오타 점검 웹 서비스 기획서

작성일: 2026년 1월 15일 **버전**: 1.0 **작성자**: 김송희

---

1. 프로젝트 개요

1.1 서비스 소개

텍스트를 입력하면 AI(Claude)가 자동으로 오타를 점검하고, 깔끔한 HTML 또는 PDF 보고서로 결과를 제공하는 웹 서비스입니다.

1.2 개발 배경

- 학술 논문, 보고서 등 긴 문서의 오타 점검에 많은 시간 소요
- 기존 맞춤법 검사기는 문맥을 고려하지 못하는 한계
- AI 기반 점검으로 더 정확하고 상세한 피드백 제공 필요

1.3 목표

- 팀/조직 내부에서 간편하게 사용 가능한 오타 점검 도구 제공
- 맞춤법, 띄어쓰기, 조사, 외래어 표기 등 종합적인 검토
- 보고서 형태로 결과 제공 (HTML/PDF 다운로드)

1.4 대상 사용자

- 팀/조직 내부 구성원 (비밀번호 인증)

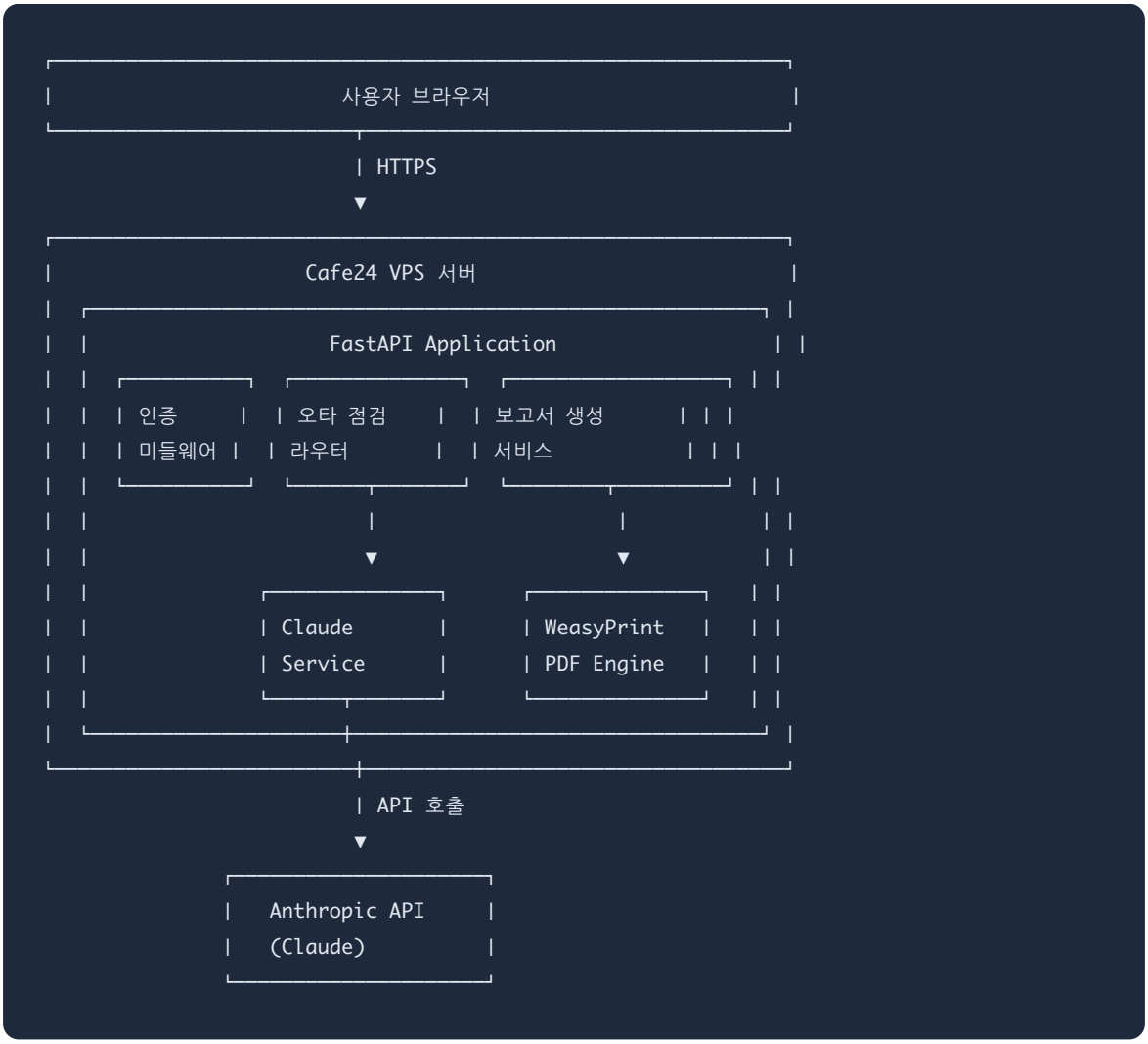
---

2. 기술 스택

구분	기술	설명
Backend	Python + FastAPI	빠르고 현대적인 웹 프레임워크
AI	Claude API (Anthropic)	고품질 한국어 오타 점검
PDF 생성	WeasyPrint	HTML → PDF 변환
Frontend	HTML + Tailwind CSS + JS	반응형 웹 UI
인증	세션 기반	간단한 비밀번호 보호
배포	Cafe24 VPS	클라우드 서버

3. 시스템 아키텍처

3.1 전체 구조



3.2 프로젝트 디렉토리 구조

```
typo-checker/
|
├── app/
|   ├── __init__.py
|   ├── main.py                 # FastAPI 앱 엔트리포인트
|   ├── config.py               # 환경 설정
|   |
|   ├── routers/
|   |   ├── __init__.py
|   |   ├── auth.py             # 인증 라우터
|   |   └── checker.py          # 오타 점검 API 라우터
|   |
|   ├── services/
|   |   ├── __init__.py
|   |   ├── claude_service.py   # Claude API 연동
|   |   └── report_service.py   # HTML/PDF 보고서 생성
|   |
|   ├── templates/
|   |   ├── base.html           # 기본 레이아웃
|   |   ├── index.html          # 메인 페이지
|   |   ├── login.html          # 로그인 페이지
|   |   └── report.html         # 보고서 템플릿
|   |
|   └── static/
|       ├── css/
|       |   └── style.css
|       └── js/
|           └── main.js
|
├── tests/
|   ├── test_checker.py
|   └── test_report.py
|
├── requirements.txt
├── .env                        # 환경 변수 (Git 제외)
├── .env.example                # 환경 변수 예시
├── .gitignore
└── README.md
```

4. 기능 명세

4.1 사용자 인터페이스

4.1.1 로그인 페이지 (`/login`)

- 비밀번호 입력 필드
- 로그인 버튼
- 세션 유지 (브라우저 종료 시까지)

4.1.2 메인 페이지 (`/`)

요소	설명
텍스트 입력 영역	대용량 텍스트 입력 가능한 textarea
출력 형식 선택	HTML / PDF 라디오 버튼
점검 버튼	"오타 점검 시작" 버튼
진행 표시	프로그레스 바 + 상태 메시지
결과 미리보기	점검 완료 후 결과 요약 표시
다운로드 버튼	HTML/PDF 파일 다운로드

4.2 API 명세

4.2.1 인증 API

POST /api/login

```
// Request
{
  "password": "팀비밀번호"
}

// Response (성공)
{
  "success": true,
  "message": "로그인 성공"
}



// Response (실패)
{
  "success": false,
  "message": "비밀번호가 올바르지 않습니다"
}
```

4.2.2 오타 점검 API

POST /api/check

```
// Request
{
  "text": "점검할 텍스트 내용...",
  "format": "html"  // "html" 또는 "pdf"
}

// Response
{
  "success": true,
  "report_id": "abc123",
  "summary": {
    "total_typos": 15,
    "by_type": {
      "맞춤법": 5,
      "띄어쓰기": 6,
      "조사": 2,
      "외래어": 1,
      "문장부호": 1
    }
  },
  "typos": [
    {
      "id": 1,
      "type": "맞춤법",
      "context": "이 문서는 맞춤핼습에 대한 내용입니다.",
      "original": "맞춤핼습",
      "suggestion": "맞춤학습",
      "explanation": "'핼'은 '학'의 오타입니다."
    }
    // ... 추가 오타들
  ]
}
```

4.2.3 보고서 다운로드 API

GET /api/report/{report_id}?format=html

- HTML 파일 반환

GET /api/report/{report_id}?format=pdf

- PDF 파일 반환 (다운로드)

4.3 오타 점검 항목

점검 항목	예시

맞춤법 오류	맞춤핼습 → 맞춤학습
띄어쓰기 오류	20236월 → 2023년 6월
조사 사용 오류	Twitter과 → Twitter와
중복 표현	목표한다 → 목표로 한다
문장 부호 오류	마침표 누락, 이중 공백
외래어 표기법	컨텐츠 → 콘텐츠

5. 핵심 구현 상세

5.1 Claude API 연동

5.1.1 프롬프트 설계

TYPO_CHECK_PROMPT = """
당신은 전문 한국어 교정 편집자입니다.
다음 텍스트의 오타를 꼼꼼히 점검해주세요.

점검 항목

- 맞춤법 오류
- 띄어쓰기 오류
- 조사 사용 오류 (은/는, 이/가, 을/를, 와/과 등)
- 중복 표현
- 문장 부호 오류
- 외래어 표기법 오류

응답 형식

반드시 아래 JSON 형식으로만 응답하세요:

```
{
  "typos": [
    {
      "type": "오류 유형",
      "context": "앞뒤 맥락을 포함한 문장. 오타부분을 볼드로 표시",
      "original": "원본 텍스트",
      "suggestion": "수정 제안",
      "explanation": "오류 이유 설명"
    }
  ]
}
```

오타가 없으면 빈 배열 {"typos": []}을 반환하세요.

```
점검할 텍스트:


{text}
"""
```

5.1.2 청크 분할 로직

```
def split_text_into_chunks(text: str, max_chars: int = 8000) -> list[str]:
    """
    긴 텍스트를 문단 단위로 청크 분할
    - 문단 경계를 유지하여 문맥 보존
    - 각 청크는 max_chars 이하로 유지
    """
    paragraphs = text.split('\n\n')
    chunks = []
    current_chunk = ""

for para in paragraphs:
        if len(current_chunk) + len(para) < max_chars:
            current_chunk += para + "\n\n"
        else:
            if current_chunk:
                chunks.append(current_chunk.strip())
            current_chunk = para + "\n\n"



if current_chunk:
        chunks.append(current_chunk.strip())



return chunks
```

5.2 보고서 생성

5.2.1 HTML 템플릿 스타일

```
/* 보고서 핵심 스타일 */
body {
    font-family: 'Apple SD Gothic Neo', sans-serif;
    line-height: 1.8;
    max-width: 800px;
    margin: 0 auto;
}

.typo-item {
    background: #fefce8;
    border-left: 4px solid #eab308;
    padding: 15px;
    margin: 15px 0;
}



.typo-original {
    color: #dc2626;
    background: #fef2f2;
    padding: 2px 6px;
    font-weight: bold;
}



.typo-suggestion {
    color: #16a34a;
    background: #f0fdf4;
    padding: 2px 6px;
}
```

5.2.2 PDF 생성

```
from weasyprint import HTML

def generate_pdf(html_content: str) -> bytes:
    """HTML을 PDF로 변환"""
    html = HTML(string=html_content)
    return html.write_pdf()
```

6. 보안 고려사항

6.1 인증

- 비밀번호는 환경 변수로 관리
- 세션 기반 인증 (쿠키 + 서버 세션)
- 브루트포스 방지: 5회 실패 시 1분 대기

6.2 API 키 보호

- Anthropic API 키는 서버 측에서만 사용
- 클라이언트에 노출되지 않음
- `.env` 파일은 Git에서 제외

6.3 입력 검증

- 텍스트 길이 제한 (최대 100,000자)
- XSS 방지를 위한 HTML 이스케이프

---

7. 배포 가이드

7.1 서버 요구사항

- OS: Ubuntu 20.04 이상
- Python: 3.10 이상
- RAM: 최소 1GB
- 저장공간: 10GB 이상

7.2 설치 순서

```
# 1. 시스템 패키지 설치
sudo apt update
sudo apt install python3 python3-pip python3-venv
sudo apt install libpango-1.0-0 libpangocairo-1.0-0  # WeasyPrint 의존성
```

2. 프로젝트 디렉토리 생성

```
mkdir -p /home/user/typo-checker
cd /home/user/typo-checker
```

3. 프로젝트 파일 업로드 (scp 또는 git clone)

```
scp -r ./typo-checker/* user@server:/home/user/typo-checker/
```

4. 가상환경 생성 및 활성화

```
python3 -m venv venv
source venv/bin/activate
```

5. 패키지 설치

```
pip install -r requirements.txt
```

6. 환경 변수 설정

```
cp .env.example .env
nano .env  # API 키와 비밀번호 입력
```

7. 서비스 실행 (개발)

```
uvicorn app.main:app --host 0.0.0.0 --port 8000 --reload
```

8. 서비스 실행 (프로덕션 - systemd)

```
sudo nano /etc/systemd/system/typo-checker.service
```

7.3 systemd 서비스 설정

```
[Unit]
Description=Typo Checker Web Service
After=network.target

[Service]
User=user
Group=user
WorkingDirectory=/home/user/typo-checker
Environment="PATH=/home/user/typo-checker/venv/bin"
ExecStart=/home/user/typo-checker/venv/bin/uvicorn app.main:app --host 0.0.0.0 --port 8000
Restart=always


[Install]
WantedBy=multi-user.target
```

```
# 서비스 활성화 및 시작
sudo systemctl enable typo-checker
sudo systemctl start typo-checker
sudo systemctl status typo-checker
```

7.4 Nginx 리버스 프록시 (선택)

```
server {
    listen 80;
    server_name typo.yourdomain.com;

location / {
        proxy_pass http://127.0.0.1:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

---

8. 필요 패키지

requirements.txt

```
fastapi==0.109.0
uvicorn[standard]==0.27.0
anthropic==0.18.0
weasyprint==60.2
python-multipart==0.0.6
jinja2==3.1.3
python-dotenv==1.0.0
itsdangerous==2.1.2
aiofiles==23.2.1
```

---

9. 환경 변수

.env.example

```
# Anthropic API
ANTHROPIC_API_KEY=sk-ant-api03-xxxxxxxx

서비스 인증

SERVICE_PASSWORD=your_team_password_here
```

서버 설정

```
DEBUG=false
HOST=0.0.0.0
PORT=8000
```

보안

```
SECRET_KEY=your_random_secret_key_here
```

---

10. 향후 개선 사항

10.1 단기

☐ 파일 업로드 지원 (txt, docx, hwp) ☐ 점검 이력 저장 및 조회 ☐ 다크 모드 UI

10.2 중기

☐ 사용자별 계정 관리 ☐ 커스텀 점검 규칙 설정 ☐ API 사용량 모니터링

10.3 장기

☐ 실시간 협업 편집 ☐ 다국어 지원 ☐ 자체 AI 모델 학습

부록 A. 예상 비용

Claude API 비용 (2026년 1월 기준)

모델	입력	출력
Claude 3.5 Sonnet	$3 / 1M tokens	$15 / 1M tokens
Claude 3 Haiku	$0.25 / 1M tokens	$1.25 / 1M tokens

예상 사용량 (월 100건, 평균 10,000자/건 기준)

- 입력: ~500K tokens → 약 $1.5
- 출력: ~100K tokens → 약 $1.5
- **월 예상 비용: 약 $3~5**

부록 B. 테스트 체크리스트

☐ 로그인 성공/실패 테스트 ☐ 빈 텍스트 입력 시 에러 처리 ☐ 최대 길이 텍스트 처리 ☐ HTML 보고서 생성 확인 ☐ PDF 보고서 생성 확인 ☐ 한글 폰트 정상 표시 (PDF) ☐ 동시 다중 요청 처리 ☐ API 키 오류 시 에러 처리

문서 끝