

# INFORMATION MANAGEMENT FINAL PROJECT

GCASH



USER INTERFACE OVERVIEW, ENTITY RELATIONSHIP DIAGRAM, AND PURPOSES

# MEET THE MAN BEHIND THE APP

RAE PAULOS



r.paulos.dev@gmail.com

National University - Manila ('27)  
Bachelor of Science in Computer Science

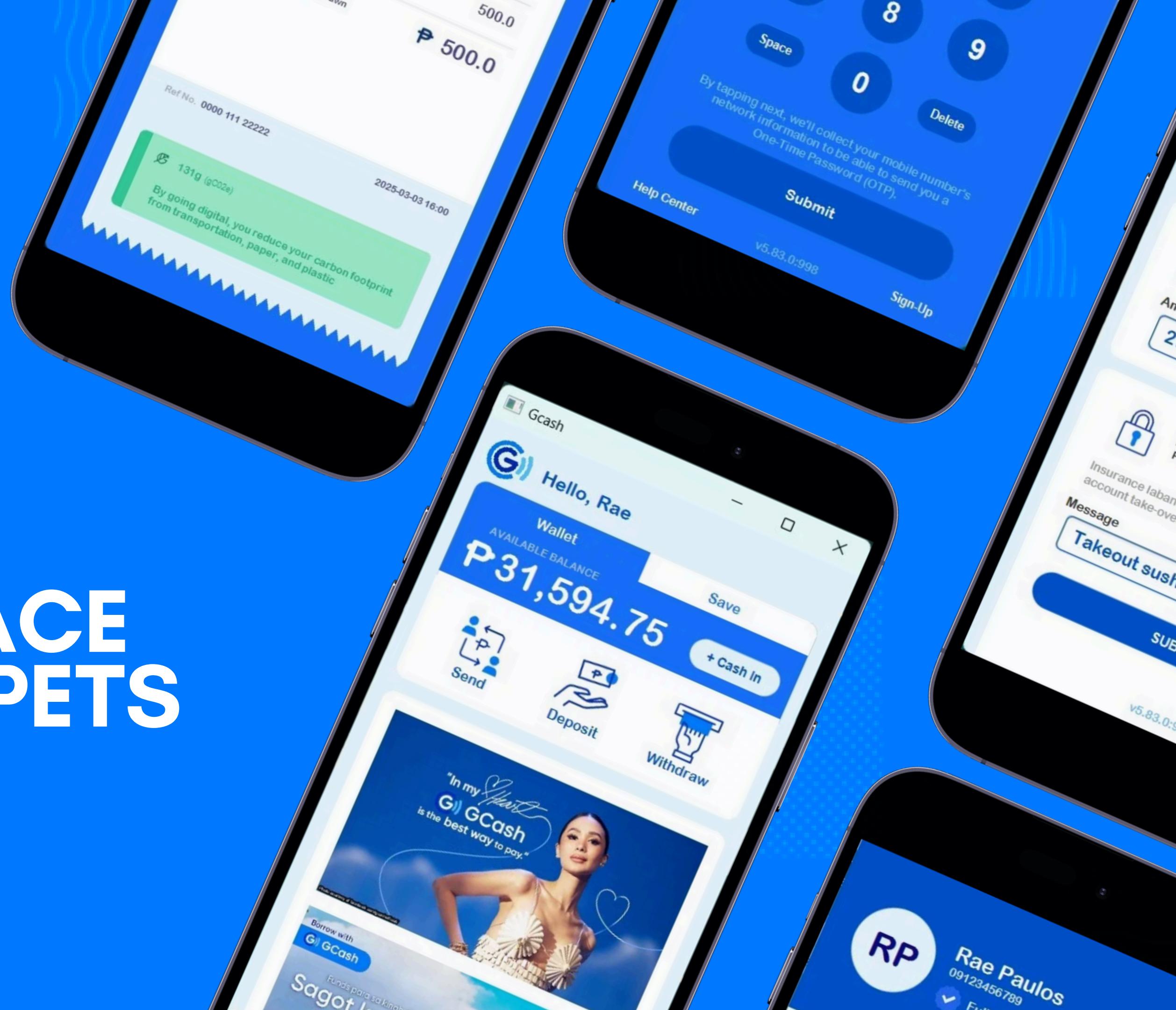
his project is a Java-based e-wallet application with an SQL database, designed to provide a seamless and efficient digital transaction experience. Inspired by the functionality of GCash, it incorporates three key features: Express Send, allowing users to transfer funds instantly; Deposit, enabling users to add money to their accounts; and Withdrawal, facilitating secure cash-out transactions.

The system is developed as part of the Information Management course, taught by `@robitussin` on GitHub. It emphasizes secure and efficient financial transactions while demonstrating the integration of Java-based application development with database management using SQL. The project aims to enhance the understanding of digital financial services and database implementation while ensuring a user-friendly experience.



[@rpaulos](https://github.com/rpaulos)

# GCASH USER INTERFACE & CODE SNIPPETS



# LOGIN PAGE

Login page is where the User logs their specific number account and PIN to log in to their Gcash. Only authorized individuals may log in and helps safeguard their financial information.





```
1 public static boolean validateMobileNumber(String phone_number){  
2  
3     getInstance();  
4     String query = "SELECT * FROM users WHERE phone_number = '" + phone_number + "'";  
5  
6     System.out.println(query);  
7  
8     ResultSet result = handler.execQuery(query);  
9     try {  
10         if (result.next()) {  
11             return true;  
12         }  
13     }  
14     catch (SQLException e){  
15         e.printStackTrace();  
16     }  
17     return false;  
18 }
```



```
1 public static boolean validateMobileNumberAndMPIN(String phone_number, String PIN){  
2     getInstance();  
3     String query = "SELECT * FROM users WHERE phone_number = '" + phone_number + "' AND PIN = '" + PIN + "'";  
4  
5     System.out.println(query);  
6  
7     ResultSet result = handler.execQuery(query);  
8     try {  
9         if (result.next()) {  
10             return true;  
11         }  
12     }  
13     catch (SQLException e){  
14         e.printStackTrace();  
15     }  
16     return false;  
17 }
```

Never share your MPIN

# SIGN UP PAGE

Is where the User creates their account as well their designated numbers.

The diagram illustrates a user registration process across four mobile phone screens. Each screen shows a registration form with the following fields:

- Mobile Number:** 1234 567 8901
- Mobile Personal Identification Number:** 1 2 3 4
- Text:** Never share your MPIN with anyone.
- Email Address:** oliviarodrigo@gmail.com
- First Name:** Olivia
- Last Name:** Rodrigo
- Birthdate:** Birthday
- Country:** Philippines
- Address:** Quezon
- Submit Button:** SUBMIT (v5.83.0:998)

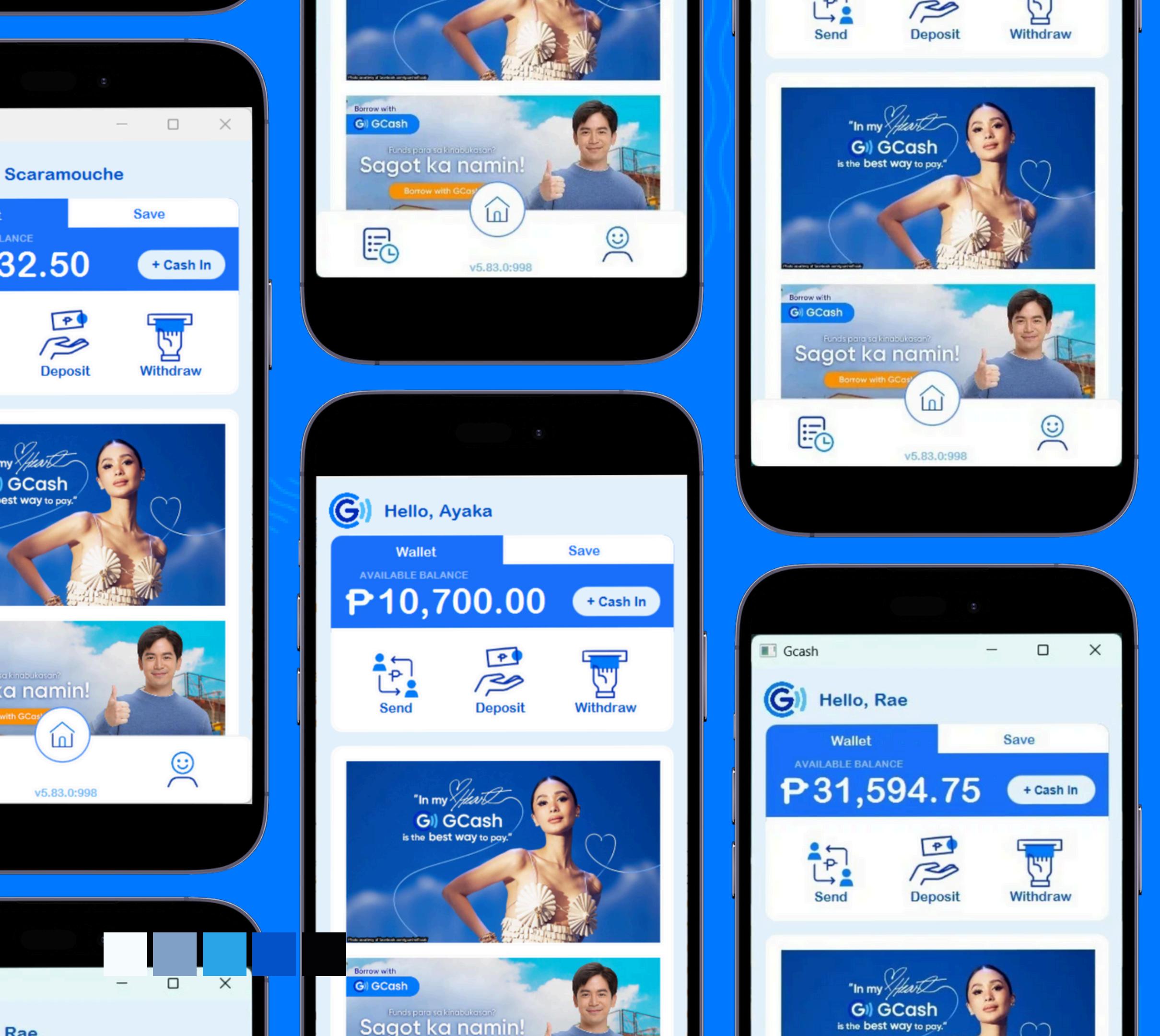
The screens are arranged vertically, showing the progression of the registration form from top to bottom. The background features a blue gradient with circular patterns and a color bar at the bottom left.



```
1 public static boolean addUser(User user) {
2     try {
3         pstatement = getDBConnection().prepareStatement("INSERT INTO users (phone_number, first
4 _name, last_name, email_address, PIN, birthdate, country, address) VALUES (?, ?, ?, ?, ?, ?, ?,
5 ?)");
6         pstatement.setString(1, user.getPhone_number());
7         pstatement.setString(2, user.getFirst_name());
8         pstatement.setString(3, user.getLast_name());
9         pstatement.setString(4, user.getEmail_address());
10        pstatement.setString(5, user.getPIN());
11        pstatement.setString(6, user.getBirthdate());
12        pstatement.setString(7, user.getCountry());
13        pstatement.setString(8, user.getAddress());
14
15        return pstatement.executeUpdate() > 0;
16    } catch (Exception e) {
17        e.printStackTrace();
18    }
19    return false;
20 }
```

# HOME PAGE

Where it serves the main important part of the UI. Providing the User information such as balance, sending money and receiving money.





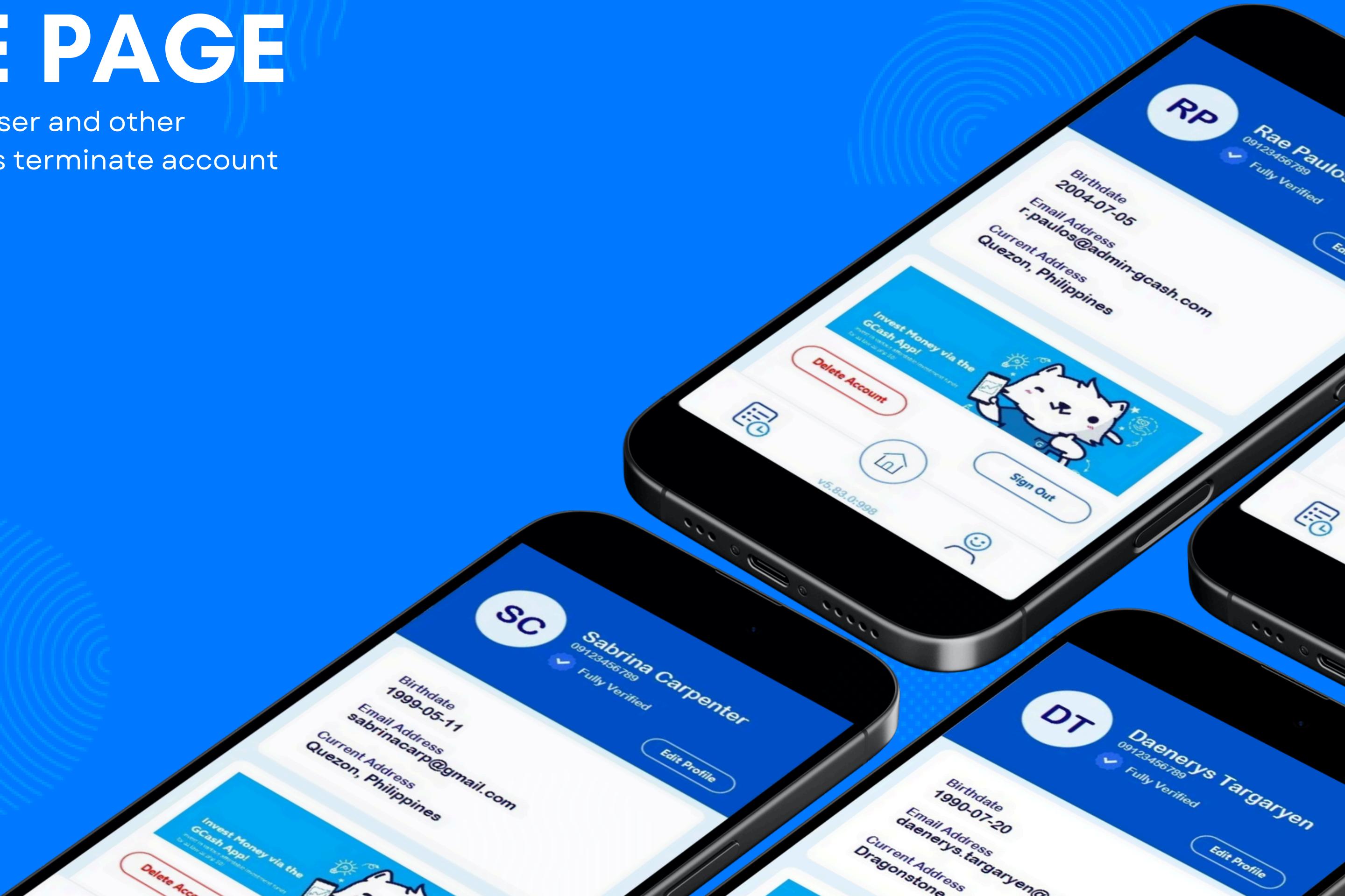
```
1 public static String getFirstName(String phone_number) {  
2     String query = "SELECT first_name FROM users WHERE phone_number = ?";  
3     String first_name = null;  
4     Connection conn = null;  
5     PreparedStatement stmt = null;  
6     ResultSet result = null;  
7  
8     try {  
9         conn = getDBConnection();  
10        stmt = conn.prepareStatement(query);  
11        stmt.setString(1, phone_number);  
12        result = stmt.executeQuery();  
13  
14        if(result.next()) {  
15            first_name = result.getString("first_name");  
16        }  
17  
18    } catch (SQLException e) {  
19        e.printStackTrace();  
20    }  
21    return first_name;  
22 }
```



```
1 public static float getUserBalance(String phone_number) {  
2     String query = "SELECT balance FROM wallet WHERE phone_number = ?";  
3     float balance = 0.0f;  
4     Connection conn = null;  
5     PreparedStatement stmt = null;  
6     ResultSet result = null;  
7  
8     try {  
9         conn = getDBConnection();  
10        stmt = conn.prepareStatement(query);  
11        stmt.setString(1, phone_number);  
12        result = stmt.executeQuery();  
13  
14        if(result.next()) {  
15            balance = result.getFloat("balance");  
16        }  
17  
18    } catch (SQLException e) {  
19        e.printStackTrace();  
20    }  
21    return balance;  
22 }
```

# PROFILE PAGE

The information of the User and other account settings such as terminate account or log out.

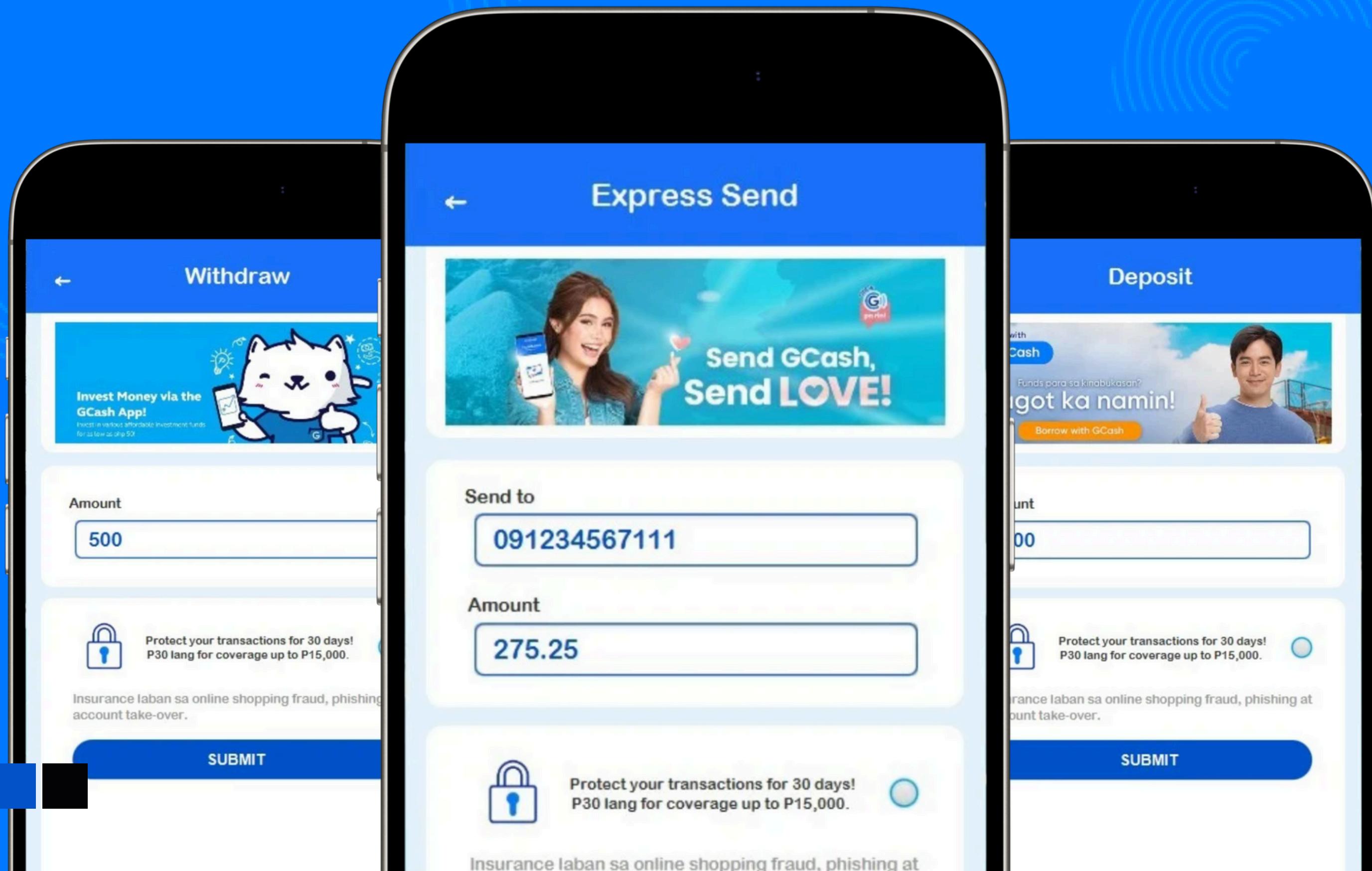


# EDIT PROFILE

The user may edit his/her account information.

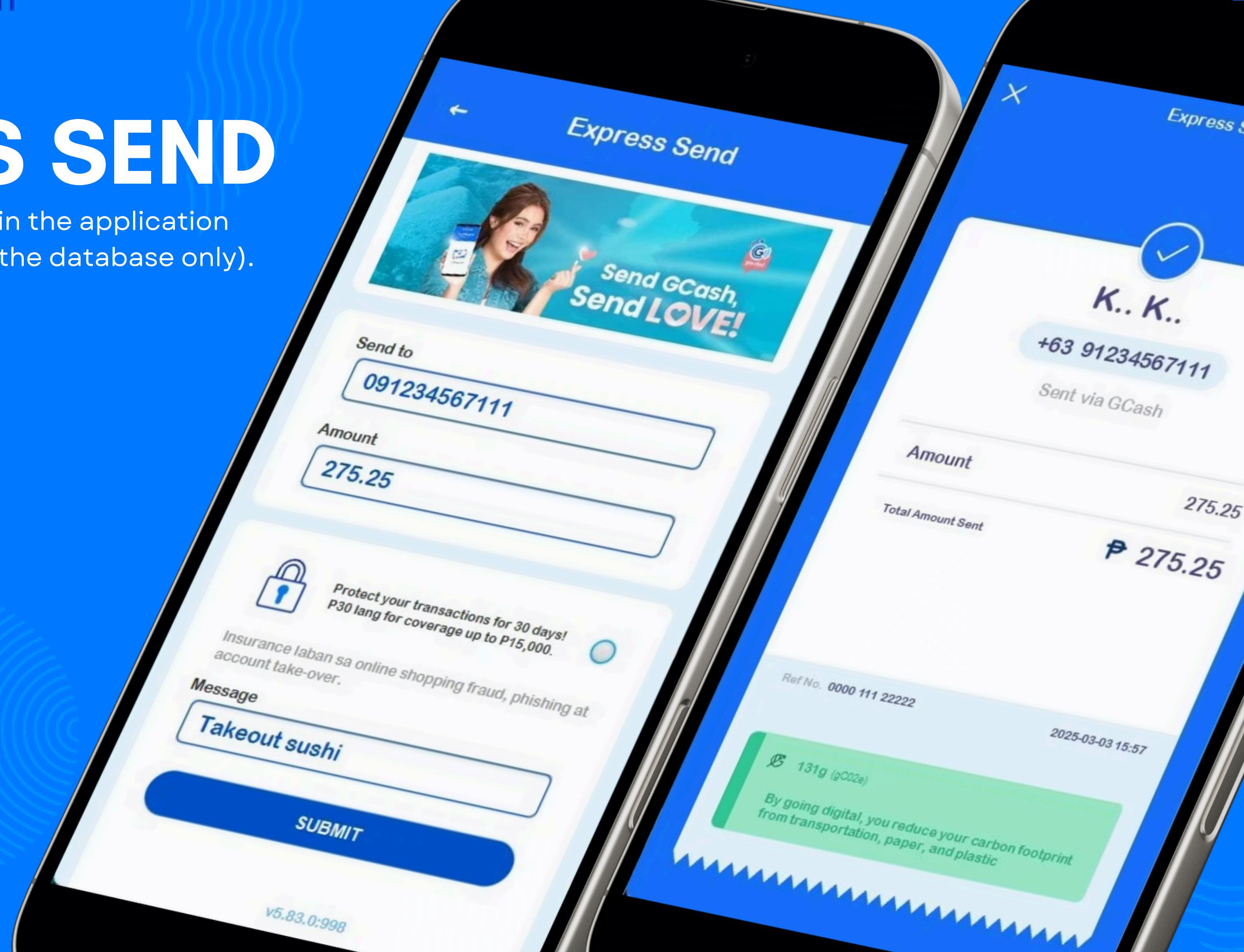


# MAIN FEATURES



# EXPRESS SEND

User can send money within the application to other accounts (Within the database only).





```
1 public static float expressSend(String numberToSendTo, float amountToSend) {  
2     String query = "UPDATE wallet SET balance = balance + ? WHERE phone_number = ?";  
3     float balance = 0.0f;  
4     Connection conn = null;  
5     PreparedStatement stmt = null;  
6  
7     try {  
8         conn = getDBConnection();  
9         stmt = conn.prepareStatement(query);  
10        stmt.setFloat(1, amountToSend);  
11        stmt.setString(2, numberToSendTo);  
12  
13        int affectedRows = stmt.executeUpdate();  
14  
15    } catch (SQLException e) {  
16        e.printStackTrace();  
17    }  
18    return balance;  
19 }
```



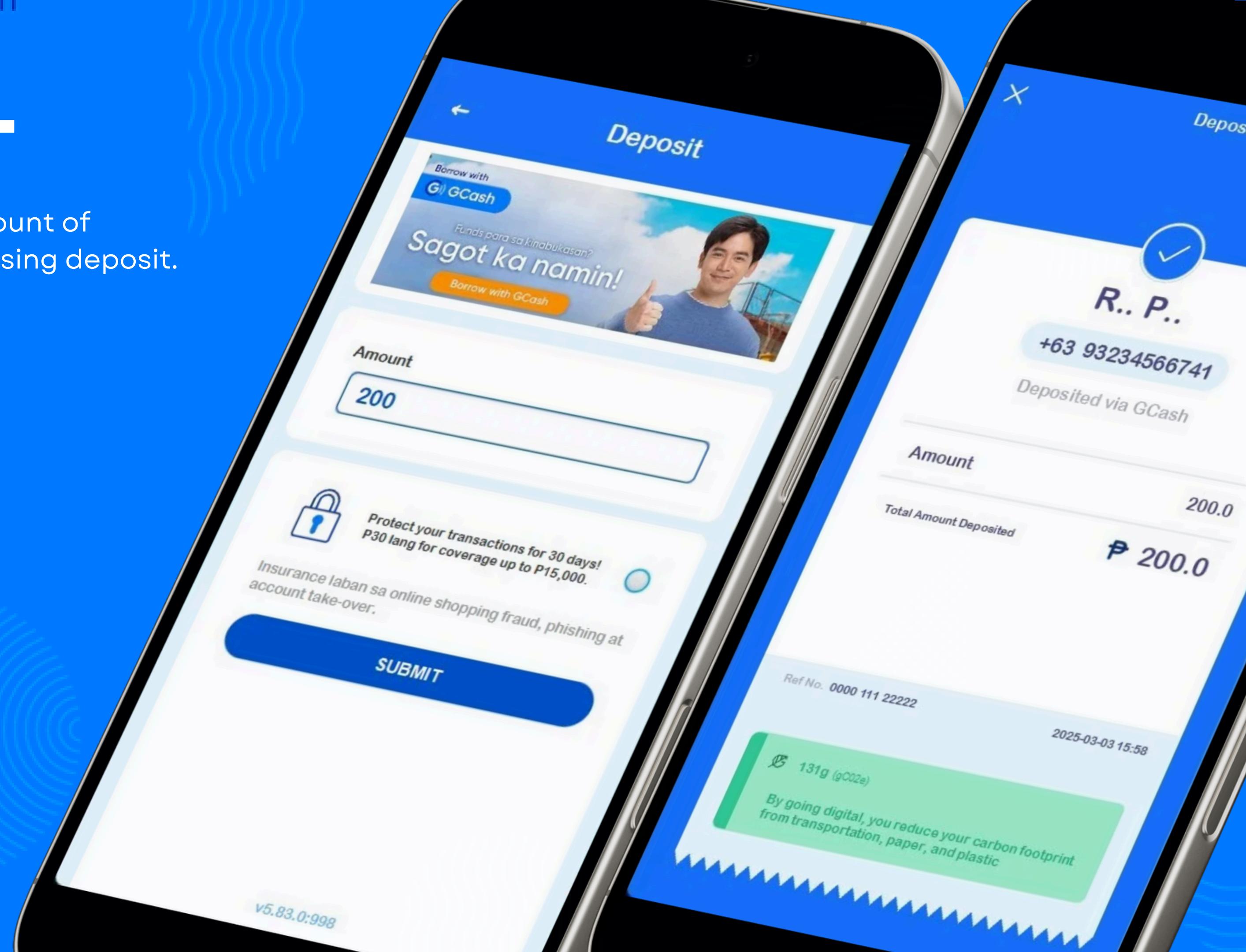
```
1 public static void negateBalance(Float negateFromBalance, String myNumber) {  
2     String query = "UPDATE wallet SET balance = balance - ? WHERE phone_number = ?";  
3     Connection conn = null;  
4     PreparedStatement stmt = null;  
5  
6     try {  
7         conn = getDBConnection();  
8         stmt = conn.prepareStatement(query);  
9         stmt.setFloat(1, negateFromBalance);  
10        stmt.setString(2, myNumber);  
11  
12        int affectedRows = stmt.executeUpdate();  
13  
14    } catch (Exception e) {  
15        e.printStackTrace();  
16    }  
17}
```



```
1 public static void recordExpressSend(String sender_number, String receiver_number, Fl
oat amount) {
2     String query = "INSERT INTO send_transactions (sender_number, receiver_number, am
ount) VALUES (?, ?, ?);";
3     Connection conn = null;
4     PreparedStatement stmt = null;
5
6     try {
7         conn = getDBConnection();
8         stmt = conn.prepareStatement(query);
9         stmt.setString(1, sender_number);
10        stmt.setString(2, receiver_number);
11        stmt.setFloat(3, amount);
12        int affectedRows = stmt.executeUpdate();
13
14    } catch (Exception e) {
15        e.printStackTrace();
16    }
17 }
```

# DEPOSIT

The User may add an amount of money to their account using deposit.





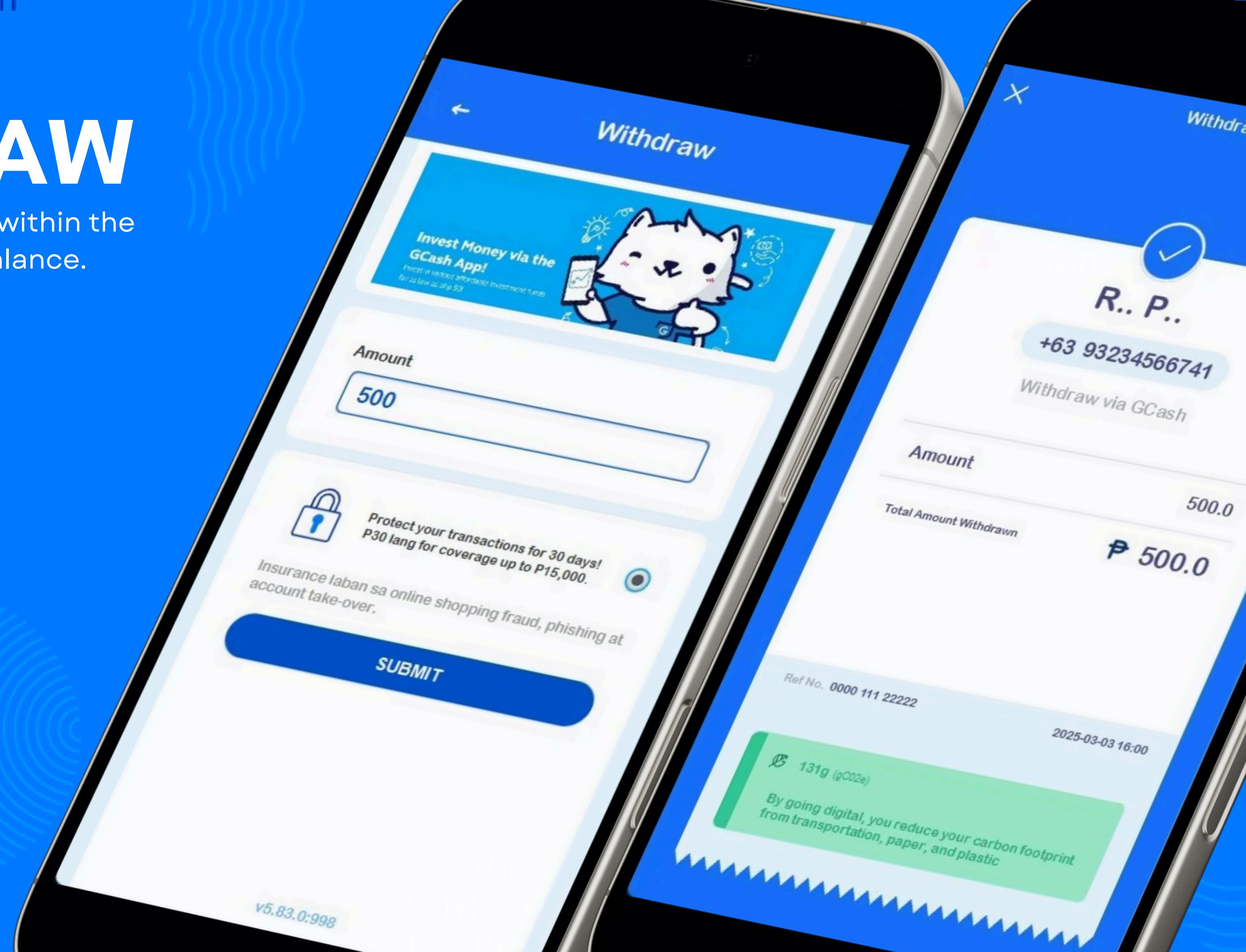
```
1 public static void deposit(Float addToBalance, String phone_number) {  
2     String query = "UPDATE wallet SET balance = balance + ? WHERE phone_number = ?";  
3     Connection conn = null;  
4     PreparedStatement stmt = null;  
5  
6     try {  
7         conn = getDBConnection();  
8         stmt = conn.prepareStatement(query);  
9         stmt.setFloat(1, addToBalance);  
10        stmt.setString(2, phone_number);  
11        int affectedRows = stmt.executeUpdate();  
12  
13    } catch (Exception e) {  
14        e.printStackTrace();  
15    }  
16 }
```



```
1 public static void recordDeposit(String phone_number, Float amount) {  
2     String query = "INSERT INTO deposit_transactions (depositor_number, amount) VALUES (?, ?);"  
3     Connection conn = null;  
4     PreparedStatement stmt = null;  
5  
6     try {  
7         conn = getDBConnection();  
8         stmt = conn.prepareStatement(query);  
9         stmt.setString(1, phone_number);  
10        stmt.setFloat(2, amount);  
11        int affectedRows = stmt.executeUpdate();  
12  
13    } catch (Exception e) {  
14        e.printStackTrace();  
15    }  
16 }
```

# WITHDRAW

User may withdraw cash within the range of their account balance.





```
1 public static void withdraw(Float negateFromBalance, String phone_number) {  
2     String query = "UPDATE wallet SET balance = balance - ? WHERE phone_number = ?";  
3     Connection conn = null;  
4     PreparedStatement stmt = null;  
5  
6     try {  
7         conn = getDBConnection();  
8         stmt = conn.prepareStatement(query);  
9         stmt.setFloat(1, negateFromBalance);  
10        stmt.setString(2, phone_number);  
11        int affectedRows = stmt.executeUpdate();  
12  
13    } catch (Exception e) {  
14        e.printStackTrace();  
15    }  
16 }
```



```
1 public static void recordWithdraw(String phone_number, Float amount) {  
2     String query = "INSERT INTO withdraw_transactions (withdrawer_number, amount) VALUES (?, ?);  
3     Connection conn = null;  
4     PreparedStatement stmt = null;  
5  
6     try {  
7         conn = getDBConnection();  
8         stmt = conn.prepareStatement(query);  
9         stmt.setString(1, phone_number);  
10        stmt.setFloat(2, amount);  
11        int affectedRows = stmt.executeUpdate();  
12  
13    } catch (Exception e) {  
14        e.printStackTrace();  
15    }  
16}
```

# TRANSACTION HISTORY

The image shows three smartphones arranged horizontally, each displaying a transaction history screen from a mobile application. The background is a solid blue color with faint, concentric circular patterns.

**Smartphone 1 (Left):** The screen title is "Transaction History". Below it is a table with columns: "Receiver", "Amount", and "Date". Two entries are visible:

Receiver	Amount	Date
Lucas Domingo	2000.50	2025-03-03
Kazuha Kaedeh...	275.25	2025-03-03

**Smartphone 2 (Middle):** The screen title is "Transaction History". Below it is a table with columns: "Amount" and "Date". Two entries are visible:

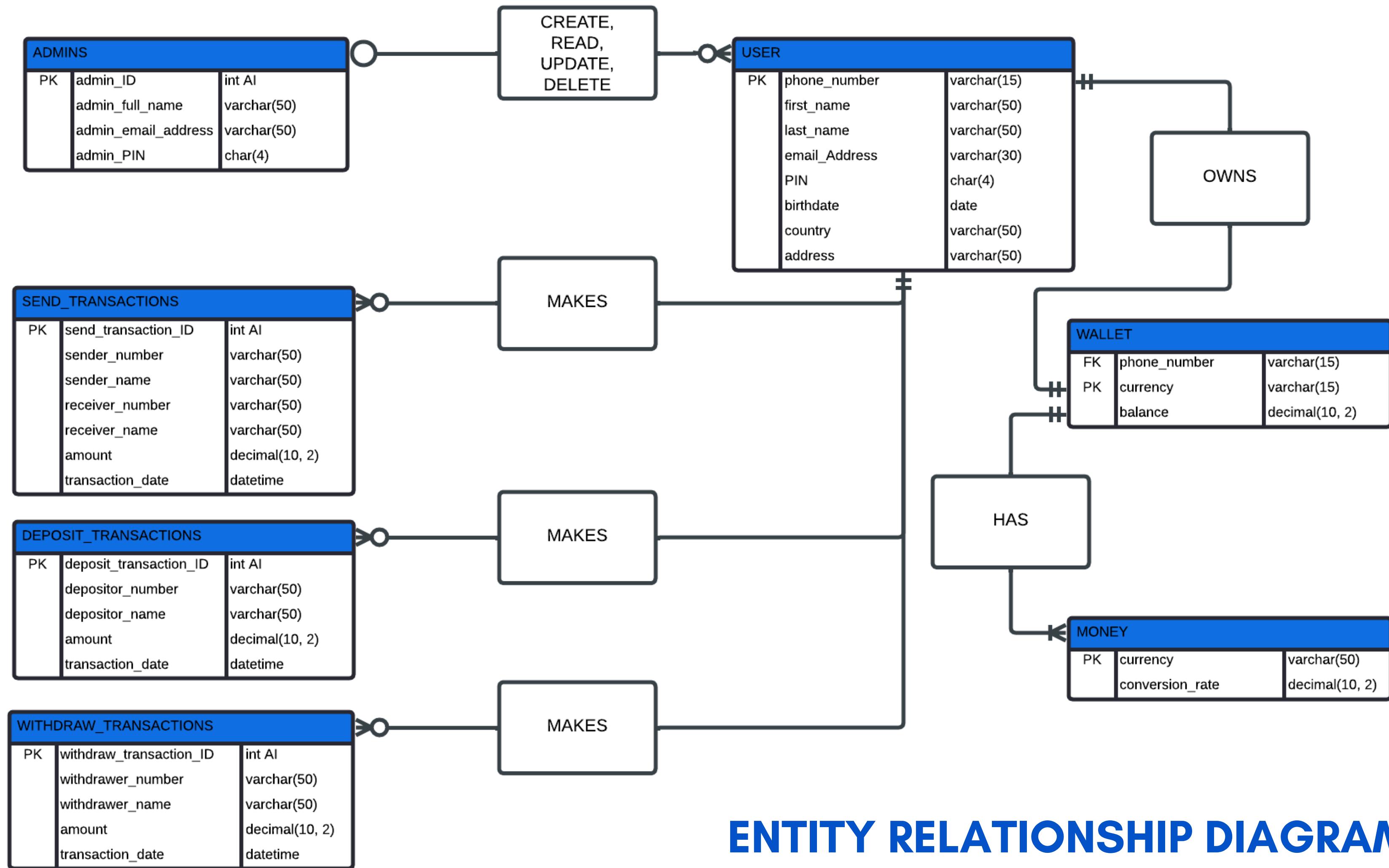
Amount	Date
200.00	2025-03-03 15:58:28
1500.25	2025-03-03 16:00:00

**Smartphone 3 (Right):** The screen title is "Transaction History". Below it is a table with columns: "Amount" and "Date". Three entries are visible:

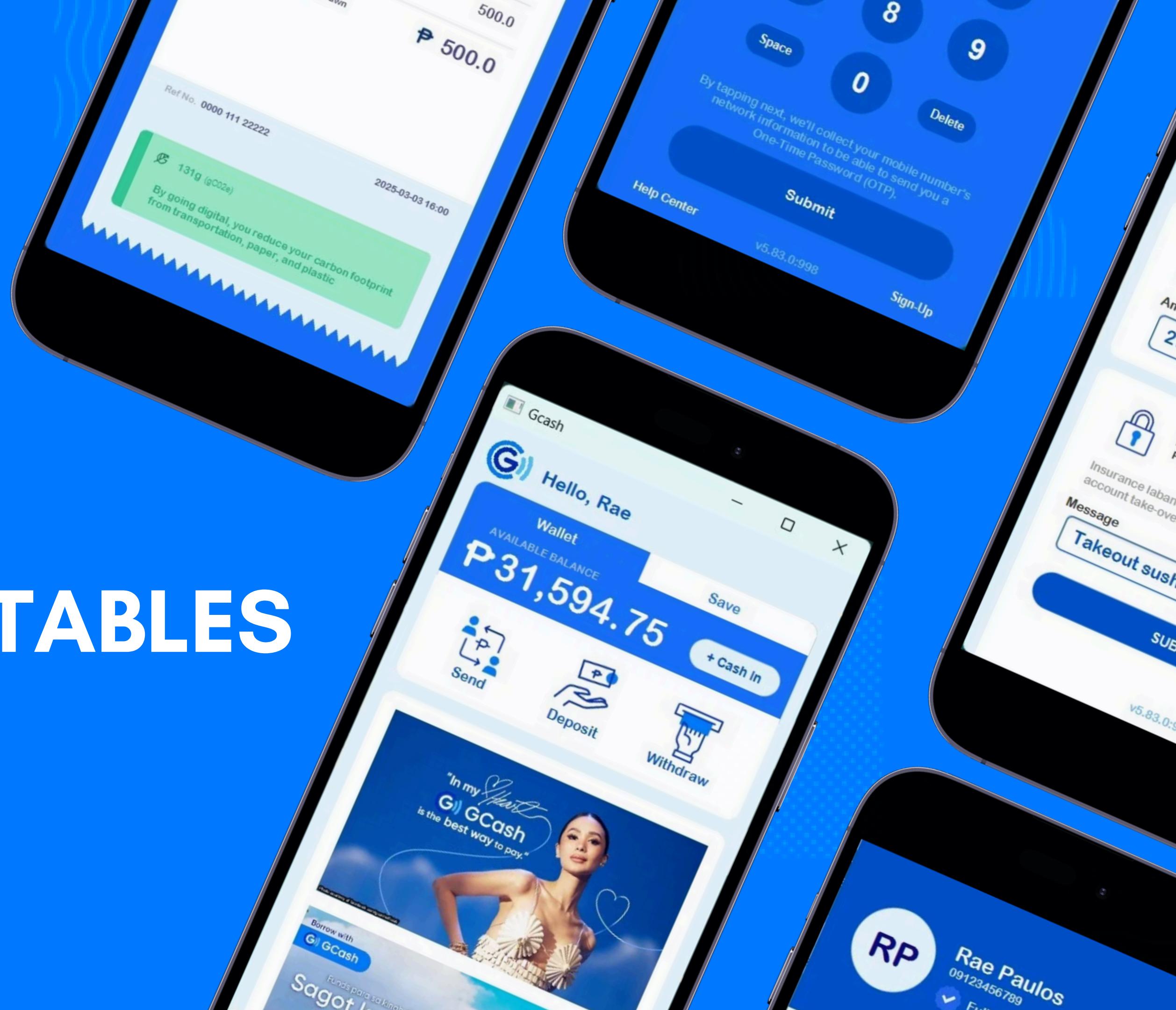
Amount	Date
100.00	2025-03-02 02:31:57
50.00	2025-03-02 10:27:51
5.00	2025-03-03 15:59:48

# GCASH ENTITY RELATIONSHIP DIAGRAM





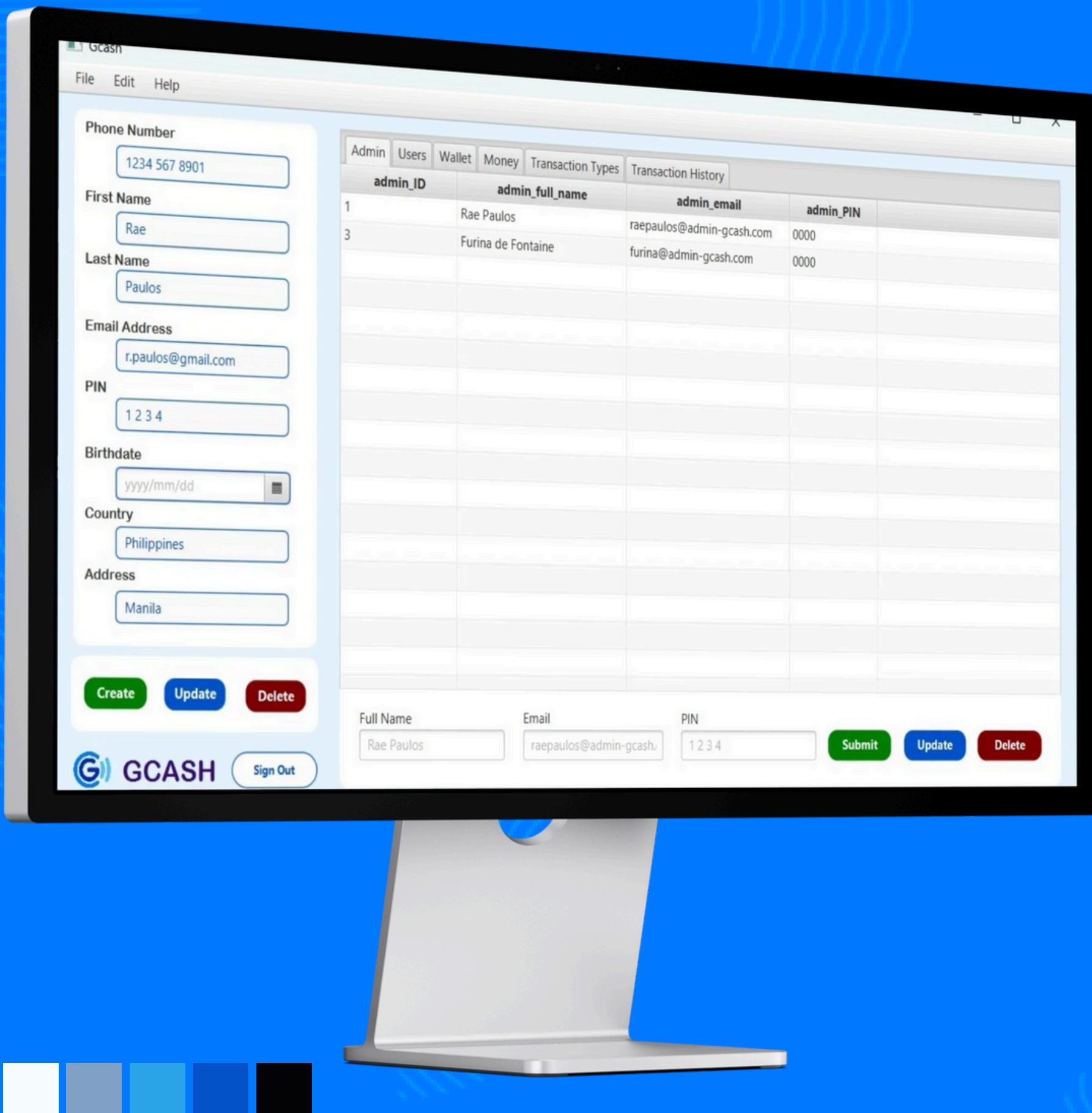
# GCASH PURPOSE OF TABLES



# ADMIN TABLE

Admins are those capable of creating, reading, updating, and deleting accounts.

They are the only ones who has access to the admin side of Gcash.



# USER TABLE

The user table are the ones who uses Gcash.  
They are the ones who uses the main three  
features of the app.

Phone Number  
1234 567 8901

First Name  
Rae

Last Name  
Paulos

Email Address  
r.paulos@gmail.com

PIN  
1 2 3 4

Birthdate  
yyyy/mm/dd

Country  
Philippines

Address  
Manila

Create   Update   Delete

**G** GCASH Sign Out

phone_number	first_name	last_name	email_address	PIN	birthdate	country	address
091122334000	Scaramouche	Wanderer	scaramouche.wanderer@...	3941	2001-01-03	Japan	Inazuma
091122334455	Veronica	Rivera	veronica.rivera@gmail.com	9871	2005-11-14	Philippines	Cavite
091234098765	Lucas	Domingo	lucas.domingo@gmail.com	6784	2003-04-27	Philippines	Pasig
091234567101	Jon	Snow	jon.snow@gmail.com	1001	1992-12-03	United King...	Winterfell
091234567111	Kazuha	Kaedehara	kazuha.kaedehara@gmai...	1023	2000-10-29	Japan	Inazuma
091234567890	Sophia	Gonzales	sophia.gonzales@icloud....	9876	2006-07-12	Philippines	Taguig
091698231945	Jared	Pilapil	jared.pilapil@example.com	4352	2005-03-09	Philippines	Quezon
092122334000	Nahida	Buer	nahida.buer@yahoo.com	0013	2002-10-27	Iran	Sumeru
092209876543	Angelica	Ramos	angelica.ramos@outlook....	6543	2005-11-10	Philippines	General Sa
092233445111	Hu	Tao	hu.tao@yahoo.com	1124	2001-07-15	China	Liyue
092345678202	Daenerys	Targaryen	daenerys targaryen@yah...	2002	1990-07-20	United King...	Dragonsto
092345678222	Xiao	Alatus	xiao.alatus@yahoo.com	5674	2000-04-17	China	Liyue
092345678901	Andrea	Reyes	andrea.reyes@yahoo.com	6789	2005-08-30	Philippines	Cebu
093234566741	Rae	Paulos	r.paulos@admin-gcash.c...	1989	2004-07-05	Philippines	Quezon
093344556222	Eula	Lawrence	eula.lawrence@outlook.c...	2235	2000-10-25	Germany	Mondstadt
093456789303	Tyrion	Lannister	tyrion.lannister@outlook....	3003	1989-06-02	United King...	Casterly Ro
093456789321	Marco	Sorriano	marco.deleon@outlook.c...	1298	2004-01-09	Philippines	San Juan
093456789333	Zhongli	Morax	zhongli.morax@outlook....	9876	1999-12-31	China	Liyue
094512346789	Miguel	Santos	miguel.santos@gmail.com	5678	2003-11-21	Philippines	Makati
094567890404	Arya	Stark	arya.stark@gmail.com	4004	1995-03-11	United King...	Winterfell
094567890444	Raiden	Ei	raiden.shogun@gmail.com	2301	2000-06-26	Japan	Inazuma

# WALLET TABLE

The wallet table is where the data for storing user balance is located.

phone_number	first_name	last_name	balance	currency
091122334000	Scaramouche	Wanderer	9032.50	Philippine Peso (₱)
091122334455	Veronica	Rivera	5229.98	Philippine Peso (₱)
091234098765	Lucas	Domingo	5631.24	Philippine Peso (₱)
091234567101	Jon	Snow	8725.71	Philippine Peso (₱)
091234567111	Kazuha	Kaedehara	10719.12	Philippine Peso (₱)
091234567890	Sophia	Gonzales	13028.40	Philippine Peso (₱)
091698231945	Jared	Pilapil	3667.35	Philippine Peso (₱)
092122334000	Nahida	Buer	9151.76	Philippine Peso (₱)
092209876543	Angelica	Ramos	63.57	Philippine Peso (₱)
092233445111	Hu	Tao	13356.45	Philippine Peso (₱)
092345678202	Daenerys	Targaryen	9497.33	Philippine Peso (₱)
092345678222	Xiao	Alatus	6755.65	Philippine Peso (₱)
092345678901	Andrea	Reyes	9654.06	Philippine Peso (₱)
093234566741	Rae	Paulos	31019.50	Philippine Peso (₱)
093344556222	Eula	Lawrence	9500.00	Philippine Peso (₱)
093456789303	Tyrion	Lannister	17000.23	Philippine Peso (₱)
093456789321	Marco	Sorriano	10250.00	Philippine Peso (₱)
093456789322	Zhenli	Moray	0.600.00	Philippine Peso (₱)

# MONEY TABLE

Contains different types of supported (and soon to be supported) currencies and its conversion rate.

The image shows a tablet device with a white bezel and a black frame. On the screen, there are two main sections. The left section is a user profile form with fields for Phone Number (1234 567 8901), First Name (Rae), Last Name (Paulos), Email Address (r.paulos@gmail.com), PIN (1234), Birthdate (yyyy/mm/dd), Country (Philippines), and Address (Manila). It includes buttons for Create (green), Update (blue), and Delete (red). The right section is a "MONEY TABLE" showing a list of currencies and their conversion rates relative to the US Dollar. The table has columns for currency and conversion\_rate. The data includes:

currency	conversion_rate
Australian Dollar (A\$)	38.20
Brazilian Real (R\$)	11.30
British Pound (£)	73.10
Canadian Dollar (\$)	40.05
Chinese Yuan (¥)	8.00
Euro (€)	62.50
Hong Kong Dollar (...)	7.40
Indian Rupee (₹)	0.70
Japanese Yen (¥)	0.39
Mexican Peso (MX\$)	3.30
New Zealand Dollar (...)	35.90
Philippine Peso (₱)	1.00
Qatari Riyal (₼)	16.00
Russian Ruble (₽)	0.65
Saudi Riyal (﷼)	15.40
Singapore Dollar (\$\$)	42.30
South Korean Won (...)	0.04
Swiss Franc (CHF)	45.00

Below the table, there is a form to add new currency entries with fields for Currency (US Dollar) and Conversion Rate (1.00), and buttons for Submit (green), Update (blue), and Delete (red).



# SEND TABLE

This table records the transactions made by the user when using the Express Send feature of the app.

The GCash app interface on a tablet. The left sidebar contains personal information and navigation buttons. The right side shows a transaction history table with columns: send\_transaction\_ID, sender\_number, sender\_name, receiver\_number, receiver\_name, amount, and transaction\_date. The table lists 16 transactions between various users like John Doe, Veronica Rivera, and Kazuha Kaedehara.

send_transaction_ID	sender_number	sender_name	receiver_number	receiver_name	amount	transaction_date
2	091122334455	John Doe	091122334000	Scaramouche Wan...	100.00	2025-03-02 01:5
3	091122334455	Veronica Rivera	091122334000	Scaramouche Wan...	200.00	2025-03-02 02:0
4	092233445111	Hu Tao	091234567111	Kazuha Kaedehara	200.00	2025-03-02 23:0
5	091122334000	Scaramouche Wan...	091122334455	Veronica Rivera	1000.00	2025-03-03 12:5
6	091122334455	Veronica Rivera	091122334000	Scaramouche Wan...	1000.00	2025-03-03 12:5
7	091122334000	Scaramouche Wan...	092345678222	Xiao Alatus	300.00	2025-03-03 12:5
8	092345678222	Xiao Alatus	091122334000	Scaramouche Wan...	300.00	2025-03-03 12:5
9	091698231945	Jared Pilapil	091234567101	Jon Snow	20.00	2025-03-03 13:0
10	091234567101	Jon Snow	091698231945	Jared Pilapil	20.00	2025-03-03 13:0
11	093234566741	Rae Paulos	091234098765	Lucas Domingo	2000.50	2025-03-03 13:2
12	091234098765	Lucas Domingo	093234566741	Rae Paulos	2000.50	2025-03-03 13:2
13	097890123707	Cersei Lannister	096677889555	Ayaka Kamisato	100.00	2025-03-03 13:2
14	096677889555	Ayaka Kamisato	097890123707	Cersei Lannister	100.00	2025-03-03 13:2
15	093234566741	Rae Paulos	091234567111	Kazuha Kaedehara	275.25	2025-03-03 15:5
16	091234567111	Kazuha Kaedehara	093234566741	Rae Paulos	275.25	2025-03-03 15:5

Number: 1234 567 8901 Send Amount: 0.00 Submit

# DEPOSIT TABLE

This table records the transactions made by the user when using the deposit feature of the app.

deposit_transaction_ID	depositor_number	depositor_name	amount	transaction_date
1	093456789333	Zhongli Morax	220.00	2025-03-02 02:30:42
2	091122334000	Scaramouche Wan...	200.20	2025-03-02 13:19:29
3	092122334000	Nahida Buer	259.72	2025-03-02 13:23:04
4	092345678202	Daenerys Targaryen	78.45	2025-03-02 13:25:04
5	093456789303	Tyrion Lannister	5000.23	2025-03-02 13:27:07
6	09876543210	Sabrina Carpenter	2500.75	2025-03-02 13:28:28
7	09876543210	Sabrina Carpenter	2500.75	2025-03-02 13:28:34
8	091234567111	Kazuha Kaedehara	50.00	2025-03-02 13:30:12
9	091698231945	Jared Pilapil	80.00	2025-03-02 13:30:39
10	096334891432	RJ Cruz	7500.25	2025-03-02 13:32:28
11	091122334000	Scaramouche Wan...	500.00	2025-03-02 20:29:28
12	099876543210	Paolo Del Rosario	156.00	2025-03-02 22:30:32
13	099876543210	Paolo Del Rosario	670.25	2025-03-02 22:33:39
14	091122334455	Veronica Rivera	252.00	2025-03-03 08:04:56
15	097198231945	Carl Anastacio	100000.00	2025-03-03 13:28:01
16	093234566741	Rae Paulos	200.00	2025-03-03 15:58:28

# WITHDRAW TABLE

This table records the transactions made by the user when using the withdraw feature of the app.

withdraw_transaction_ID	withdrawer_number	withdrawer_name	amount	transaction_date
1	093234566741	Rae Paulos	800.00	2025-03-02 02:31:57
2	093234566741	Rae Paulos	250.00	2025-03-02 10:27:51
3	094567890404	Arya Stark	-560.70	2025-03-02 13:45:21
4	091122334000	Scaramouche Wan...	-500.00	2025-03-02 20:30:00
5	099876543210	Paolo Del Rosario	299.00	2025-03-02 22:34:43
6	091122334455	Veronica Rivera	100.00	2025-03-03 08:05:58
7	093234566741	Rae Paulos	1500.25	2025-03-03 15:59:48
8	093234566741	Rae Paulos	500.00	2025-03-03 16:00:30

# INFORMATION MANAGEMENT FINAL PROJECT

GCASH



USER INTERFACE OVERVIEW, ENTITY RELATIONSHIP DIAGRAM, AND PURPOSES