SWAP Database Design

Isabelle May | Founder & DB Engineer

Platform Overview

- College marketplace platform
- Enabling students to…
    - post and browse items for sale
    - offer freelance services to students or community members
    - message other users
    - rate items and services they have received

Overview of Use Cases

01

Login/Sign Up

02

View/Modify Listings

03

Search for Listings & Record Transactions

04

View/Send Messages

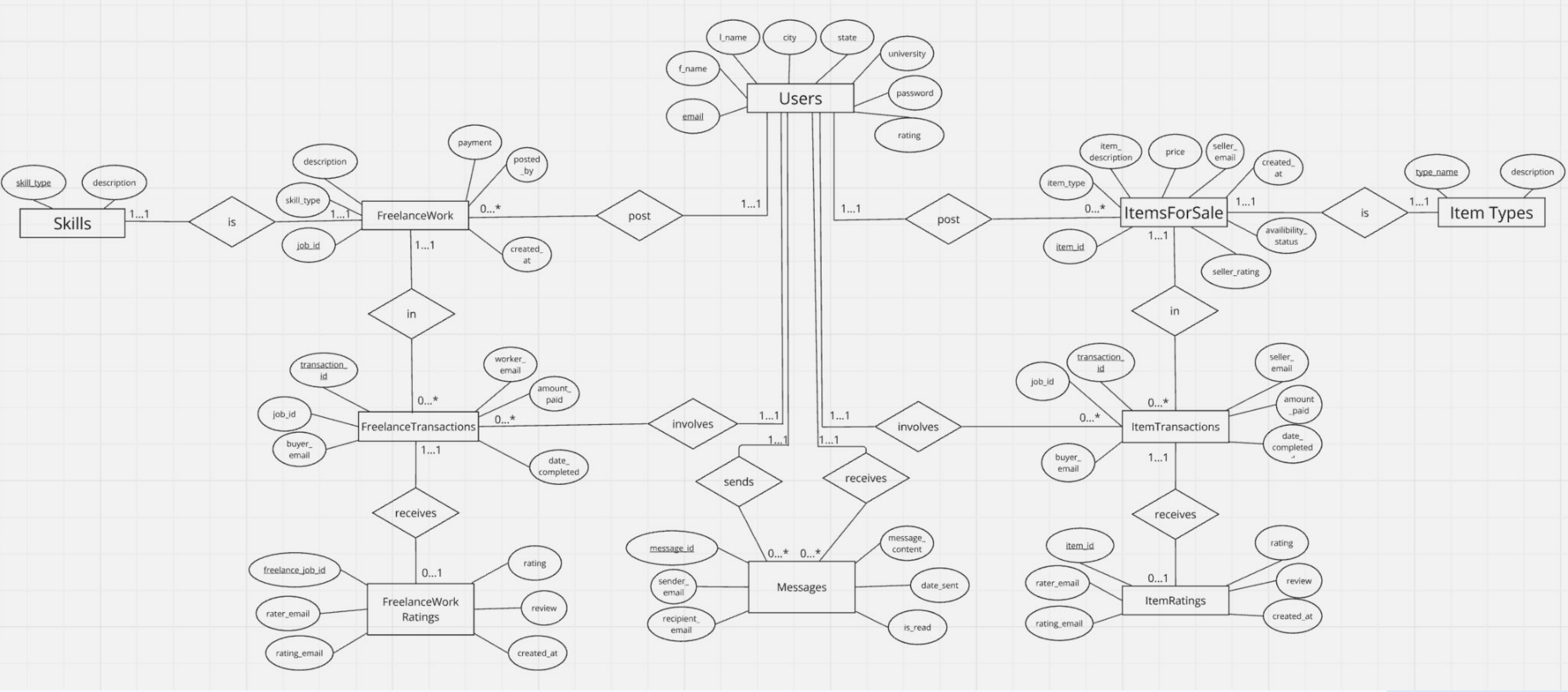
05

Rating Items/Services

06

Platform Analytics

Entity-Relationship Diagram



Data Population Techniques

Method 1: Automated Generation via ChatGPT

Quickly populate large datasets for tables such as Users, Items, and Freelance Work.

Method 2: SQL Query-Based Random Population

Populate dependent tables like Transactions and Ratings using pre-existing data and logical relationships.

EXAMPLE

SQL Query-Based Random Population

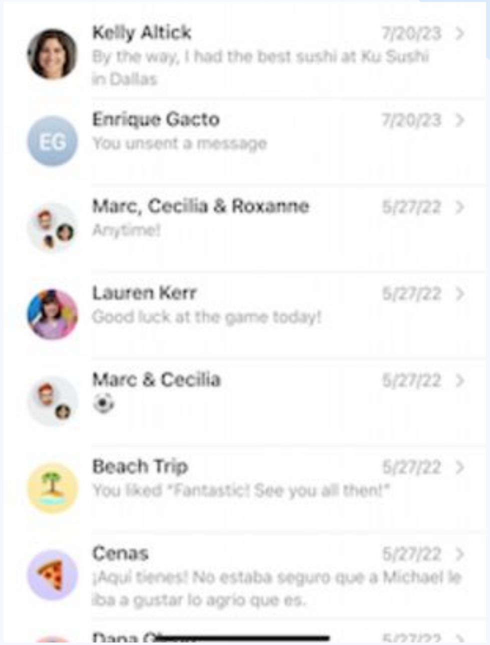
```
INSERT INTO ItemTransactions (item_id, buyer_email, seller_email, amount_paid, date_completed)
SELECT
    random_transactions.item_id,
    random_transactions.buyer_email,
    random_transactions.seller_email,
    random_transactions.price AS amount_paid,
    NOW() -- Use current time for simplicity
FROM (
    SELECT
        i.item_id,
        u1.email AS buyer_email,
        u2.email AS seller_email,
        i.price
    FROM
        ItemsForSale i
    CROSS JOIN
        Users u1 -- Join to allow any user to be a potential buyer
    INNER JOIN
        Users u2 ON u2.email = i.seller_email -- Ensure seller matches item
    WHERE
        u1.email != u2.email -- Ensure buyer is not the seller
    ORDER BY
        RANDOM() -- Shuffle rows to add randomness
    LIMIT 45 -- Limit to 45 transactions
) AS random_transactions;
```

Interesting Query #1

Messages: find unique emails with whom the user has had conversations

```
SELECT DISTINCT
        CASE
                WHEN sender_email = %s THEN recipient_email
                ELSE sender_email
        END AS other_email
        FROM Messages
        WHERE sender_email = %s OR recipient_email = %s;
```

Inspiration:



Kelly Altick 7/20/23 >
By the way, I had the best sushi at Ku Sushi in Dallas

Enrique Gacto 7/20/23 >
You unsent a message

Marc, Cecilia & Roxanne 5/27/22 >
Anytime!

Lauren Kerr 5/27/22 >
Good luck at the game today!

Marc & Cecilia 5/27/22 >
⚽

Beach Trip 5/27/22 >
You liked "Fantastic! See you all then!"

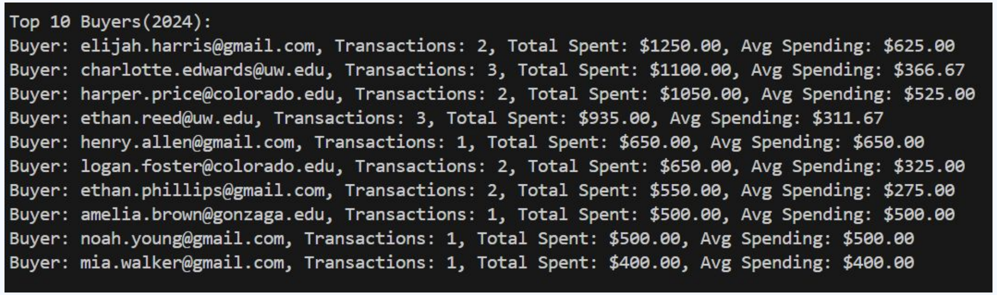
Cenas 5/27/22 >
¡Aquí tienes! No estaba seguro que a Michael le iba a gustar lo agrio que es.

Dana O[redacted] 5/27/22 >

Interesting Query #2

Analytics: Top Buyers

```
WITH BuyerSpending AS (
    SELECT
        it.buyer_email,
        COUNT(it.transaction_id) AS transaction_count,
        SUM(it.amount_paid) AS total_spent,
        AVG(it.amount_paid) AS avg_spent_per_transaction,
        RANK() OVER (ORDER BY SUM(it.amount_paid) DESC) AS rank
    FROM ItemTransactions it
    WHERE it.date_completed BETWEEN '2024-01-01' AND '2024-12-31'
    GROUP BY it.buyer_email
    )
    SELECT
        buyer_email,
        transaction_count,
        total_spent,
        avg_spent_per_transaction
    FROM BuyerSpending
    WHERE rank <= 10
    ORDER BY rank;
```

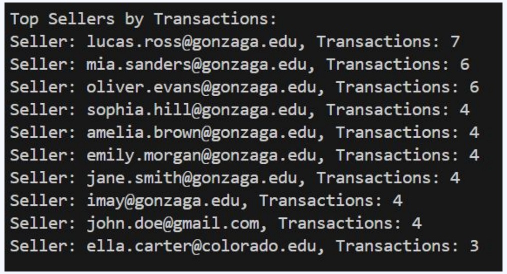


```
Top 10 Buyers(2024):
Buyer: elijah.harris@gmail.com, Transactions: 2, Total Spent: $1250.00, Avg Spending: $625.00
Buyer: charlotte.edwards@uw.edu, Transactions: 3, Total Spent: $1100.00, Avg Spending: $366.67
Buyer: harper.price@colorado.edu, Transactions: 2, Total Spent: $1050.00, Avg Spending: $525.00
Buyer: ethan.reed@uw.edu, Transactions: 3, Total Spent: $935.00, Avg Spending: $311.67
Buyer: henry.allen@gmail.com, Transactions: 1, Total Spent: $650.00, Avg Spending: $650.00
Buyer: logan.foster@colorado.edu, Transactions: 2, Total Spent: $650.00, Avg Spending: $325.00
Buyer: ethan.phillips@gmail.com, Transactions: 2, Total Spent: $550.00, Avg Spending: $275.00
Buyer: amelia.brown@gonzaga.edu, Transactions: 1, Total Spent: $500.00, Avg Spending: $500.00
Buyer: noah.young@gmail.com, Transactions: 1, Total Spent: $500.00, Avg Spending: $500.00
Buyer: mia.walker@gmail.com, Transactions: 1, Total Spent: $400.00, Avg Spending: $400.00
```

Interesting Query #3

Analytics: Top Sellers by Transaction

```
WITH ItemSellerTransactions AS (
        SELECT it.seller_email AS email, COUNT(it.transaction_id) AS transactions_count
        FROM ItemTransactions it
        GROUP BY it.seller_email
    ),
    FreelanceWorkerTransactions AS (
        SELECT ft.worker_email AS email, COUNT(ft.transaction_id) AS transactions_count
        FROM FreelanceTransactions ft
        GROUP BY ft.worker_email
    ),
    CombinedTransactions AS (
        SELECT email, SUM(transactions_count) AS total_transactions
        FROM (
            SELECT * FROM ItemSellerTransactions
            UNION ALL
            SELECT * FROM FreelanceWorkerTransactions
        ) subquery
        GROUP BY email
    )
    SELECT email, total_transactions
    FROM CombinedTransactions
    ORDER BY total_transactions DESC
    LIMIT 10;
```



```
Top Sellers by Transactions:
Seller: lucas.ross@gonzaga.edu, Transactions: 7
Seller: mia.sanders@gonzaga.edu, Transactions: 6
Seller: oliver.evans@gonzaga.edu, Transactions: 6
Seller: sophia.hill@gonzaga.edu, Transactions: 4
Seller: amelia.brown@gonzaga.edu, Transactions: 4
Seller: emily.morgan@gonzaga.edu, Transactions: 4
Seller: jane.smith@gonzaga.edu, Transactions: 4
Seller: imay@gonzaga.edu, Transactions: 4
Seller: john.doe@gmail.com, Transactions: 4
Seller: ella.carter@colorado.edu, Transactions: 3
```

DEMO TIME

Lets see the database in action

Challenges & Future Expansions

Creating Distinct Interfaces

Developing different user interfaces for students and community members, ensuring both experiences are tailored while maintaining consistent functionality.

Sequential Sorting and Filtering

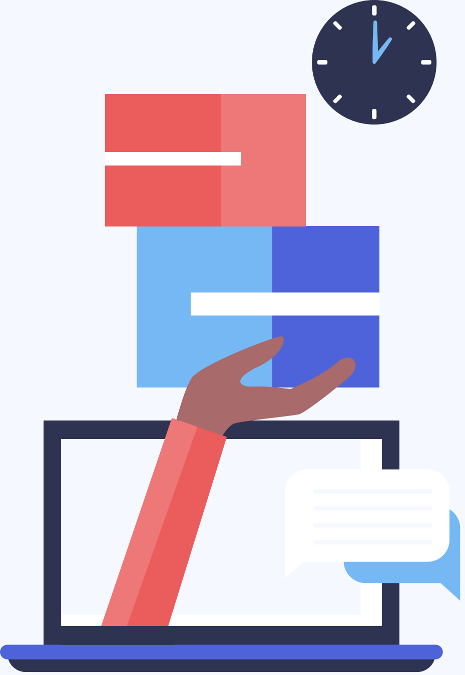
Allowing users to sort or filter by multiple criteria (e.g., category, then rating) by executing one query, saving the result, and then executing the next.

User Experience for Different Groups

Continue refining the user interface to ensure that students and community members have relevant options and easy access to features specific to their needs.

Advanced Search and Sort Features

Implement more flexible and sequential search and sort functionality, where users can adjust multiple criteria without confusion.



Thanks!

SWAP | imay2@gonzaga.edu