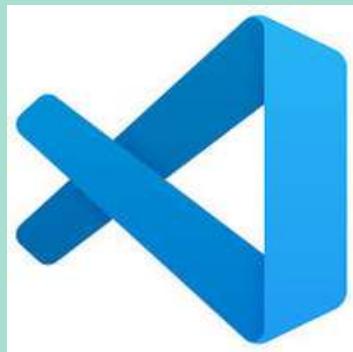
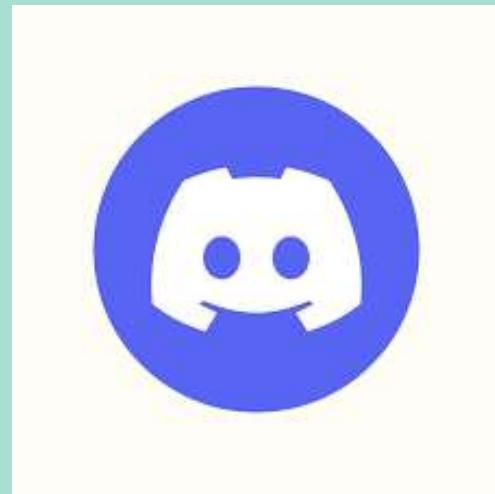
# OBSTACULOS IMPORTANTES

COMO AFRONTAR EL TRABAJO

FUSIONAR O UNIFICAR EL CODIGO



# HERRAMIENTAS CONSIDERADAS UTILIZADAS



# MÉTRICAS analizadas



1. FRECUENCIA DE USO DEL SERVICIO
2. TASAS DE INCIDENTES
3. INGRESOS TOTALES GENERADOS POR LAS COHORTE
4. USUARIOS ÚNICOS ACTIVOS POR COHORT
5. TIEMPO PROMEDIO DE PAGO SEGÚN INCIDENTE
6. INGRESOS PROMEDIO GENERADOS POR USUARIO ACTIVO

# CODIGO

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.patches import Patch
```

```
# =====
# CARGAR DATOS CSV Cash y Fees
# =====
df_cash = pd.read_csv('extract - cash request - data analyst.csv')
df_fees = pd.read_csv('extract - fees - data analyst -.csv')
```

Python

# CODIGO

```
df_cash.info()  
df_cash.head()
```

		<b>id</b>	<b>amount</b>	<b>status</b>	<b>created_at</b>	<b>updated_at</b>	<b>user_id</b>
0		5	100.0	rejected	2019-12-10 19:05:21.596873+00	2019-12-11 16:47:42.40783+00	804.0
1		70	100.0	rejected	2019-12-10 19:50:12.34778+00	2019-12-11 14:24:22.900054+00	231.0
2		7	100.0	rejected	2019-12-10 19:13:35.82546+00	2019-12-11 09:46:59.779773+00	191.0
3		10	99.0	rejected	2019-12-10 19:16:10.880172+00	2019-12-18 14:26:18.136163+00	761.0
4		1594	100.0	rejected	2020-05-06 09:59:38.877376+00	2020-05-07 09:21:55.34008+00	7686.0

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 23970 entries, 0 to 23969  
Data columns (total 16 columns):  
 #   Column           Non-Null Count  Dtype     
 ---  --              --              --  
 0   id              23970 non-null   int64    
 1   amount          23970 non-null   float64  
 2   status          23970 non-null   object    
 3   created_at      23970 non-null   object    
 4   updated_at      23970 non-null   object    
 5   user_id         21867 non-null   float64  
 6   moderated_at    16035 non-null   object    
 7   deleted_account_id  2104 non-null   float64  
 8   reimbursement_date  23970 non-null   object    
 9   cash_request_received_date  16289 non-null   object    
 10  money_back_date   16543 non-null   object    
 11  transfer_type    23970 non-null   object    
 12  send_at          16641 non-null   object    
 13  recovery_status   3330 non-null   object    
 14  reco_creation    3330 non-null   object    
 15  reco_last_update  3330 non-null   object    
 dtypes: float64(3), int64(1), object(12)  
memory usage: 2.9+ MB
```

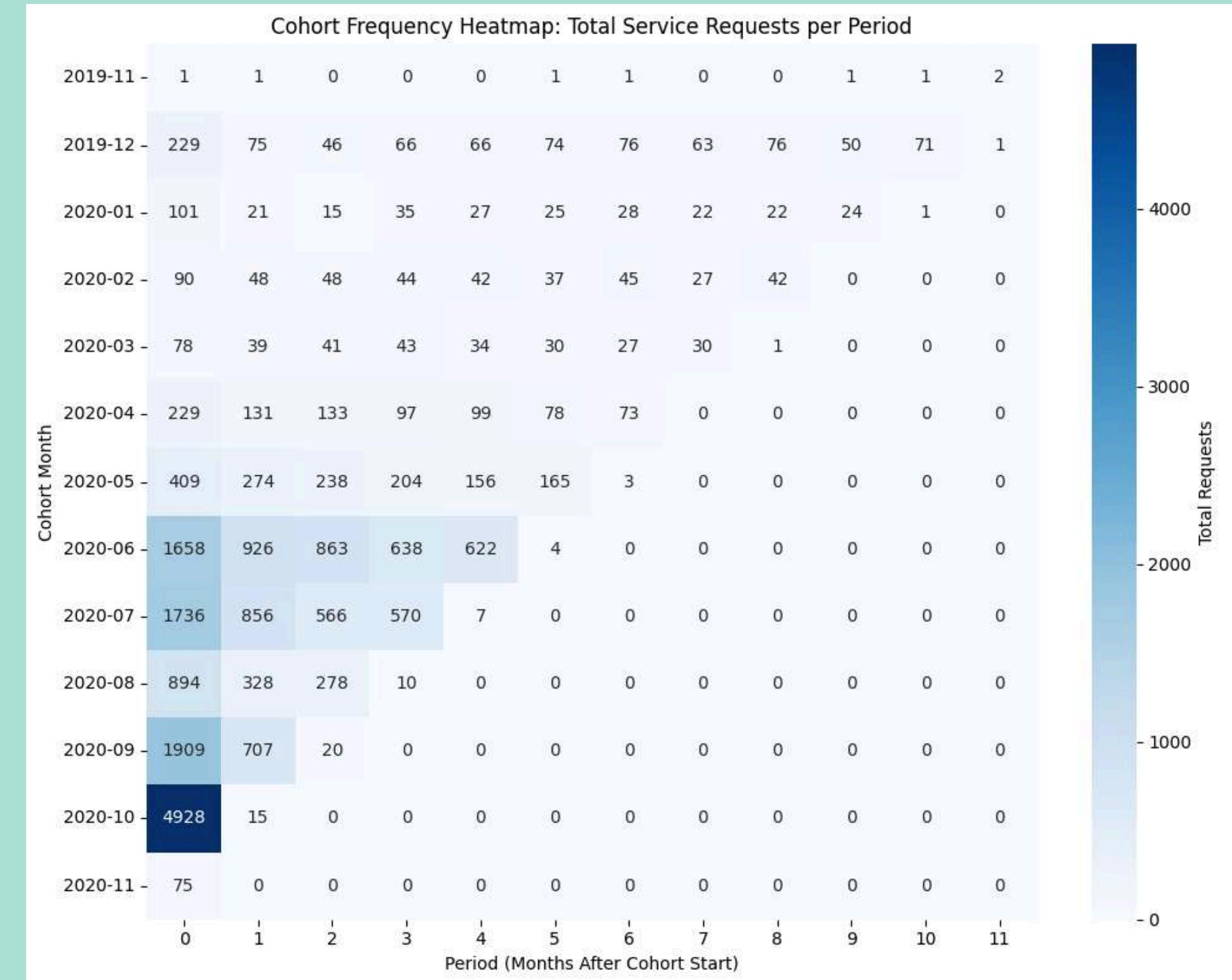
# PYTHON DEMO

# MÉTRICAS

# #M1

# Frecuencia de uso del servicio

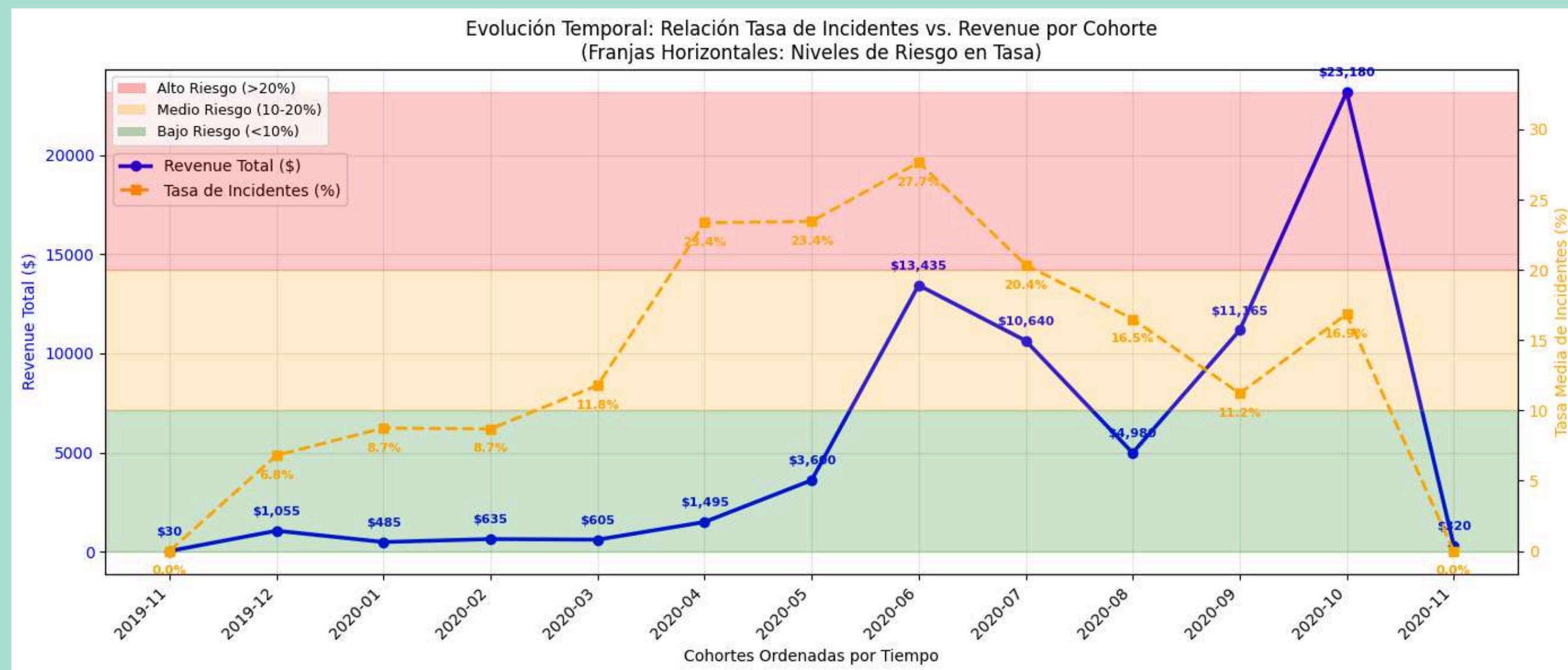
**Objetivo:** Analizar frecuencia de usuarios por solicitudes y cohorte utilizando el servicio IronHack Payments, para entender patrones de comportamiento recurrente.



# #M2 & #M3

## Tasa de Incidentes vs Revenue total

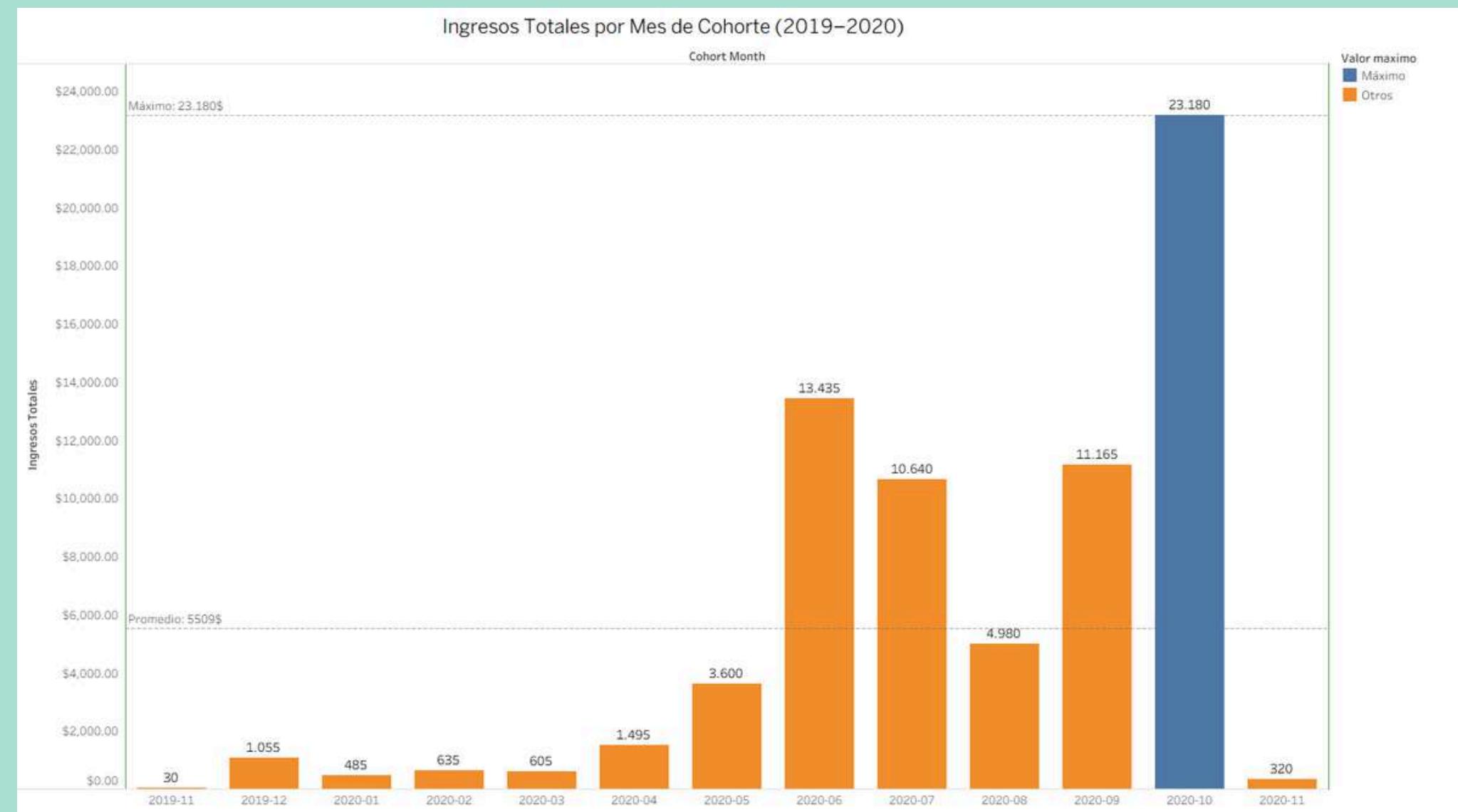
Objetivo: Calcular la tasa de incidentes, enfocándonos en problemas de pago, por cohorte y detectar variaciones entre ellas, con el fin de identificar riesgos y mejorar la retención.



# #M3

## Ingresos Generados por la Cohorte

**Objetivo:** Computar los ingresos totales generados por cada cohorte mes a mes, evaluando el impacto financiero del comportamiento de los usuarios y la rentabilidad de los servicios.

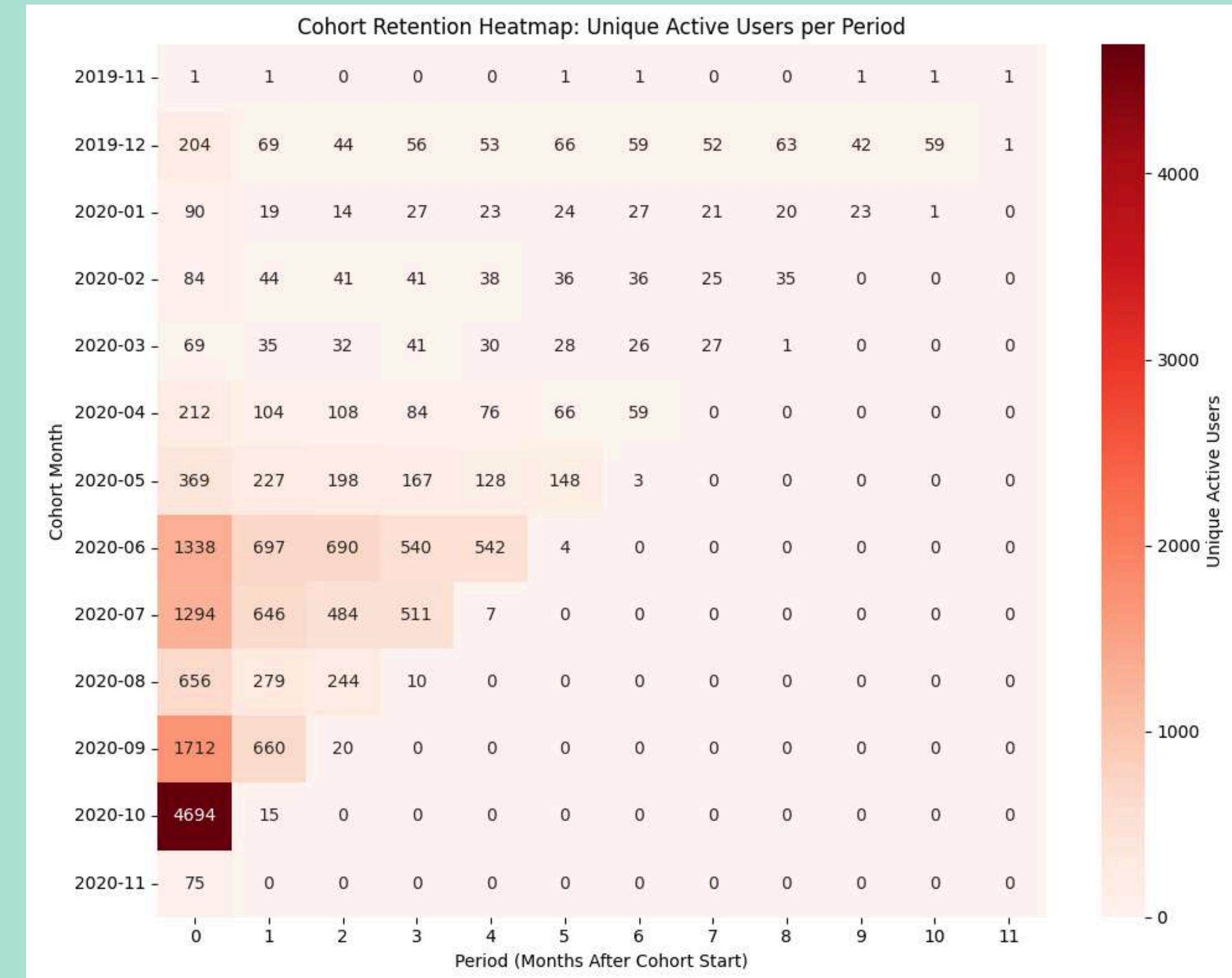


(Tableau - Gráfica)

# #Bonus (M1)

## Retención por usuarios únicos

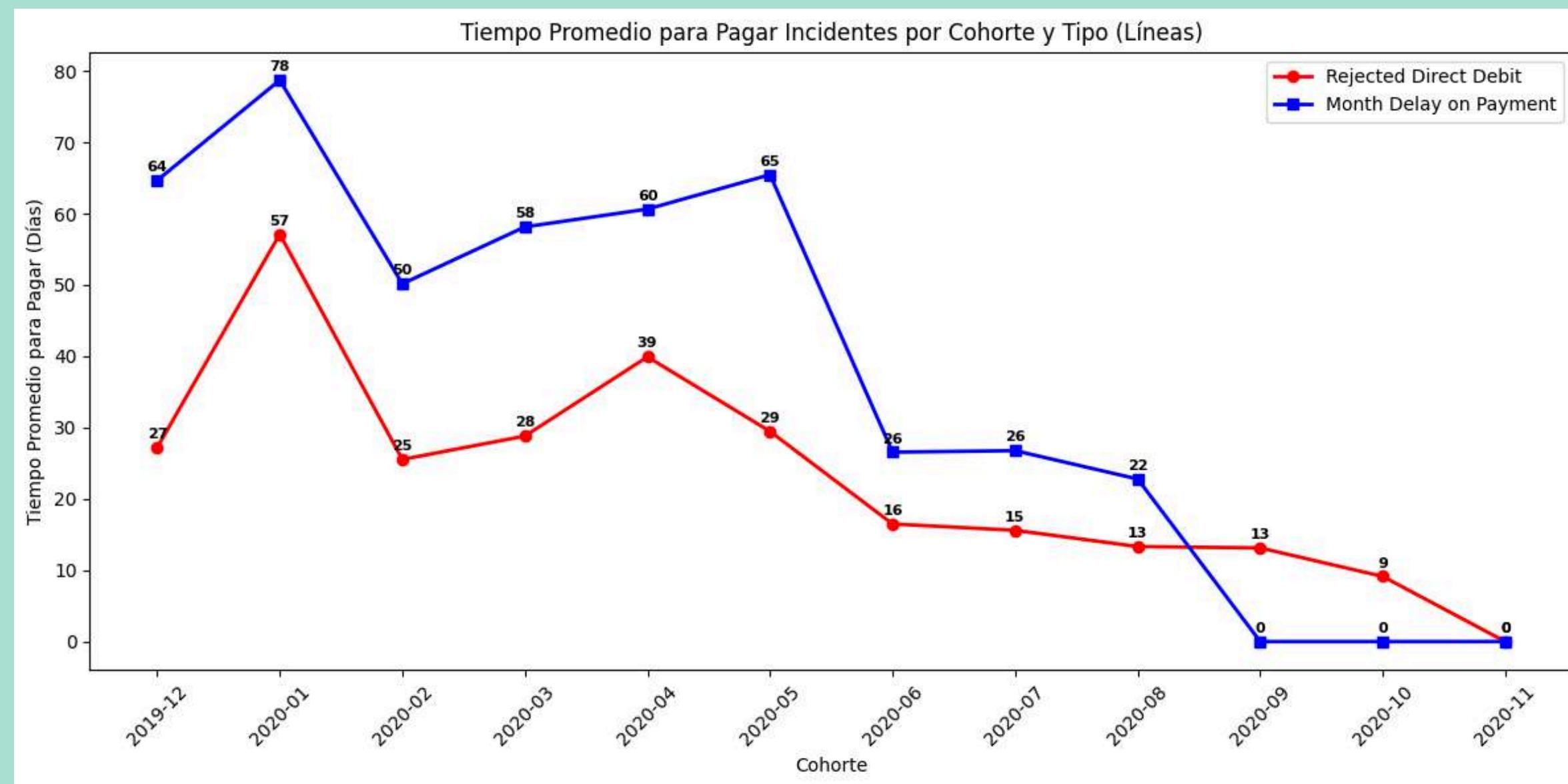
**Objetivo:** Analizar los usuarios únicos por períodos, evaluando así la lealtad y la capacidad de IronHack Payments para mantener a sus clientes activos a lo largo del tiempo.



# #Bonus (M2)

## Tiempo promedio de pago según Incidente

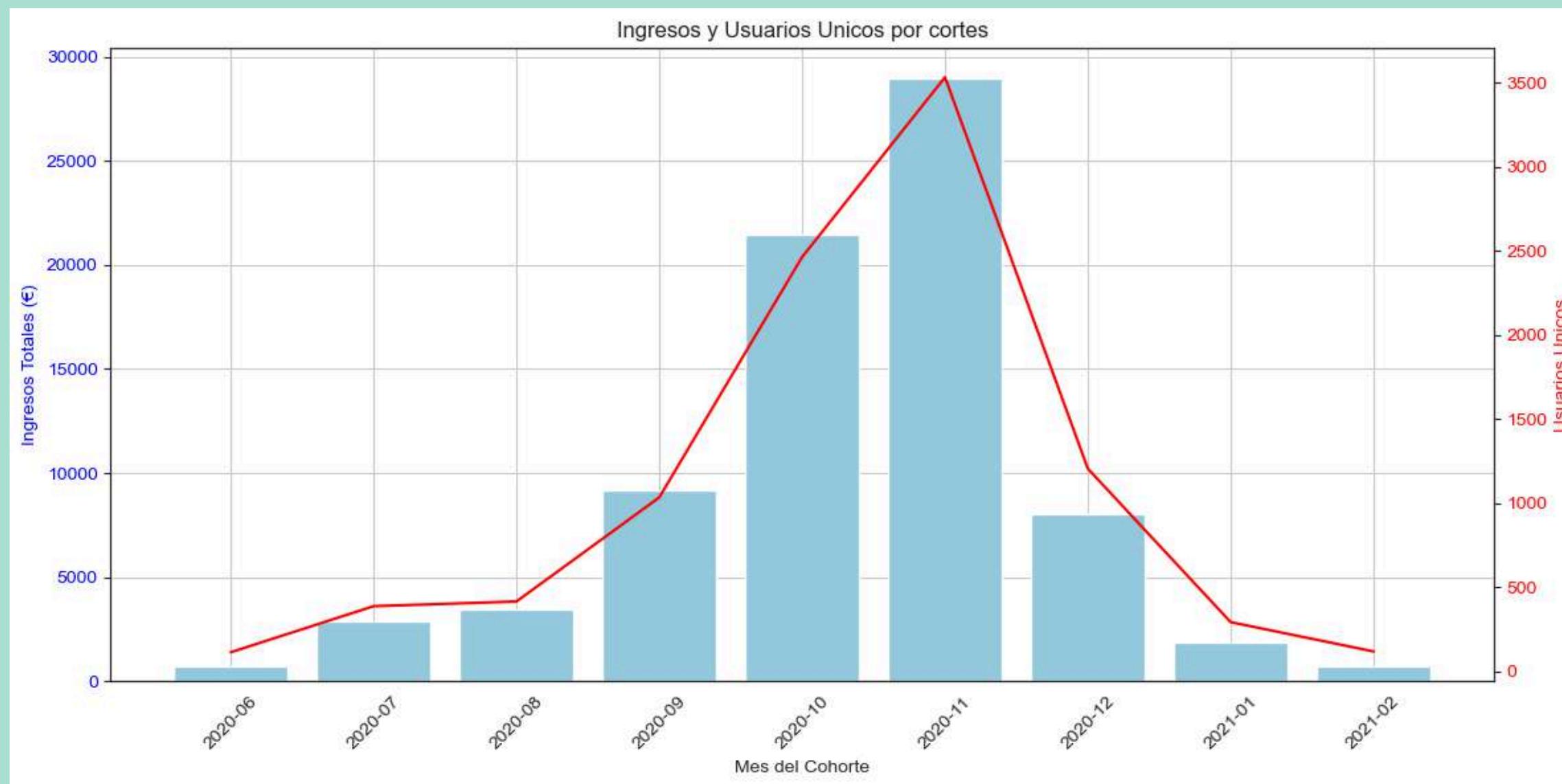
Objetivo: cuánto tardan en pagar según tipo de incidente y cómo repercute esto en el revenue futuro



# #Bonus (M3)

## Ingreso total (usuario único por mes)

Objetivo: Computar los ingresos totales generados por cada cohorte mes a mes, evaluando el impacto financiero del comportamiento por usuarios únicos y la rentabilidad de los servicios.



# CONCLUSIONES



# iMUCHAS GRACIAS!

Presentado por:

Alvaro Reyes  
Angel Vergara  
David Arenas  
Wilmer Acosta