오늘점심뭐먹을까?

20221806 이정원

<개발 동기>

“2교시 수업을 듣고 난 12시,, 이제 점심시간인데 뭐 먹지?? “

“오늘은 뭘 먹어야 하지??”

항상 하던 고민들!! 도움을 주기 위한 프로젝트

<구현 방법>

[제공할 기능]

1. 학식이나 학교 근처 음식점들을 리스트 업 하여 메뉴를 선정
2. 단순 랜덤 음식 제안에서 부터 날씨에 따라 제안을 하거나(계획),
3. 몇 개의 메뉴 중 뽑아주는 기능

[구현]

1. 기본 기능들은 C언어를 사용한다.
2. 음식이나 음식점 정보는 CSV파일을 이용하여 저장하고 불러온다.
3. GUI 구현
   = EasyWin32를 이용하여 GUI를 구현

<구현 방법>

[제공할 기능]

1. 학식이나 학교 근처 음식점들을 리스트 업 하여 메뉴를 선정
2. 단순 랜덤 음식 제안에서 부터 날씨에 따라 제안을 하거나(계획),
3. 몇 개의 메뉴 중 뽑아주는 기능

[날씨 추천 관련]

날씨 및 요일 특성이 음식점 메뉴 검색시스템 이용에 미치는 영향에 관한 실증 연구

조찬열 (식신(주) 시스템개발본부) ; 정구임 (식신(주) 서비스사업본부) ; 서양민 (식신(주) 서비스사업본부 사업팀) ; 최혜림 (식신(주) 서비스사업본부 기획팀)

https://scienceon.kisti.re.kr/srch/selectPORSrchArticle.do?cn=JAKO201720636501085&dbt=NART

“본 논문은 날씨 및 요일 변수가 사용자의 메뉴 검색에 영향이 있는지에 대해 연구를 하였는데 영향이 있다는 것을 확인할 수 있었다.”

<개발 환경>

Windows
11
버전 22H2

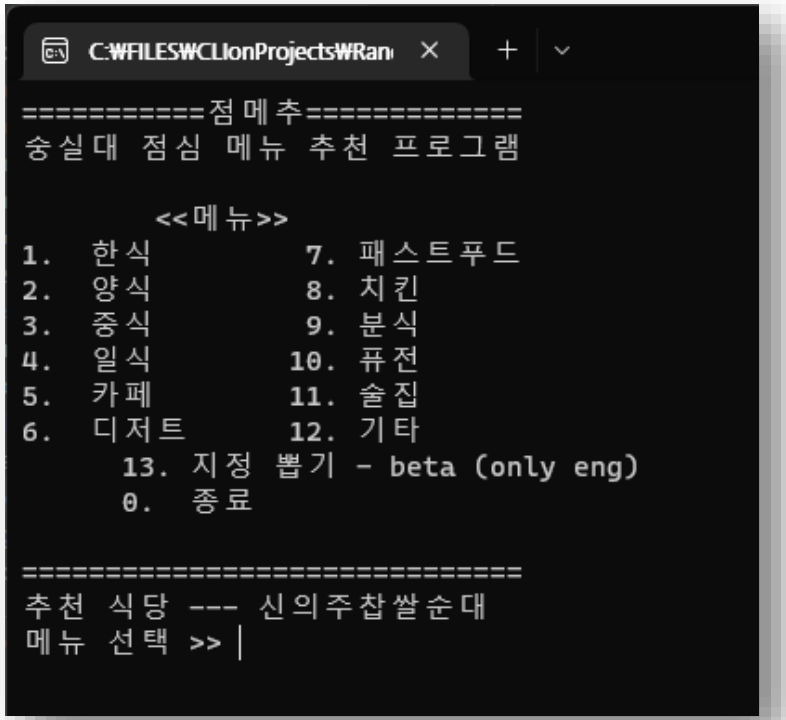
Jetbrains
Clion
22.1.1

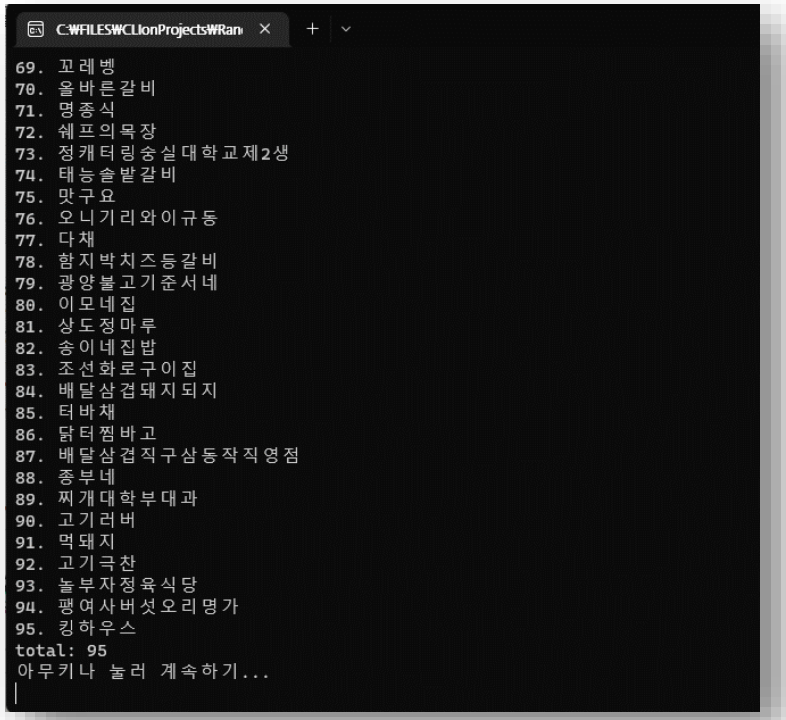
Jetbrains
Pycharm
22.2.1

Visual Studio
2022

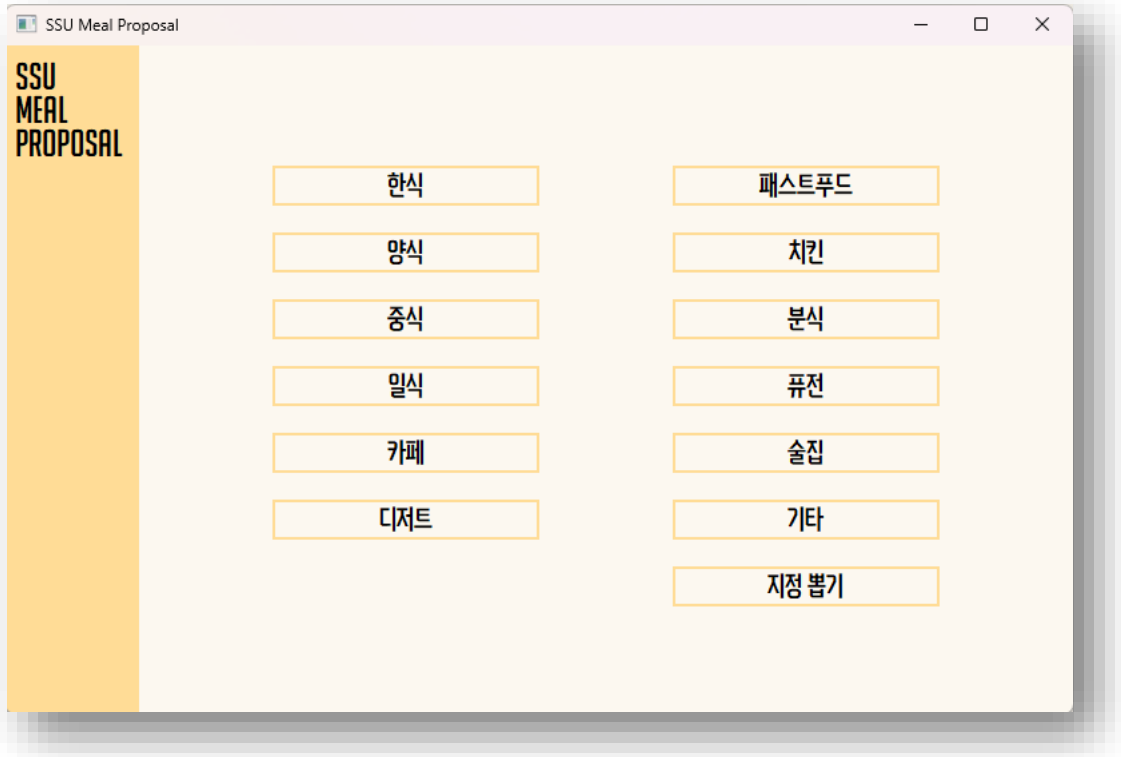
<UI – User Interface>

콘솔 프로그램 (기본 설계)

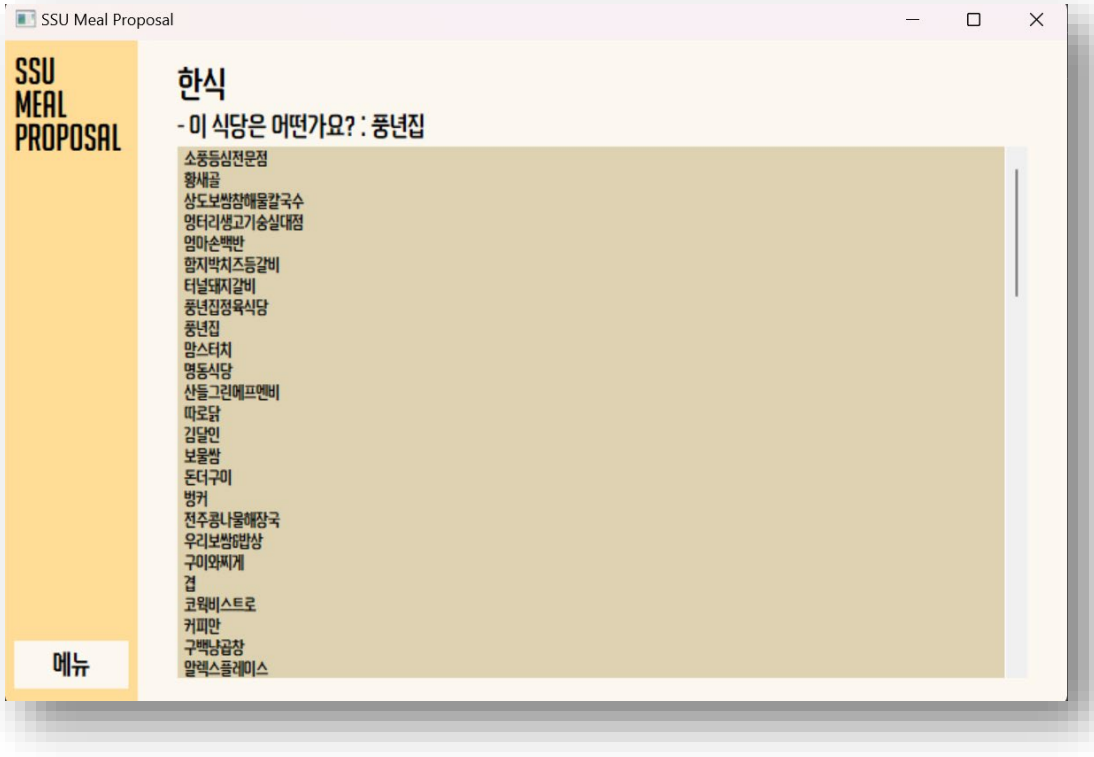




[실행 화면]

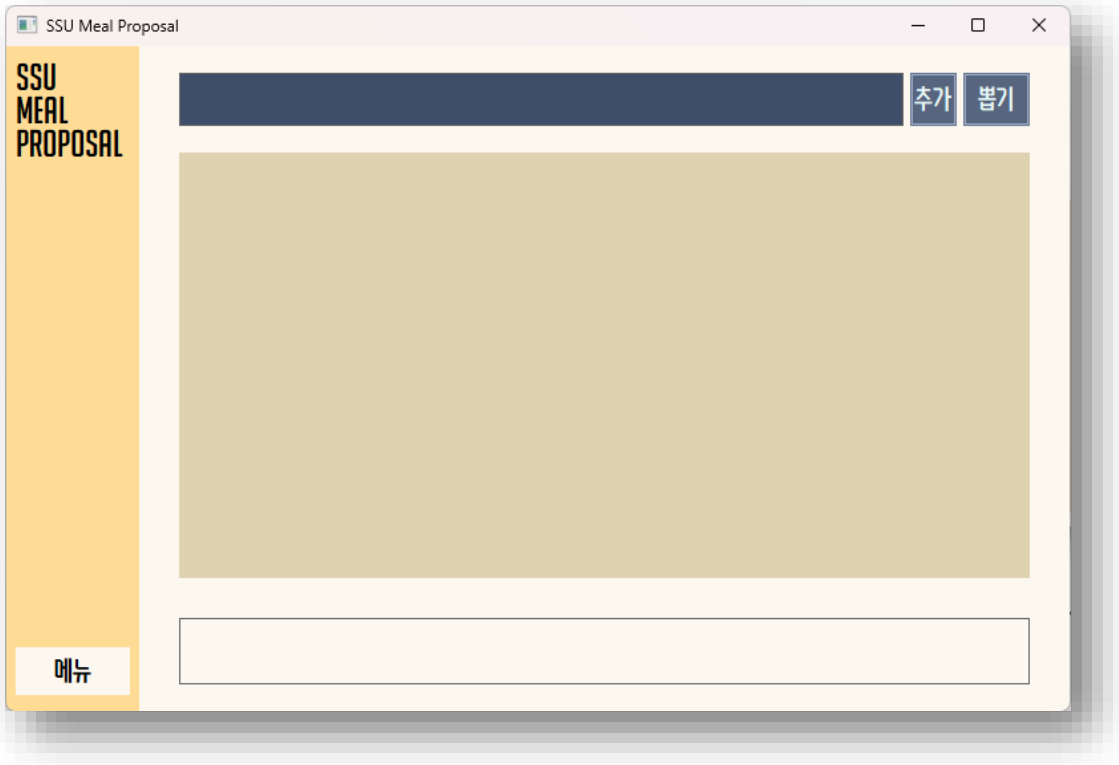


메인 페이지

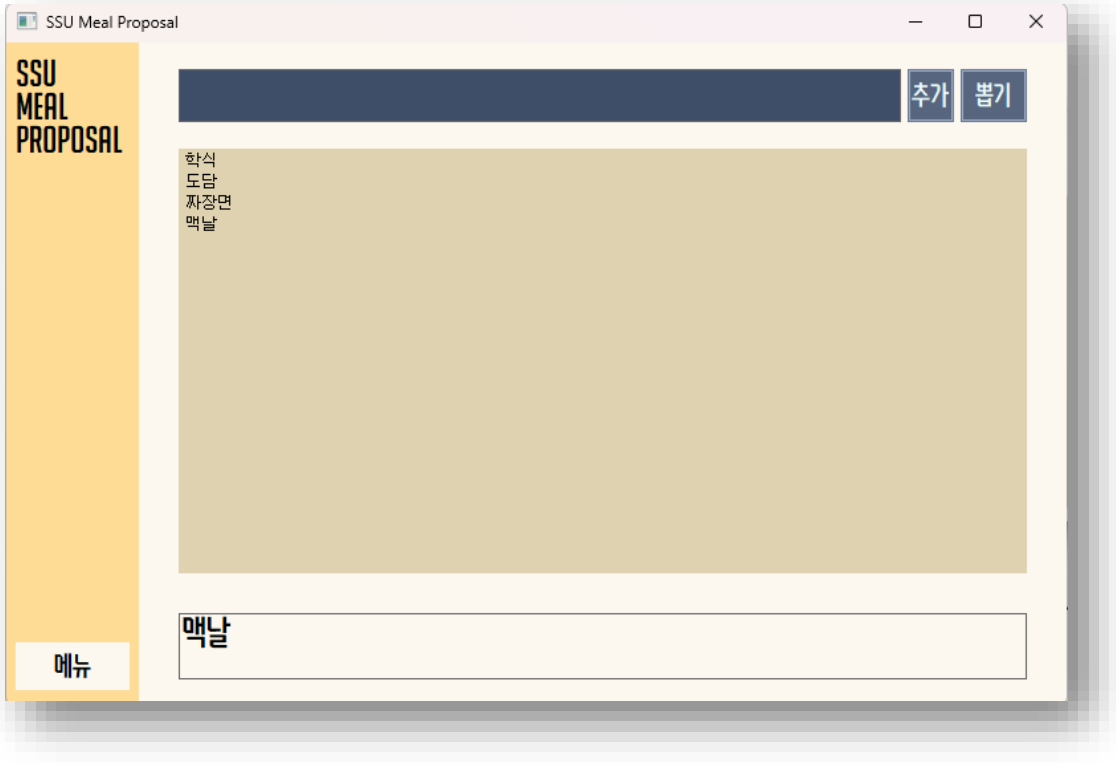


“한식“ 메뉴 클릭 시

[실행 화면]



‘지정 뽑기’ 페이지



‘지정 뽑기’ 페이지

[구현]

- 설계 (콘솔 버전)

rdm.h

rdm_menu_recomnd (전역 범위)

```
1   #pragma once
2   #pragma warning(disable:4996) // C4996 에러를 무시
3
4   #include <stdbool.h>
5   #include <stdio.h>
6   #include <stdlib.h>
7   #include <string.h>
8   #include <time.h>
9   #include <windows.h>
10
11  #define MAX_NAME_LEN 50
12  #define MAX_MENU_LEN 400
13  #define _CRT_NOT_SECURE_NO_WARNINGS
14
15  typedef struct MenuData {
16      char name[MAX_NAME_LEN];// 가게이름
17  } M_DATA;
18
19  char* GetNextString(char* ap_src_str, char a_delimiter, char* ap_buffer); // CSV에서 ','구분해주는 함수
20  int ReadData(const char* ap_file_name, M_DATA* ap_data, unsigned int* ap_data_count); // 외부 파일을 읽어오는 함수
21
22  void show_menu_all(M_DATA* ap_data, unsigned int a_count); // 모든 메뉴 출력 (음식점 리스트)
23  char* random_pick(M_DATA* ap_data, unsigned int a_count); // 랜덤 메뉴 선택
24  void print_main_page(); // 1 ~ 13의 선택지 (메뉴) 화면 출력 - 메인화면
```

[구현]

- 데이터 준비 공공 데이터 포탈 – 소상공인시장진흥공단_상가(상권)정보

```python
import pandas as pd

df = pd.read_csv('shops.csv')

# 음식점 데이터만 쓸 겁니다
df = df.loc[df['상권업종대분류명'] == '음식']

# 다음과 같은 칼럼만 있으면 됩니다
df = df[['상호명', '상권업종중분류명', '상권업종소분류명', '표준산업분류명', '행정동명', '위도', '경도']]

# 그 중에서도 흑석동과 상도1동만 쓸 겁니다.
df = df.loc[(df['행정동명'] == '상도1동')]

# 칼럼명 단순화

df.columns = ['name',  # 상호명
              'cate_1',  # 중분류명
              'cate_2',  # 소분류명
              'cate_3',  # 표준산업분류명
              'dong',  # 행정동명
              'lon',  # 위도
              'lat'  # 경도
              ]
```

```python
# path = 'stores.csv'
# df.to_csv(path_or_buf=path, header=True, index=False)

csv_for_clang = df[['name', 'cate_1', 'cate_2']]
path = "ssu_restaurant.csv"
csv_for_clang.to_csv(path_or_buf=path, header=True, index=False)
# print(csv_for_clang)


###### 각 카테고리별 csv를 만들기 위한 부분 ######
kor = csv_for_clang.loc[df['cate_1'] == '한식']
cafe = csv_for_clang.loc[df['cate_1'] == '커피점/카페']
chn = csv_for_clang.loc[df['cate_1'] == '중식']
cake = csv_for_clang.loc[df['cate_1'] == '제과제빵떡케익']
fast = csv_for_clang.loc[df['cate_1'] == '패스트푸드']
jpn = csv_for_clang.loc[df['cate_1'] == '일식/수산물']
alc = csv_for_clang.loc[df['cate_1'] == '유흥주점']
wst = csv_for_clang.loc[df['cate_1'] == '양식']
mil = csv_for_clang.loc[df['cate_1'] == '분식']
fusion = csv_for_clang.loc[df['cate_1'] == '별식/퓨전요리']
chick = csv_for_clang.loc[df['cate_1'] == '닭/오리요리']
etc = csv_for_clang.loc[df['cate_1'] == '기타음식업']

categories = ['kor', 'cafe', 'chn', 'cake', 'fast',
              'jpn', 'alc', 'wst', 'mil', 'fusion', 'chick', 'etc']
data_cate = [kor, cafe, chn, cake, fast,
             jpn, alc, wst, mil, fusion, chick, etc]


# csv 출력
for cate in range(len(categories)):
    path = 'cate/'
    path += (categories[cate] + '.csv')
    data = data_cate[cate]
    data = data[['name']]
    data.to_csv(path_or_buf=path, header=True, index=False)
```

[구현]

- 코드 (GUI 버전)

```
#include "pch.h"
#include <stdio.h>  // printf, fopen_s, fgets, fclose 함수를 사용하기 위해!
#include <stdlib.h> // srand
#include <time.h>
#include "tipsware.h"

#define MAX_NAME_LEN 50
#define MAX_MENU_LEN 400
#define FONT_OVERWATCH "koverwatch"

#define GOTO_MAIN_BTN 9999

typedef struct MenuData {
    char name[MAX_NAME_LEN];// 가게이름
} M_DATA;
```

```
#include “pch.h”
#include “tipsware.h”
```

GUI를 위해

[구현]

- 코드 (GUI 버전) – 메인화면의 버튼들

```
void print_main_buttons() {
    const char* p_btn_name[13] = {"한식", "양식", "중식", "일식", "카페", "디저트", "패스트푸드", "치킨", "분식",
"퓨전", "술집", "기타", "지정 뽑기"};
    unsigned int x = 200, y = 90;
    unsigned int btn_ids = 1000; // 메뉴 버튼의 id는 1000 ~ 1012 로 사용
    for (int i = 0; i < 13; i++, btn_ids++, y += 50) {
        if (i == 6) { x = 500; y = 90; }
        CreateButton(p_btn_name[i], x, y, 200, 30, btn_ids);

        // 버튼의 외관
        void* btn_id = FindControl(btn_ids);
        SetCtrlFont(btn_id, FONT_OVERWATCH, 20, 0);
        ChangeCtrlColor(btn_id, RGB(252, 248, 240), RGB(255, 220, 150), RGB(255, 220, 150), RGB(0, 0,
0));
    }
    ShowDisplay();
}
```

[구현]

- 코드 (GUI 버전) – 사이드 바 구성

```
void side_menu() {
    ChangeWorkSize(800, 500); // 작업 영역을 설정한다.
    Clear(0, RGB(252, 248, 240));  // 작업 영역을 RGB(252, 248, 240) 색으로 채운다!
    Rectangle(0, 0, 100, 500, RGB(255, 220, 150), RGB(255, 220, 150));
    SelectFontObject(FONT_OVERWATCH, 26, 0);
    TextOut(7, 10, RGB(0, 0, 0), "SSU");
    TextOut(7, 35, RGB(0, 0, 0), "Meal");
    TextOut(7, 60, RGB(0, 0, 0), "Proposal");

    ShowDisplay(); // 정보를 윈도우에 출력한다.
}
```

[구현]

- 코드 (GUI 버전)

> CSV에서 데이터 읽어오는 함수

```
void ReadFileDataToListBox(void* ap_list_box, const char* ap_file_name)
{
    FILE* p_file = NULL;  // 파일을 열어서 사용할 파일 포인터!
    // fopen_s 함수를 사용하여 파일을 텍스트 형식의 읽기 모드로 연다!
    // 이 함수는 파일 열기에 성공했다면 0을 반환한다.
    srand(time(NULL));

    if (0 == fopen_s(&p_file, ap_file_name, "rt")) {
        // 파일에서 한 줄의 정보를 읽어서 저장할 변수와
        // 쉼표를 기준으로 분리한 문자열을 저장할 변수
        char one_line_string[256], str[64], * p_pos;
        // 이름을 저장할 변수와 ListBox에 추가할 문자열을 구성할 배열
        char name[64], insert_str[256];

        int i, rdm = 0;
        // 파일에서 한 줄의 데이터를 읽는다.
        // 하지만 첫 줄은 타이틀 정보라서 처리하지 않고 넘어간다.
        if (NULL != fgets(one_line_string, 256, p_file)) {
            // 파일의 끝까지 학생별 성적 정보를 읽어들인다.
            for (i = 0; NULL != fgets(one_line_string, 256, p_file); i++) {
                p_pos = GetNextString(one_line_string, ',', str); // 학번을 읽는다.
                strcpy_s(name, 64, str);
                // ListBox에 추가할 문자열을 구성한다. 화면에 출력되지 않고 insert_str 배열에
                // 문자열이 저장됩니다.
                sprintf_s(insert_str, 256, "%s", name);
                // insert_str에 저장된 문자열을 ListBox에 추가한다.
                ListBox_InsertString(ap_list_box, i, insert_str, 0);
            }

        }
        fclose(p_file);  // 파일을 닫는다.
    }
}
```

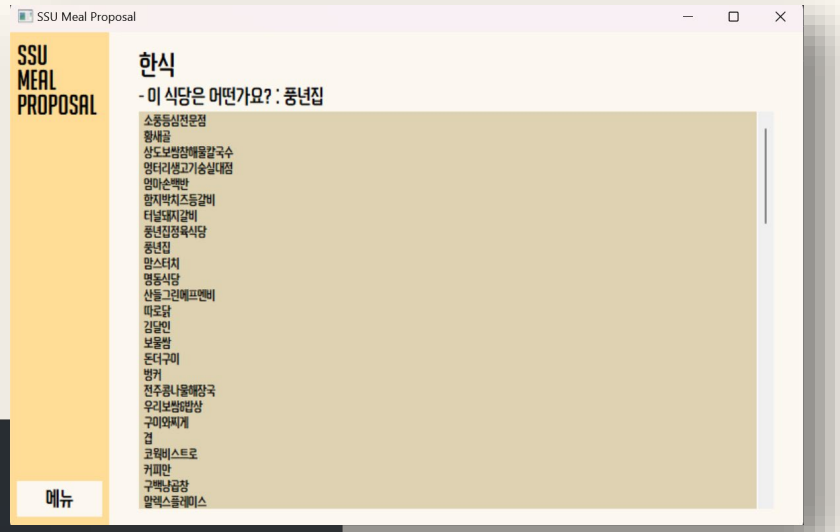
[구현]

- 코드 (GUI 버전) – 버튼 눌렀을 때 발생하는 이벤트 함수

```
void OnCommand(INT32 a_ctrl_id, INT32 a_notify_code, void* ap_ctrl) {

    if (a_ctrl_id >= 1000 && a_ctrl_id <= 1012) { // 메뉴 버튼일 경우
        const char* csv_name[12] = { "cate/kor.csv", "cate/wst.csv", "cate/chn.csv", "cate/jpn.csv",
"cate/cafe.csv", "cate/cake.csv", "cate/fast.csv",
                        "cate/chick.csv", "cate/mil.csv", "cate/fusion.csv", "cate/alc.csv",
"cate/etc.csv" };
        void* p = CreateListBox(130, 80, 640, 400, 2000, NULL);  // 리스트 박스를 생성한다.
        ChangeCtrlColor(p, RGB(222, 210, 177), RGB(252, 248, 240), RGB(252, 248, 240), RGB(0, 0, 0));
        SetCtrlFont(p, FONT_OVERWATCH, 13, 0);
```

[구현]

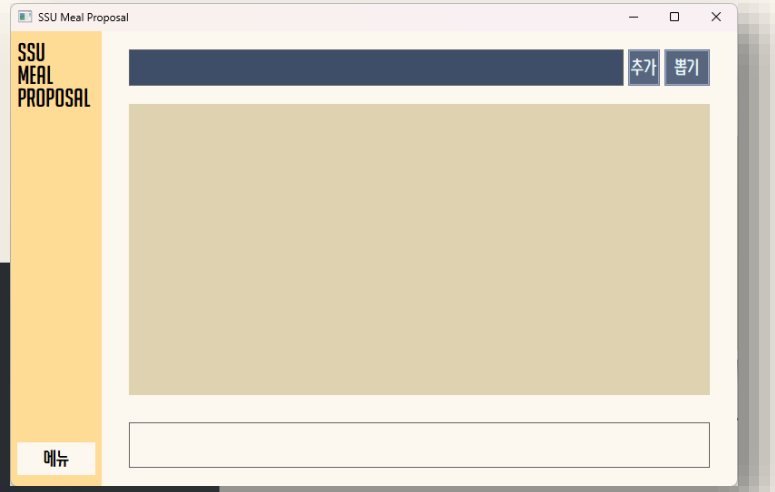
- 코드 (GUI 버전)




```
if (a_ctrl_id != 1012) // 지정 뽑기가 아닌 경우
        {
            ReadFileDataToListBox(p, csv_name[a_ctrl_id - 1000]); // 파일에서 성적 정보를 읽어서 ListBox에 추
가한다.
            int num_of_box = ListBox_GetCount(p), idx = 0;
            char rdm_select[256];
            srand(time(NULL));
            idx = rand() % num_of_box;
            ListBox_GetText(p, idx, rdm_select, 128);

            char cate[64];
            GetCtrlName(FindControl(a_ctrl_id), cate, 64);
            TextOut(130, 20, "%s", cate);
            SelectFontObject(FONT_OVERWATCH, 20, 0);
            TextOut(130, 55, "- 이 식당은 어떤가요? : %s ", rdm_select);
        }
```

[구현]

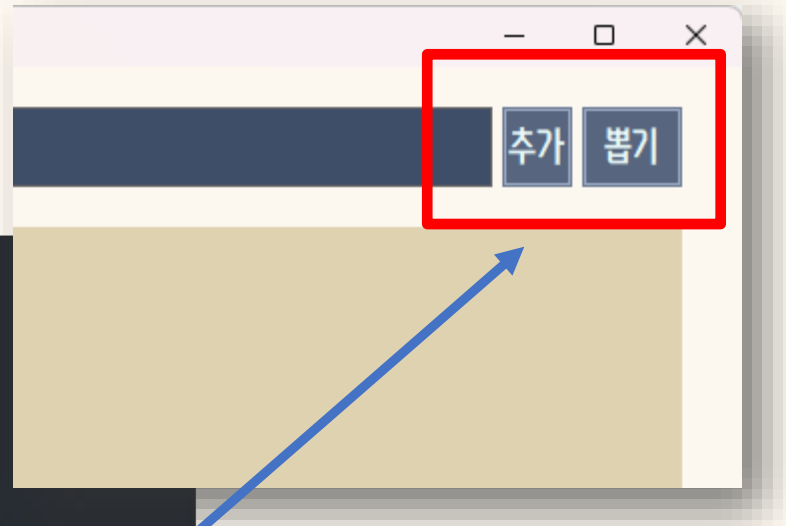
- 코드 (GUI 버전)




```
else { // 지정 뽑기 메뉴일 경우
            DestroyControl(FindControl(2000));
            void* p = CreateListBox(130, 80, 640, 320, 2000, NULL);  // 리스트 박스를 생성한다.
            CreateEdit(130, 430, 640, 50, 2003, 0);
            ChangeCtrlColor(p, RGB(222, 210, 177), RGB(252, 248, 240), RGB(252, 248, 240), RGB(0, 0, 0));
            ChangeCtrlColor(FindControl(2003), RGB(252, 248, 240), RGB(255, 220, 150), RGB(255, 220,
150), RGB(0, 0, 0));
            void *edit_p = CreateEdit(130, 20, 545, 40, 2001, 0);
            void *btn_p_add = CreateButton("추가", 680, 20, 35, 40, 1030); // 1030: 추가 버튼 id
            void *btn_p_result = CreateButton("뽑기", 720, 20, 50, 40, 1031); // 1031: 추출 버튼 id
            SetCtrlFont(edit_p, FONT_OVERWATCH, 24, 0);
            SetCtrlFont(btn_p_add, FONT_OVERWATCH , 20, 0);
            SetCtrlFont(btn_p_result, FONT_OVERWATCH, 20, 0);
        }
        unsigned int btn_ids = 1000;
        for (int i = 0; i < 13; i++, btn_ids++) {
            DestroyControl(FindControl(btn_ids));
        }
        void* btn_p = CreateButton("메뉴", 5, 450, 90, 40, GOTO_MAIN_BTN); // 메인화면으로 돌아가는 버튼
        SetCtrlFont(btn_p, FONT_OVERWATCH, 20, 0);
        ChangeCtrlColor(btn_p, RGB(252, 248, 240), RGB(255, 220, 150), RGB(255, 220, 150), RGB(0, 0, 0));

        ShowDisplay();
    }
```

[구현]

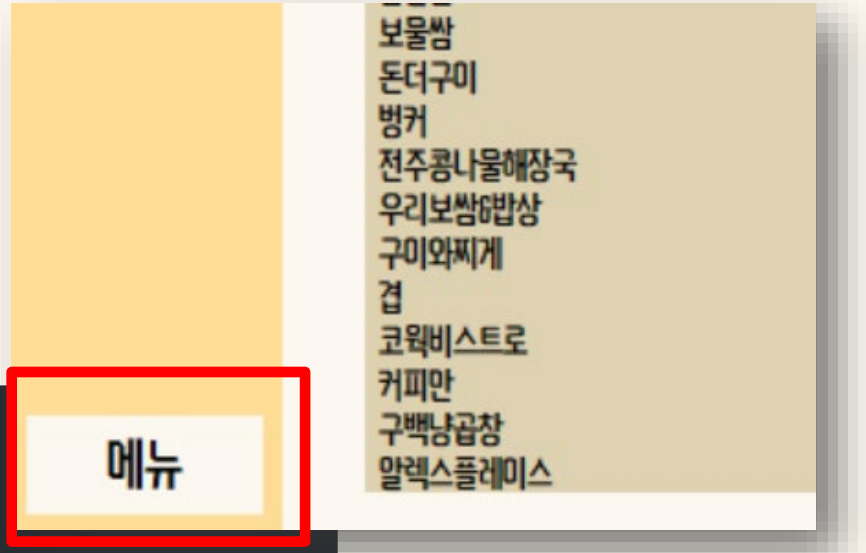
- 코드 (GUI 버전)




```
if (a_ctrl_id == 1030) { // 지정뽑기 - 추가 버튼
        char str[256];
        GetCtrlName(FindControl(2001), str, 64);
        SetCtrlName(FindControl(2001), "");
        ListBox_InsertString(FindControl(2000), -1, str, 0);
    }
    if (a_ctrl_id == 1031) { // 지정뽑기 - 결과 버튼
        int num_of_box = ListBox_GetCount(FindControl(2000)), idx = 0;
        char rdm_select[256];
        srand(time(NULL));
        idx = rand() % num_of_box;
        ListBox_GetText(FindControl(2000), idx, rdm_select, 128);
        SetCtrlName(FindControl(2003), rdm_select);
        SetCtrlFont(FindControl(2003), FONT_OVERWATCH, 25, 0);
    }
```

[구현]

- 코드 (GUI 버전)




```
if (a_ctrl_id == 9999) {
        Clear();
        side_menu();
        DestroyControl(FindControl(9999)); // 왼쪽 하단 메뉴 버튼 삭제
        DestroyControl(FindControl(2000)); // ListBox 삭제
        DestroyControl(FindControl(2001)); // 지정뽑기 - 에딧컨트롤 삭제
        DestroyControl(FindControl(1030)); // 지정뽑기 - 추가 버튼
        DestroyControl(FindControl(1031)); // 지정뽑기 - 결과 버튼
        DestroyControl(FindControl(2003)); // 지정뽑기 - 결과 에딧컨트롤 삭제
        print_main_buttons();
    }
```

[구현]

- 코드 (GUI 버전)

```
CMD_MESSAGE(OnCommand)
```

```
int main()
{
    side_menu();
    print_main_buttons();
    ShowDisplay(); // 정보를 윈도우에 출력한다.
    return 0;
}
```

[구현]

- 코드 (beta – 날씨 반영)

<getWeather.py>

```python
from bs4 import BeautifulSoup
from pprint import pprint
import requests


def get_naver_weather():
    try:
        html = requests.get('https://search.naver.com/search.naver?query=날씨')
        # pprint(html.text)
    except Exception as e:
        return f'error: {e}'

    soup = BeautifulSoup(html.text, 'html.parser')

    data1 = soup.find('section', {'class': 'sc_new cs_weather_new _cs_weather'})

    # pprint(data1)

    differ_yesterday = data1.find('p', {'class': 'summary'})
    # print(differ_yesterday.text) # "어제보다 4.5° 높아요  흐림 "
    differ_yesterday = differ_yesterday.text
    differ = differ_yesterday[5:9]
    which = differ_yesterday[10:13]
    status = differ_yesterday[-3:].strip()

    # print(f"differ : {differ} | which : {which} | status : {status}")

    now_weather = data1.find('div', {'class': 'temperature_text'})
    # print(now_weather.text) # " 현재 온도2.3°"
    now_weather = now_weather.text[6:-1]
    # print(now_weather)
    datas = {"now_degree": now_weather, "differ": differ, "which": which, "status": status}
    return datas


if __name__ == "__main__":
    print(get_naver_weather())
```

[구현]

- 코드 (beta – 날씨 반영) **<guessMenu.py>**

```python
import getWeather
import pandas as pd
import datetime
import random


def settings():
    try:
        now = datetime.datetime.now()

        weather_data = getWeather.get_naver_weather()  # 맑음, 흐림, 구름 많음, 흐리고 비/눈, 흐리고 눈, 흐리고 비, 비, 눈,
        return now, weather_data
    except Exception as e:
        return f"error, {e}"


def make_list():
    try:
        now, weather_data = settings()
        df = pd.read_csv('ssu_restaurant.csv')
        status = weather_data["status"]
    except Exception as e:
        return 'error'
    # 운량
    cloudy = ["한식", "고기"]
    sunny = ["한식", "양식", "고기", "구이류", "일식", "중식", "세계음식"]

    now_cloud = cloudy  # default cloud
    if status == "흐림":
        now_cloud = cloudy
    elif status == "맑음":
        now_cloud = sunny

    # 계절
    spring = ["한식", "갈비", "고기", "구이", "불고기"]
    summer = ["삼계탕", "아이스크림", "냉면"]
    fall = ["호프", "브런치"]
    winter = ["빠", "해물"]
    season = winter  # default season
    # 파이썬은 &&기호로 연결할 필요 없이 두 개의 범위 구분을 한 번에 쓸 수 있다.
    if 3 <= now.month <= 5:
        season = spring

    if 6 <= now.month <= 8:
        season = summer

    if 9 <= now.month <= 11:
        season = fall

    if now.month <= 2 or now.month == 12:
        season = winter

    search_List = []
    for i in range(len(df)):
        for kw in now_cloud:  # 운량에 따른 분류
            if (kw in df.loc[i]['name']) or (kw in df.loc[i]['cate_1']) or (kw in df.loc[i]['cate_2']):
                search_List.append(df.loc[i]['name'])

        for kw in season:  # 계절에 따른 분류
            if (kw in df.loc[i]['name']) or (kw in df.loc[i]['cate_1']) or (kw in df.loc[i]['cate_2']):
                search_List.append(df.loc[i]['name'])

    search_List = list(sorted(search_List))
    # print(search_List)
    return search_List


def get_rdm_from_list():
    search_list = make_list()
    if search_list == 'error':
        return 'network is not connected'
    length = len(search_list)
    rdm_idx = random.randint(0, length)
    return search_list[rdm_idx]


if __name__ == "__main__":
    print(get_rdm_from_list())
```

[구현]

- 코드 (beta – 날씨 반영) **<index.py>**

```
import customtkinter
import os
from PIL import Image
import getWeather, guessMenu

customtkinter.set_appearance_mode("System")  # Modes: "System" (standard), "Dark", "Light"
customtkinter.set_default_color_theme("green")  # Themes: "blue" (standard), "green", "dark-blue"


class App(customtkinter.CTk):
    def __init__(self):
        super().__init__()

        self.title("SSU Meal Proposal")
        self.geometry(f"{800}x{500}")

        # set grid layout 1x2
        self.grid_rowconfigure(0, weight=1)
        self.grid_columnconfigure(1, weight=1)

        # load images with light and dark mode image
        image_path = os.path.join(os.path.dirname(os.path.realpath(__file__)), "test_images")
        self.logo_image = customtkinter.CTkImage(Image.open(os.path.join(image_path, "cloudy.png")),
                                                 size=(26, 26))

        self.image_icon_image = customtkinter.CTkImage(Image.open(os.path.join(image_path,
"cloudy.png")),
                                                       size=(20, 20))
        self.home_image = customtkinter.CTkImage(light_image=Image.open(os.path.join(image_path,
```

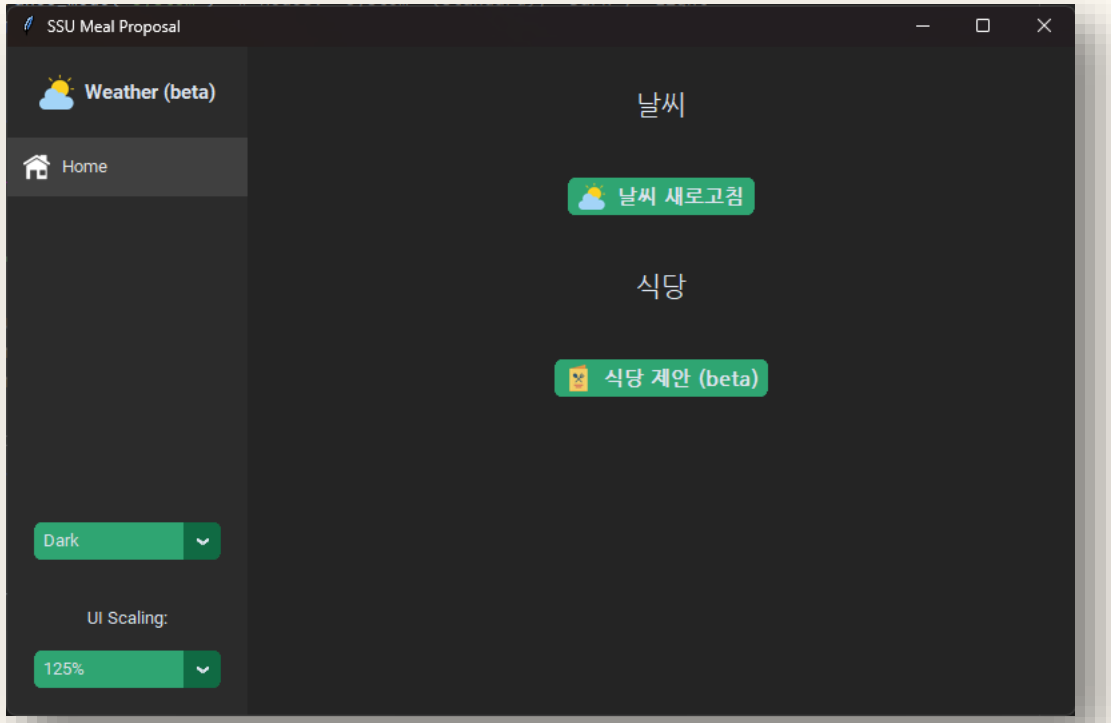
[구현]

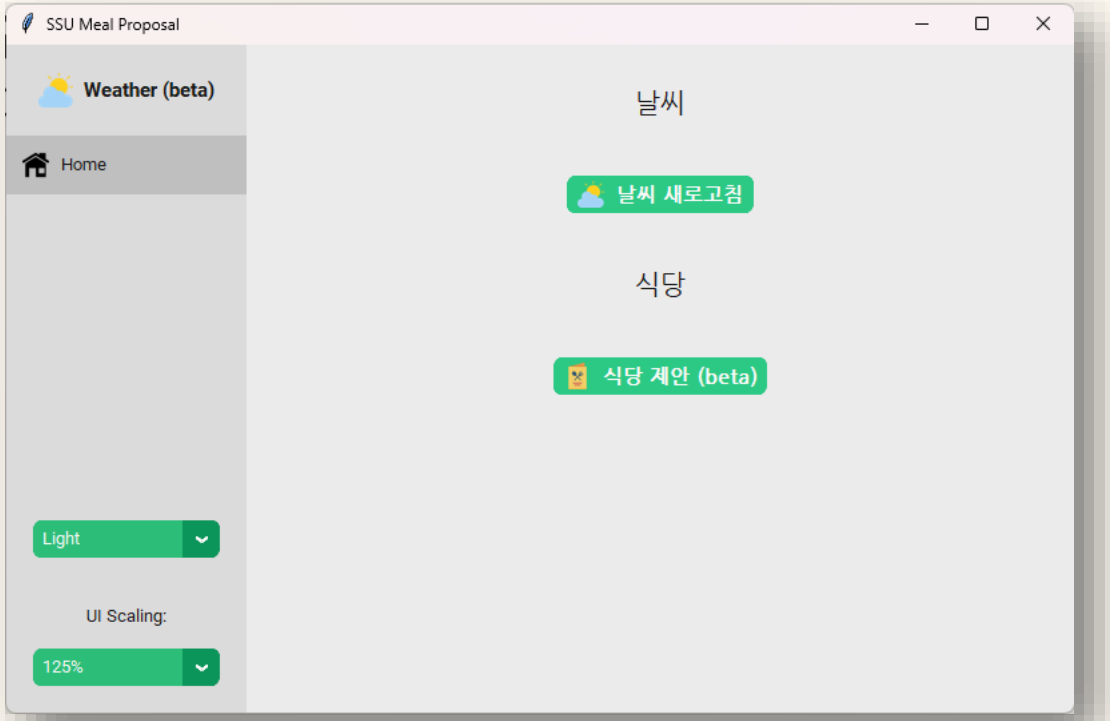
- 코드 (beta – 날씨 반영) **<index.py>**

```
    def select_frame_by_name(self, name):
        # set button color for selected button
        self.home_button.configure(fg_color=("gray75", "gray25") if name == "home" else "transparent")

        # show selected frame
        if name == "home":
            self.home_frame.grid(row=0, column=1, sticky="nsew")
        else:
            self.home_frame.grid_forget()

    def home_button_event(self):
        self.select_frame_by_name("home")

    def change_appearance_mode_event(self, new_appearance_mode):
        customtkinter.set_appearance_mode(new_appearance_mode)

    def change_scaling_event(self, new_scaling: str):
        new_scaling_float = int(new_scaling.replace("%", "")) / 100
        customtkinter.set_widget_scaling(new_scaling_float)

    def get_weather(self):
        celcius = getWeather.get_naver_weather()
        if 'error' in celcius:
            self.home_frame_label.configure(text="현재 날씨: network is not connected")
        else:
            celcius = celcius["now_degree"]
            self.home_frame_label.configure(text=f"현재 날씨: {celcius}")


    def get_menu(self):
        proposal = guessMenu.get_rdm_from_list()
        self.home_frame_menu_proposal.configure(text=f"이곳은 어떠신가요?\n\n{proposal}")
```

[구현]

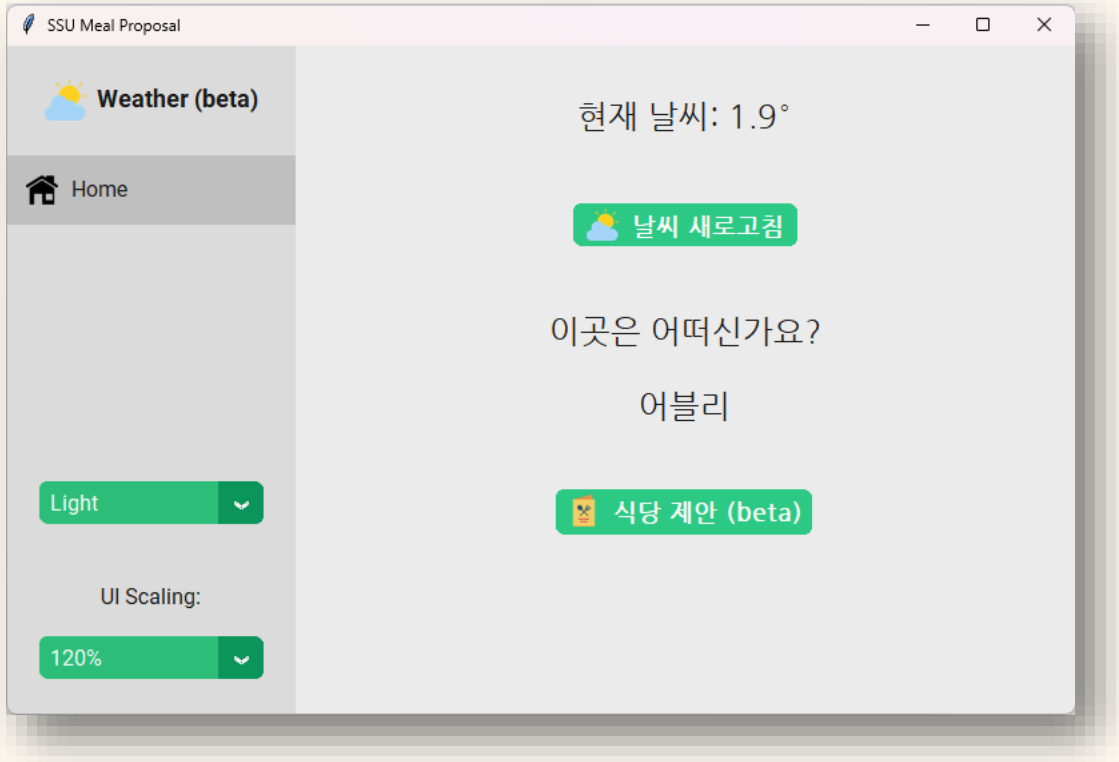
- GUI (beta – 날씨 반영)

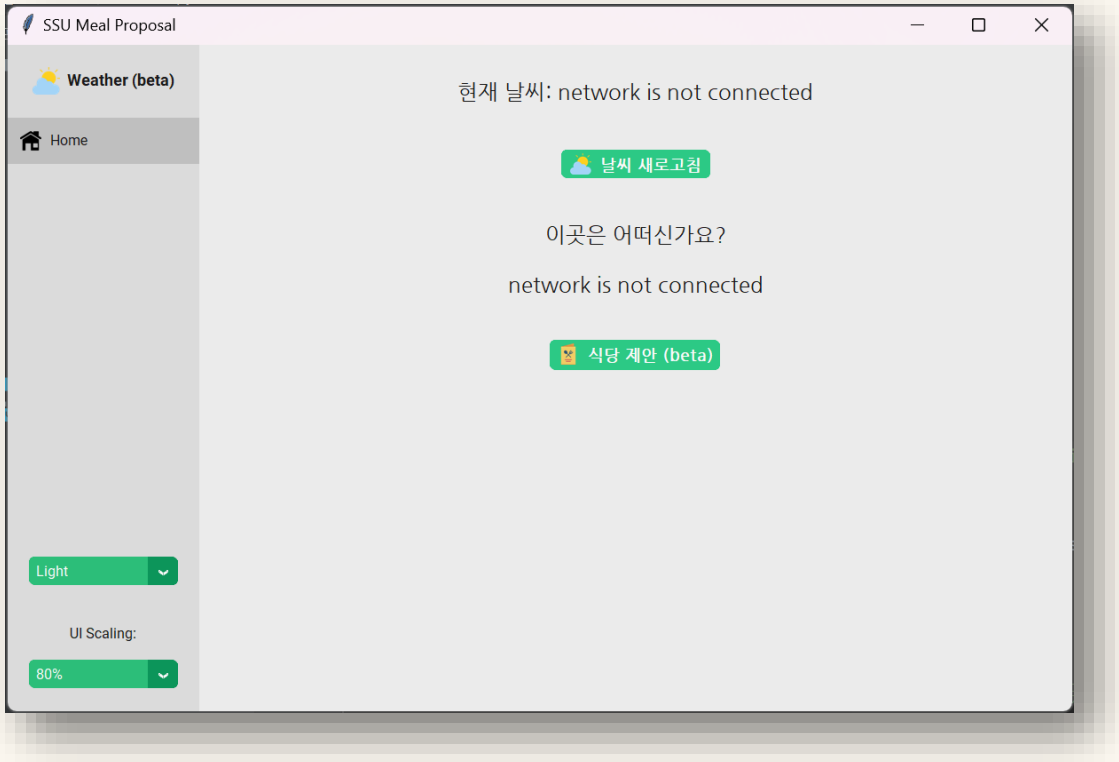




[구현]

- GUI (beta – 날씨 반영)





[기대 효과]

숭실대 근처에서 밥을 해결해야 한다;
하지만 어디를 가야할 지 모르겠다;

⇒ 무작위로 지정된 곳에 갈 수도 있고, 카테고리별 가고 싶은 곳을 정할 수도 있다;
⇒ 여러명이 의견이 갈릴 때! 뽑기를 할 수 있고, 추가로 다른 식당을 찾아 볼 수 도 있다;

[느낀점]

= 프로젝트를 처음 계획했던대로 진행했다는 점에서 성취감을 느낌

= C로 GUI를 하면서 쉽지 않음을 느꼈지만, 결국 기본적인 기능들을 GUI로 구현해 내서 기쁨

= 추가적인 기능들이나 소소한 부분들: C로 GUI를 만들면서 기능 추가 하기에는 능력 부족이라 느꼈음. 추후 Python 등을 이용하여 업그레이드를 진행 하고 싶음

= 파이썬이 매우 편한 도구였다는 점을 절실히 깨달음

= 논문을 기반으로 rule-based 알고리즘을 짜려 했지만, 가지고 있는 데이터가 그정도 까지 라벨링이 되어 있는 데이터가 아니라 세세한 분류는 실패해서 아쉽다

= 파이썬으로 만든 프로그램을 exe로 만들어 c프로그램이랑 연결 시켰으면 더 좋았을거라 생각

REF.

Easywin32 네이버 카페, https://cafe.naver.com/easywin32

우리 동네 맛집 추천엔진 직접, 쉽게 만들기 (크롤링과 코사인 유사도), https://data101.oopy.io/recommendation-engine-cosine-similarity

날씨 및 요일 특성이 음식점 메뉴 검색시스템 이용에 미치는 영향에 관한 실증 연구, 2017, https://scienceon.kisti.re.kr/srch/selectPORSrchArticle.do?cn=JAKO201720636501085&dbt=NART

Customtkinter, 2022, https://github.com/TomSchimansky/CustomTkinter

Q&A

깃헙 주소