자가용이용 여비정산신청서 자동 작성 스킬

개요

출장 후 제출하는 **자가용이용 여비정산신청서(유류비, 통행료)** 엑셀 파일의 새 시트를 자동으로 생성하고, 오피넷과 고속도로 통행료 홈페이지에서 실시간 데이터를 조회하여 입력하는 스킬이다.

입력 데이터

사용자로부터 받아야 하는 정보: 1. **출장일자** (예: 2026.1.30.) 2. **업체(기관)명** (예: (주)세아림테크) 3. **지역** (예: 김천) 4. **첨부 파일**: `__양식__출장여비.xlsx`

고속도로 통행료 로그인 정보 (기본값, 변경 시 사용자 확인): - ID: `●●●●●●●●●●` - PW: `●●●●●●●●●●`

작업 순서

```
1. 엑셀 파일 구조 파악 → 마지막 시트 확인
2. Playwright: 오피넷 → 대구 보통휘발유 평균가격 조회 (출장일자 기준)
3. Playwright: 고속도로 통행료 홈페이지 → 로그인 → 사용내역 조회 (출장일자 기준)
4. 통행료1 (대구→출장지), 통행료2 (출장지→대구) 구분하여 추출
5. 엑셀: 마지막 시트 복사 → 새 시트 생성 → 데이터 입력 → 저장
```

Step 1: 엑셀 구조 파악

```
from openpyxl import load_workbook

wb = load_workbook('파일경로/__양식__출장여비.xlsx')
last_sheet = wb[wb.sheetnames[-1]]
```

입력자료 셀 위치 (마지막 시트 기준, 변경 없음):

셀	내용
O2	출장자명 (차대호) — 수정 불필요
O3	차종/유종 (토레스/휘발유) — 수정 불필요
O4	**출장일자** (datetime)
O5	**업체(기관)명**
O8	**유류비** (보통휘발유 평균가격, 원/ℓ)
O9	**지역**
O10	**통행료1** (대구→출장지)
O11	**통행료2** (출장지→대구)

본문 셀들은 수식으로 O열을 참조하므로 **O열만 수정**하면 자동 계산된다.

Step 2: 오피넷 유류비 조회

`scripts/fetch_opinet.py` 참고. 핵심 로직:

```
from playwright.sync_api import sync_playwright
import re, time

def fetch_daegu_gasoline_price(date_str: str) -> float:
    """
    date_str: 'YYYYMMDD' 형식 (예: '20260130')
    returns: 대구 보통휘발유 평균가격 (float, 원/ℓ)
    """
    year = date_str[:4]
    month = date_str[4:6].lstrip('0') or '1'
    day   = date_str[6:8].lstrip('0') or '1'
```

```
    with sync_playwright() as p:
        browser = p.chromium.launch(headless=True,
                                    args=['--no-sandbox', '--disable-setuid-sandbox'])
        page = browser.new_page()
        page.goto("https://www.opinet.co.kr/user/dopospdrg/dopOsPdrgAreaView.do",
                  timeout=30000)
        time.sleep(4)

        # 일간 선택 및 날짜 설정
        page.evaluate("document.querySelector('input[name=\"TERM\"][value=\"D\"]').click()")
        time.sleep(0.5)
        page.evaluate(f"""
            document.getElementById('STA_Y').value = '{year}';
            document.getElementById('STA_M').value = '{month}';
            document.getElementById('STA_D').value = '{day}';
            document.getElementById('END_Y').value = '{year}';
            document.getElementById('END_M').value = '{month}';
            document.getElementById('END_D').value = '{day}';
            document.getElementById('sta_dt').value = '{date_str}';
            document.getElementById('end_dt').value = '{date_str}';
        """)

        # 폼 제출
        page.evaluate("""
            var form = document.getElementById('search_form');
            form.action = '/user/dopospdrg/dopOsPdrgAreaSelect.do';
            form.target = '_self';
            form.submit();
        """)
        time.sleep(8)

        content = page.content()
        browser.close()

    # ⚠ 반드시 HTML 테이블에서 행 단위로 파싱 — JSON 패턴 사용 금지
    # JSON 구조의 지역 순서가 테이블과 다를 수 있어 오파싱 위험
    # (실제 발생 사례: 대구 JSON 블록 다음이 인천 데이터여서 가격 오파싱됨)
    #
    # 오피넷 테이블 컬럼 순서 (전체 제품 조회 기준):
    #   [0]지역명  [1]고급휘발유  [2]보통휘발유  [3]자동차용경유  [4]실내등유
    tbodies = re.findall(r'<tbody[^>]*>(.*?)</tbody>', content, re.DOTALL)
    for tb in tbodies:
        rows = re.findall(r'<tr[^>]*>(.*?)</tr>', tb, re.DOTALL)
        for row in rows:
            tds = re.findall(r'<td[^>]*>(.*?)</td>', row, re.DOTALL)
            clean = [re.sub(r'<[^>]+>', '', td).replace(',', '').strip()
                     for td in tds]
            clean = [c for c in clean if c]
            if len(clean) >= 3 and clean[0] == '대구':
                # clean[1] = 고급휘발유, clean[2] = 보통휘발유
                return float(clean[2])
    raise ValueError("대구 보통휘발유 가격을 찾을 수 없습니다.")
```

> **주의**: 오피넷 테이블 컬럼 순서 (왼쪽→오른쪽): `지역명` | `고급휘발유` | `보통휘발유` | `자동차용경유` | `실내등유` → `clean[1]` 이 **보통휘발유**

Step 3: 고속도로 통행료 조회

`scripts/fetch_toll.py` 참고. 핵심 로직:

```
import urllib.request, urllib.parse
from playwright.sync_api import sync_playwright
import re, time

HIPASS_ID = "●●●●●●●●●●"
HIPASS_PW = "●●●●●●●●●●"

def fetch_toll_fees(date_str: str, dest_region: str):
```

```
"""
date_str: 'YYYYMMDD' 형식
dest_region: 출장 지역명 (예: '김천')
returns: (toll1: int, toll2: int)
         toll1 = 대구→출장지 요금
         toll2 = 출장지→대구 요금
"""
with sync_playwright() as p:
    browser = p.chromium.launch(headless=True,
                                args=['--no-sandbox', '--disable-setuid-sandbox'])
    ctx = browser.new_context()
    page = ctx.new_page()

    # 로그인 페이지 접근 → 세션 쿠키 획득
    page.goto("https://www.hipass.co.kr/comm/lginpg.do", timeout=30000)
    time.sleep(2)

    # 아이디 입력
    page.fill("#per_user_id", HIPASS_ID)

    # 비밀번호: 보안키보드 우회 (MagicLine4Web 환경)
    pw_field = page.query_selector("#per_passwd")
    pw_field.click()
    page.keyboard.type(HIPASS_PW)
    time.sleep(0.3)

    # IdPwLogin.do AJAX 로그인 (result > 0 이면 성공)
    result = page.evaluate("""
        (function() {
            var r = null;
            $.ajax({
                url: '/comm/IdPwLogin.do', type: 'post', async: false,
                data: {
                    user_id: document.getElementById('per_user_id').value,
                    passwd:  document.getElementById('per_passwd').value
                },
                dataType: 'json',
                success: function(d) { r = d; }
            });
            return r;
        })()
    """)
    if not result or result.get('result', 0) <= 0:
        raise ValueError(f"로그인 실패: {result}")
    time.sleep(1)

    # 쿠키 수집
    cookies = ctx.cookies()
    cookie_str = "; ".join([f"{c['name']}={c['value']}" for c in cookies])

    # 사용내역 페이지 접근 (세션 유지용)
    page.goto("https://www.hipass.co.kr/usepculr/InitUsePculrTabSearch.do",
              timeout=30000)
    time.sleep(2)
    browser.close()

# 직접 POST 요청으로 사용내역 조회
post_data = urllib.parse.urlencode({
    'w': '', 'h': '',
    'card_kind': 'all', 'card_com': 'all', 'ecd_no': 'all',
    'sDate': date_str, 'eDate': date_str,
    'rdo_date_type': 'work', 'date_type': 'work',
    'biz_type': '', 'receipt_time_type': 'display',
    'inc_vat': 'nodisplay', 'in_ic_nm': '', 'out_ic_nm': '',
    'pageSize': '100', 'order_type': 'desc', 'order_item': 'date'
}).encode()

req = urllib.request.Request(
    'https://www.hipass.co.kr/usepculr/UsePculrTabSearchList.do',
    data=post_data, method='POST'
)
```

```
    req.add_header('Cookie', cookie_str)
    req.add_header('Content-Type', 'application/x-www-form-urlencoded')
    req.add_header('Referer',
                   'https://www.hipass.co.kr/usepculr/InitUsePculrTabSearch.do')
    req.add_header('User-Agent', 'Mozilla/5.0 Chrome/120')

    with urllib.request.urlopen(req, timeout=15) as resp:
        html = resp.read().decode('utf-8', errors='ignore')

    # 행 파싱: [번호, 일시, 카드, 차종, 입구IC, 출구IC, 결제방식, 금액, 청구일]
    trs = re.findall(r'<tr[^>]*>(.*?)</tr>', html, re.DOTALL)
    records = []
    for tr in trs:
        tds = re.findall(r'<td[^>]*>(.*?)</td>', tr, re.DOTALL)
        clean = [re.sub(r'<[^>]+>', '', td).strip() for td in tds]
        clean = [re.sub(r'\s+', ' ', c).replace(',', '').replace('원', '')
                 for c in clean if c]
        if len(clean) >= 8:
            try:
                amt = int(clean[7])
                records.append({
                    'time': clean[1],
                    'in_ic':  clean[4],
                    'out_ic': clean[5],
                    'amount': amt
                })
            except (ValueError, IndexError):
                pass

    return _classify_tolls(records, dest_region)


def _classify_tolls(records, dest_region):
    """
    통행료1 = 대구 출발 → 출장지 방향 (시간순 앞쪽)
    통행료2 = 출장지 출발 → 대구 방향 (시간순 뒤쪽)
    출장지 IC는 dest_region 키워드 포함 여부로 판단.
    """
    toll1, toll2 = 0, 0
    for rec in records:
        in_ic  = rec['in_ic']
        out_ic = rec['out_ic']
        amt    = rec['amount']

        # 대구→출장지: 입구가 대구 계열, 출구가 출장지 계열
        daegu_keywords = ['대구', '서대구', '북대구', '남대구', '달성',
                          '다사', '화원', '현풍']
        dest_in_out = dest_region in out_ic or dest_region in in_ic

        is_daegu_start = any(k in in_ic for k in daegu_keywords)
        is_daegu_end   = any(k in out_ic for k in daegu_keywords)

        if is_daegu_start and not is_daegu_end:
            toll1 = amt
        elif is_daegu_end and not is_daegu_start:
            toll2 = amt

    return toll1, toll2
```

> **IC 명칭 참고 (대구 계열)**: 서대구, 북대구, 남대구, 달성, 동대구, 다사, 화원, 현풍, 북달성 출장지 방향은 출구 IC에 지역명 키워드가 포함되는지로 판단

Step 4: 엑셀 새 시트 생성 및 입력

```
import openpyxl, datetime

def write_expense_sheet(xlsx_path, output_path,
                        date_obj,         # datetime.datetime
```

```
                        company_name,     # str
                        region,           # str
                        fuel_price,       # float (원/ℓ)
                        toll1,            # int (원)
                        toll2,            # int (원)
                        sheet_name=None): # 기본값: 업체명 약어
    wb = openpyxl.load_workbook(xlsx_path)
    last_sheet = wb[wb.sheetnames[-1]]

    name = sheet_name or company_name.replace('(주)', '').replace(' ', '')[:6]
    new_ws = wb.copy_worksheet(last_sheet)
    new_ws.title = name

    new_ws['O4'] = date_obj       # 출장일자
    new_ws['O5'] = company_name   # 업체(기관)명
    new_ws['O8'] = fuel_price     # 유류비
    new_ws['O9'] = region         # 지역
    new_ws['O10'] = toll1         # 통행료1
    new_ws['O11'] = toll2         # 통행료2

    wb.save(output_path)
    return output_path
```

시트 이름: 기존에 같은 지역명 시트가 있으면 업체명 약어로 대체.

전체 실행 흐름

```
import datetime

# 1. 입력 파싱
date_str  = "20260130"      # YYYYMMDD
company   = "(주)세아림테크"
region    = "김천"

# 2. 오피넷 유류비 조회
fuel_price = fetch_daegu_gasoline_price(date_str)

# 3. 통행료 조회
toll1, toll2 = fetch_toll_fees(date_str, region)

# 4. 엑셀 입력
date_obj = datetime.datetime.strptime(date_str, "%Y%m%d")
write_expense_sheet(
    xlsx_path     = "원본.xlsx",
    output_path   = "수정본.xlsx",
    date_obj      = date_obj,
    company_name  = company,
    region        = region,
    fuel_price    = fuel_price,
    toll1         = toll1,
    toll2         = toll2
)
```

오류 처리 가이드

상황	조치
오피넷 `대구` 행 미발견	3회 재시도, 실패 시 사용자에게 수동 확인 요청
통행료 로그인 실패 (result ≤ 0)	비밀번호 잠금 여부 확인 후 사용자에게 알림
통행료 toll1/toll2 = 0	사용내역 원문 출력, 사용자가 직접 금액 확인
엑셀 수식 미계산	openpyxl은 수식 저장만 함. Excel 열면 자동 계산됨

환경 요구사항

```
pip install playwright openpyxl --break-system-packages
python -m playwright install chromium
```

- Python 3.10+
- Chromium (headless)
- 네트워크: opinet.co.kr, hipass.co.kr 접속 가능