Regex-Based Pattern Matching in Digital Receipts: Case Study on Split Bill Automation from GoFood Orders

Carlo Angkisan - 13523091[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13523091@std.stei.itb.ac.id, [2]carloangkisan21@gmail.com

***Abstract*—Group food ordering through services like GoFood often presents challenges in fairly splitting the total cost, especially when receipts include discounts, service charges, and complex item details. This paper proposes an automated bill-splitting system that utilizes regular expression (regex)-based pattern matching to extract structured information from GoFood digital receipts in PDF format. The system processes each line of text to identify item quantities, names, unit prices, and total costs, even when the data spans multiple lines. Extracted data is then allocated to each participant, and the system proportionally distributes additional fees based on individual consumption. Built using Python, FastAPI, PyMuPDF, and ReportLab, the system enables users to upload receipts, assign items, and download a PDF summary of individual payments. Testing shows the system is accurate and efficient when applied to GoFood's consistent receipt format. In the future, this approach can be extended to support receipts from other platforms by dynamically adjusting regex patterns.**

Keywords—regular expression; digital receipt; pattern matching; bill splitting; GoFood

I. INTRODUCTION

In today's digital era, nearly all aspects of human life have been digitized, including food purchasing transactions. Food delivery applications such as GoFood have become an integral part of modern lifestyles. The convenience of ordering food from the palm of one's hand has made these platforms widely popular across various demographics, from students to professionals. Transactions that once required direct interaction have now transformed into quick and automated digital processes.

This condition becomes even more relevant during social gatherings with friends or family. In such situations, it is common for people to place a group order through GoFood. Besides being practical, this choice is often driven by the reluctance to leave the house or by time and energy constraints. With just one device and one account, an entire group's food needs can be fulfilled in a single transaction.

However, collective ordering often comes with a recurring problem, splitting the bill. A GoFood digital receipt contains not only the price of each food item, but also additional elements such as delivery fees, service charges, and applied discounts. These factors introduce a level of complexity to the cost-sharing process, especially when done manually. Users must calculate each individual's share based on their respective orders while also proportionally distributing the additional fees and discounts.

If not handled properly, this process can result in miscalculations and unclear responsibilities, which may lead to discomfort or even misunderstandings among group members. Thus, there is a need for a smart solution that can extract relevant information from digital receipts and automate the bill-splitting process.

This paper aims to explore the use of regex-based pattern matching as a method for extracting information from GoFood digital receipts and applying it to an automated split-bill system. By leveraging the consistent textual patterns found in these receipts, details such as item names, quantities, unit prices, and total payments can be accurately identified. This approach is expected to simplify the cost-sharing process in group transactions and provide a practical solution for users of digital food delivery services.

II. THEORETICAL BASIS

2.1. Pattern Matching

Pattern matching is the process of identifying a specific pattern within a given text. Formally, it involves:

- T: a long string (the *text*) consisting of n characters,
- P: a shorter string (the *pattern*) consisting of m characters, where m << n.

  The goal is to find the positions within T where P appears.

Example:

- T: *The quick brown fox **jumps** over the lazy dog*
- P: ***jumps***

Pattern matching is widely used in numerous applications, including text editors, web search engines, image analysis, bioinformatics, spam detection, and increasingly, in the processing of digital documents such as electronic purchase receipts.

Pattern matching can be grouped into three main approaches which are exact matching, approximate matching, and regex-based matching. Exact matching finds patterns that match the text exactly without any differences. For example, searching the word "jumps" in the sentence "The quick brown fox jumps over the lazy dog". Common algorithms used include Naive Search, Knuth-Morris-Pratt (KMP), and Boyer-Moore. Approximate matching allows for minor differences, such as typos or spelling variations. For example, the pattern "color" may still match "colour". Algorithm such as Levenshtein Distance and Edit Distance are commonly used. In regex-based matching patterns are defined using regular expressions to match more complex structures, such as \d+ for sequences of digits. This method is highly flexible and useful for extracting information from unstructured text, such as digital payment receipts discussed in this paper.

2.2. Regular Expression (Regex)

A regular expression, often shortened to regex, is a symbolic method used to search, identify, and process parts of text based on defined patterns. Regex is commonly used in string processing because it can effectively detect both simple and complex text patterns in a flexible way. In the context of digital receipts, regex is especially useful for extracting important information such as the number of items, item names, unit prices, and total costs, even when the text layout is inconsistent or unstructured.

A regex pattern consists of a combination of normal characters and special symbols, known as metacharacters. When used together, they form rules that define what kind of text should be matched. These patterns allow systems to scan through large amounts of text and accurately extract the needed details.

1. Metacharacters and Special Symbols

   Metacharacters are symbols with special meanings in regex. Commonly used metacharacters include:

   - . : matches any single character except newline
   - | : logical OR between patterns
   - * : matches zero or more occurrences
   - + : matches one or more occurrences
   - ? : matches zero or one occurrence
   - ^ : indicates the beginning of a line
   - $: indicates the end of a line
   - () : groups part of a pattern
   - [] : defines a character class

2. Character Classes

   Character classes allow matching one character from a specified set:

Construct	Description
[abc]	matches one of: a, b, or c
[^abc]	matches any character except a, b, or c
[a-zA-Z]	matches any uppercase or lowercase letter
[0-9]	matches any digit

3. Predefined Character Classes

   To simplify expressions, several character classes are predefined:

Construct	Description
.	matches any character (except newline)
\d	matches digits (0–9)
\D	matches non-digit characters
\w	matches alphanumeric characters and underscores
\W	matches non-alphanumeric characters
\s	matches whitespace (spaces, tabs, newlines, etc.)
\S	matches non-whitespace characters

4. Quantifiers and Repetition Operators

   Quantifiers specify how many times an element in a pattern must occur:

Construct	Description
X?	matches X zero or one time
X*	matches X zero or more times
X+	matches X one or more times
X{n}	matches X exactly *n* times
X{n,}	matches X *n* or more times
X{n,m}	matches X between *n* and *m* times

5. Boundary Matchers

   Boundary matchers are used to define the position of patterns in a line or word:

Construct	Description
^	matches the beginning of a line
$	matches the end of a line
\b	matches a word boundary
\B	matches a position that is not a word boundary

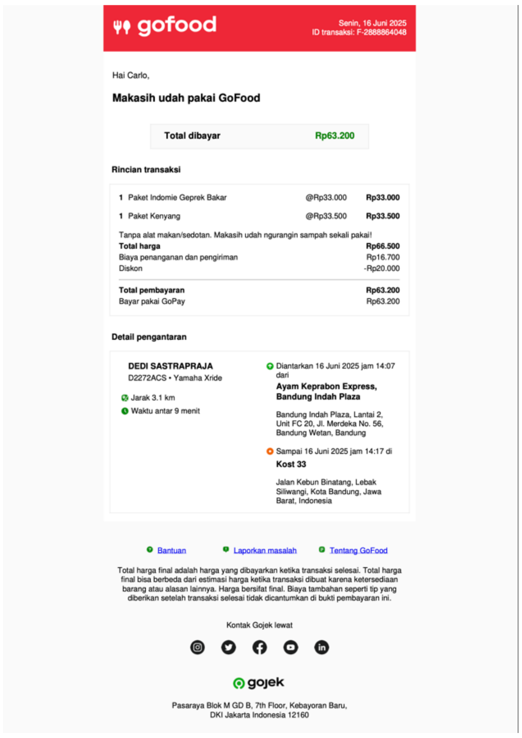
2.3. Digital Receipts in GoFood Service

Digital receipts are electronic proof of payment provided to users after completing a transaction online. Unlike traditional paper receipts, digital receipts are sent through digital platforms such as email or mobile apps and can be accessed anytime. These receipts contain essential details including order summary, item prices, discounts, delivery information, and payment methods.

GoFood is a food delivery service by Gojek that connects users to nearby restaurants through its mobile application. Every completed order on GoFood is accompanied by a digital receipt as a form of documentation and transaction validation.

A typical digital receipt from GoFood is highly structured and includes:

- Item names and quantities
- Unit prices and total cost
- Discounts and additional charges
- Delivery details (address, distance, estimated time)
- Courier information
- Payment method



gofood

Senin, 16 Juni 2025
ID transaksi: F-2888864048

Hai Carlo,

Makasih udah pakai GoFood

Total dibayar	**Rp63.200**

Rincian transaksi

1	Paket Indomie Geprek Bakar	@Rp33.000	**Rp33.000**
1	Paket Kenyang	@Rp33.500	**Rp33.500**

Tanpa alat makan/sedotan. Makasih udah ngurangin sampah sekali pakai!

Total harga	**Rp66.500**
Biaya penanganan dan pengiriman	Rp16.700
Diskon	-Rp20.000
Total pembayaran	**Rp63.200**
Bayar pakai GoPay	Rp63.200

Detail pengantaran

DEDI SASTRAPRAJA
D2272ACS • Yamaha Xride

Jarak 3.1 km
Waktu antar 9 menit

Diantarkan 16 Juni 2025 jam 14:07 dari
Ayam Keprabon Express, Bandung Indah Plaza
Bandung Indah Plaza, Lantai 2, Unit FC 20, Jl. Merdeka No. 56, Bandung Wetan, Bandung

Sampai 16 Juni 2025 jam 14:17 di
Kost 33
Jalan Kebun Binatang, Lebak Siliwangi, Kota Bandung, Jawa Barat, Indonesia

Bantuan Laporkan masalah Tentang GoFood

Total harga final adalah harga yang dibayarkan ketika transaksi selesai. Total harga final bisa berbeda dari estimasi harga ketika transaksi dibuat karena ketersediaan barang atau alasan lainnya. Harga bersifat final. Biaya tambahan seperti tip yang diberikan setelah transaksi selesai tidak dicantumkan di bukti pembayaran ini.

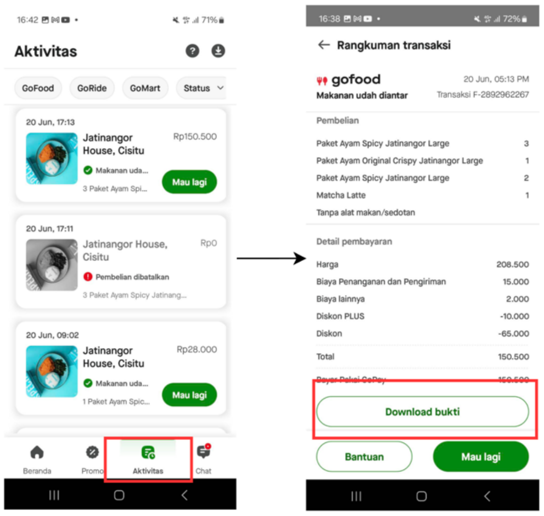
Kontak Gojek lewat

gojek

Pasaraya Blok M GD B, 7th Floor, Kebayoran Baru, DKI Jakarta Indonesia 12160

Fig. 1 GoFood Digital Receipt
(Source: Author's documentation)

Digital receipts are also accessible directly through the Gojek app. Users can open the *Aktivitas* tab, select a completed order, and tap the *Download bukti* button to retrieve the receipt in digital format.



16:42 71%

Aktivitas

GoFood GoRide GoMart Status

20 Jun, 17:13
Jatinangor House, Cisitu Rp150.500
Makanan uda...
3 Paket Ayam Spi...
Mau lagi

20 Jun, 17:11
Jatinangor House, Cisitu Rp0
Pembelian dibatalkan
3 Paket Ayam Spicy Jatinang...

20 Jun, 09:02
Jatinangor House, Cisitu Rp28.000
Makanan uda...
1 Paket Ayam Spic...
Mau lagi

Beranda Promo Aktivitas Chat

16:38 72%

Rangkuman transaksi

gofood 20 Jun, 05:13 PM
Makanan udah diantar Transaksi F-2892962267

Pembelian

Paket Ayam Spicy Jatinangor Large	3
Paket Ayam Original Crispy Jatinangor Large	1
Paket Ayam Spicy Jatinangor Large	2
Matcha Latte	1
Tanpa alat makan/sedotan	

Detail pembayaran

Harga	208.500
Biaya Penanganan dan Pengiriman	15.000
Biaya lainnya	2.000
Diskon PLUS	-10.000
Diskon	-65.000
Total	150.500
Bayar Pakai GoPay	150.500

Download bukti

Bantuan Mau lagi

Fig. 2 Accessing and Downloading Digital Receipts from the Gojek App
(Source: Author's documentation)

2.4. Split Bill Automation

A split bill is the process of dividing the total cost of a transaction among several individuals who share in the purchase. This practice is common in everyday situations such as ordering food together, splitting transportation fees, or participating in group shopping. The goal is to ensure that each person pays only for the portion they used or agreed upon.

However, manually splitting a bill often presents various challenges. Users must calculate the total cost, include additional fees such as delivery or taxes, and divide item prices based on quantity and consumption. Errors in calculation or allocation are frequent, especially when there are many items or shared orders.

To address these challenges, automated split bill systems have been developed. In such systems, users can upload a receipt, such as a digital receipt from a food delivery service. The system extracts item names, quantities, and prices using techniques like regular expressions (regex). Users can then assign items to participants, and the system automatically calculates each person's total payment. Split bill automation offers benefits such as time efficiency, improved accuracy, and greater transparency in group transactions, making cost-sharing more practical and equitable.

III. IMPLEMENTATION

3.1. Research Limitation

This paper specifically limits its scope to the processing of digital receipts in PDF format obtained from the GoFood service. GoFood is selected due to its relatively consistent receipt structure, which makes it easier to identify patterns such as item quantity, product name, unit price, and total amount. This limitation is applied because receipts from other services or payment systems often exhibit highly varied and non-standardized text structures, making it difficult to generalize the extraction process using Regular Expression (regex)-based methods.

3.2. Automated Split Bill Workflow

The implementation of the automated split bill system is carried out through a structured series of processes, beginning with the user uploading a digital receipt in PDF format. The system will process the receipt and extract important information using regular expression (regex)-based pattern matching techniques. Extracted information includes item names, purchase quantities, unit prices, and total prices. Once the data is obtained, users can add a list of participants involved in the transaction and assign each item to one or more participants. The final step of this system is to calculate the total cost to be paid by each participant based on the items consumed, and to display and export the results in the form of a bill summary.

In its implementation, this system utilizes several supporting Python libraries. The fitz library from PyMuPDF is used to extract text from PDF files. The re library is used for regex-based pattern matching. FastAPI is used to build a fast and lightweight backend API. Pydantic is used for data validation through input and output models. Meanwhile, ReportLabis utilized to generate PDF documents as a summary of the bill split.

The system's workflow is designed to be user-friendly and minimize manual intervention. With the relatively consistent GoFood receipt patterns, the extraction and bill splitting processes can be performed accurately and quickly. **Figure 3** illustrates the workflow of this system.

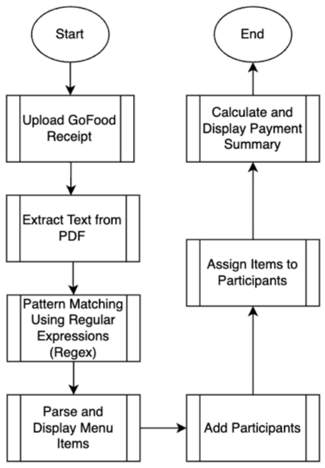
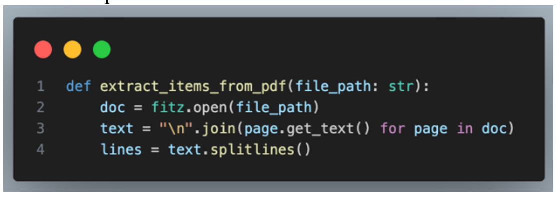

Start
Upload GoFood Receipt
Extract Text from PDF
Pattern Matching Using Regular Expressions (Regex)
Parse and Display Menu Items
Add Participants
Assign Items to Participants
Calculate and Display Payment Summary
End

Fig. 3 Workflow of the automated split bill system
(Source: Author's documentation)

3.3. Text Extraction from PDF Files

After the user uploads a digital receipt in PDF format, the system performs an initial process of extracting the text content from the document. This process is crucial because the necessary information, such as menu names, purchase quantities, unit prices, and total prices, is arranged in text format scattered across multiple lines within the document.

To convert the visual layout of the PDF into a text format that can be further processed, the system uses the PyMuPDF library via the fitz module. Each PDF page is read, then all found text is combined into a single string with line separators. Subsequently, this text is re-separated into a list of lines using the splitlines() function. This structure allows the system to systematically analyze the text line by line. The following code snippet is used to perform this process:



```python
def extract_items_from_pdf(file_path: str):
    doc = fitz.open(file_path)
    text = "\n".join(page.get_text() for page in doc)
    lines = text.splitlines()
```

Fig. 4 Text Extraction Process from PDF Using PyMuPDF
(Source: Author's documentation)

The splitlines() function is used to separate the string based on newline characters, resulting in a list of strings where each element represents one line of extracted text. This is vital as it allows the system to read and analyze the text sequentially line by line.

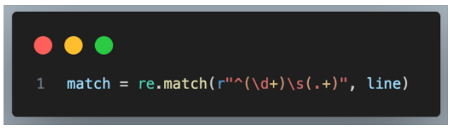
With such a line-by-line list data structure, the system can detect patterns representing important information like item quantities, menu names, unit prices, and total prices. This extraction format serves as the foundation for the next stage, which is pattern matching using regular expressions (regex), enabling the system to automatically recognize and group data based on the typical semi-structured format of GoFood receipts.

3.4. Pattern Matching with Regular Expressions

This step is the core of the data extraction process. Once the text has been successfully extracted from the PDF file, the system must identify key components such as the item name, quantity, unit price, and total price. To do this, the system first examines the consistent structure commonly found in GoFood digital receipts.

In general, each item on a GoFood receipt is presented in the following order: the quantity (a number), the menu name (which may span one or more lines), the unit price (preceded by the symbol @Rp), and the total price (preceded by Rp). Based on this structure, specific regex patterns are created to match the data accurately.

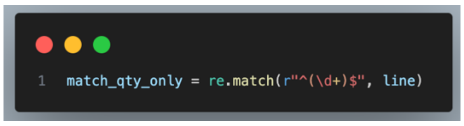
1. **Quantity and Item Name**: Due to irregular spacing in text extracted from PDFs, two patterns are used to handle different cases:
   - The first pattern ^(\d+)\s+(.+) is applied when the quantity and item name are on the same line. In this pattern, the caret symbol ^ matches the start of the line. Then, (\d+) captures one or more digits as the quantity, \s+ matches one or more spaces, and (.+) captures the remaining characters as the item name.



```python
match = re.match(r"^(\d+)\s(.+)", line)
```

Fig. 5 Pattern for Quantity and Item Name in One Line
(Source: Author's documentation)

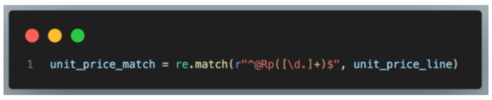
   - The second pattern ^(\d+)$ is used when the line contains only the quantity. The caret ^ still represents the beginning of the line, (\d+) captures the numeric value, and the dollar sign $ ensures that no other characters follow on that line.



```python
match_qty_only = re.match(r"^(\d+)$", line)
```

Fig. 6 Pattern for Quantity in a Separate Line
(Source: Author's documentation)

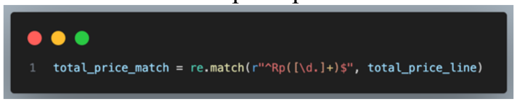
2. **Unit Price**: The pattern ^@Rp([\d.]+)$ is used to match the unit price indicated by the @Rp symbol at the beginning. The dot is used as a thousands separator, so it needs to be cleaned when converted to a number.



```python
unit_price_match = re.match(r"^@Rp([\d.]+)$", unit_price_line)
```

Fig. 7 Regular Expression for Matching Unit Price in GoFood Receipt
(Source: Author's documentation)

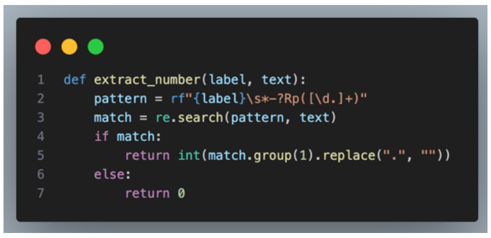
3. **Total Price**: The pattern ^Rp([\d.]+)$ is similar to the previous one, but without the @ symbol. This is used to match the total price per item.



```python
total_price_match = re.match(r"^Rp([\d.]+)$", total_price_line)
```

Fig. 8 Regular Expression for Matching Total Price in GoFood Receipt
(Source: Author's documentation)

4. **Extraction of Specific Values**: For values such as total payment, discount, and handling fees, the system uses fixed labels found on the receipt, such as "Total pembayaran, "Diskon", or "Biaya lainnya" The following function is used to capture these values:



```python
def extract_number(label, text):
    pattern = rf"{label}\s*-?Rp([\d.]+)"
    match = re.search(pattern, text)
    if match:
        return int(match.group(1).replace(".", ""))
    else:
        return 0
```

Fig. 9 Function for Extracting Values Based on Fixed Labels in Receipt
(Source: Author's documentation)

With these patterns, the system can accurately recognize important parts of the digital GoFood receipt, as long as there are no significant format changes. The results of this regex process will be used in the subsequent stages for calculation and cost allocation.

3.5. Implementation of Parsing and Menu Display

After Regex successfully matches patterns, the data is stored in a Python dictionary structure by the extract_items_from_pdf function within parser.py. This data is then serialized into JSON for the frontend. The primary data structure is a list of dictionaries for each item, containing name, quantity, and unit_price. Additionally, the dictionary also includes total_price, handling_fee, other_fee, discount, discount_plus, and total_payment.

Example item data structure from parser.py

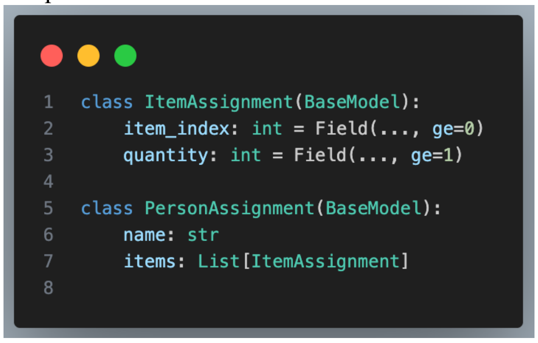
```
items.append({
    "name": name,
    "quantity": qty,
    "unit_price": unit_price
})
```

JSON Payload transmission to the *frontend* from main.py

```
return JSONResponse(content={
    "items": items,
    "total_price": total_price,
    "handling_fee": handling_fee,
    "other_fee": other_fee,
    "discount": discount,
    "discount_plus": discount_plus,
    "total_payment": total_payment,
})
```

3.6. Implementation of Adding People and Assigning Items

Users can add people and assign items through the SplitRequest data structure in models.py, specifically its assignments field. The SplitRequest includes a session_id, a list of parsed Item objects, a list of PersonAssignmentobjects, and overall cost and discount details. Each PersonAssignment contains the name of the person and a list of ItemAssignment objects, which indicate the item_index and quantity of the item taken. The frontend is responsible for user interaction in adding people and assigning items, then sending the complete SplitRequest data to the backend.



```python
class ItemAssignment(BaseModel):
    item_index: int = Field(..., ge=0)
    quantity: int = Field(..., ge=1)

class PersonAssignment(BaseModel):
    name: str
    items: List[ItemAssignment]

```

Fig. 10 Data Structure for Assigning Items to Participants in the Split Bill System
(Source: Author's documentation)

3.7. Implementation of Split Calculation

The bill split calculation is performed by the calculate_split function within the split.py file. This function takes a SplitRequest object as input and returns a list of PersonSplitResult objects. The calculation steps are follows:

1. **Individual Subtotal Calculation**: For each person in data.assignments, the system iterates through their item_assignment. The subtotal is calculated by multiplying the item.unit_price by the item_assignment.quantity. Validation ensures the assigned quantity does not exceed the available item quantity.

$$Subtotal_{person} = \sum (unit\_price_{item} \times quantity_{assigned})$$

2. **Total Subtotal Calculation**: The total subtotal of all items assigned to all people (subtotal_sum) is calculated for proportional fee and discount distribution.

$$Subtotal_{all} = \sum_{all\ persons} Subtotal_{person}$$

3. **Proportional Fee & Discount Calculation**: Handling fees (handling_fee), other fees (other_fee), and total discounts (total_discount = discount + discount_plus) are

proportionally divided based on each individual's subtotal ratio to the overall total subtotal.

$$Propotional\ Ratio = \frac{Subtotal_{person}}{Subtotal_{all}}$$

$$Handling\ Share = round(Propotional\ Ratio \times Handling\ Fee)$$
$$Other\ Share = round(Propotional\ Ratio \times Other\ Fee)$$
$$Discount\ Share = round(Propotional\ Ratio \times Total\ Discount)$$

4. **Final Individual Payment Calculation**: Each person's final payment is calculated by adding their individual subtotal to their proportional share of handling and other fees, then subtracting their proportional discount share.

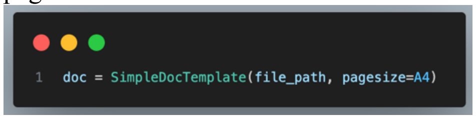
$$Final\ Payment_{person} = Subtotal_{person} + Handling\ Share + Other\ Share - Discount\ Share$$

The result is a PersonSplitResult object containing the person's name, their total amount due, and the items assigned to them.

3.8. Implementation of Extracting Results into a PDF

Once the bill-splitting calculations are completed, the system generates a downloadable summary in PDF format. This functionality is implemented in the generate_split_pdf function located in the pdf_generator.py file. The PDF is created using the ReportLab library, and the process consists of several structured steps:

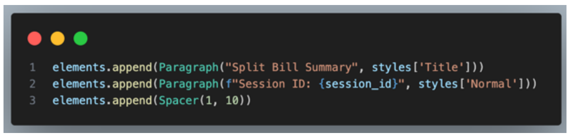
1. **Document Initialization**: A SimpleDocTemplate object is created, specifying the output file path (e.g., output/split_summary_{session_id}.pdf) and A4 page size.



```python
doc = SimpleDocTemplate(file_path, pagesize=A4)
```

Fig. 11 Initializing the PDF Document with SimpleDocTemplate
(Source: Author’s documentation)

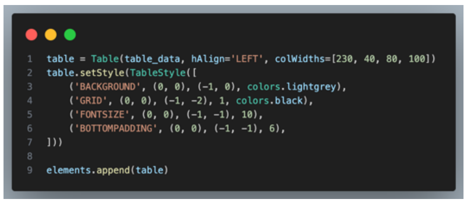
2. **Title and Session Information**: A "Split Bill Summary" title and the session ID are added to the document.



```python
elements.append(Paragraph("Split Bill Summary", styles['Title']))
elements.append(Paragraph(f"Session ID: {session_id}", styles['Normal']))
elements.append(Spacer(1, 10))
```

Fig. 12 Adding Title and Session ID to the PDF Output
(Source: Author’s documentation)

3. **Per-Person Split Details**: Each person gets a section with their name, an item table (item name, qty, unit price, subtotal). Proportional fee shares and discounts are listed, and a bold red "Total Bayar" (Total Payment) is displayed.

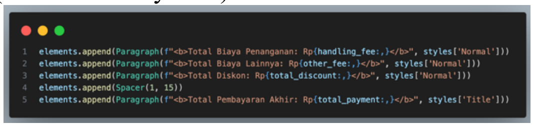


```python
table = Table(table_data, hAlign='LEFT', colWidths=[230, 40, 80, 100])
table.setStyle(TableStyle([
    ('BACKGROUND', (0, 0), (-1, 0), colors.lightgrey),
    ('GRID', (0, 0), (-1, -2), 1, colors.black),
    ('FONTSIZE', (0, 0), (-1, -1), 10),
    ('BOTTOMPADDING', (0, 0), (-1, -1), 6),
]))

elements.append(table)
```

Fig. 13 Styling and Displaying Per-Person Item Table in the PDF
(Source: Author’s documentation)

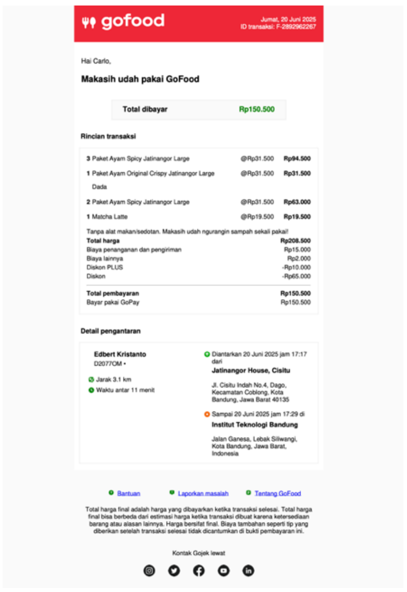
4. **Final Summary**: Displays total handling fee, other fee, total discount, and "Total Pembayaran Akhir" (Final Total Payment) for the whole bill.



```python
elements.append(Paragraph(f"<b>Total Biaya Penanganan: Rp{handling_fee:,}</b>", styles['Normal']))
elements.append(Paragraph(f"<b>Total Biaya Lainnya: Rp{other_fee:,}</b>", styles['Normal']))
elements.append(Paragraph(f"<b>Total Diskon: Rp{total_discount:,}</b>", styles['Normal']))
elements.append(Spacer(1, 15))
elements.append(Paragraph(f"<b>Total Pembayaran Akhir: Rp{total_payment:,}</b>", styles['Title']))
```

Fig. 14 Final Summary Section in the PDF Output
(Source: Author’s documentation)

IV. RESULT AND DISCUSSION



gofood

Jumat, 20 Juni 2025
ID transaksi: F-2892962267

Hai Carlo,

Makasih udah pakai GoFood

Total dibayar	**Rp150.500**

Rincian transaksi

3 Paket Ayam Spicy Jatinangor Large	@Rp31.500	**Rp94.500**
1 Paket Ayam Original Crispy Jatinangor Large Dada	@Rp31.500	**Rp31.500**
2 Paket Ayam Spicy Jatinangor Large	@Rp31.500	**Rp63.000**
1 Matcha Latte	@Rp19.500	**Rp19.500**

Tanpa alat makan/sedotan. Makasih udah ngurangin sampah sekali pakai!

Total harga	**Rp208.500**
Biaya penanganan dan pengiriman	Rp15.000
Biaya lainnya	Rp2.000
Diskon PLUS	-Rp10.000
Diskon	-Rp65.000
Total pembayaran	**Rp150.500**
Bayar pakai GoPay	Rp150.500

Detail pengantaran

Edbert Kristanto
D2077OM •
Jarak 3.1 km
Waktu antar 11 menit

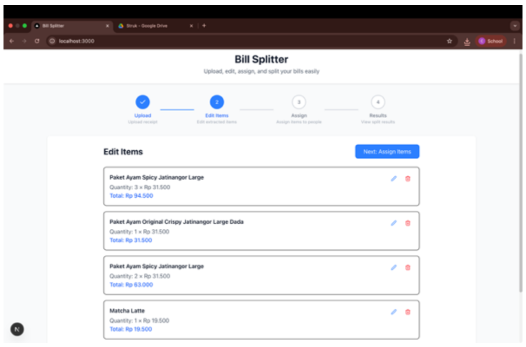
Diantarkan 20 Juni 2025 jam 17:17 dari
Jatinangor House, Cisitu
Jl. Cisitu Indah No.4, Dago, Kecamatan Coblong, Kota Bandung, Jawa Barat 40135

Sampai 20 Juni 2025 jam 17:29 di
Institut Teknologi Bandung
Jalan Ganesa, Lebak Siliwangi, Kota Bandung, Jawa Barat, Indonesia

Total harga final adalah harga yang dibayarkan ketika transaksi selesai. Total harga final bisa berbeda dari estimasi harga ketika transaksi dibuat karena ketersediaan barang atau alasan lainnya. Harga bersifat final. Biaya tambahan seperti tip yang diberikan setelah transaksi selesai tidak dicantumkan di bukti pembayaran ini.

Kontak Gojek lewat

Fig. 12 Sample of GoFood Receipt Used in Multi-Line Pattern Matching Test
(Source: Author’s documentation)



Bill Splitter

Upload, edit, assign, and split your bills easily

Edit Items

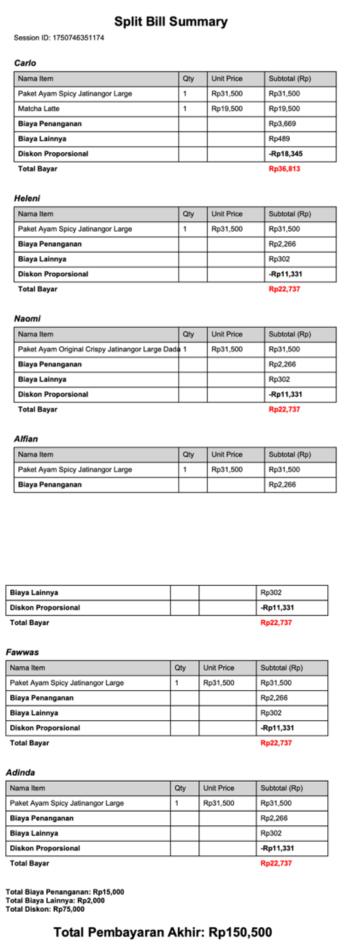
Paket Ayam Spicy Jatinangor Large
Quantity: 3 × Rp 31.500
Total: Rp 94.500

Paket Ayam Original Crispy Jatinangor Large Dada
Quantity: 1 × Rp 31.500
Total: Rp 31.500

Paket Ayam Spicy Jatinangor Large
Quantity: 2 × Rp 31.500
Total: Rp 63.000

Matcha Latte
Quantity: 1 × Rp 19.500
Total: Rp 19.500

Fig. 13 Extracted Menu Items Displayed After Successful Multi-Line Pattern Matching
(Source: Author’s documentation)



Split Bill Summary

Session ID: 1750746351174

Carlo

Nama Item	Qty	Unit Price	Subtotal (Rp)
Paket Ayam Spicy Jatinangor Large	1	Rp31,500	Rp31,500
Matcha Latte	1	Rp19,500	Rp19,500
Biaya Penanganan			Rp3,669
Biaya Lainnya			Rp489
Diskon Proporsional			**-Rp18,345**
Total Bayar			**Rp36,813**

Heleni

Nama Item	Qty	Unit Price	Subtotal (Rp)
Paket Ayam Spicy Jatinangor Large	1	Rp31,500	Rp31,500
Biaya Penanganan			Rp2,266
Biaya Lainnya			Rp302
Diskon Proporsional			**-Rp11,331**
Total Bayar			**Rp22,737**

Naomi

Nama Item	Qty	Unit Price	Subtotal (Rp)
Paket Ayam Original Crispy Jatinangor Large Dada	1	Rp31,500	Rp31,500
Biaya Penanganan			Rp2,266
Biaya Lainnya			Rp302
Diskon Proporsional			**-Rp11,331**
Total Bayar			**Rp22,737**

Alfian

Nama Item	Qty	Unit Price	Subtotal (Rp)
Paket Ayam Spicy Jatinangor Large	1	Rp31,500	Rp31,500
Biaya Penanganan			Rp2,266
Biaya Lainnya			Rp302
Diskon Proporsional			**-Rp11,331**
Total Bayar			**Rp22,737**

Fawwas

Nama Item	Qty	Unit Price	Subtotal (Rp)
Paket Ayam Spicy Jatinangor Large	1	Rp31,500	Rp31,500
Biaya Penanganan			Rp2,266
Biaya Lainnya			Rp302
Diskon Proporsional			**-Rp11,331**
Total Bayar			**Rp22,737**

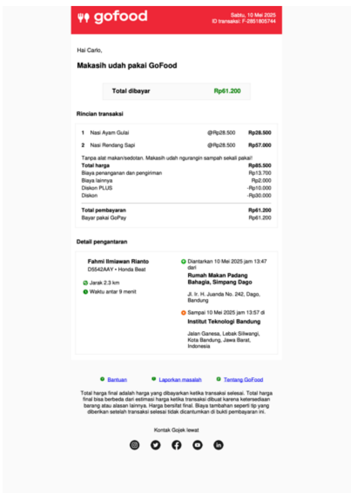
Adinda

Nama Item	Qty	Unit Price	Subtotal (Rp)
Paket Ayam Spicy Jatinangor Large	1	Rp31,500	Rp31,500
Biaya Penanganan			Rp2,266
Biaya Lainnya			Rp302
Diskon Proporsional			**-Rp11,331**
Total Bayar			**Rp22,737**

Total Biaya Penanganan: Rp15,000
Total Biaya Lainnya: Rp2,000
Total Diskon: Rp75,000

Total Pembayaran Akhir: Rp150,500

Fig. 14 Final PDF Output of Bill Splitting from Multi-Line Receipt Format (Scenario 1)
(Source: Author's documentation)



gofood
Sabtu, 10 Mei 2025
ID transaksi: F-2851805744

Hai Carlo,

Makasih udah pakai GoFood

Total dibayar Rp61.200

Rincian transaksi

1	Nasi Ayam Gulai	@Rp28.500	**Rp28.500**
2	Nasi Rendang Sapi	@Rp28.500	**Rp57.000**

Tanpa alat makan/sedotan. Makasih udah ngurangin sampah sekali pakai!

Total harga	**Rp85.500**
Biaya penanganan dan pengiriman	Rp13.700
Biaya lainnya	Rp2.000
Diskon PLUS	-Rp10.000
Diskon	-Rp30.000
Total pembayaran	**Rp61.200**
Bayar pakai GoPay	Rp61.200

Detail pengantaran

Fahmi Ilmiawan Rianto
D5542AAY • Honda Beat
Jarak 2.3 km
Waktu antar 9 menit

Diantarkan 10 Mei 2025 jam 13:47 dari
Rumah Makan Padang Bahagia, Simpang Dago
Jl. Ir. H. Juanda No. 242, Dago, Bandung

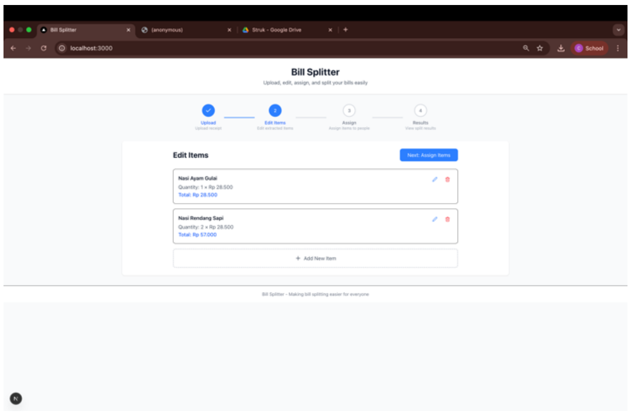
Sampai 10 Mei 2025 jam 13:57 di
Institut Teknologi Bandung
Jalan Ganesa, Lebak Siliwangi, Kota Bandung, Jawa Barat, Indonesia

Bantuan Laporkan masalah Tentang GoFood

Total harga final adalah harga yang dibayarkan ketika transaksi selesai. Total harga final bisa berbeda dari estimasi harga ketika transaksi dibuat karena ketersediaan barang atau alasan lainnya. Harga bersifat final. Biaya tambahan seperti tip yang diberikan setelah transaksi selesai tidak dicantumkan di bukti pembayaran ini.

Kontak Gojek lewat

Fig. 15 Sample of GoFood Receipt Used in Single-Line Pattern Matching Test
(Source: Author's documentation)



Bill Splitter
Upload, edit, assign, and split your bills easily

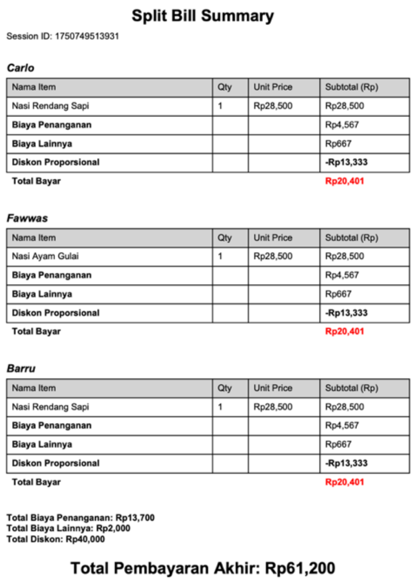
Edit Items

Nasi Ayam Gulai
Quantity: 1 × Rp 28.500
Total: Rp 28.500

Nasi Rendang Sapi
Quantity: 2 × Rp 28.500
Total: Rp 57.000

Bill Splitter - Making bill splitting easier for everyone

Fig. 16 Extracted Menu Items Displayed After Successful Single-Line Pattern Matching
(Source: Author's documentation)



Split Bill Summary

Session ID: 1750749513931

Carlo

Nama Item	Qty	Unit Price	Subtotal (Rp)
Nasi Rendang Sapi	1	Rp28,500	Rp28,500
Biaya Penanganan			Rp4,567
Biaya Lainnya			Rp667
Diskon Proporsional			**-Rp13,333**
Total Bayar			**Rp20,401**

Fawwas

Nama Item	Qty	Unit Price	Subtotal (Rp)
Nasi Ayam Gulai	1	Rp28,500	Rp28,500
Biaya Penanganan			Rp4,567
Biaya Lainnya			Rp667
Diskon Proporsional			**-Rp13,333**
Total Bayar			**Rp20,401**

Barru

Nama Item	Qty	Unit Price	Subtotal (Rp)
Nasi Rendang Sapi	1	Rp28,500	Rp28,500
Biaya Penanganan			Rp4,567
Biaya Lainnya			Rp667
Diskon Proporsional			**-Rp13,333**
Total Bayar			**Rp20,401**

Total Biaya Penanganan: Rp13,700
Total Biaya Lainnya: Rp2,000
Total Diskon: Rp40,000

Total Pembayaran Akhir: Rp61,200

Fig. 17 Final PDF Output of Bill Splitting from Single-Line Receipt Format (Scenario 2)
(Source: Author's documentation)

Testing was conducted on a system developed to extract menu-related information using regular expressions (regex), evaluated under two scenarios based on the layout of digital receipts from GoFood. The objective of this testing was to assess the accuracy of the regex patterns in capturing key details, whether the data appeared on a single line or was spread across multiple lines. The main focus of this discussion is to evaluate the validity of the regex patterns and the system's ability to manage variations in the textual layout of the receipt.

In the first scenario, all components quantity, item name, unit price, and total price, were presented on the same line, as shown in **Figure 15**. The testing results (**Figure 16**) showed that the primary regex pattern used, namely ^(\d+)\s(.+?)\s+@Rp([\d.]+)\s+Rp([\d.]+)$, was able to correctly identify and extract each element. The pattern successfully captured the quantity, followed by the item name, and then the pricing components marked by @Rp and Rp. The final result of the bill-splitting process for this scenario is shown in **Figure 17**. These findings indicate that the pattern is valid and effective for standard single-line formats commonly found in GoFood digital receipts.

The second scenario involved more complex layouts in which the menu information was distributed across several lines, as illustrated in **Figure 12**. For example, the quantity might appear first, followed by the item name on the next line, and pricing information on the line after that. Despite this separation, the system was able to accurately recognize and reconstruct the complete menu entry. The extracted results are presented in **Figure 13**, and the final PDF summary is shown in **Figure 14**. This was achieved by designing the regex logic not to rely strictly on line-by-line parsing, but rather on a contextual multi-line approach. The system retains relevant data from previous lines and intelligently combines them, identifying numeric values as quantities, subsequent lines as item names, and following lines as pricing details. This approach allows the system to accurately process menu data even in receipts with non-uniform layouts.

The system's success in both scenarios highlights the robustness of the regex-based approach in handling various receipt formats. Rather than depending solely on exact line matches, the system incorporates a buffering strategy that enables it to merge information across multiple lines based on logical structure. This method greatly reduces the likelihood of parsing errors as long as the data follows a consistent and interpretable format.

Beyond accurate menu extraction, the system is also capable of proportionally distributing additional costs such as discounts, handling fees, and other charges. This allocation is based on each user's share of the total purchase, ensuring fairness and accuracy. As a result, the system not only assigns menu items correctly but also handles all related financial components in a transparent and equitable manner.

Despite these positive outcomes, it is important to recognize the system's limitations. It was specifically built to handle the structure of GoFood receipts, and therefore may not perform as effectively on receipts from other platforms with different layouts or formatting. The reliability of the extraction process relies heavily on the consistency of the receipt format. Nevertheless, within the scope of this testing, the system has demonstrated strong performance and high accuracy in extracting the intended information.

V. CONCLUSION AND RECOMMENDATION

Regex based pattern matching has proven to be an effective method for extracting structured information from GoFood digital receipts in the context of automated bill splitting. By using carefully designed regular expressions and contextual multiline parsing, the system can accurately identify item quantities, names, unit prices, and total costs, even when the receipt layout is not uniform. The implementation is able to handle both single line and multiline formats reliably, ensuring fair cost distribution including proportional allocation of additional fees and discounts. Moreover, the user friendly interface allows for smooth interaction, from uploading receipts to exporting the final result in a clear PDF summary. These features together simplify the process of bill splitting and reduce the risk of manual calculation errors in group transactions.

Although the current system is specifically designed for GoFood receipts, the approach used has strong potential to be applied to other platforms. Future development is recommended to expand support for receipts from services like GrabFood, ShopeeFood, or retail stores, by extending the regex pattern collection or implementing more flexible matching strategies. To further enhance its capabilities, the system should also be equipped with Optical Character Recognition (OCR). This addition would allow the system to read receipts in image format, such as photos or scans, making it more flexible and useful in various situations.

VI. APPENDIX

The complete source code related to this paper can be accessed through the following GitHub repository:
https://github.com/carllix/split-bill-app

A video explanation of the project is available at the following link:
https://youtu.be/w-jEL1HM5_A

VII. ACKNOWLEDGMENT

The author would like to sincerely thank God Almighty for His blessings and grace during the writing of this paper, which made its completion possible. The author also wishes to express deep appreciation to Dr. Ir. Rinaldi Munir, M.T., lecturer of the IF2211 Algorithm Strategy course, Class K02, for his valuable guidance and support. The online learning resources he provided were especially helpful in improving the author's understanding of the course material.

The author is also very grateful to their parents for their continuous support, encouragement, and prayers, which have been a great source of motivation and strength throughout this journey.

Finally, the author would like to thank their friends, especially the Split Bill group, for the enjoyable moments spent together. Those experiences brought renewed energy and made the process of writing this paper more enjoyable.

REFERENCES

[1] Munir, Rinaldi. 2024. "Pencocokan String (String/Pattern Matching)". https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/23-Pencocokan-string-(2025).pdf (Diakses pada 23 Juni 2025)
[2] Khodra, Masayu Leylia. 2024. "String Matching dengan Regular Expression". https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-

2025/24-String-Matching-dengan-Regex-(2025).pdf (Diakses pada 23 Juni 2025).

[3] Wibisono, Yudi dan Khodra, Masayu Leylia. 2020. "Modul Praktikum Kuliah Pengantar Regular Expression". https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/Modul-Praktikum-NLP-Regex.pdf (Diakses pada 23 Juni 2025)

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 24 Juni 2025

[illegible]

Carlo Angkisan
13523091