Penerapan Algoritma String Matching dan Regular Expression pada Bukti Transaksi Digital untuk Otomatisasi Pencatatan Keuangan

Alfian Hanif Fitria Yustanto - 13523073
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: alfian.yustanto@gmail.com , 13523073@std.stei.itb.ac.id

***Abstract*—Transaksi digital menjadi tren yang terus meningkat seiring berkembangnya teknologi dan kemudahan akses layanan keuangan digital. Perilaku masyarakat yang beralih dari transaksi tunai ke transaksi digital mendorong kebutuhan akan sistem pencatatan keuangan yang lebih cerdas dan otomatis. Salah satu pendekatan untuk mencapai hal tersebut adalah melalui penerapan algoritma string matching dan *regular expression* dalam pengolahan bukti pembayaran digital. Dengan memanfaatkan teknik ini, sistem dapat secara otomatis mengenali dan mengekstraksi informasi penting seperti nominal, tanggal transaksi, dan pihak tujuan dari berbagai format bukti pembayaran digital. Hal ini tidak hanya meningkatkan efisiensi pencatatan, tetapi juga meminimalisasi kesalahan input manual dan mendukung pengelolaan keuangan pribadi yang lebih baik.**

Keywords—string matching; regular expression; Otomatisasi pengelolaan keuangan; bukti pembayaran digital; ekstraksi informasi; pencatatan informasi;

I. PENDAHULUAN

Perkembangan teknologi informasi telah membawa perubahan signifikan dalam berbagai aspek kehidupan, termasuk cara masyarakat melakukan transaksi keuangan. Digitalisasi sistem pembayaran memungkinkan pengguna untuk bertransaksi melalui berbagai platform seperti dompet digital, internet banking, hingga layanan pembayaran berbasis kode QR (QRIS). Akibatnya, bukti transaksi tidak lagi berbentuk fisik, melainkan dalam bentuk digital seperti tangkapan layar atau file PDF yang dikirim secara otomatis setelah transaksi berhasil dilakukan.

Seiring dengan maraknya transaksi digital, kebutuhan akan sistem pencatatan keuangan yang efisien dan otomatis juga diperlukan, terutama ketika harus menangani bukti transaksi dari berbagai platform dengan format yang berbeda-beda. Untuk itu diperlukan cara untuk mendapatkan informasi dari berbagai format bukti transaksi dari berbagai platform.

Ekstraksi informasi pada transaksi digital dapat dilakukan pada bukti transaksi yang memiliki pola tertentu. Bukti transaksi digital memiliki sedikitnya informasi mengenai jumlah transaksi, tanggal transaksi, dan penerima transaksi. Bukti transaksi digital memiliki banyak bentuk peletakkan dan cara menampilkan informasi transaksi. String mathcing dapat digunakan untuk mengambil informasi pada bukti transaksi digital. String matching terutama pattern matching dapat mengambil informasi-informasi utama pada sebuah transaksi digital dan melakukan pengolahan untuk kepentingan pencatatan keuangan.

Algoritma string matching dan *regular expression* merupakan dua teknik penting dalam bidang pemrosesan teks yang dapat dimanfaatkan untuk mengekstraksi informasi spesifik dari dokumen tidak terstruktur. Dengan menggunakan teknik ini, sistem dapat mengenali pola seperti nominal uang, tanggal transaksi, dan pihak penerima atau pengirim dari teks hasil OCR (Optical Character Recognition) pada bukti pembayaran.

II. TEORI DASAR

A. Transaksi Digital dan Bukti Transaksi

Transaksi digital adalah proses pertukaran barang, jasa, atau pembayaran yang dilakukan secara elektronik melalui perangkat digital seperti ponsel, komputer, atau aplikasi berbasis internet. Dalam transaksi digital, pembayaran dilakukan secara online atau menggunakan teknologi elektronik lainnya yang memungkinkan pengguna menyelesaikan proses transaksi dengan cepat dan mudah, tanpa perlu bertemu langsung atau menggunakan uang tunai.

Beberapa contoh transaksi digital meliputi penggunaan e-wallet (dompet digital), mobile banking, payment gateway, hingga QR code sebagai media pembayaran. Transaksi digital juga mencakup aktivitas jual beli di platform e-commerce, pengiriman dana, top-up saldo, dan berbagai layanan keuangan digital lainnya.

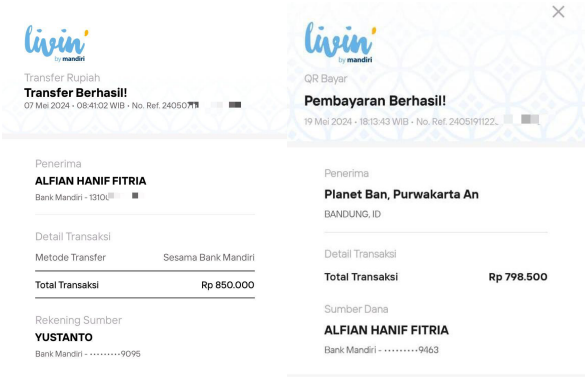
Transaksi digital pada umumnya akan memberikan sebuah bukti transaksi. Bukti transaksi berisi informasi dari suatu transaksi yang telah terjadi. Bukti transaksi dapat berbentuk apapun tergantung dari penyedia layanan. Bukti transaksi dari layanan A dapat berbeda dengan bukti layanan B begitu pula dengan layanan C.

Walau memiliki format atau bentuk berbeda, setiap bukti transaksi memiliki setidaknya informasi mengenai jumlah nominal transaksi, nama penerima, dan tanggal terjadinya

transaksi. Berikut contoh bukti transaksi beberapa *e-commerce* di Indonesia

Gambar 2.1 Bukti transaksi melalui m-banking BCA

Sumber : Dokumen penulis



Gambar 2.1 Bukti transaksi melalui livin Mandiri

Sumber : Dokumen penulis

Perbedaan bentuk dan format bukti transaksi yang dikeluarkan oleh berbagai penyedia layanan keuangan digital menimbulkan tantangan dalam proses pencatatan keuangan, khususnya ketika pencatatan tersebut ingin dilakukan secara otomatis. Setiap penyedia layanan memiliki struktur penyajian informasi yang berbeda-beda, baik dari segi urutan maupun format penulisan data transaksi. Sebagai contoh, informasi nominal transaksi pada suatu layanan dapat ditampilkan di awal dengan format mata uang tertentu, sementara pada layanan lain informasi yang sama diletakkan di bagian bawah dengan format penulisan yang berbeda.

B. *Otomatisasi Pencatatan Keuangan*

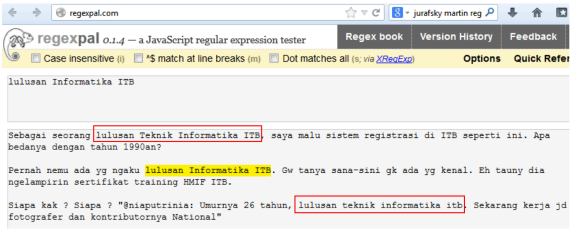
Pencatatan keuangan adalah proses mencatat proses merekam dan mencatat setiap kegiatan keuangan. Tujuan utama dari pencatatan keuangan adalah untuk menciptakan catatan yang akurat, terperinci, dan terstruktur mengenai semua kegiatan keuangan yang terjadi.

Otomatisasi pencatatan keuangan merupakan proses mencatat transaksi keuangan secara otomatis melalui teknologi. Pencatatan keuangan otomatis dapat membaca bukti transaksi dari berbagai penyedia layanan dan mengambil serta menyimpan informasi penting berdasarkan bukti transaksi tersebut. Tujuan utama dari otomatisasi ini adalah meningkatkan efisiensi, mengurangi kemungkinan kesalahan pencatatan, serta mempercepat proses pelaporan dan analisis keuangan, baik untuk kepentingan individu maupun entitas bisnis. Pada manajemen keuangan pribadi, otomatisasi pencatatan keuangan dapat memudahkan serta mempercepat pengolahan atas pengeluaran yang dilakukan selama rentang waktu tertentu.

C. *String/Pattern Matching*

Pencocokan string merupakan suatu proses yang terdiri dari dua komponen utama yaitu text dan pattern. text merupakan sederet string panjang yang berisi kumpulan informasi tertentu sedangkan pattern merupakan string yang ingin dicari pada text. Tujuan dari proses string matching adalah mencari keberadaan suatu pattern pada text. Terdapat dua contoh string matching yaitu exact matching dan regex matching

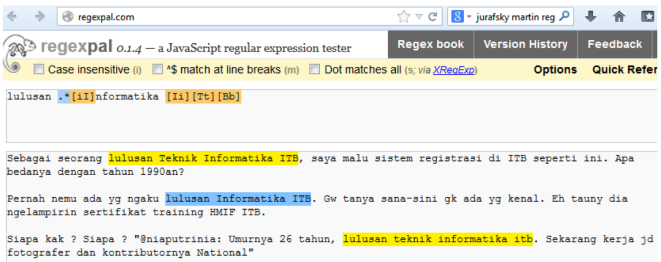
Exact matching berfokus pada pencarian pattern pada suatu text secara pasti. Proses ini hanya akan mencari string pada text yang benar-benar serupa dengan pattern yang ingin dicari.



Gambar 2.3 Contoh exact matching

Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/24-String-Matching-dengan-Regex-(2025).pdf

Regex matching berfokus pada pencarian pola string. Regex matching tidak mencari string pada text yang serupa dengan pattern, namun mencari string yang bersesuaian dengan notasi regex pada pattern.



Gambar 2.4 Contoh regex matching

Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/24-String-Matching-dengan-Regex-(2025).pdf

D. *Regular Expression*

Regular expression (regex) adalah suatu notasi yang digunakan untuk mendeskripsikan pola dari kata yang ingin dicari. Regex dapat digunakan untuk mencari pattern dalam

bentuk tertentu dan tidak harus serupa. Regex memiliki beberapa pola seperti berikut.

expression	matches...
abc	abc (that exact character sequence, but anywhere in the string)
^abc	abc at the *beginning* of the string
abc$	abc at the *end* of the string
a|b	either of a and b
^abc|abc$	the string abc at the beginning or at the end of the string
ab{2,4}c	an a followed by two, three or four b's followed by a c
ab{2,}c	an a followed by at least two b's followed by a c
ab*c	an a followed by any number (zero or more) of b's followed by a c
ab+c	an a followed by one or more b's followed by a c
ab?c	an a followed by an optional b followed by a c; that is, either abc or ac
a.c	an a followed by any single character (not newline) followed by a c
a\.c	a.c exactly
[abc]	any one of a, b and c
[Aa]bc	either of Abc and abc
[abc]+	any (nonempty) string of a's, b's and c's (such as a, abba, acbabcacaa)
[^abc]+	any (nonempty) string which does *not* contain any of a, b and c (such as defg)
\d\d	any two decimal digits, such as 42; same as \d{2}
\w+	a "word": a nonempty sequence of alphanumeric characters and low lines (underscores), such as foo and 12bar8 and foo_1
100\s*mk	the strings 100 and mk optionally separated by any amount of white space (spaces, tabs, newlines)
abc\b	abc when followed by a word boundary (e.g. in abc! but not in abcd)
perl\B	perl when *not* followed by a word boundary (e.g. in perlert but not in perl stuff)

Gambar 2.5 Notasi regex

Sumber : https://devopedia.org/regular-expression

Notasi regex dapat digunakan untuk membentuk pola string yang ingin dicari. Sebagai contoh untuk mencari kata yang memiliki awalan "me" dapat menggunakan pola " $me\backslash w+$ ". Pola tersebut akan mencari semua kata yang berawalan "me" pada suatu teks seperti "memakan" , "memakai", "melempar" , dan lainnya. notasi me akan memastikan pola memiliki karakter "m" dan "e" secara terurut kemudian notasi $\backslash w+$ akan memperbolehkan karakter apapun dengan jumlah berapapun. sehingga $me\backslash w+$ dapat merepresentasikan notasi regex untuk mencari kata yang diawali awalah "me". Sehingga notasi regex dapat digunakan untuk mencari informasit tertentu pada suatu teks.

III. IMPLEMENTASI

Implementasi program otomatisasi pencatatan keuangan dapat memanfaatkan string matching dan regular expression. Program ini dapat membaca berkas bukti transaksi dari berbagai platform dan mengambil serta mengolah informasi dari berkas tersebut. Untuk membuat program ini diperlukan beberapa tahapan yaitu proses pembacaan bukti transaksi, ekstraksi informasi, pengolahan transaksi, dan program utama.

A. Pembacaan Bukti Transfer

Bukti transfer yang akan dibaca dapat berbentuk file PDF maupun gambar seperti .jpg, .png, .jpeg , atau format gambar lainnya. Untuk implementasi pembacaan gambar, penulis memanfaatkan library PIL dan pytesseract. PIL merupakan library python yang digunakan untuk melakukan pemrosesan gambar sedangkan pytesseract merupakan library python yang membungkus Tesseract OCR. OCR (Optical Character Recognition) merupakan suatu library yang dapat membaca teks didalam sebuah gambar.

```python
from PIL import Image, ImageEnhance
import pytesseract


class ImageAdapter:

    @staticmethod
    def ToString(path):
        try:
            image = Image.open(path)

            # === PREPROCESSING ===
            image = image.convert("L")  # Grayscale
            image = ImageEnhance.Contrast(image).enhance(2.0)  # Tingkatkan kontras
            image = image.point(lambda x: 0 if x < 160 else 255, "1")  # Thresholding
            image = image.resize((image.width * 2, image.height * 2))  # Perbesar gambar

            return pytesseract.image_to_string(image, config="--psm 6")
        except Exception as e:
            raise RuntimeError(f"Gagal membaca atau OCR gambar: {e}")
```

Gambar 3.1 Implementasi pembacaan bukti transaksi dalam format gambar/*image*

Sumber : Dokumentasi penulis

Kelas ImageAdapter bertugas mengubah gambar menjadi teks menggunakan OCR. Pada metode ToString, gambar pertama-tama dibuka lalu dikonversi ke skala abu-abu (grayscale) agar lebih optimal. Kemudian gambar yang sudah dalam bentuk grayscale akan ditingkatkan kontrasnya agar huruf terlihat lebih jelas. Treshold digunakan untuk memberi penegasan terhadap perbedaan latar gambar dengan teks. Kemudian ukuran gambar diperbesar agar memeduhkan dan mengoptimalkan pembacaan gambar oleh OCR. Setelah proses ini, pytesseract.image_to_string digunakan untuk mengekstrak teks dari gambar. Jika terjadi kesalahan selama proses, akan dilemparkan exception dengan pesan yang sesuai.

Bukti transfer dapat juga berbentuk sebagai file PDF. Untuk mengimplementasikan pembacaan file PDF penulis menggunakan library fitz atau pymupdf.

```python
import fitz  # PyMuPDF


class PDFAdapter:

    @staticmethod
    def ToString(pdf_path):
        try:
            doc = fitz.open(pdf_path)
            all_text = ""
            for page in doc:
                all_text += page.get_text()
            return all_text
        except Exception as e:
            raise RuntimeError(f"Gagal membaca atau OCR PDF: {e}")
```

Gambar 3.2 Implementasi pembacaan bukti transaksi dalam format PDF

Sumber : Dokumentasi penulis

Kelas PDFAdapter berfungsi sebagai pembaca file PDF. Seperti ImageAdapter, PDFAdapter akan membaca file PDF menggunakan OCR dan akan mengembalikan hasil pembacaan dalam bentuk string.

Setelah membuat adapter untuk PDF dan Image, data terkait bukti transfer perlu disimpan oleh program. Untuk menyimpan

hasil secara terstruktur dibuat suatu model kelas Transaksi yang memiliki atribut yang akan diperlukan.

```
class Transaksi:
    def __init__(
        self, bukti: str, nominal: int = 0, tujuan: str = "", tanggal: str = ""
    ):
        self._bukti = bukti
        self._nominal = nominal
        self._tujuan = tujuan
        self._tanggal = tanggal
```

Gambar 3.3 Implementasi kelas Transaksi

Sumber : Dokumentasi penulis

Kelas tersebut memiliki atribut bukti, nominal,tujuan, dan tanggal. Adapter akan menghasilkan string yang kemudian akan disimpan pada atribut bukti. Untuk atribut nominal, tujuan, dan tanggal akan diisi setelah proses ekstrasi informasi.

B. Ekstraksi Informasi

Setelah berkas bukti transaksi dibaca dan diubah dalam bentuk string. Program diharuskan dapat mengambil informasi-informasi penting yang diperlukan untuk pencatatan keuagan. Implementasi ekstrasi informasi memanfaatkan notasi regex untuk mencari informasi yang diperlukan.

```
class Extractor:
    @staticmethod
    def ekstrak(transaksi: Transaksi) -> Transaksi:
        text = transaksi.bukti
```

Gambar 3.4 Implementasi kelas Extractor

Sumber : Dokumentasi penulis

Kelas extractor dibuat untuk mengimplementasikan proses ekstraksi informasi pada string bukti transaksi menggunakan regex. Kelas Extractor memiliki method ekstrak yang digunakan untuk mengisi atribut nominal, pengirim, dan tanggal pada suatu objek kelas Transaksi yang telah memiliki atribut string bukti. Method ini akan mengembalikan objek Transaksi yang sudah terisi semua atributnya.

```
# === 1. Ekstrak Nominal ===
nominal_match = re.search(
    r"(?:Rp\.?|IDR)[\s]*([0-9a-zA-Z.,]+)", text, re.IGNORECASE
)
if nominal_match:
    raw = nominal_match.group(1).strip()

    # Ganti karakter OCR salah seperti 'd' → '0'
    cleaned = re.sub(r"[^0-9.,]", "0", raw)

    # Hapus akhiran desimal .xx atau ,xx (jika tepat 2 digit)
    cleaned = re.sub(r"[.,](\d{2})$", "", cleaned)

    # Hapus semua titik dan koma pemisah ribuan
    cleaned_int = re.sub(r"[.,]", "", cleaned)

    try:
        transaksi.nominal = int(cleaned_int)
    except ValueError:
        pass
```

Gambar 3.5 Implementasi ekstraksi informasi nominal

Sumber : Dokumentasi penulis

Informasi nominal didapat melalui notasi regex (?:Rp\.?|IDR)[\s]*([0-9a-zA-Z.,]+) notasi tersebut akan mencari string yang diawali oleh "Rp" ,"Rp." atau "IDR" dan diukuti angka 0 — 9 atau karakter a — Z. Contoh string yang dapat diambil notasi tersebut adalah "Rp 10.000,00" , "IDR 100.000 " , dan string lain yang sesuai dengan notasi regex tersebut. Hasil yang berupa string tersebut disimpan pada variabel nominal.

Langkah selanjutnya adalah proses *cleaning*. Hasil string bukti transaksi yang dihasilkan oleh OCR belum tentu sempurna, pada kasus tertentu OCR mungkin melakukan kesalahan dalam pembacaan teks pada suatu gambar. Untuk mengatasai hal ini dilakukan *cleaning* pada string yang sudah disimpan pada variabel nominal. Proses ini akan menghilangkan spasi yang terdapat pada string, menghapus desimal pada hasil, dan melakukan konversi kedalam bentuk integer untuk kemudian disimpan pada atribut nominal pada objek kelas Transaksi.

```
# === 2. Ekstrak dan Normalisasi Tanggal ===
bulan_map = {
    # Bahasa Indonesia (lengkap)
    "januari": 1, "februari": 2, "maret": 3, "april": 4,
    "mei": 5, "juni": 6, "juli": 7, "agustus": 8,
    "september": 9, "oktober": 10, "november": 11, "desember": 12,

    # Bahasa Indonesia (singkatan)
    "jan": 1, "feb": 2, "mar": 3, "apr": 4, "mei": 5,
    "jun": 6, "jul": 7, "agu": 8, "sep": 9,
    "okt": 10, "nov": 11, "des": 12,

    # Bahasa Inggris (singkatan)
    "may": 5, "aug": 8, "oct": 10, "dec": 12,
}


# Format: 20 Mei 2024
date_match = re.search(r"\b(\d{1,2})\s+([A-Za-z]+)\s+(\d{4})", text)
if date_match:
    day, month_str, year = date_match.groups()
    month_str = month_str.lower()
    if month_str in bulan_map:
        day_int = int(day)
        month_int = bulan_map[month_str]
        year_int = int(year)
        transaksi.tanggal = f"{day_int}-{month_int}-{year_int}"
else:
    # Fallback: 20/05/2024 atau 20-05-2024
    fallback = re.search(r"\b(\d{1,2})[/-](\d{1,2})[/-](\d{4})", text)
    if fallback:
        d, m, y = fallback.groups()
        transaksi.tanggal = f"{int(d)}-{int(m)}-{int(y)}"
```

Gambar 3.6 Implementasi ekstraksi informasi tanggal

Sumber : Dokumentasi penulis

Informasi tanggal diperoleh melalui notasi regex (\d{1,2})\s+([A-Za-z]+)\s+(\d{4}) yang dirancang untuk menangkap pola tanggal dalam format seperti "20 Mei 2024". Notasi ini akan mencari string yang terdiri dari dua digit pertama sebagai hari, diikuti nama bulan dalam huruf, dan empat digit terakhir sebagai tahun. Contoh string yang dapat ditangkap oleh pola ini antara lain "7 Maret 2021", "15 April 2020", atau "1 Januari 2023".

Setelah string tanggal berhasil ditangkap, nama bulan akan dikonversi menjadi angka melalui pencocokan terhadap dictionary bulan_map yang berisi kumpulan nama bulan (baik versi lengkap maupun singkatan, dalam bahasa Indonesia dan Inggris). Kemudian, tanggal tersebut disusun ulang dalam format dd-mm-yyyy dan disimpan pada atribut tanggal dari objek kelas Transaksi. Apabila pencocokan format tersebut gagal, dilakukan pencarian alternatif dengan format numerik seperti "20/05/2024" atau "20-05-2024", sehingga memastikan bahwa informasi tanggal tetap dapat diekstrak meskipun ditulis dengan gaya yang berbeda.

```python
# === 3. Ekstrak Tujuan ===
tujuan = ""
tujuan_patterns = [
    r"(?:Penerima|Ke|Pembayaran Ke|Merchant(?: Name)?)[\s:\-]+([A-Z][A-Za-z0-9 .,&'\-]+)",
    r"(?<=\n)([A-Z][A-Z ]{3,})\n(?:Citibank|Bank|Rekening|[A-Z])",
]
for pattern in tujuan_patterns:
    match = re.search(pattern, text, re.IGNORECASE)
    if match:
        tujuan = match.group(1).strip()
        break
if tujuan:
    transaksi.tujuan = tujuan
```

Gambar 3.7 Implementasi ekstraksi informasi Tujuan

Sumber : Dokumentasi penulis

Informasi tujuan diperoleh dengan mencocokkan teks terhadap beberapa pola regex yang telah disusun dalam daftar tujuan_patterns. Pola pertama dirancang untuk menangkap entitas tujuan yang biasanya muncul setelah kata kunci seperti "Penerima", "Ke", "Pembayaran Ke", atau "Merchant Name", yang kemudian diikuti oleh teks kapital yang mewakili nama atau identitas tujuan.

Pola kedua digunakan sebagai alternatif untuk menangkap nama tujuan yang ditulis dalam huruf kapital dan berada di antara dua baris, biasanya diikuti oleh kata seperti "Citibank", "Bank", atau "Rekening". Proses pencarian dilakukan secara berurutan. Jika salah satu pola berhasil menemukan kecocokan, maka bagian yang ditangkap akan dibersihkan dari spasi berlebih dan disimpan ke dalam variabel tujuan kemudian nilai akan disimpan pada atribut tujuan objek kelas Transaksi, yang akan merepresentasikan pihak penerima atau tujuan dari suatu transaksi.

C. Pengolahan Transaksi

Pengolahan informasi dari berkas bukti transaksi dapat diimplementasikan dengan membuat suatu kelas penyimpan dan pengolah objek kelas Transaksi. Kelas ini bertugas membaca seluruh berkas dari folder data, melakukan ekstraksi informasi seperti nominal, tanggal, dan tujuan, lalu menyimpannya dalam bentuk terstruktur. Selain itu, kelas ini juga dapat menyediakan fitur analisis sederhana seperti total pengeluaran dan rekap transaksi berdasarkan bulan.

```python
class TransaksiRepository:
    _instance = None

    def __new__(cls, data_folder="data"):
        if cls._instance is None:
            cls._instance = super(TransaksiRepository, cls).__new__(cls)
            cls._instance._initialized = False
        return cls._instance

    def __init__(self, data_folder="data"):
        if self._initialized:
            return
        self.data_folder = data_folder
        self.list_transaksi = []
        self._initialized = True
```

Gambar 3.8 Implementasi Kelas TransaksiRepository

Sumber : Dokumentasi penulis

Kelas TransaksiRepository merupakan singleton yang berfungsi sebagai penanggung jawab atas segala proses penyimpanan dan pengolahan objek dari kelas Transaksi yang terbentuk. Kelas ini dapat diakses secara global dalam program untuk menyediakan data terkait Transaksi yang dilakukan. Kelas ini memiliki method load data, filter, dan analisis.

```python
def load_data(self):
    self.list_transaksi.clear()
    if not os.path.exists(self.data_folder):
        print(f"📂 Folder '{self.data_folder}' tidak ditemukan.")
        return

    for filename in os.listdir(self.data_folder):
        filepath = os.path.join(self.data_folder, filename)
        transaksi = None

        if filename.lower().endswith(".pdf"):
            try:
                adapter = PDFAdapter(filepath)
                text = adapter.to_string()
                transaksi = Transaksi(bukti=text)
                print(f"✅ PDF dimuat: {filename}")
            except Exception as e:
                print(f"⚠️ Gagal memproses PDF {filename}: {e}")

        elif filename.lower().endswith((".png", ".jpg", ".jpeg")):
            try:
                text = ImageAdapter.ToString(filepath)
                transaksi = Transaksi(bukti=text)
                print(f"✅ Gambar dimuat: {filename}")
            except Exception as e:
                print(f"⚠️ Gagal memproses gambar {filename}: {e}")

        if transaksi:
            transaksi = Extractor.ekstrak(transaksi)
            self.list_transaksi.append(transaksi)
```

Gambar 3.9 Implementasi load data

Sumber : Dokumentasi penulis

Metode load_data digunakan untuk memuat seluruh file PDF dan gambar dari folder data, lalu mengonversinya menjadi objek Transaksi. File PDF diproses menggunakan PDFAdapter, sedangkan gambar diproses dengan ImageAdapter untuk memperoleh teks dari bukti transaksi. Setelah itu, teks diekstrak informasinya (seperti nominal, tanggal, tujuan) menggunakan Extractor, dan hasilnya disimpan ke dalam list_transaksi. Jika folder tidak ditemukan atau file gagal diproses, pesan peringatan akan ditampilkan.

```python
def get_all(self):
    return self.list_transaksi

def filter_per_bulan(self, bulan: int, tahun: int):
    hasil = []
    for transaksi in self.list_transaksi:
        if transaksi.tanggal:
            try:
                dt = datetime.strptime(transaksi.tanggal, "%d-%m-%Y")
                if dt.month == bulan and dt.year == tahun:
                    hasil.append(transaksi)
            except ValueError:
                pass
    return hasil
```

Gambar 3.10 Implementasi get_all dan filter_per_bulan

Sumber : Dokumentasi penulis

Method get_all akan menngembalikan seluruh Transaksi yang telah disimpan sedangkan method filter_per_bulan akan mengembalikan Transaksi yang terjadi pada bulan dan tahun tertentu.

```python
def analisis(self):
    total_semua = 0
    jumlah_transaksi = 0
    nominal_tertinggi = None
    nominal_terendah = None
    bulanan = defaultdict(int)
    jumlah_tidak_lengkap = 0
    list_tidak_lengkap = []
    for t in self.list_transaksi:
        incomplete = (
            t.nominal <= 0
            or not t.tujuan
            or not t.tanggal
            or not self._is_valid_date(t.tanggal)
        )
        if incomplete:
            jumlah_tidak_lengkap += 1
            list_tidak_lengkap.append(t)
            continue
        total_semua += t.nominal
        jumlah_transaksi += 1
        if nominal_tertinggi is None or t.nominal > nominal_tertinggi:
            nominal_tertinggi = t.nominal
        if nominal_terendah is None or t.nominal < nominal_terendah:
            nominal_terendah = t.nominal
        dt = datetime.strptime(t.tanggal, "%d-%m-%Y")
        key = dt.strftime("%Y-%m")
        bulanan[key] += t.nominal
    rata_rata = total_semua / jumlah_transaksi if jumlah_transaksi > 0 else 0
    bulanan = dict(bulanan)
    return {
        "total": total_semua, "jumlah": jumlah_transaksi, "rata_rata": rata_rata,
        "tertinggi": nominal_tertinggi or 0, "terendah": nominal_terendah or 0,
        "bulanan": bulanan,
        "tidak_lengkap": { "jumlah": jumlah_tidak_lengkap,"daftar": list_tidak_lengkap,},
    }
```

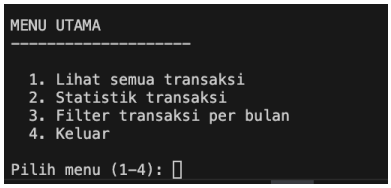
Gambar 3.11 Implementasi analisis

Sumber : Dokumentasi penulis

Method analisis akan mengembalikan data hasil analisis berupa total nominal transaksi, jumlah transaksi, rata-rata transaksi, nominal transaksi tertinggi, nominal transaksi terendah, dan data transaksi per bulan.

D. Program Utama

Program utama dari Pencatatan keuangan otamatis merupakan implementasi dari penggunaan kelas-kelas yang telah dibuat sebelumnya seperti Transaksi, TransaksiRepository, PDFAdapter, ImageAdapter, dan Extractor. Program utama dibuat pada file main.py dan didesain agar mudah digunakan secara interaktif oleh pengguna.

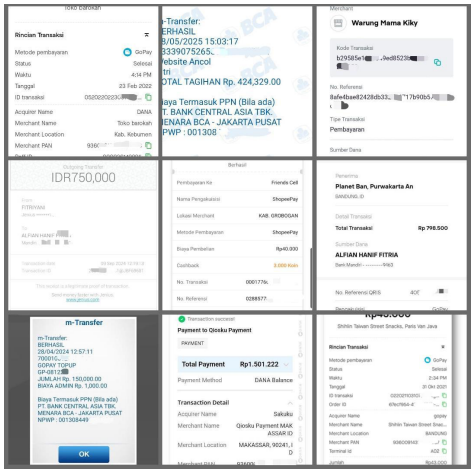


Gambar 3.12 Tampilan awal program utama

Sumber : Dokumentasi penulis

IV. HASIL DAN PEMBAHASAN

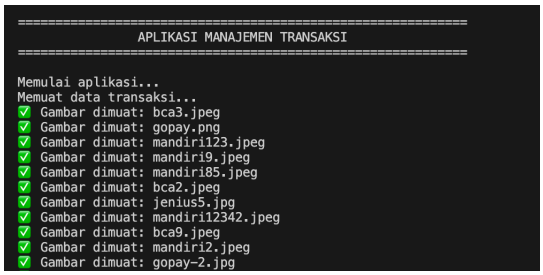
Hasil pengujian dilakukan dengan menyiapkan sekumpulan berkas yang akan digunakan sebagai data test. Berkas berupa gambar dari bukti transaksi yang berasal dari berbagai platform penyedia layanan transaksi digital.



Gambar 4.1 Kumpulan format berkas bukti transfer yang akan diuji

Sumber : Dokumentasi penulis

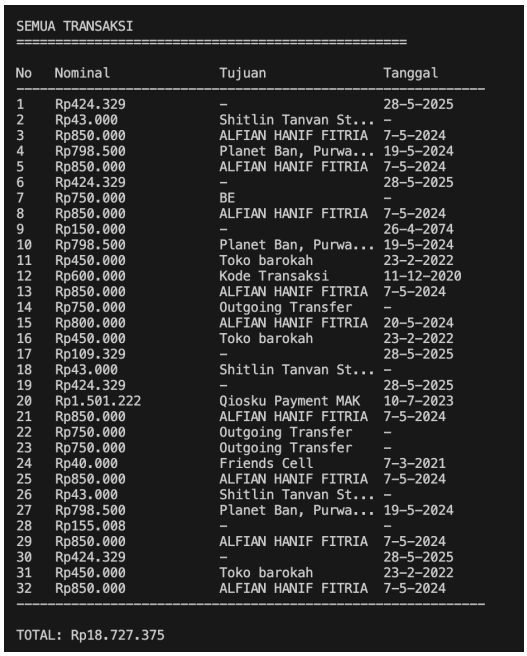
Proses pengujian pertama dilakukan dengan menjalankan program utama untuk melakukan pembacaan terhadap seluruh berkas bukti transaksi yang sudah disiapkan.



Gambar 4.2 Hasil pengujian pembacaan file gambar

Sumber : Dokumentasi penulis

Pembacaan gambar berhasil dilakukan dan telah berhasil menyimpan data transaksi pada TransaksiRepository. Pengujian selanjutnya adalah memerika seluruh transaksi yang telah dimuat.



Gambar 4.3 Hasil pengujian pemeriksaan hasil penyimpanan data transaksi

Sumber : Dokumentasi penulis

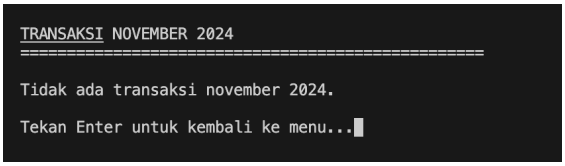
Berdasarkan hasil penyimpanan transaksi. Terdapat beberapa Transaksi yang nilai atributnya tidak lengkap. Hal ini dapat disebabkan oleh beberapa hal seperti format bukti transaksi yang sangat berbeda dari format pada umumnya atau kegagalan OCR dalam mendeteksi teks pada sebuah image sehingga informasi yang didapat menjadi tidak lengkap.

Pengujian selanjutnya adalah melakukan filter terhadap transaksi berdasarkan bulan dan tahun tertentu. Pengujian akan dilakukan dua kali pengujian pertama akan mencari transaksi pada bulan Mei tahun 2024 dan pengujian ke dua akan mencari transaksi pada bulan November 2024.

```
TRANSAKSI MEI 2024
==================================================

No   Nominal             Tujuan              Tanggal
------------------------------------------------------------
1    Rp850.000           ALFIAN HANIF FITRIA  7-5-2024
2    Rp798.500           Planet Ban, Purwa... 19-5-2024
3    Rp850.000           ALFIAN HANIF FITRIA  7-5-2024
4    Rp850.000           ALFIAN HANIF FITRIA  7-5-2024
5    Rp798.500           Planet Ban, Purwa... 19-5-2024
6    Rp850.000           ALFIAN HANIF FITRIA  7-5-2024
7    Rp800.000           ALFIAN HANIF FITRIA  20-5-2024
8    Rp850.000           ALFIAN HANIF FITRIA  7-5-2024
9    Rp850.000           ALFIAN HANIF FITRIA  7-5-2024
10   Rp798.500           Planet Ban, Purwa... 19-5-2024
11   Rp850.000           ALFIAN HANIF FITRIA  7-5-2024
12   Rp850.000           ALFIAN HANIF FITRIA  7-5-2024
------------------------------------------------------------

TOTAL: Rp9.995.500
```

Gambar 4.4 Hasil pengujian pertama filter transaksi

Sumber : Dokumentasi penulis

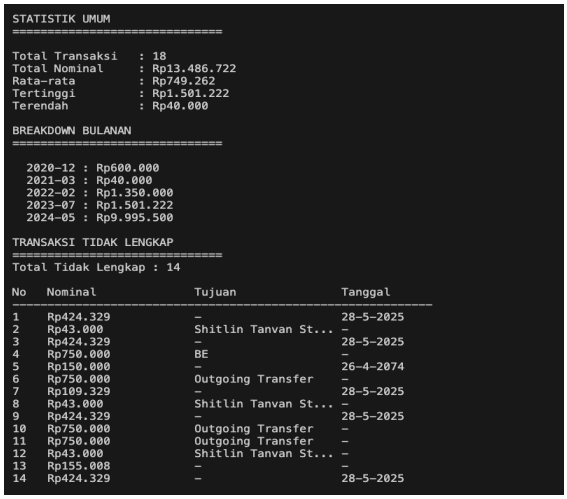


```
TRANSAKSI NOVEMBER 2024
==================================================

Tidak ada transaksi november 2024.

Tekan Enter untuk kembali ke menu...
```

Gambar 4.5 Hasil pengujian ke dua filter transaksi

Sumber : Dokumentasi penulis

Berdasarkan hasil pengujian dapat dilihat bahwa filter transaksi berdasarkan bulan berhasil dilakukan. Pengujian selanjutnya akan melihat hasil analisis program terhadap kumpulan transaksi yang telah disimpan.



```
STATISTIK UMUM
==========================

Total Transaksi   : 18
Total Nominal     : Rp13.486.722
Rata-rata         : Rp749.262
Tertinggi         : Rp1.501.222
Terendah          : Rp40.000

BREAKDOWN BULANAN
==========================

  2020-12 : Rp600.000
  2021-03 : Rp40.000
  2022-02 : Rp1.350.000
  2023-07 : Rp1.501.222
  2024-05 : Rp9.995.500

TRANSAKSI TIDAK LENGKAP
==========================
Total Tidak Lengkap : 14

No   Nominal             Tujuan              Tanggal
------------------------------------------------------------
1    Rp424.329           -                   28-5-2025
2    Rp43.000            Shitlin Tanvan St... -
3    Rp424.329           -                   28-5-2025
4    Rp750.000           BE                  -
5    Rp150.000           -                   26-4-2074
6    Rp750.000           Outgoing Transfer   -
7    Rp109.329           -                   28-5-2025
8    Rp43.000            Shitlin Tanvan St... -
9    Rp424.329           -                   28-5-2025
10   Rp750.000           Outgoing Transfer   -
11   Rp750.000           Outgoing Transfer   -
12   Rp43.000            Shitlin Tanvan St... -
13   Rp155.008           -                   -
14   Rp424.329           -                   28-5-2025
```

Gambar 4.6 Hasil pengujian analisis transaksi

Sumber : Dokumentasi penulis

Program berhasil menampilkan analisis terhadap transaksi yang memiliki informasi atribut lengkap dan transaksi yang tidak memiliki nilai atribut lengkap. Analisis yang ditampilkan berupa statistik umum yang berisi total transaksi lengkap, total nominal, rata-rata, nominal tertinggi, dan nominal terendah. Breakdown bulanan merupakan jumlah nominal transaksi setiap bulannya dan data transaksi tidak lengkap berisi data transaksi yang tidak lengkap.

V. Kesimpulan dan Saran

String matchin dan regular expression dapat diimplementasikan untuk membentuk program pencatatan keuangan otomatis. Notasi regex dapat dibentuk sedemikian rupa sehingga informasi-informasi yang terdapat pada bukti transaksi digital dapat diambil. Masih terdaoat beberapa kasus transaksi yang tidak dapat dipenuhi informasinya. Untuk itu diperlukan optimasi dan pelengkapan terhadap notasi regex yang digunakan agar program dapat membaca lebih banyak format bukti transaksi digital.

VI. Ucapan Terimakasih

Penulis mengucapkan puji syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya yang melimpah dalam proses penulisan makalah ini sehingga dapat diselesaikan tepat waktu. Penulis juga mengucapkan terima kasih kepada Bapak Dr. Rinaldi Munir, sebagai dosen mata kuliah Strategi Algoritma K02. Penulis juga berterima kasih kepada orang tua, keluarga, dan teman yang selalu memberikan dukungan moril dan spiritual, serta menjadi sumber inspirasi dan motivasi selama proses penulisan makalah.

Lampiran

- Pranala github : https://github.com/AlfianHanifFY/Pencatatan_Keuangan.git

Referensi

[1] Khodra, M. L., *"String Matching dengan Regular Expression,"* Bahan kuliah STMIK, [Online]. Tersedia: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/24-String-Matching-dengan-Regex-(2025).pdf. [Diakses pada 24 Juni 2025].K. Elissa, “Title of paper if known,” unpublished.R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.

[2] Diva, M., dan Anshori, M. I., *"Penggunaan E-Wallet Sebagai Inovasi Transaksi Digital: Literatur Review,"* Universitas Trunojoyo Madura, Indonesia. [Online]. Tersedia: https://journal.institercom-edu.org/index.php/multiple/article/download/435/296/1561. [Diakses pada 24 Juni 2025].

[3] SOCS BINUS, *"Regular Expression,"* Artikel Website, [Online]. Tersedia: https://socs.binus.ac.id/2018/11/26/regular-expression/. [Diakses pada 24 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

Alfian Hanif Fitria Yustanto 13523073