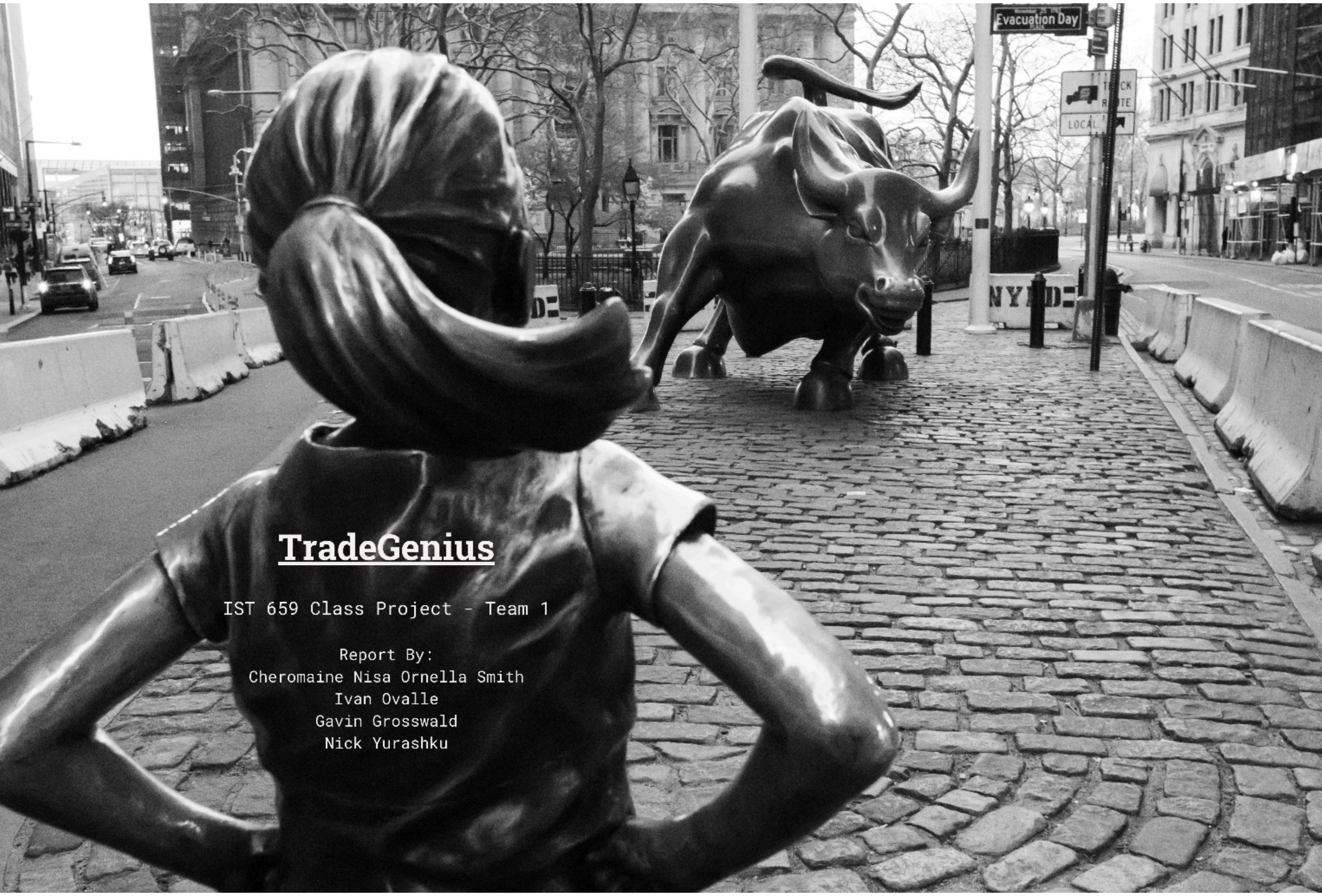


TradeGenius

IST 659 Class Project - Team 1

Report By:
Cheromaine Nisa Ornella Smith
Ivan Ovalle
Gavin Grosswald
Nick Yurashku

“Never ask anyone for their opinion, forecast, or recommendation. Just ask them what they have - or don't have - in their portfolio.”

- Nassim Nicholas Taleb

TradeGenius

TradeGenius Narrative Overview

The complex financial services landscape and industry is often made more complex than it needs to be in order to maintain an edge against the everyday investor. This is why there should be a push for innovation which can drive a transformative change. TradeGenius is focused on redefining the boundaries of financial research and data analysis, and revolutionizing research methodologies within the financial sector.

TradeGenius was founded on the idea of creating a cutting-edge research tool tailored for the unique needs of hedge funds (CTA's), venture capital firms (VC's), esteemed financial publications (WSJ, FT) and a space for retail traders to have a fair edge in the market. This tool is not just a collection of data like platforms such as Yahoo Finance; it is an intuitive platform designed to provide insights into market trends and help generate groundbreaking trading ideas through TradeGenius' unique white glove service in financial research.

The TradeGenius platform is a state-of-the-art research tool that delivers a game-changer for all institutions and retail alike. At its core, the platform's functionality lies in delivering real-time insights into market trends, economic shifts (micro and macro), and offering a foundation for generating informed trading ideas. TradeGenius is focused on delivering white-glove customer service and high-end research to CTA's, Venture Firms and retail investors. We pride ourselves in democratizing financial research for all no matter the user's background.

The solution lies in a strategically designed database architecture consisting of key tables outlined later in the report. Each table is data engineered to capture and analyze data ensuring users are equipped with holistic and up-to-date information to drive investment decisions. As the platform grows, the TradeGenius team will also be growing key tables and pertinent data on the platform.

Platform accessibility is crucial to TradeGenius' success and business goals. To achieve widespread reach, TradeGenius will operate under a subscription-based model - an approach that aligns with our vision to 'democratize' access to high-quality financial research. Retail traders can now save big by purchasing a subscription instead of spending thousands of dollars on expensive books and research articles - typically not available to the public. This model not only ensures inclusivity but also highlights our commitment to making impactful contributions to the democratization of financial knowledge.

TradeGenius Business Requirements

- Each user can get access to different research for each company and create watchlists with more than one company.
- User accounts have different privileges depending on status of whether they are retail, CTA, journalist, etc.
- TradeGenius will be able to do an API integration with user’s brokerage where trades can be programmed based on ‘event signals’, price movement and more.

TradeGenius Stakeholders

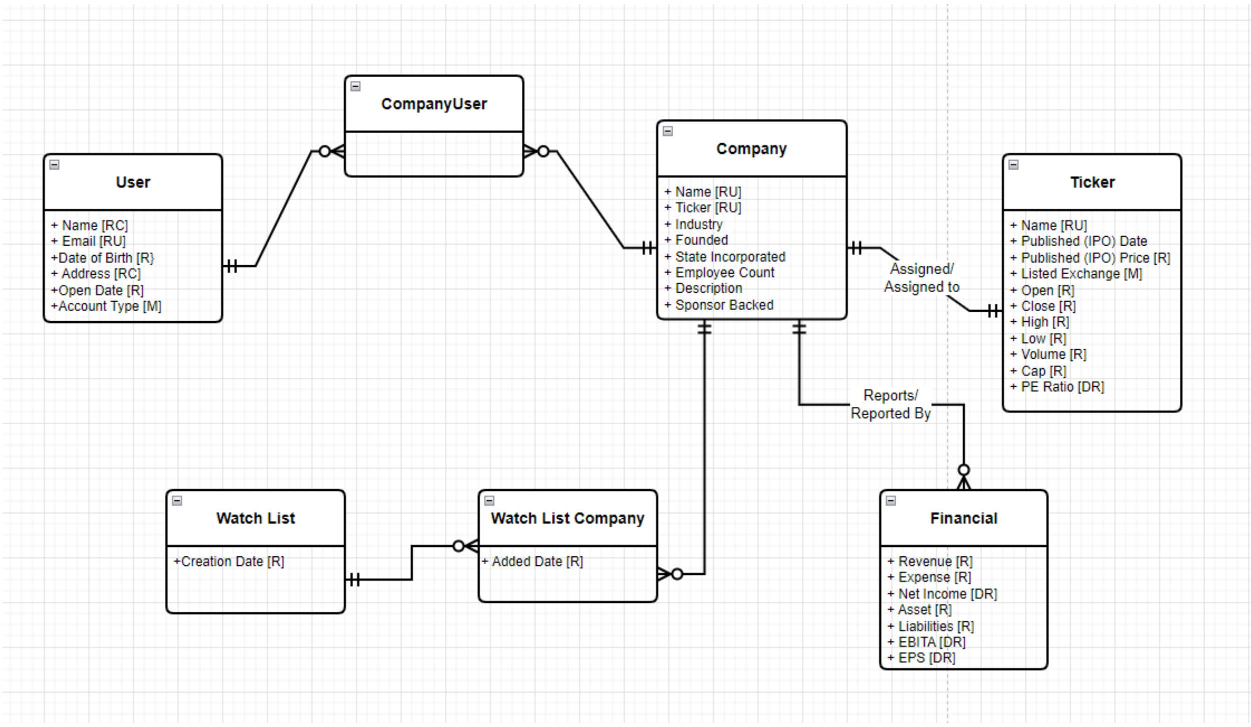
- Management Team: Responsible for strategic decisions, overseeing operations, and ensuring that the TradeGenius platform aligns with business goals of democratizing finance.
- Product Development Team: Includes software developers, database administrators, and IT specialists who build and maintain the platform.
- Data Analysts/Scientists: Analyze data to provide the promised TradeGenius research insights, improve user experience, and develop new features.
- Quality Assurance Team: Ensures the platform is reliable, has UI/UX that meets quality standards, and functions as intended.
- Marketing and Sales Team: Promotes the platform, identifies market needs, and attracts new users.
- Customer Support Team: Provides support to users, gathers feedback, and helps in resolving issues.
- Finance Department: Manages financial aspects, budgeting, and economic feasibility of the platform.
- Users/Clients: Individual investors, traders, or financial analysts who use the platform for research and decision-making.

- Investors and Shareholders: Provide capital and are interested in the profitability and growth of the platform.
- Regulatory Bodies: Ensure compliance with financial regulations and standards.
- Technology Partners and Vendors: Provide tools, technologies, or services essential for the platform’s operation.
- Academic Institutions: Might use the platform for educational purposes or collaborate on research.
- Media and Industry Analysts: Report on the platform's performance and influence public perception.
- General Public: Especially if the platform impacts broader financial markets or public investing trends.

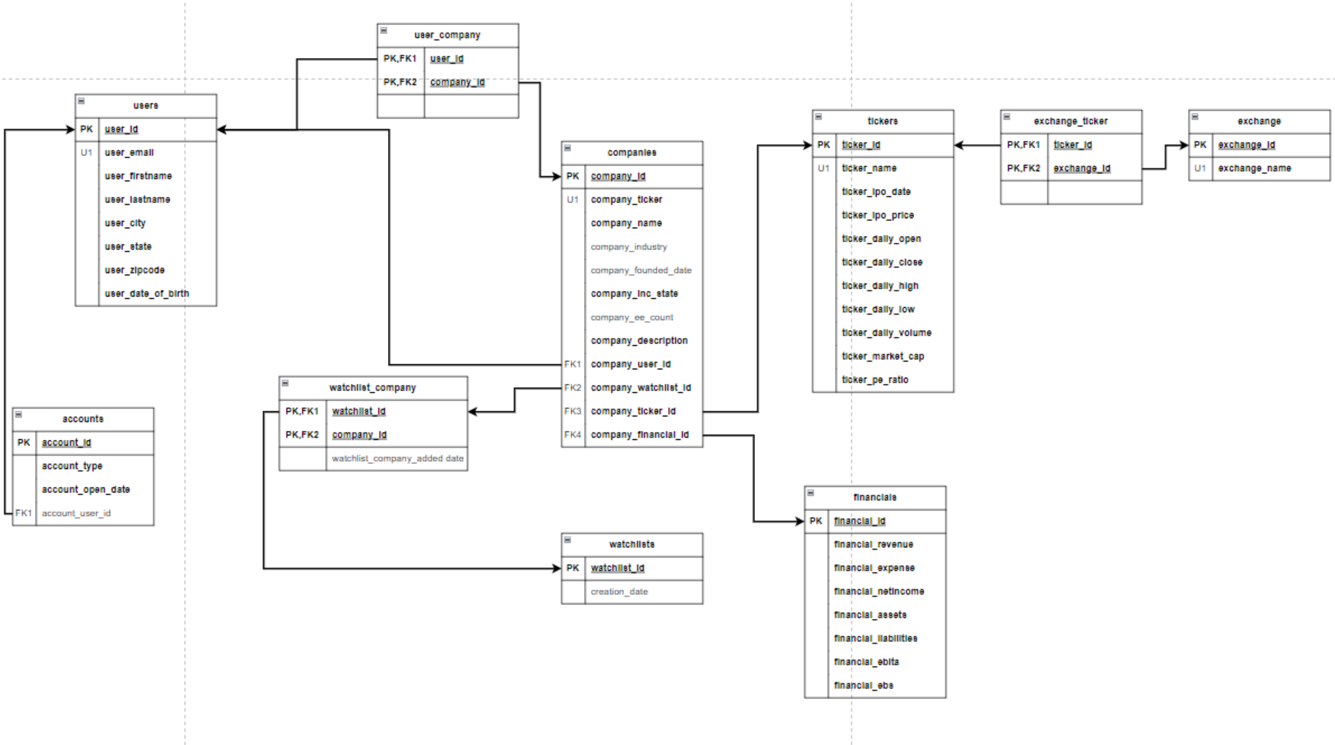
Glossary

CTA - “Commodity Trading Advisor” which is the US regulatory term for hedge funds.

TradeGenius Conceptual Model

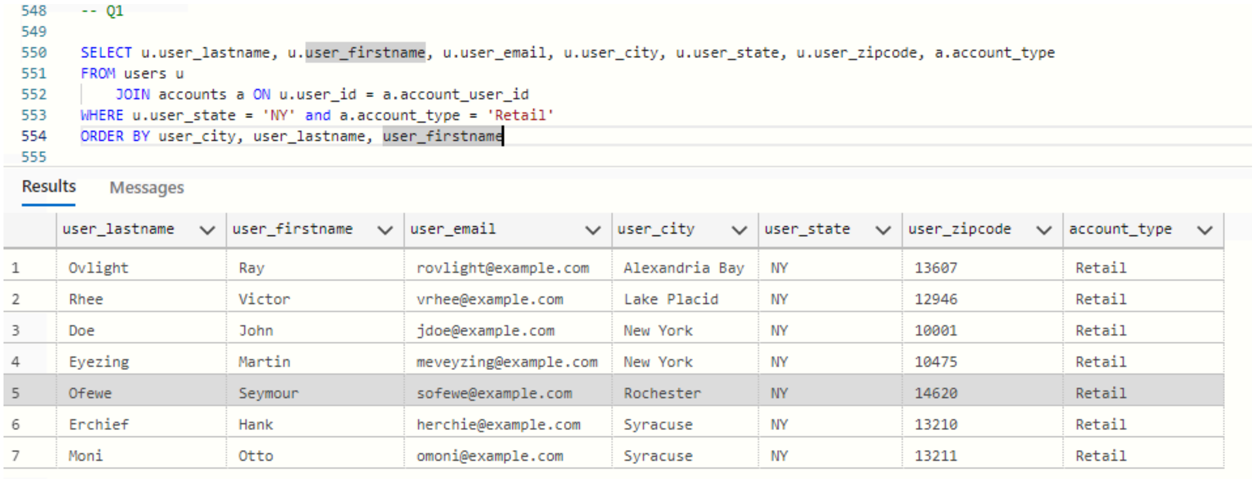


TradeGenius Physical Model



TradeGenius Business Questions

1. Find all the users from the state of New York with a Retail account. Print their names and emails along with their city, state, Zip code and account type. Sort by city, then by user's last /first name

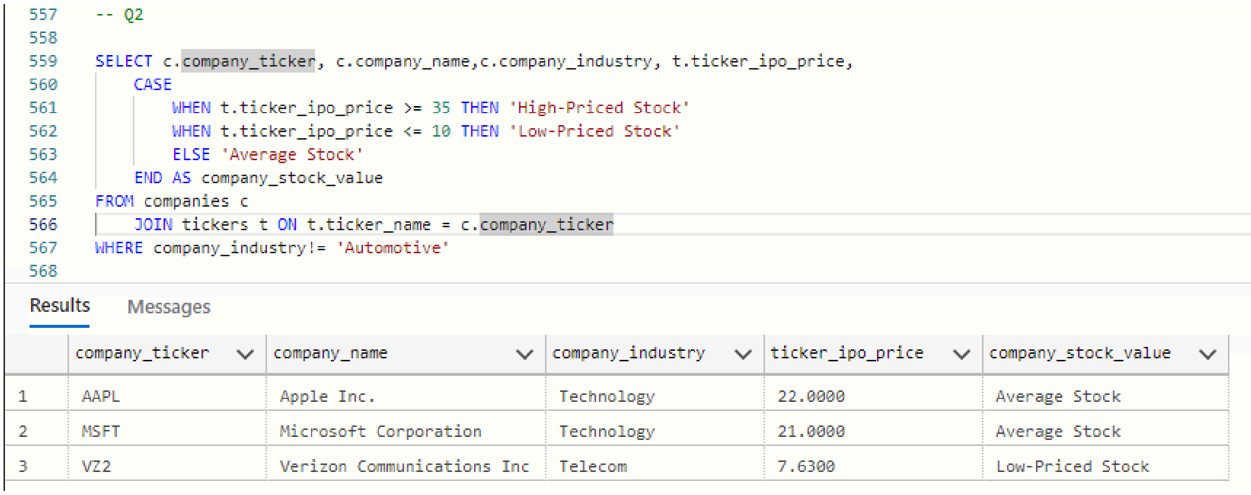


```
-- Q1

SELECT u.user_lastname, u.user_firstname, u.user_email, u.user_city, u.user_state, u.user_zipcode, a.account_type
FROM users u
    JOIN accounts a ON u.user_id = a.account_user_id
WHERE u.user_state = 'NY' and a.account_type = 'Retail'
ORDER BY user_city, user_lastname, user_firstname
```

	user_lastname	user_firstname	user_email	user_city	user_state	user_zipcode	account_type
1	Ovlight	Ray	rovlight@example.com	Alexandria Bay	NY	13607	Retail
2	Rhee	Victor	vrhee@example.com	Lake Placid	NY	12946	Retail
3	Doe	John	jdoe@example.com	New York	NY	10001	Retail
4	Eyezing	Martin	meveyzing@example.com	New York	NY	10475	Retail
5	Ofewe	Seymour	sofewe@example.com	Rochester	NY	14620	Retail
6	Erchief	Hank	herchie@example.com	Syracuse	NY	13210	Retail
7	Moni	Otto	omoni@example.com	Syracuse	NY	13211	Retail

2. Ticker categories. Include the ID, name, industry, and ticker IPO Price. Do not include items of industry "Automotive". Create a category column based on IPO price. When the

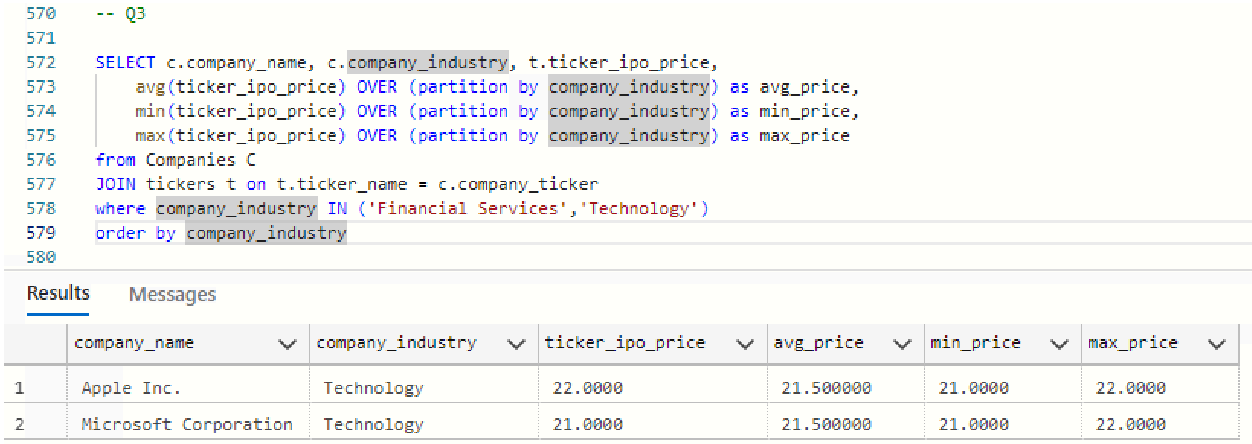
ticker price is 35 or more, it is a high-priced stock. When the ticker price is 10 or less, it is a low-priced stock. Everything else is an average stock.



```
-- Q2

SELECT c.company_ticker, c.company_name,c.company_industry, t.ticker_ipo_price,
    CASE
        WHEN t.ticker_ipo_price >= 35 THEN 'High-Priced Stock'
        WHEN t.ticker_ipo_price <= 10 THEN 'Low-Priced Stock'
        ELSE 'Average Stock'
    END AS company_stock_value
FROM companies c
    JOIN tickers t ON t.ticker_name = c.company_ticker
WHERE company_industry!= 'Automotive'
```

	company_ticker	company_name	company_industry	ticker_ipo_price	company_stock_value
1	AAPL	Apple Inc.	Technology	22.0000	Average Stock
2	MSFT	Microsoft Corporation	Technology	21.0000	Average Stock
3	VZ2	Verizon Communications Inc	Telecom	7.6300	Low-Priced Stock

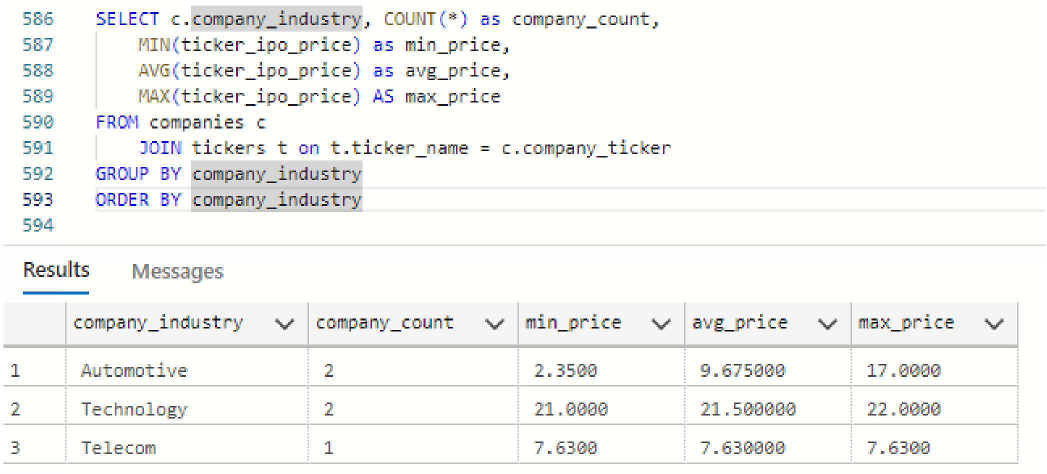
3. Perform an analysis of companies in the “Technology” and “Financial Services” account types. For each company, display the name, industry type, and employee count. Include the minimum, maximum, and average ipo price over each industry type so that the current ipo price can be compared to these values.



```
-- Q3

SELECT c.company_name, c.company_industry, t.ticker_ipo_price,
    avg(ticker_ipo_price) OVER (partition by company_industry) as avg_price,
    min(ticker_ipo_price) OVER (partition by company_industry) as min_price,
    max(ticker_ipo_price) OVER (partition by company_industry) as max_price
from Companies C
JOIN tickers t on t.ticker_name = c.company_ticker
where company_industry IN ('Financial Services','Technology')
order by company_industry
```

	company_name	company_industry	ticker_ipo_price	avg_price	min_price	max_price
1	Apple Inc.	Technology	22.0000	21.500000	21.0000	22.0000
2	Microsoft Corporation	Technology	21.0000	21.500000	21.0000	22.0000

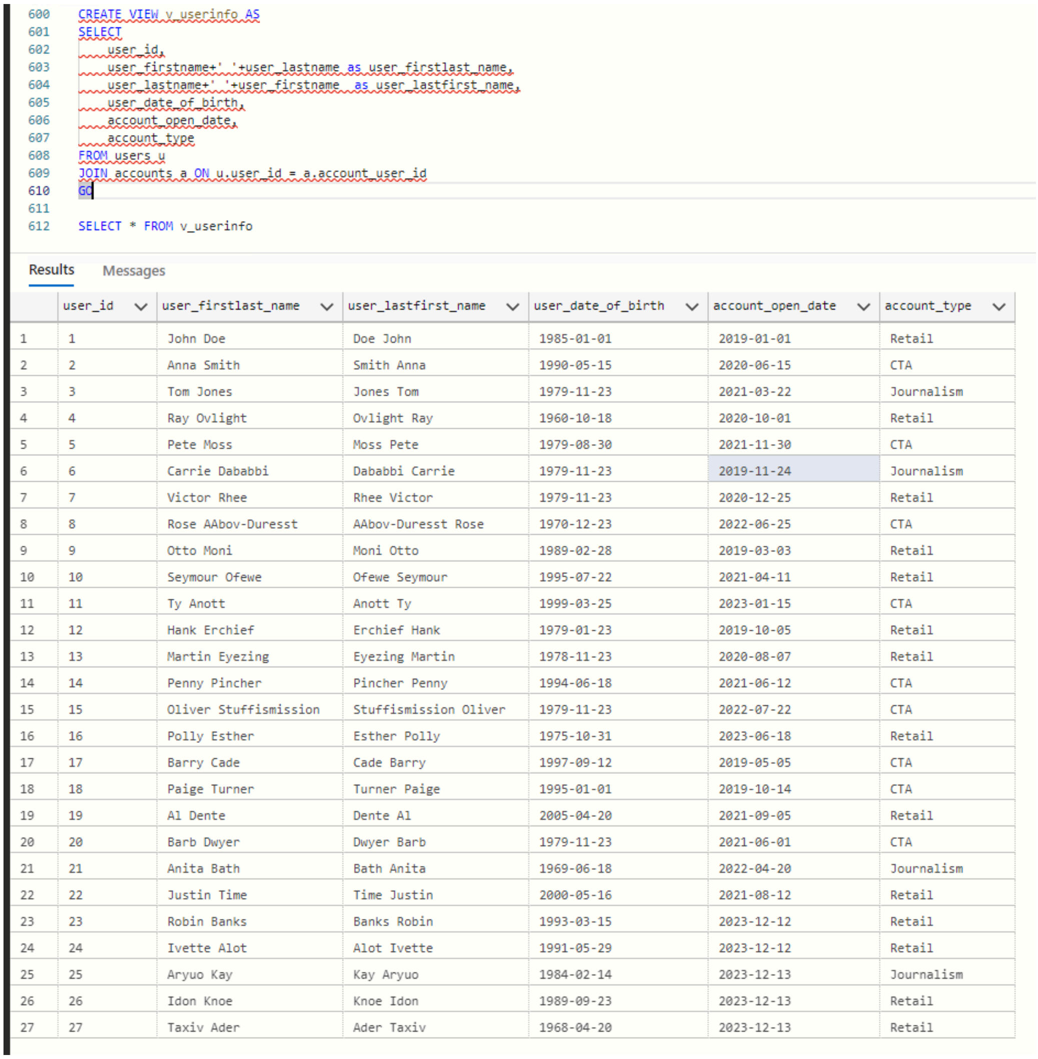
4. How many industry types are there? For each industry type, provide the count of companies in that type and the minimum, average, and maximum ipo prices for that type. Sort the output by item type.



```sql
SELECT c.company_industry, COUNT(*) as company_count,
    MIN(ticker_ipo_price) as min_price,
    AVG(ticker_ipo_price) as avg_price,
    MAX(ticker_ipo_price) AS max_price
FROM companies c
    JOIN tickers t on t.ticker_name = c.company_ticker
GROUP BY company_industry
ORDER BY company_industry
```

	company_industry	company_count	min_price	avg_price	max_price
1	Automotive	2	2.3500	9.675000	17.0000
2	Technology	2	21.0000	21.500000	22.0000
3	Telecom	1	7.6300	7.630000	7.6300

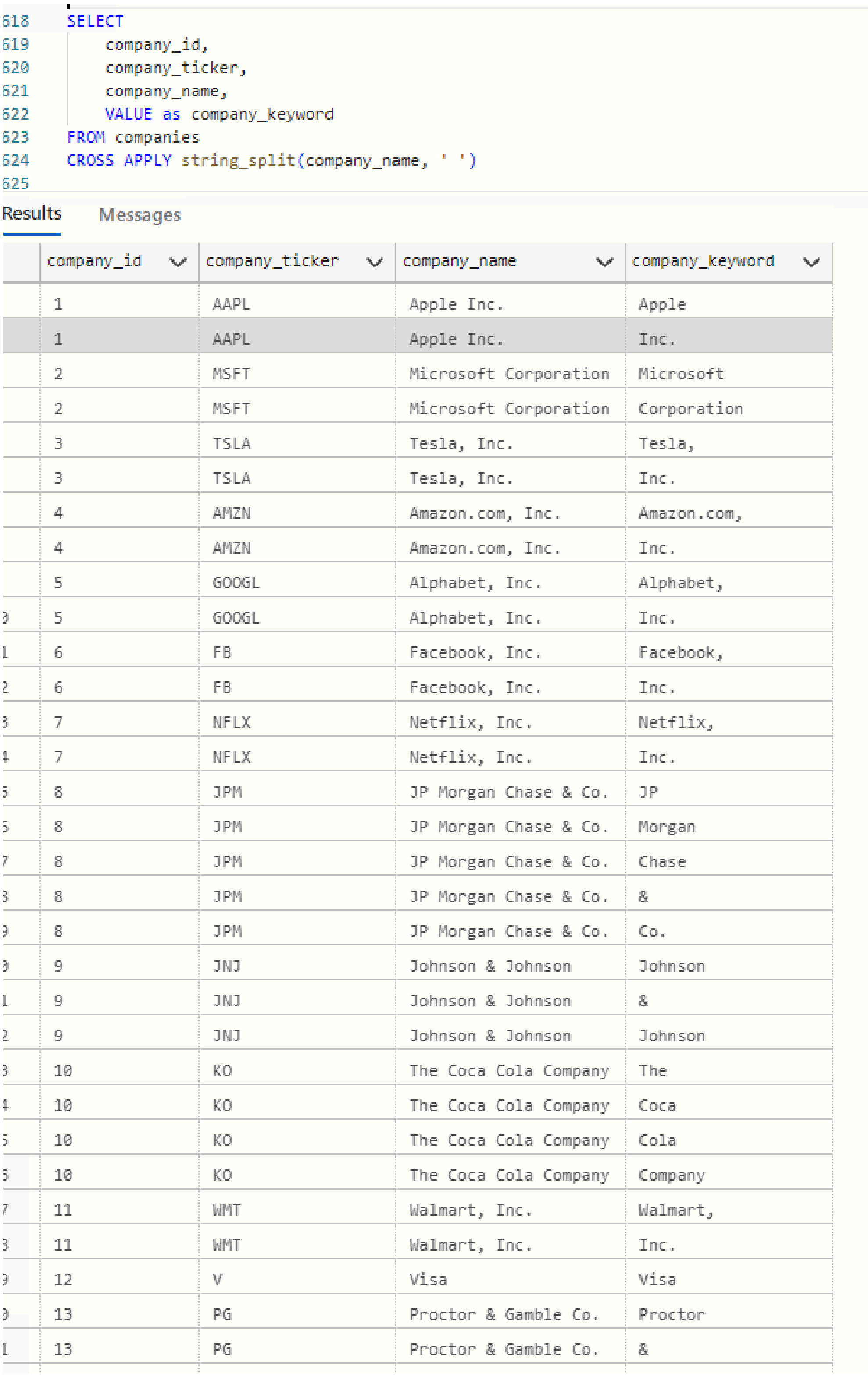
5. Create a view called v_userinfo that displays the user_id, user name (first last), user name (last, first), date of birth, account creation date and account type.



```sql
CREATE VIEW v_userinfo AS
SELECT
    user_id,
    user_firstname+' '+user_lastname as user_firstlast_name,
    user_lastname+' '+user_firstname  as user_lastfirst_name,
    user_date_of_birth,
    account_open_date,
    account_type
FROM users u
JOIN accounts a ON u.user_id = a.account_user_id
GO

SELECT * FROM v_userinfo
```

	user_id	user_firstlast_name	user_lastfirst_name	user_date_of_birth	account_open_date	account_type
1	1	John Doe	Doe John	1985-01-01	2019-01-01	Retail
2	2	Anna Smith	Smith Anna	1990-05-15	2020-06-15	CTA
3	3	Tom Jones	Jones Tom	1979-11-23	2021-03-22	Journalism
4	4	Ray Ovlight	Ovlight Ray	1960-10-18	2020-10-01	Retail
5	5	Pete Moss	Moss Pete	1979-08-30	2021-11-30	CTA
6	6	Carrie Dababbi	Dababbi Carrie	1979-11-23	2019-11-24	Journalism
7	7	Victor Rhee	Rhee Victor	1979-11-23	2020-12-25	Retail
8	8	Rose AAbov-Duresst	AAbov-Duresst Rose	1970-12-23	2022-06-25	CTA
9	9	Otto Moni	Moni Otto	1989-02-28	2019-03-03	Retail
10	10	Seymour Ofewe	Ofewe Seymour	1995-07-22	2021-04-11	Retail
11	11	Ty Anott	Anott Ty	1999-03-25	2023-01-15	CTA
12	12	Hank Erchief	Erchief Hank	1979-01-23	2019-10-05	Retail
13	13	Martin Eyezing	Eyezing Martin	1978-11-23	2020-08-07	Retail
14	14	Penny Pincher	Pincher Penny	1994-06-18	2021-06-12	CTA
15	15	Oliver Stuffismission	Stuffismission Oliver	1979-11-23	2022-07-22	CTA
16	16	Polly Esther	Esther Polly	1975-10-31	2023-06-18	Retail
17	17	Barry Cade	Cade Barry	1997-09-12	2019-05-05	CTA
18	18	Paige Turner	Turner Paige	1995-01-01	2019-10-14	CTA
19	19	Al Dente	Dente Al	2005-04-20	2021-09-05	Retail
20	20	Barb Dwyer	Dwyer Barb	1979-11-23	2021-06-01	CTA
21	21	Anita Bath	Bath Anita	1969-06-18	2022-04-20	Journalism
22	22	Justin Time	Time Justin	2000-05-16	2021-08-12	Retail
23	23	Robin Banks	Banks Robin	1993-03-15	2023-12-12	Retail
24	24	Ivette Alot	Alot Ivette	1991-05-29	2023-12-12	Retail
25	25	Aryuo Kay	Kay Aryuo	1984-02-14	2023-12-13	Journalism
26	26	Idon Knoe	Knoe Idon	1989-09-23	2023-12-13	Retail
27	27	Taxiv Ader	Ader Taxiv	1968-04-20	2023-12-13	Retail

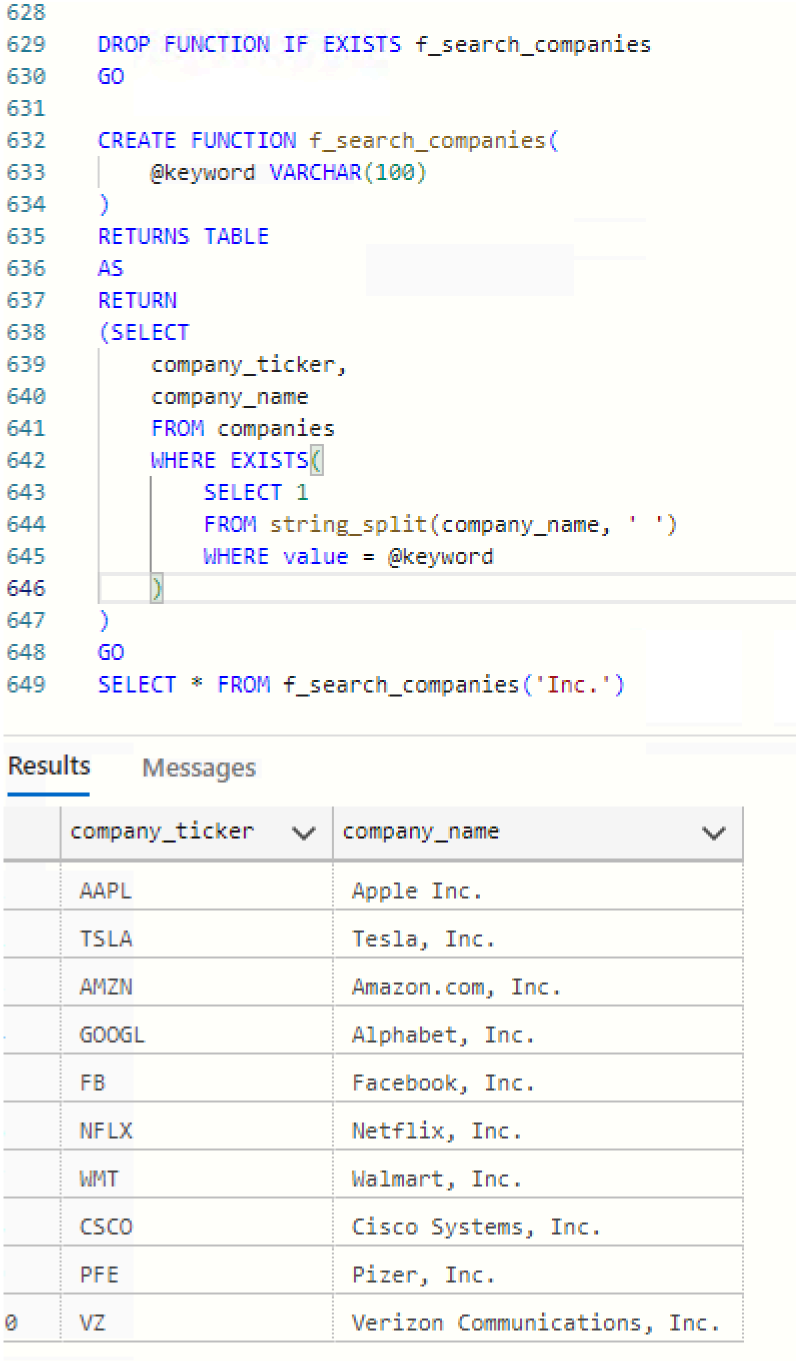
6. Write a query on the companies table so that the company_name is broken up into keywords, one per row.



```
SELECT
    company_id,
    company_ticker,
    company_name,
    VALUE as company_keyword
FROM companies
CROSS APPLY string_split(company_name, ' ')
```

company_id	company_ticker	company_name	company_keyword
1	AAPL	Apple Inc.	Apple
1	AAPL	Apple Inc.	Inc.
2	MSFT	Microsoft Corporation	Microsoft
2	MSFT	Microsoft Corporation	Corporation
3	TSLA	Tesla, Inc.	Tesla,
3	TSLA	Tesla, Inc.	Inc.
4	AMZN	Amazon.com, Inc.	Amazon.com,
4	AMZN	Amazon.com, Inc.	Inc.
5	GOOGL	Alphabet, Inc.	Alphabet,
5	GOOGL	Alphabet, Inc.	Inc.
6	FB	Facebook, Inc.	Facebook,
6	FB	Facebook, Inc.	Inc.
7	NFLX	Netflix, Inc.	Netflix,
7	NFLX	Netflix, Inc.	Inc.
8	JPM	JP Morgan Chase & Co.	JP
8	JPM	JP Morgan Chase & Co.	Morgan
8	JPM	JP Morgan Chase & Co.	Chase
8	JPM	JP Morgan Chase & Co.	&
8	JPM	JP Morgan Chase & Co.	Co.
9	JNJ	Johnson & Johnson	Johnson
9	JNJ	Johnson & Johnson	&
9	JNJ	Johnson & Johnson	Johnson
10	KO	The Coca Cola Company	The
10	KO	The Coca Cola Company	Coca
10	KO	The Coca Cola Company	Cola
10	KO	The Coca Cola Company	Company
11	WMT	Walmart, Inc.	Walmart,
11	WMT	Walmart, Inc.	Inc.
12	V	Visa	Visa
13	PG	Proctor & Gamble Co.	Proctor
13	PG	Proctor & Gamble Co.	&

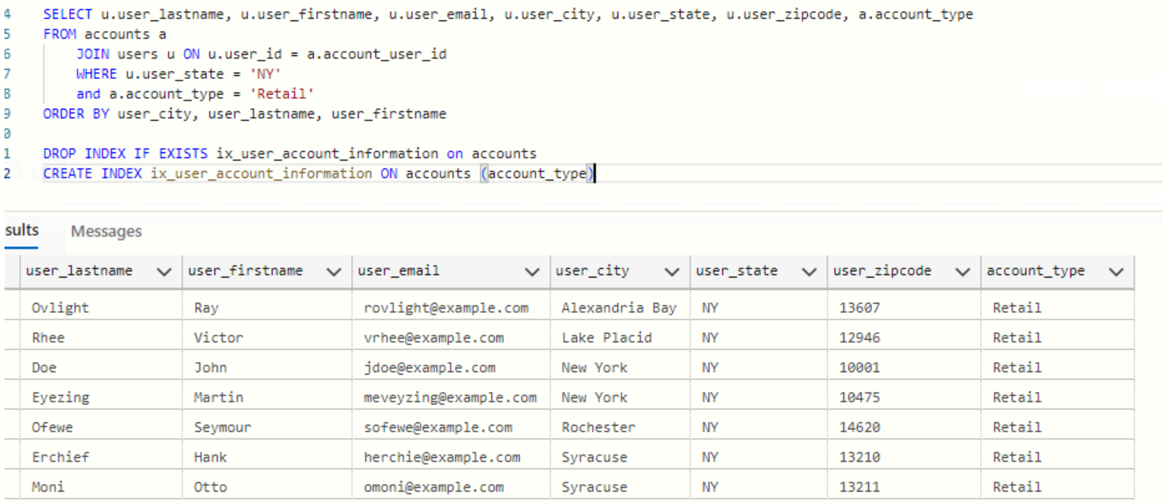
7. Using the query in 6, create a table-valued function f_search_companies that allows you to search the companies by keyword.



```
DROP FUNCTION IF EXISTS f_search_companies
GO

CREATE FUNCTION f_search_companies(
    @keyword VARCHAR(100)
)
RETURNS TABLE
AS
RETURN
(SELECT
    company_ticker,
    company_name
    FROM companies
    WHERE EXISTS(
        SELECT 1
        FROM string_split(company_name, ' ')
        WHERE value = @keyword
    )
)
GO
SELECT * FROM f_search_companies('Inc.')
```

	company_ticker	company_name
	AAPL	Apple Inc.
	TSLA	Tesla, Inc.
	AMZN	Amazon.com, Inc.
	GOOGL	Alphabet, Inc.
	FB	Facebook, Inc.
	NFLX	Netflix, Inc.
	WMT	Walmart, Inc.
	CSCO	Cisco Systems, Inc.
	PFE	Pizer, Inc.
0	VZ	Verizon Communications, Inc.

8. Write an index to improve performance of the query from question 1:

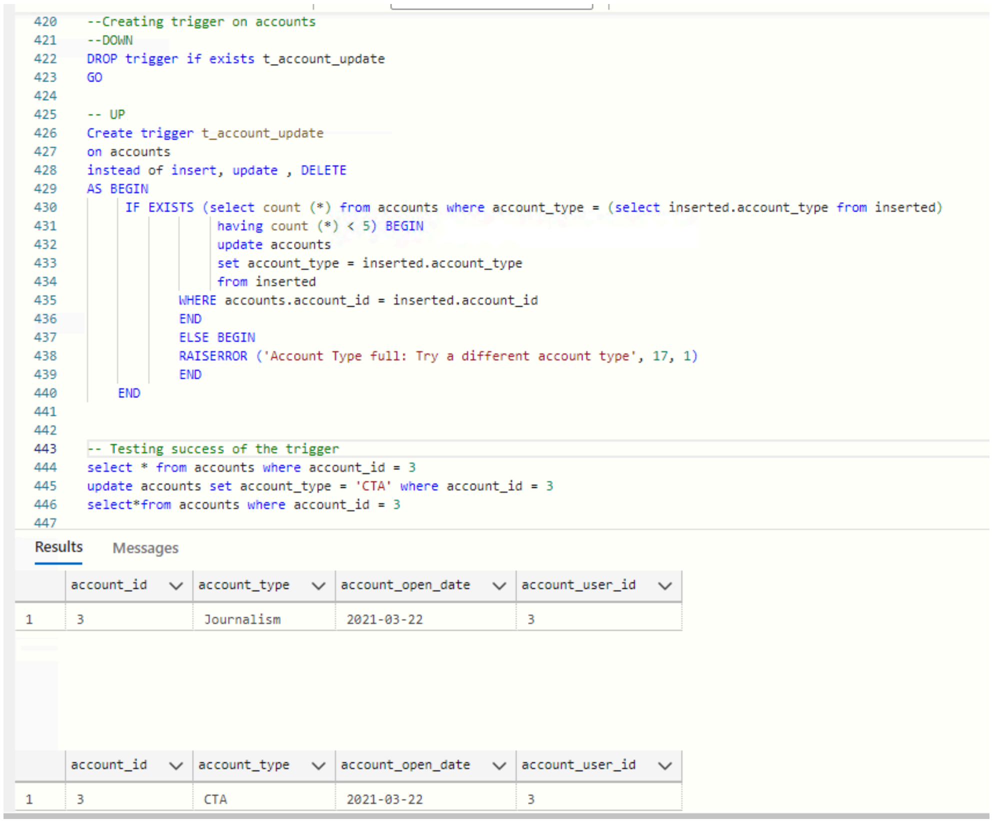


```
SELECT u.user_lastname, u.user_firstname, u.user_email, u.user_city, u.user_state, u.user_zipcode, a.account_type
FROM accounts a
    JOIN users u ON u.user_id = a.account_user_id
    WHERE u.user_state = 'NY'
    and a.account_type = 'Retail'
ORDER BY user_city, user_lastname, user_firstname

DROP INDEX IF EXISTS ix_user_account_information on accounts
CREATE INDEX ix_user_account_information ON accounts (account_type)
```

user_lastname	user_firstname	user_email	user_city	user_state	user_zipcode	account_type
Ovlight	Ray	rovlight@example.com	Alexandria Bay	NY	13607	Retail
Rhee	Victor	vrhee@example.com	Lake Placid	NY	12946	Retail
Doe	John	jdoe@example.com	New York	NY	10001	Retail
Eyezing	Martin	meveyzing@example.com	New York	NY	10475	Retail
Ofewe	Seymour	sofewe@example.com	Rochester	NY	14620	Retail
Erchief	Hank	herchie@example.com	Syracuse	NY	13210	Retail
Moni	Otto	omoni@example.com	Syracuse	NY	13211	Retail

9. Assuming there's a conceptual data requirement that needs to be met, for example no more than 5 accounts can be under the same account type for this platform. Write data logic using an instead-of-trigger to do this.

When the update is successfully completed:

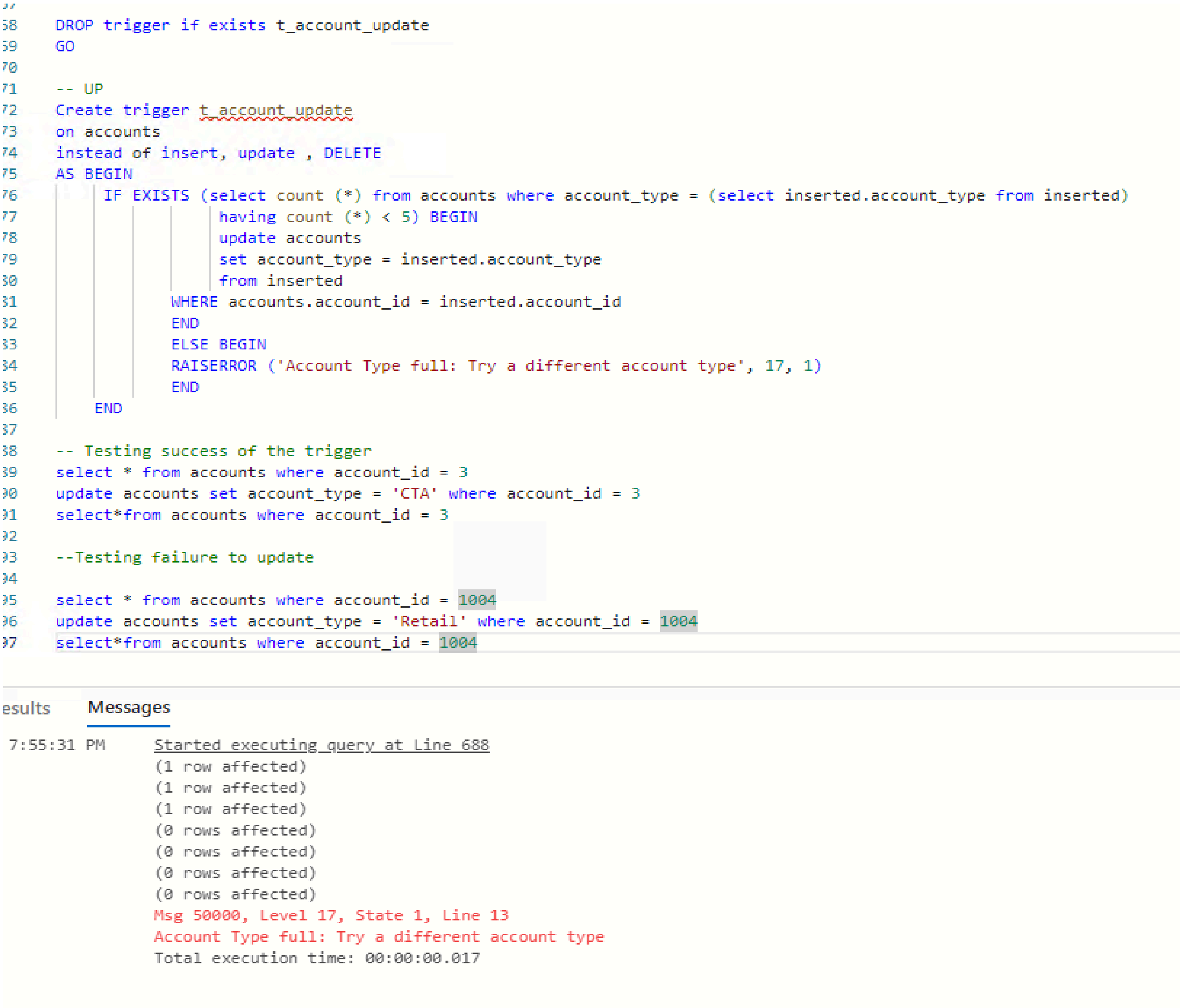


```
--Creating trigger on accounts
--DOWN
DROP trigger if exists t_account_update
GO

-- UP
Create trigger t_account_update
on accounts
instead of insert, update , DELETE
AS BEGIN
    IF EXISTS (select count (*) from accounts where account_type = (select inserted.account_type from inserted)
              having count (*) < 5) BEGIN
              update accounts
              set account_type = inserted.account_type
              from inserted
         WHERE accounts.account_id = inserted.account_id
         END
         ELSE BEGIN
         RAISERROR ('Account Type full: Try a different account type', 17, 1)
         END
    END


-- Testing success of the trigger
select * from accounts where account_id = 3
update accounts set account_type = 'CTA' where account_id = 3
select*from accounts where account_id = 3
```

	account_id	account_type	account_open_date	account_user_id
1	3	Journalism	2021-03-22	3

	account_id	account_type	account_open_date	account_user_id
1	3	CTA	2021-03-22	3

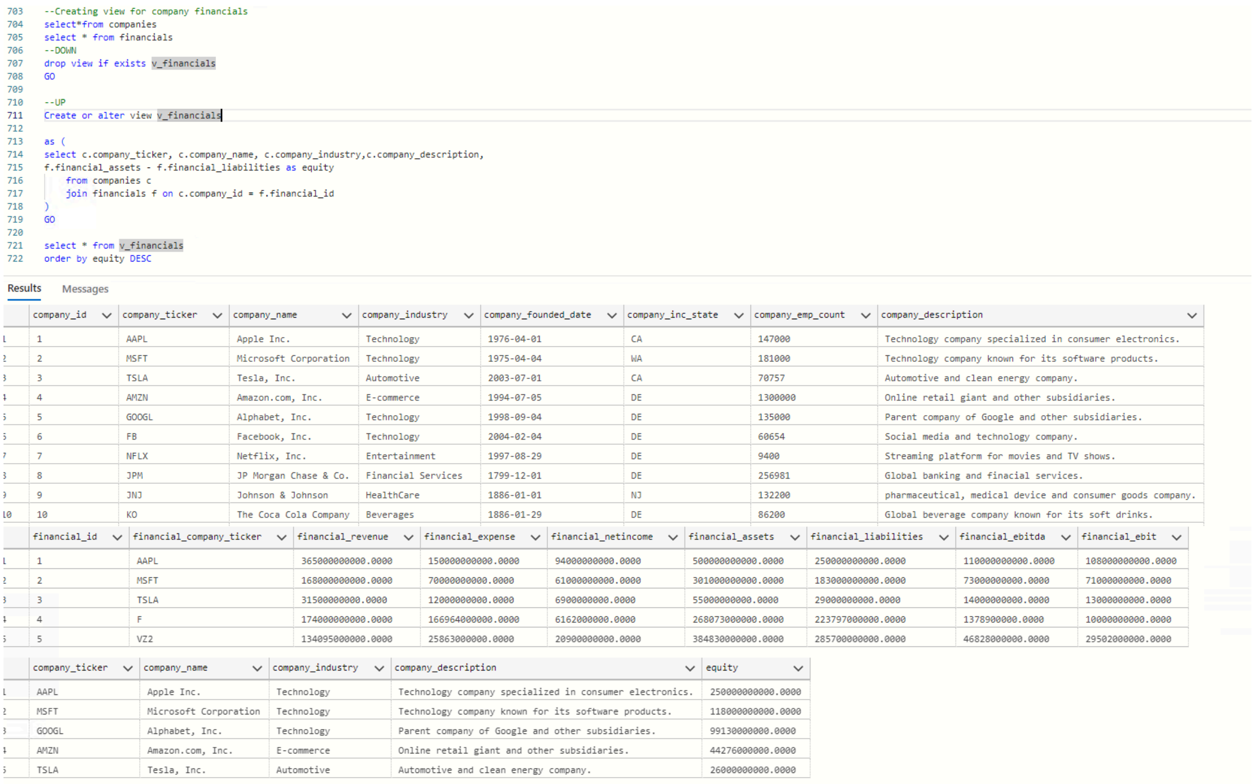
When the trigger prevents the update into a full account type:



```
DROP trigger if exists t_account_update
GO

-- UP
Create trigger t_account_update
on accounts
instead of insert, update , DELETE
AS BEGIN
    IF EXISTS (select count (*) from accounts where account_type = (select inserted.account_type from inserted)
                having count (*) < 5) BEGIN
                update accounts
                set account_type = inserted.account_type
                from inserted
            WHERE accounts.account_id = inserted.account_id
            END
            ELSE BEGIN
            RAISERROR ('Account Type full: Try a different account type', 17, 1)
            END
    END

-- Testing success of the trigger
select * from accounts where account_id = 3
update accounts set account_type = 'CTA' where account_id = 3
select*from accounts where account_id = 3

--Testing failure to update

select * from accounts where account_id = 1004
update accounts set account_type = 'Retail' where account_id = 1004
select*from accounts where account_id = 1004
```

```
7:55:31 PM    Started executing query at Line 688
              (1 row affected)
              (1 row affected)
              (1 row affected)
              (0 rows affected)
              (0 rows affected)
              (0 rows affected)
              (0 rows affected)
              Msg 50000, Level 17, State 1, Line 13
              Account Type full: Try a different account type
              Total execution time: 00:00:00.017
```

10. Create a stored procedure using a view that allows the user to see how the companies are doing in terms of assets and liabilities, the difference of assets minus liability will be stored as equity and then it will be sorted from best to worst.



```sql
--Creating view for company financials
select*from companies
select * from financials
--DOWN
drop view if exists v_financials
GO

--UP
Create or alter view v_financials

as (
select c.company_ticker, c.company_name, c.company_industry,c.company_description,
f.financial_assets - f.financial_liabilities as equity
    from companies c
    join financials f on c.company_id = f.financial_id
)
GO

select * from v_financials
order by equity DESC
```

	company_id	company_ticker	company_name	company_industry	company_founded_date	company_inc_state	company_emp_count	company_description
1	1	AAPL	Apple Inc.	Technology	1976-04-01	CA	147000	Technology company specialized in consumer electronics.
2	2	MSFT	Microsoft Corporation	Technology	1975-04-04	WA	181000	Technology company known for its software products.
3	3	TSLA	Tesla, Inc.	Automotive	2003-07-01	CA	70757	Automotive and clean energy company.
4	4	AMZN	Amazon.com, Inc.	E-commerce	1994-07-05	DE	1300000	Online retail giant and other subsidiaries.
5	5	GOOGL	Alphabet, Inc.	Technology	1998-09-04	DE	135000	Parent company of Google and other subsidiaries.
6	6	FB	Facebook, Inc.	Technology	2004-02-04	DE	60654	Social media and technology company.
7	7	NFLX	Netflix, Inc.	Entertainment	1997-08-29	DE	9400	Streaming platform for movies and TV shows.
8	8	JPM	JP Morgan Chase & Co.	Financial Services	1799-12-01	DE	256981	Global banking and finacial services.
9	9	JNJ	Johnson & Johnson	HealthCare	1886-01-01	NJ	132200	pharmaceutical, medical device and consumer goods company.
10	10	KO	The Coca Cola Company	Beverages	1886-01-29	DE	86200	Global beverage company known for its soft drinks.

	financial_id	financial_company_ticker	financial_revenue	financial_expense	financial_netincome	financial_assets	financial_liabilities	financial_ebitda	financial_ebit
1	1	AAPL	365000000000.0000	150000000000.0000	94000000000.0000	500000000000.0000	250000000000.0000	110000000000.0000	108000000000.0000
2	2	MSFT	168000000000.0000	70000000000.0000	61000000000.0000	301000000000.0000	183000000000.0000	73000000000.0000	71000000000.0000
3	3	TSLA	31500000000.0000	12000000000.0000	6900000000.0000	55000000000.0000	29000000000.0000	14000000000.0000	13000000000.0000
4	4	F	174000000000.0000	166964000000.0000	6162000000.0000	268073000000.0000	223797000000.0000	1378900000.0000	10000000000.0000
5	5	VZ2	134095000000.0000	25863000000.0000	20900000000.0000	384830000000.0000	285700000000.0000	46828000000.0000	29502000000.0000

	company_ticker	company_name	company_industry	company_description	equity
1	AAPL	Apple Inc.	Technology	Technology company specialized in consumer electronics.	250000000000.0000
2	MSFT	Microsoft Corporation	Technology	Technology company known for its software products.	118000000000.0000
3	GOOGL	Alphabet, Inc.	Technology	Parent company of Google and other subsidiaries.	99130000000.0000
4	AMZN	Amazon.com, Inc.	E-commerce	Online retail giant and other subsidiaries.	44276000000.0000
5	TSLA	Tesla, Inc.	Automotive	Automotive and clean energy company.	26000000000.0000

TradeGenius Dashboard Mock-ups

Exhibit I: Home Screen, Portfolio Page, Details Page, Research, Profile Page

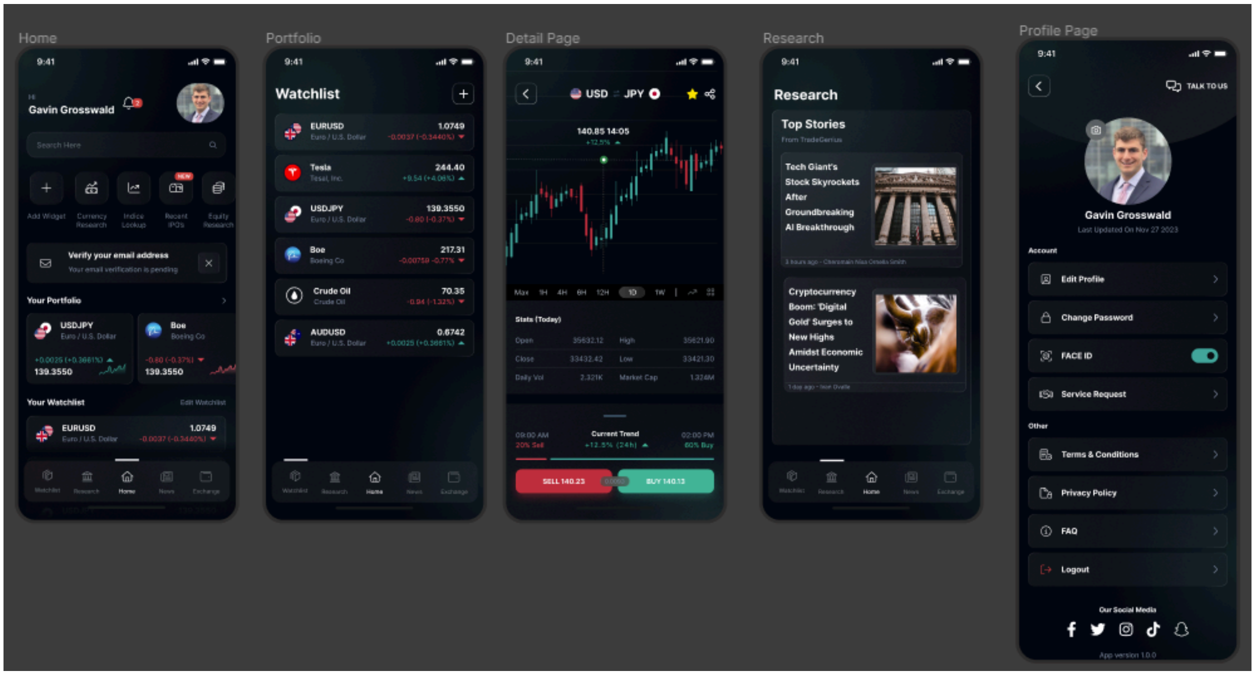


Exhibit II: Home Screen

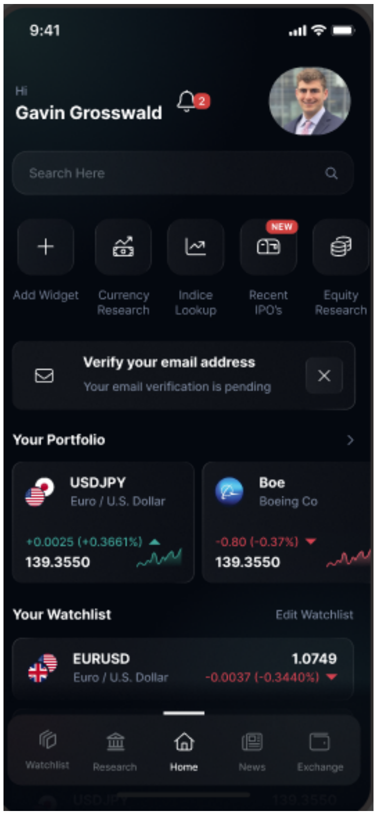
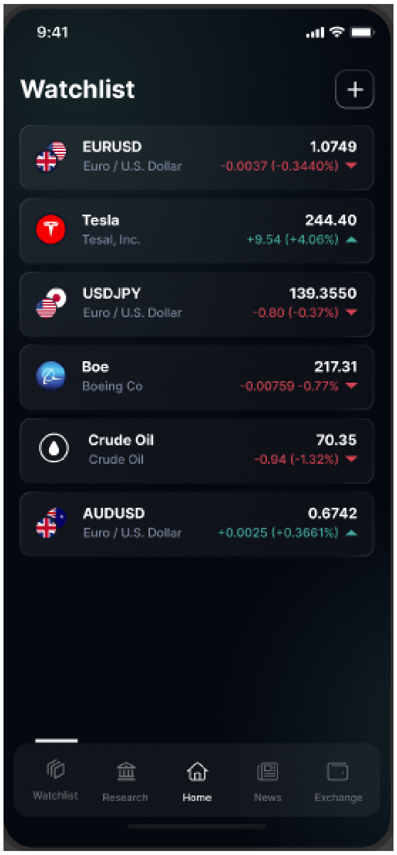


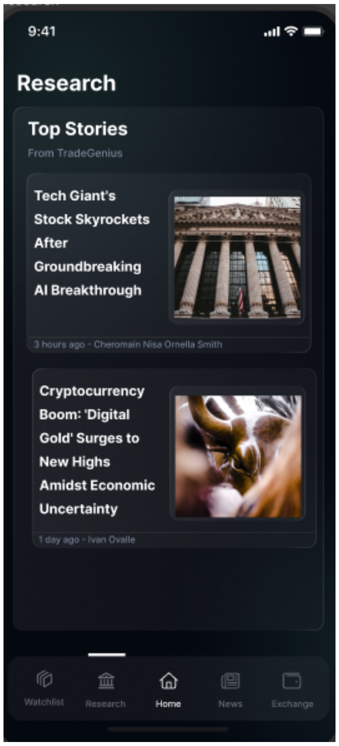
Exhibit III: Watchlist



Watchlist

EURUSD Euro / U.S. Dollar	1.0749 -0.0037 (-0.3440%) ▼
Tesla Tesal, Inc.	244.40 +9.54 (+4.06%) ▲
USDJPY Euro / U.S. Dollar	139.3550 -0.80 (-0.37%) ▼
Boe Boeing Co	217.31 -0.00759 -0.77% ▼
Crude Oil Crude Oil	70.35 -0.94 (-1.32%) ▼
AUDUSD Euro / U.S. Dollar	0.6742 +0.0025 (+0.3661%) ▲

Exhibit IIII: Research



Research

Top Stories

From TradeGenius

Tech Giant's Stock Skyrockets After Groundbreaking AI Breakthrough

3 hours ago - Cheromain Nisa Ornella Smith

Cryptocurrency Boom: 'Digital Gold' Surges to New Highs Amidst Economic Uncertainty

1 day ago - Ivan Ovalle

Summary

TradeGenius is a research based platform created and built on a premise of delivering high quality financial research that is unattainable elsewhere at the subscription price offered by TradeGenius. The TradeGenius team has worked hard on ensuring a platform that is user friendly, modern, and has the data engineering back-end to support institutions and retail traders. We look forward to attending pitch competitions with our research platform and producing a game changing platform that will disrupt the financial industry.

Reflections

Resources that immensely helped our team are listed below. We highly recommend teams utilize these tools and resources.

Task Management: Trello

Diagrams: Draw.io

App Design: Figma (link to our TradeGenius project)

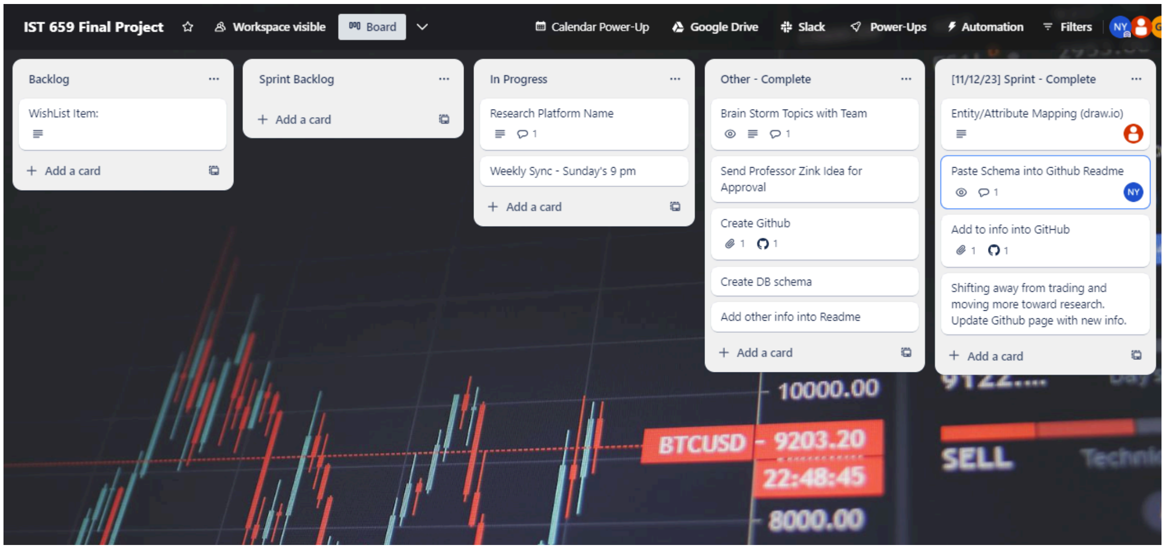
Code Repository: Github (link to our Github)

Logo Design and Powerpoint Design: Canva

TradeGenius Team Log

Everyone had equal participation in the project and every task was assigned accordingly after every team meeting. The team worked in sprints in order to finish tasks every day. Tasks and timelines were logged using a platform called Trello. Screenshots of the tasks shown down bellow:

Exhibit I:



Backlog
- WishList Item:
- + Add a card

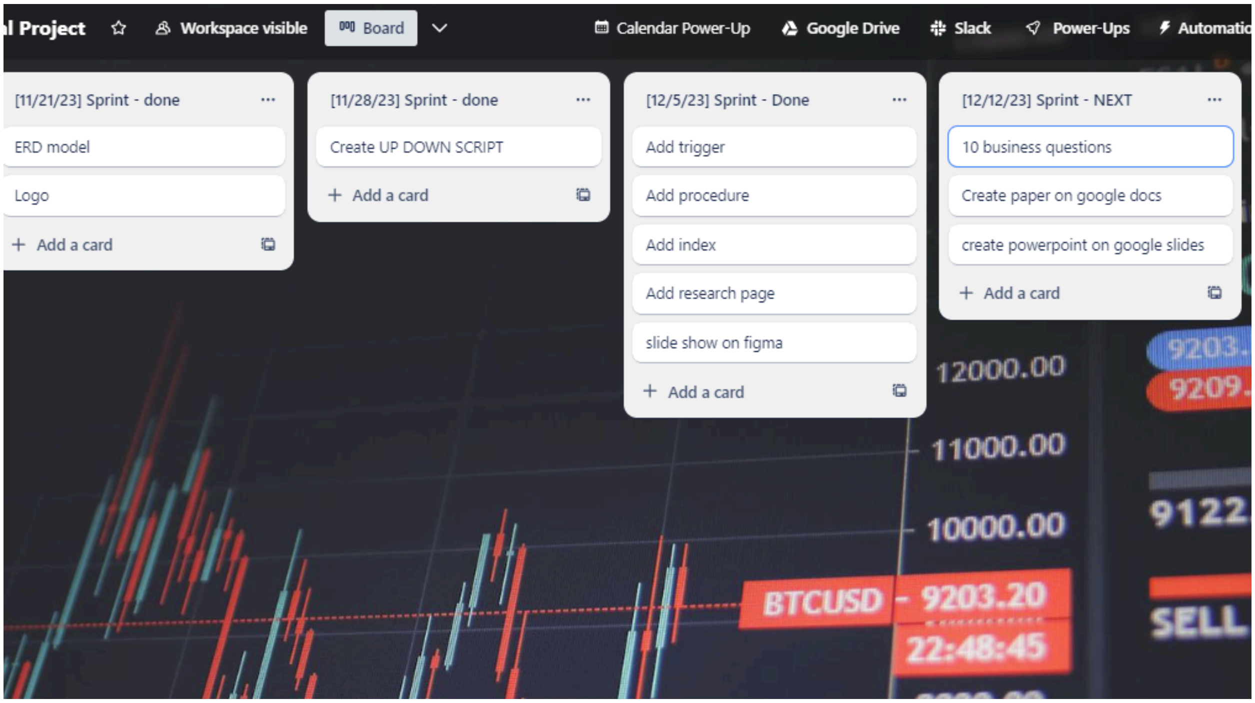
Sprint Backlog
- + Add a card

In Progress
- Research Platform Name 1
- Weekly Sync - Sunday's 9 pm
- + Add a card

Other - Complete
- Brain Storm Topics with Team 1
- Send Professor Zink Idea for Approval
- Create Github 1 1
- Create DB schema
- Add other info into Readme
- + Add a card

[11/12/23] Sprint - Complete
- Entity/Attribute Mapping (draw.io)
- Paste Schema into Github Readme 1 NY
- Add to info into GitHub 1 1
- Shifting away from trading and moving more toward research. Update Github page with new info.
- + Add a card

Exhibit II:



[11/21/23] Sprint - done
- ERD model
- Logo
- + Add a card

[11/28/23] Sprint - done
- Create UP DOWN SCRIPT
- + Add a card

[12/5/23] Sprint - Done
- Add trigger
- Add procedure
- Add index
- Add research page
- slide show on figma
- + Add a card

[12/12/23] Sprint - NEXT
- 10 business questions
- Create paper on google docs
- create powerpoint on google slides
- + Add a card