

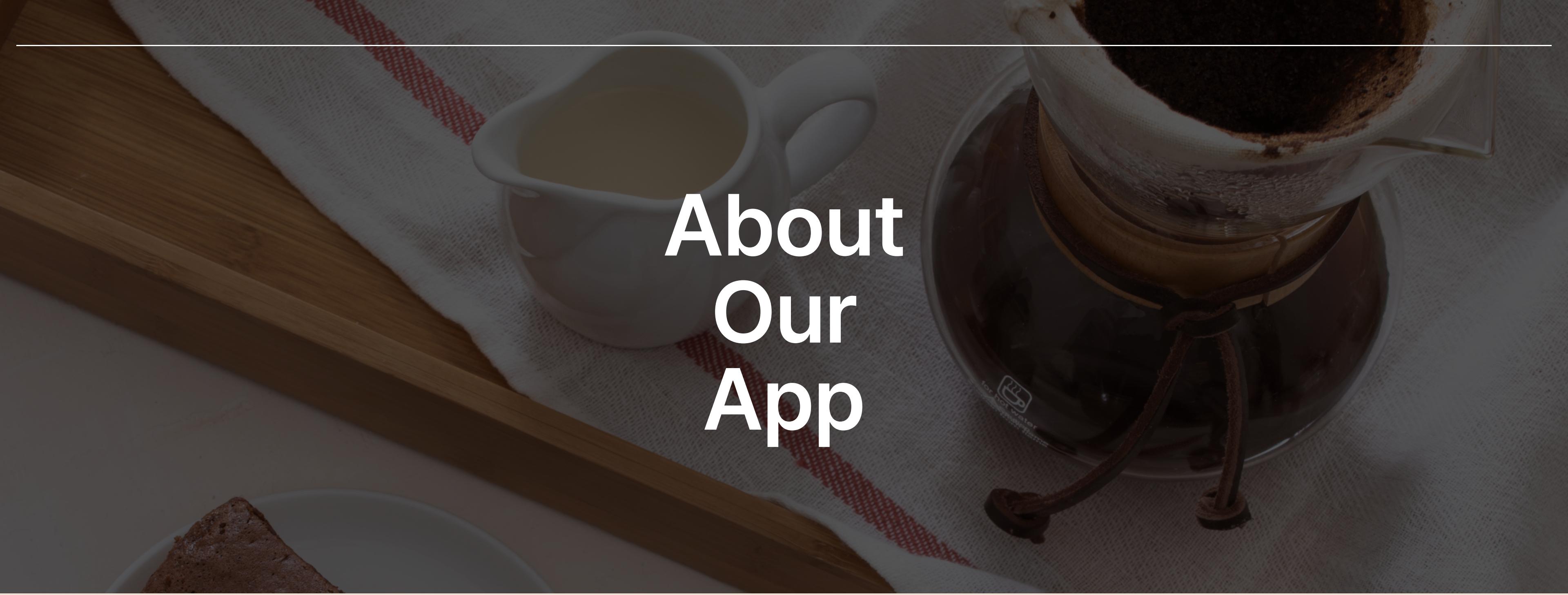
Caffeine



카페인 과다섭취 모니터링 앱

하루 카페인 섭취를 스마트하게 관리해, 숙면을 지키는 개인 맞춤 건강 도우미.





About Our App

개인의 카페인-수면 패턴을 분석하여 예상 수면 질을 예측하고, 목표 수면 질 달성을 위한 과다 섭취 경고를 제공하는 맞춤형 AI 서비스입니다



1. 공감

페르소나
What-How-Why 관찰법

2. 문제 정의

사용자 스토리
사용자 경험 감정 분석 그래프
WHAT WHY TREE

3. 아이디어 제작

브레인스토밍
아이디어 평가표

4. 프로토타입

앱 사용 시나리오
예측 모델
앱 한계점



페르소나

[이름] : 김승실

[성별] : 남자

[나이] : 18살

[직업] : 고등학생

[성격] : 섬세함, 성실함

[주요 활동 장소]: 학교, 학원, 스터디 카페

[특징] : 공부를 열심히 함.



WHAT-HOW-WHY

[WHAT] : 문제가 무엇인가?

- 자신의 적정 카페인 섭취량을 모름.
- 수면의 질이 낮아서 수업시간에 집중못하고 졸음.
- 이를 해결하려고 다시 카페인을 섭취하며 악순환이 반복.

[HOW] : 문제를 해결할 수 있는 방법은 무엇인가?

- 식곤증이 올때 카페인을 섭취하지 않고 다른 방법(스트레칭 등)으로 잠깨기.
- 자신의 적정 카페인 섭취량 알기.
- 수면의 질 높이기.

[WHY] : 왜 문제가 되는가?

- 졸음 → 카페인 섭취 → 수면 저하 → 다음날 졸음의 무한 루프는 일상 생산성을 지속적으로 떨어뜨림.
- 장기간 카페인 조절 실패는 불면증, 심박 이상, 위장 장애 등 건강 악화로 이어질 수 있으며, 카페인 의존증이나 정신적 불안정까지 유발할 수 있음

[걱정거리]

- 식후에 식곤증에 의해서 졸림.
- 저녁시간대에 커피를 마셔서 늦게 잠들음.
- 학교 수업시간에 졸릴때가 많음.

2. 문제정의

User Story

Caffeine · Sleep · Different



김승실은 성실하고 계획적인 고등학생이다.

학원 수업과 스터디 일정이 빽빽한 하루 속에서, 저녁을 먹은 뒤면 **식곤증에 의해 어김없이 졸음이 찾아왔다.**

그럴 때마다 그는 카페에서 **아메리카노 한 잔**을 마시며 잠을 쫓곤 했다.

어느 날, 친구들과 시험 공부를 하던 중 승실이는 흥미로운 이야기를 들었다.

한 친구는 밤 늦게 커피를 마셔도 아무 문제 없이 끊 잔다고 했고,

다른 친구는 카페인을 조금만 섭취해도 심장이 두근거리고 매스꺼워서 못 마신다고 했다.

그때 승실이도 깨달았다.

"나는 커피를 마시면 확실히 집중은 잘 되지만, 어떤 날은 밤늦게까지 뒤척이며 잠을 못 자기도 해."

그는 처음으로 사람마다 카페인에 반응하는 방식이 다르다는 사실을 체감했다.

그리고 자신도 카페인을 효과적으로 활용하면서도 수면을 해치지 않는 방법을 알아야겠다는 생각을 하게 되었다.

단순히 "마시면 잠 깬다"는 수준이 아니라,

언제 얼마나 마셔야 나한테 맞는지를 파악하는 게 필요하다는 자각이 생긴 순간이었다.

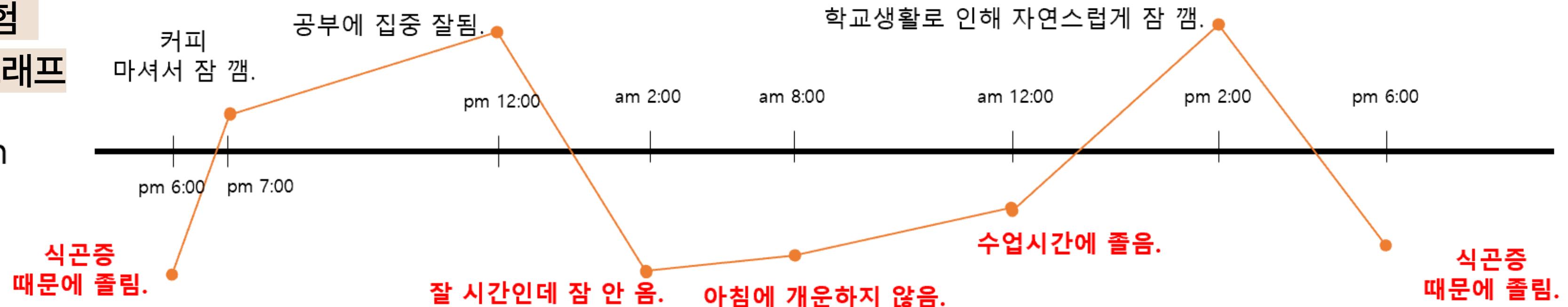
사용자 경험

감정 분석 그래프

(+)

Emotion

(-)



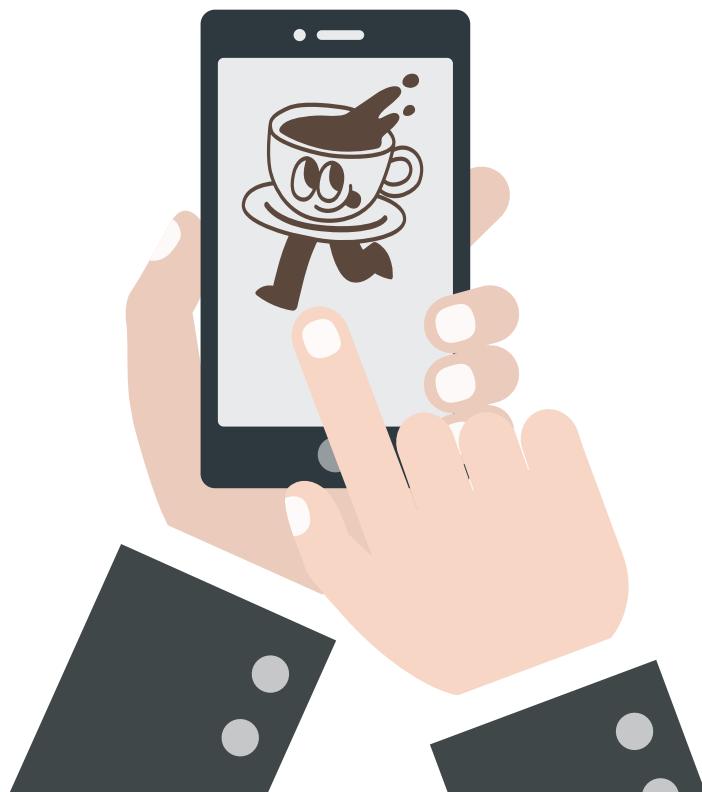
What Why Tree

카페인 과다섭취



Brainstorming

BEST IDEA 01



카페인 섭취 모니터링 앱

이 앱은 하루 동안의 카페인 섭취 이력과 수면 목표를 바탕으로, 사용자에게 남은 섭취 가능량을 알려주는 개인 맞춤형 건강 관리 도구입니다. 올바른 카페인 활용 습관을 형성하도록 돋는 스마트한 수면 도우미입니다.

BEST IDEA 02



청소년 대상 '카페인 감수성 테스트' 캠페인

학교나 보건소와 연계해서 간단한 문답/앱 기반 설문을 통해 “너는 카페인에 민감한 타입이야” 같은 결과를 제공 → 개인별 섭취 기준을 알게 하고 조절할 수 있도록 유도.

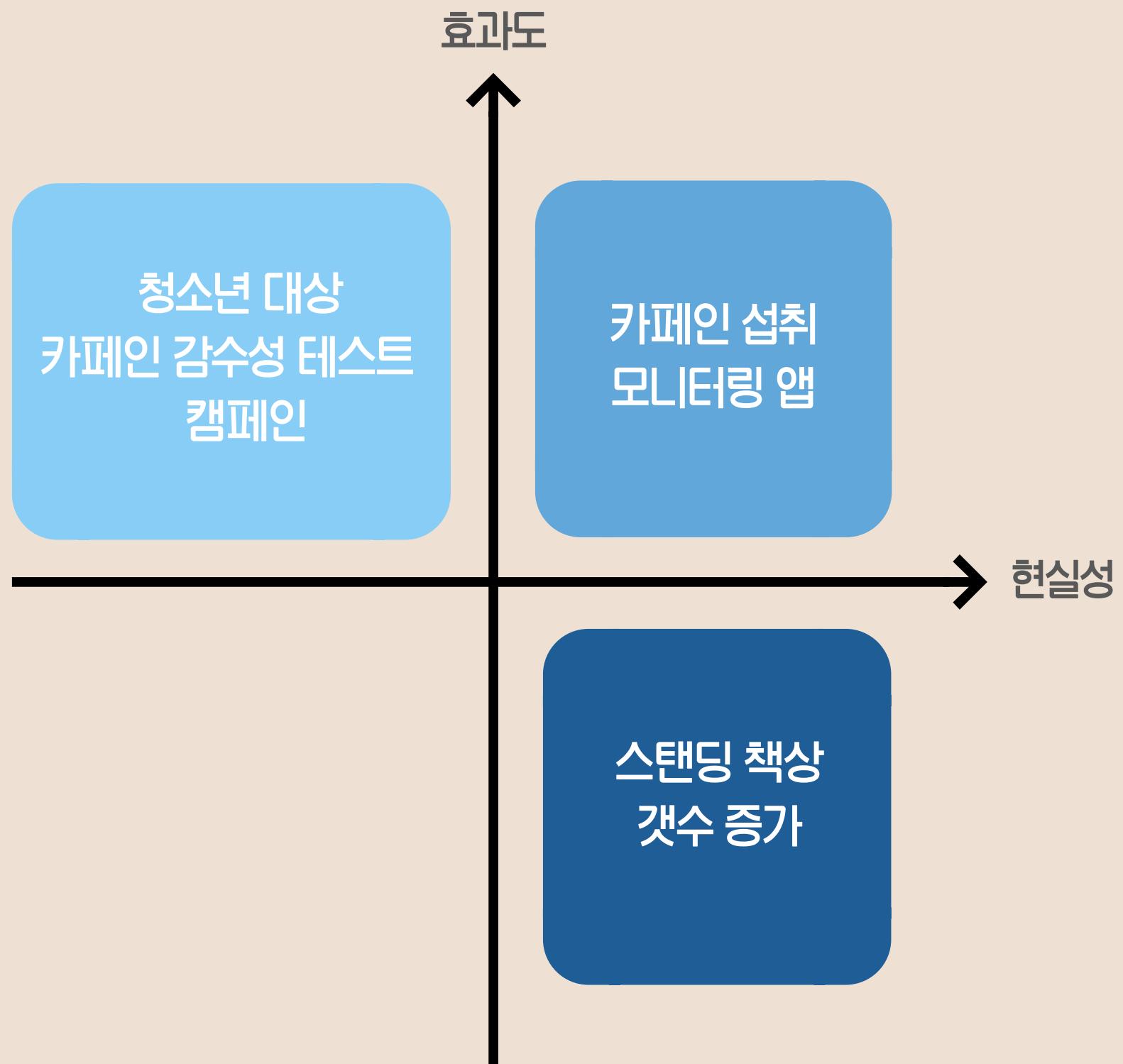
BEST IDEA 03



스탠딩 책상 갯수 증가

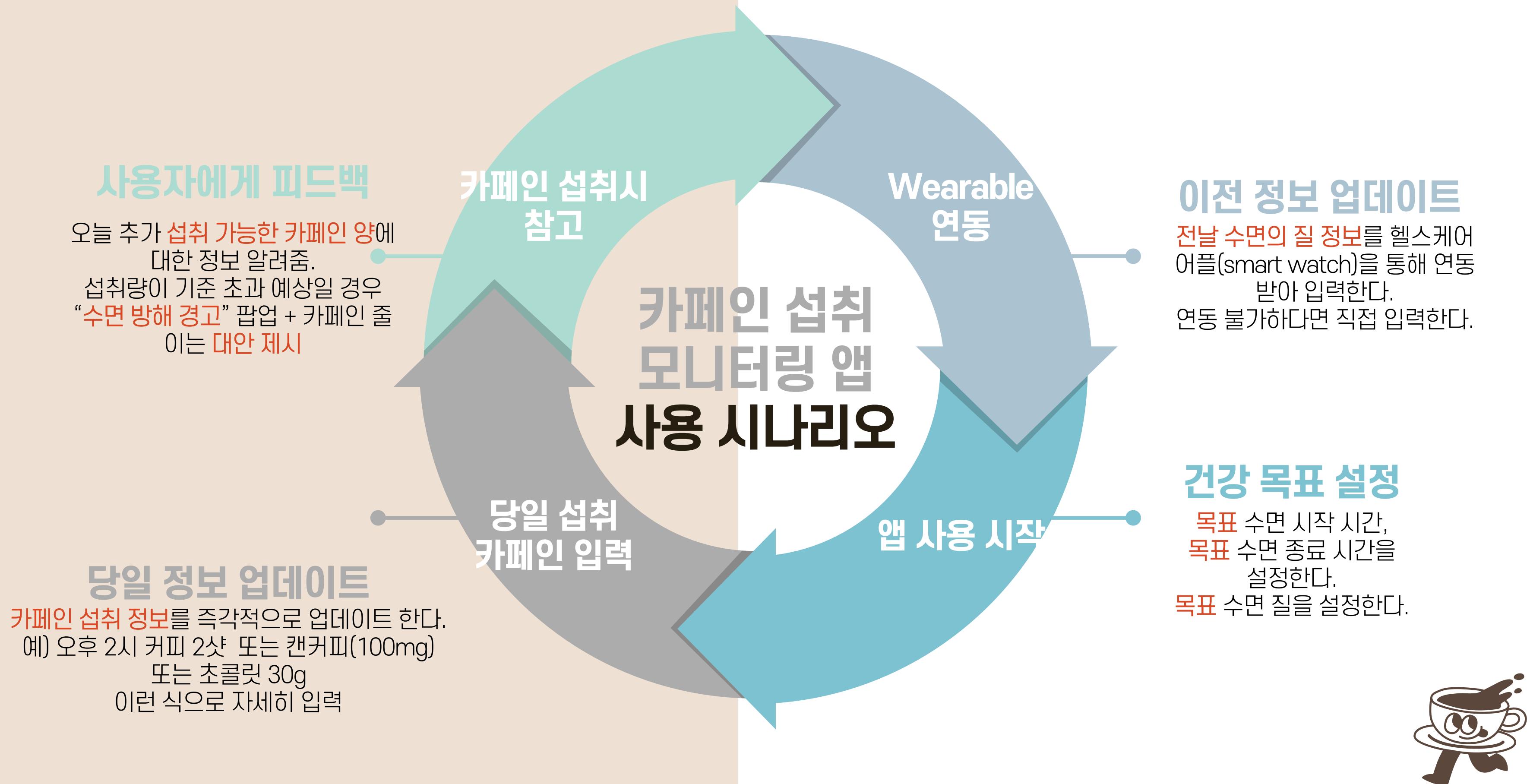
교실 내 책상의 약 1/4을 스탠딩 책상으로 도입해, 학생들이 자유롭게 서서 공부할 수 있는 환경을 조성합니다. 서 있는 자세는 졸음을 줄이고 혈액순환을 도와 집중력 향상에 도움을 줍니다.

아이디어 평가표

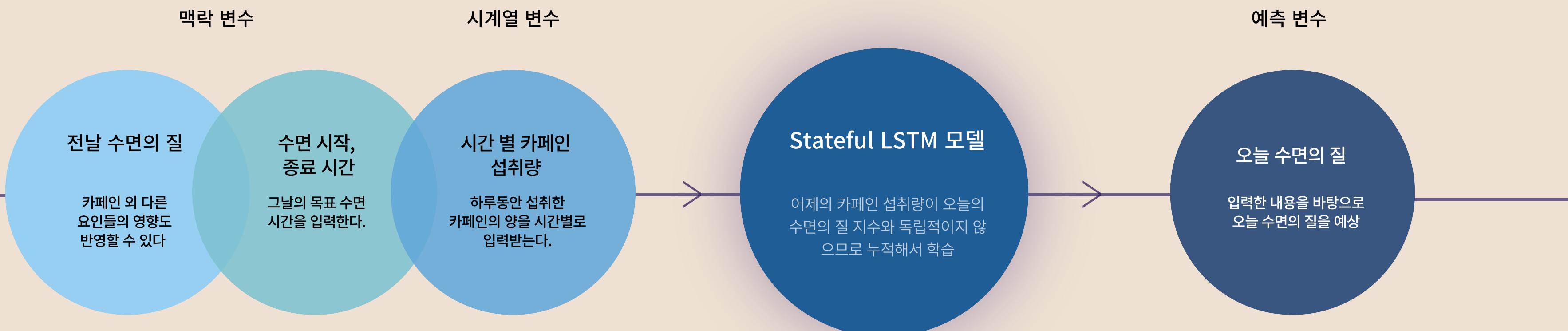


카페인 섭취
모니터링 앱

- 개인맞춤** 사람마다 다른 카페인 민감도를 고려
- 현실성** 앱 개발과 활용이 비교적 쉽고 실현 가능성 높음
- 지속성** 일회성 캠페인보다 일상 속에서 꾸준히 활용 가능
- 예방효과** 수면장애·중독 등 건강 문제를 사전에 방지
- 확장성** 웨어러블, 학교 교육, 헬스케어와 연계 가능한 발전 가능성



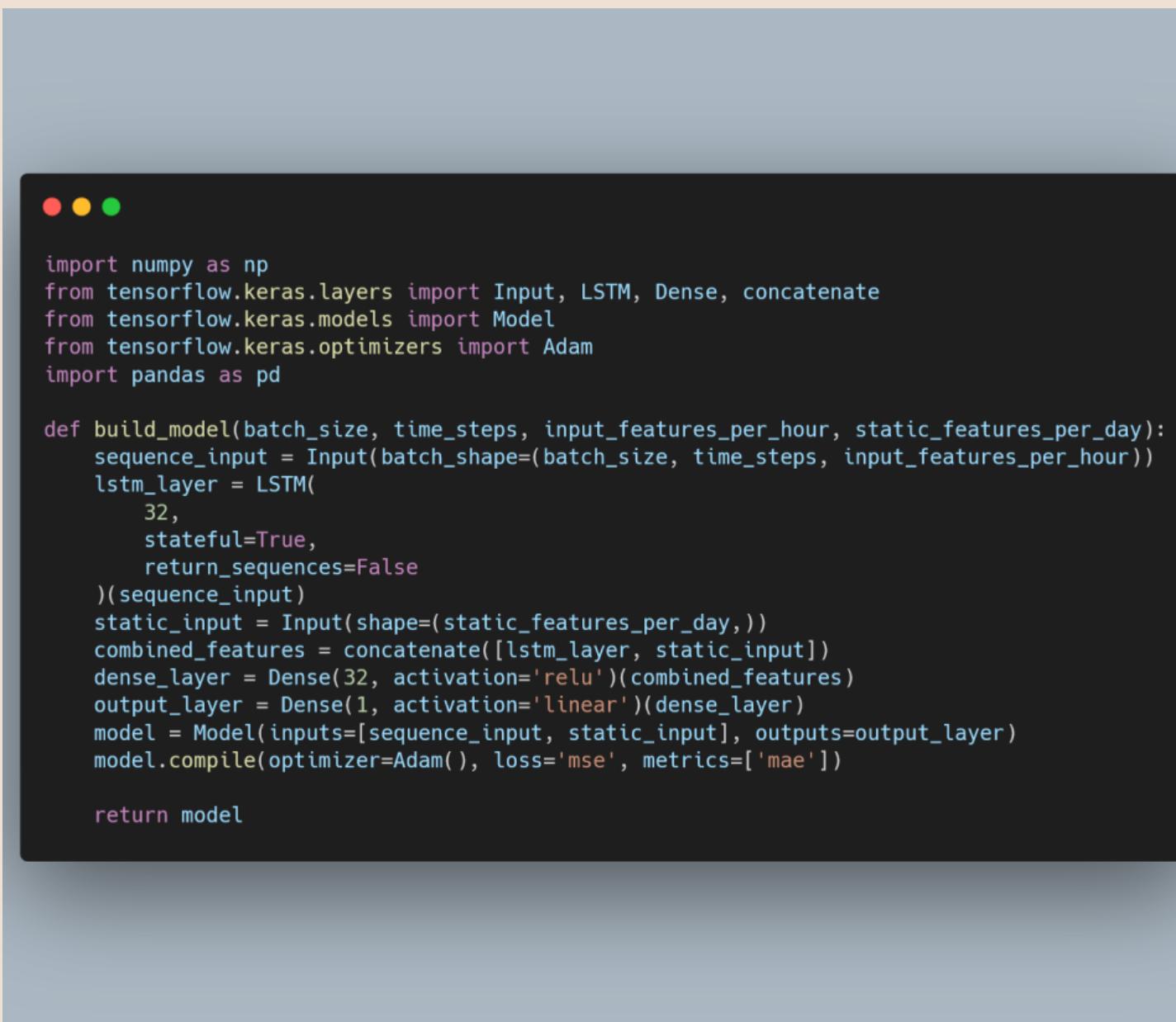
예측 모델 - 개요



예측 모델 기반 서비스 제공

개인마다 카페인 민감도가 다르다는 점을 고려하여, 각 사용자에게 맞춰 학습하는 인공지능 모델 기반의 서비스를 제공 한다. 이러한 모델을 활용하여, 어플리케이션은 사용자의 카페인 섭취량으로 인해 예측되는 수면의 질 지수가 설정한 목 표치에 미달할 것으로 예상될 때, 알림으로 경고하는 기능을 제공한다.

예측 모델 - 구현



```

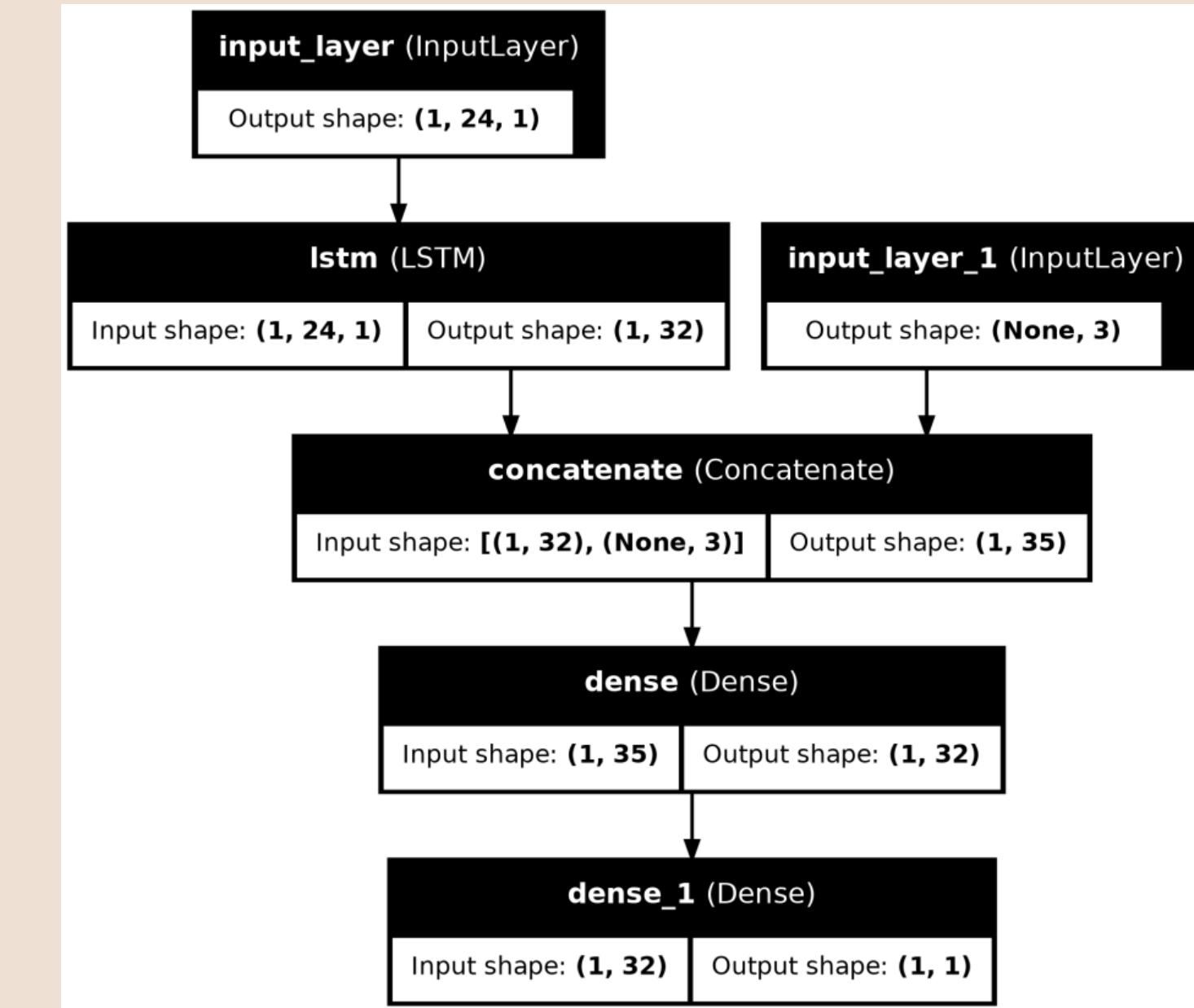
import numpy as np
from tensorflow.keras.layers import Input, LSTM, Dense, concatenate
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
import pandas as pd

def build_model(batch_size, time_steps, input_features_per_hour, static_features_per_day):
    sequence_input = Input(batch_shape=(batch_size, time_steps, input_features_per_hour))
    lstm_layer = LSTM(
        32,
        stateful=True,
        return_sequences=False
    )(sequence_input)
    static_input = Input(shape=(static_features_per_day,))
    combined_features = concatenate([lstm_layer, static_input])
    dense_layer = Dense(32, activation='relu')(combined_features)
    output_layer = Dense(1, activation='linear')(dense_layer)
    model = Model(inputs=[sequence_input, static_input], outputs=output_layer)
    model.compile(optimizer=Adam(), loss='mse', metrics=['mae'])

    return model

```

모델 학습



모델 다이어그램

예측 모델 - 구현

Date	Hour_00	Hour_01	Hour_02	Hour_03	Hour_04	Hour_05	Hour_06	Hour_07	Hour_08	Hour_09	Hour_10	Hour_11	Hour_12	Hour_13	Hour_14	Hour_15	Hour_16	Hour_17	Hour_18	Hour_19	Hour_20	Hour_21	Hour_22	Hour_23	Sleep_End_Time	Prev_Day_SQ	Sleep_Quality								
2025-05-01	0	0	0	0	0	0	0	134	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23.25	6.75	7.00	7.53							
2025-05-02	0	0	0	0	0	0	0	0	149	120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.12	7.9	7.53	7.76							
2025-05-03	0	0	0	0	0	0	0	0	0	0	131	0	0	0	0	0	0	0	0	0	0	0	0	0	23.61	7.11	7.76	7.56							
2025-05-04	0	0	0	0	0	0	0	0	107	0	0	0	148	0	0	0	0	0	0	0	0	0	0	0	0	23.18	6.88	7.56	7.43						
2025-05-05	0	0	0	0	0	0	0	0	0	118	0	0	0	137	0	0	0	0	0	0	0	0	0	0	0	24.4	7.1	7.43	7.60						
2025-05-06	0	0	0	0	0	0	0	0	0	0	0	0	0	103	141	0	0	0	0	0	0	0	0	0	0	0	23.03	6.53	7.60	7.32					
2025-05-07	0	0	0	0	0	0	0	0	0	129	0	0	0	106	0	0	0	0	0	0	0	0	0	0	0	0	23.88	7.68	7.32	7.79					
2025-05-08	0	0	0	0	0	0	0	0	0	145	111	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.5	8.3	7.79	7.97						
2025-05-09	0	0	0	0	0	0	0	0	0	0	123	0	0	0	147	0	0	0	0	0	0	0	0	0	0	0	23.31	6.81	7.97	7.59					
2025-05-10	0	0	0	0	0	0	0	0	0	101	0	0	0	108	0	136	0	0	0	0	0	0	0	0	0	0	0	23.7	7.2	7.59	7.71				
2025-05-11	0	0	0	0	0	0	0	0	0	0	114	144	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.0	7.8	7.71	7.90					
2025-05-12	0	0	0	0	0	0	0	0	0	0	100	0	0	0	117	0	0	0	0	0	0	0	0	0	0	0	0	23.55	7.05	7.90	7.55				
2025-05-13	0	0	0	0	0	0	0	0	0	130	0	0	0	140	0	0	0	0	0	0	0	0	0	0	0	0	0	23.92	7.72	7.55	7.83				
2025-05-14	0	0	0	0	0	0	0	0	0	0	146	119	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.6	8.1	7.83	8.00				
2025-05-15	0	0	0	0	0	0	0	0	0	0	0	127	0	0	0	150	0	0	0	0	0	0	0	0	0	0	0	0	23.4	6.9	8.00	7.57			
2025-05-16	0	0	0	0	0	0	0	0	0	104	0	0	0	110	0	0	0	139	0	0	0	0	0	0	0	0	0	0	0	23.75	7.25	7.57	7.73		
2025-05-17	0	0	0	0	0	0	0	0	0	0	116	143	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.18	7.98	7.73	7.93				
2025-05-18	0	0	0	0	0	0	0	0	0	0	0	102	0	0	0	112	0	0	0	0	0	0	0	0	0	0	0	0	23.6	7.1	7.93	7.58			
2025-05-19	0	0	0	0	0	0	0	0	0	0	0	132	0	0	0	146	0	0	0	0	0	0	0	0	0	0	0	0	23.95	7.75	7.58	7.85			
2025-05-20	0	0	0	0	0	0	0	0	0	0	0	140	116	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.25	8.05	7.85	7.97			
2025-05-21	0	0	0	0	0	0	0	0	0	0	0	122	0	0	0	149	0	0	0	0	0	0	0	0	0	0	0	0	23.5	7.0	7.97	7.56			
2025-05-22	0	0	0	0	0	0	0	0	0	0	105	0	0	0	107	0	0	0	138	0	0	0	0	0	0	0	0	0	0	23.8	7.3	7.56	7.71		
2025-05-23	0	0	0	0	0	0	0	0	0	0	0	110	145	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.05	7.85	7.71	7.88			
2025-05-24	0	0	0	0	0	0	0	0	0	0	0	104	0	0	0	120	0	0	0	0	0	0	0	0	0	0	0	0	0	23.65	7.15	7.88	7.54		
2025-05-25	0	0	0	0	0	0	0	0	0	0	0	133	0	0	0	140	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23.9	7.6	7.54	7.78	
2025-05-26	0	0	0	0	0	0	0	0	0	0	0	141	115	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.2	7.9	7.78	7.90	
2025-05-27	0	0	0	0	0	0	0	0	0	0	0	126	0	0	0	148	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23.7	7.2	7.90	7.56	
2025-05-28	0	0	0	0	0	0	0	0	0	0	0	108	0	0	0	106	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23.98	7.78	7.56	7.79	
2025-05-29	0	0	0	0	0	0	0	0	0	0	0	113	142	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24.1	7.8	7.79	7.9	
2025-05-30	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	119	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23.75	7.25	7.91	7.57

```
BATCH_SIZE = 1
STATEFUL_TIME_STEPS = 24
INPUT_FEATURES_PER_HOUR = 1
STATIC_FEATURES_PER_DAY = 3

df = pd.read_csv('test.csv')
NUM_DAYS_DATA = df.shape[0]

hourly_cols = [f'Hour_{h:02d}' for h in range(24)]
hourly_data = df[hourly_cols].values.reshape(NUM_DAYS_DATA, STATEFUL_TIME_STEPS, INPUT_FEATURES_PER_HOUR)

static_cols = ['Sleep_Start_Time', 'Sleep_End_Time']
static_data = df[static_cols].values
prev_day_sq_data = df['Prev_Day_SQ'].values.reshape(NUM_DAYS_DATA, 1)
static_data = np.concatenate([static_data, prev_day_sq_data], axis=-1)

quality_data = df['Sleep_Quality'].values.reshape(NUM_DAYS_DATA, 1)

train_split = int(NUM_DAYS_DATA * 0.8)
hourly_train, hourly_test = hourly_data[:train_split], hourly_data[train_split:]
static_train, static_test = static_data[:train_split], static_data[train_split:]
quality_train, quality_test = quality_data[:train_split], quality_data[train_split:]

model = build_model(
    batch_size=BATCH_SIZE,
    time_steps=STATEFUL_TIME_STEPS,
    input_features_per_hour=INPUT_FEATURES_PER_HOUR,
    static_features_per_day=STATIC_FEATURES_PER_DAY
)
model.fit(
    [hourly_train, static_train], quality_train,
    batch_size=BATCH_SIZE,
    epochs=150,
    shuffle=False,
    validation_data=(hourly_test, static_test), quality_test),
    verbose=0
)
```

테스트 학습 데이터 (test.csv)

데이터 전처리 및 모델 학습

예측 모델 - 구현

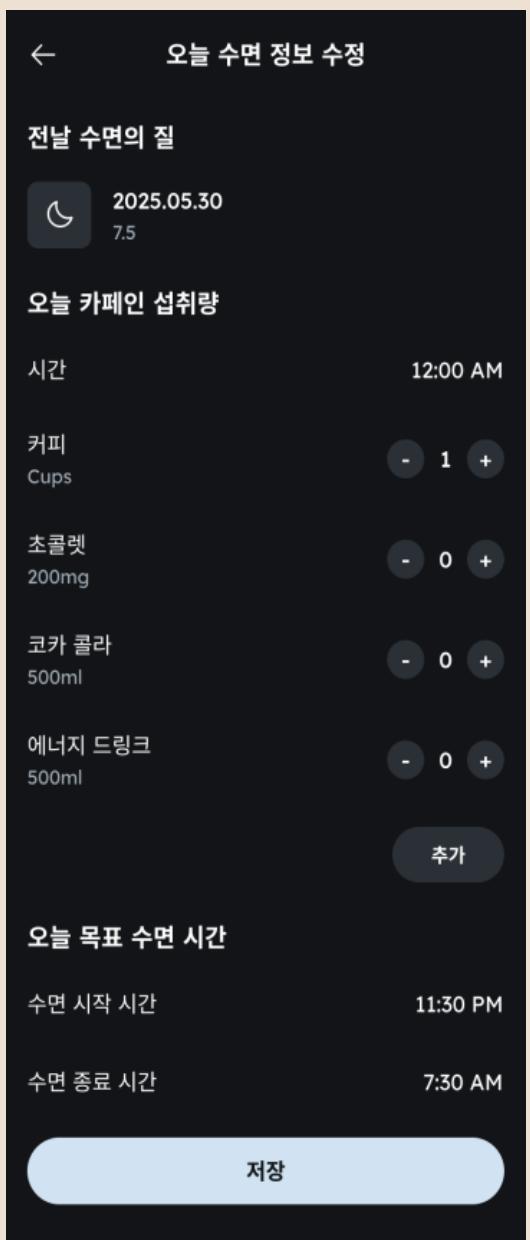
```
● ● ●  
import json  
  
json_input_str = """  
{  
    "prev_sq": 7.5,  
    "caffeine_intake": {  
        "8": 120,  
        "12": 150,  
        "20": 50  
    },  
    "target_sleep_start": 23.5,  
    "target_sleep_end": 7.5  
}"  
  
input_data = json.loads(json_input_str)  
  
prev_sq = input_data['prev_sq']  
caffeine_dict = input_data.get('caffeine_intake', {})  
target_sleep_start = input_data['target_sleep_start']  
target_sleep_end = input_data['target_sleep_end']  
  
hourly_full_day = np.zeros(24)  
for hour, amount in caffeine_dict.items():  
    hour_int = int(hour)  
    hourly_full_day[hour_int] = amount  
hourly_input_batch = hourly_full_day.reshape(1, STATEFUL_TIME_STEPS, INPUT_FEATURES_PER_HOUR)  
static_input_batch = np.array([[target_sleep_start, target_sleep_end, prev_sq]]).reshape(1, STATIC_FEATURES_PER_DAY)  
  
lstm_layer_instance.reset_states()  
predicted_sq = model.predict([hourly_input_batch, static_input_batch], verbose=0)[0, 0]  
  
output_json = {"predicted_sleep_quality": round(float(predicted_sq), 4)}  
print(json.dumps(output_json))
```

예시 데이터 입력, 출력 로직



결과

어플리케이션 프로토타입



사용자 입력 화면



결과 화면

앱 한계점



변수에 대한 고려

카페인 외 수면에 영향을 미치는 요인(오늘 기준)은 고려하지 않았고, 생물학적 메커니즘을 학습한 게 아니라, 단순히 카페인 섭취량과 수면의 질 변수의 관계를 바탕으로 판단하고 있음.



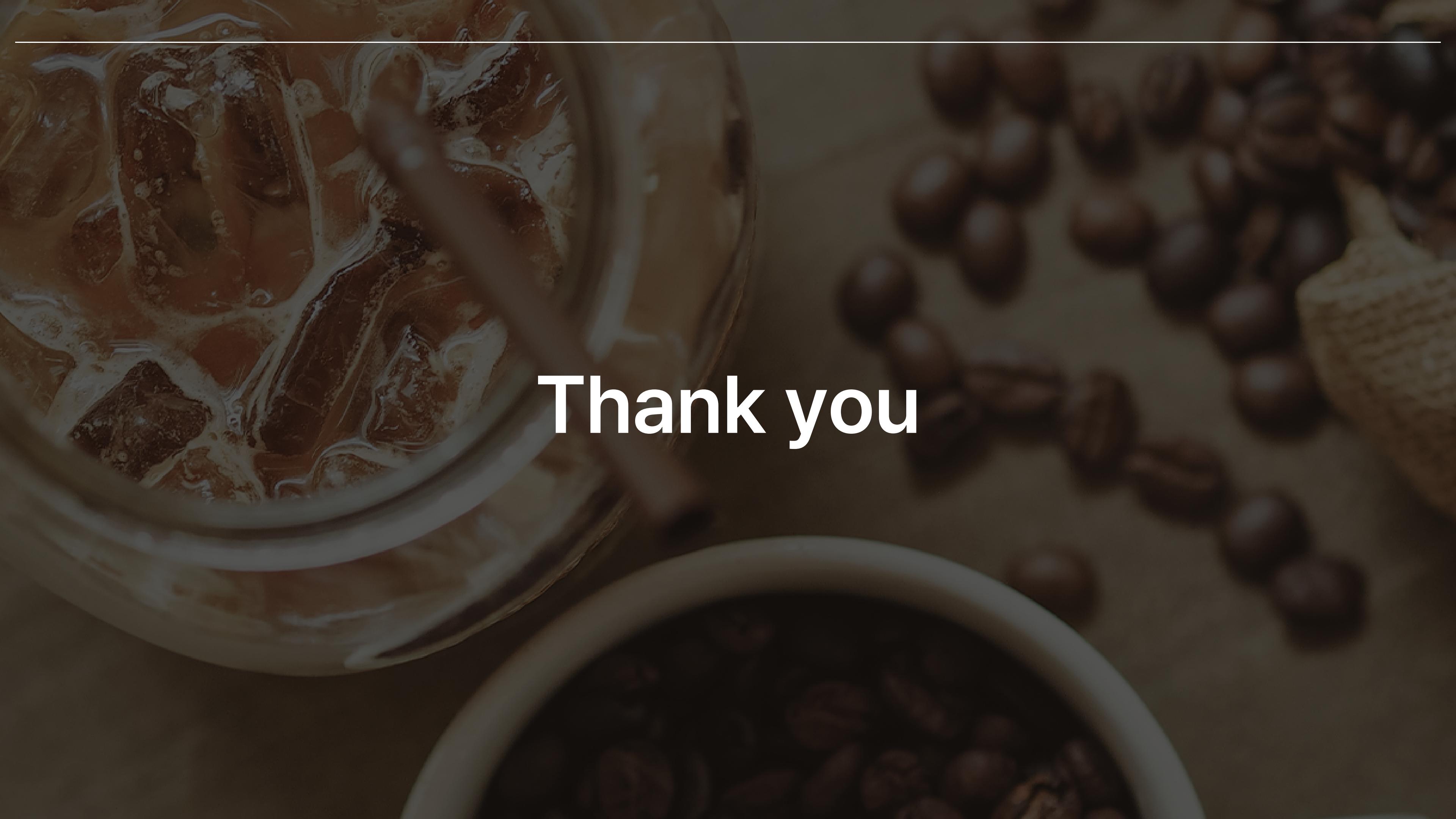
빈약한 예측력

사용자가 꾸준히 데이터를 입력하지 않거나 사용자의 패턴이 갑자기 변하면 예측력이 떨어질 수 있음.



서버 부담 증가

사용자 별 맞춤 학습을 하려면 서버 과부화가 생길 수 있음



Thank you