PT 회원 관리 솔루션

다이어트 기록 시스템

팀명 : 창규바라기 (최창규, 박현명, 진소연)

목차

① 프로그램 기획 의도

HOME > 헬스산업포커스

올해 글로벌 헬스케어 시장 2조 2840억 달러 규모

문윤희 기자 | 승인 2022.06.23 06:52 | 댓글 0

헬스케어 시장 중 디지털헬스는 전년 대비 15.0%로 가장 빠르게 성장하는 반면 규모가 가장 큰 의약품 시장은 3.8%로 가장 낮은 성장세를 보인 것으로 파악됐다.

보고서는 "코로나19가 헬스케어 산업의 혁신을 가속화하고 재정비할 수 있는 기회를 제공했다"고 평가하면서 "코로나19 대유행, 의학의 급격한 발전, 디지털 기술 · 데이터 액세스 및 분석의 폭발적 증가, 질병 관리 중심에서 예방과 웰빙으로의 전환 등 여러 트렌드가 대격변을 일으키면서 헬스케어 산업의 의학, 재무 및 경영 혁신의 촉매제로 작용했다"고 분석했다.

그러면서 "2022년은 코로나19 팬데믹이 시작된 두 번째 해로, 아직도 진행 중인 팬데믹에 지속적인 관심과 자원을 집중될 것으로 예상된다"고 내다봤다.

	2021년	2022년(Aspirational)	2022년(Conservative)	전년대비 성장율 (낙관적 전망)
Pharmaceuticals	1,429	1,483	1,476	3.8%
Medical devices	425	466	460	9.3%
Digiatl health	178	205	203	15.0%
Next-generation diagnostics	99	107	106	7.5%
Medical imaging and Informatics	38	41	39	8.6%
합계	2,169	2,302	2,284	

¶ Aspirational(낙관적 전망), Conservative(보수적 전망)

출처 : 프로스트앤설리반, Global Pharmaceuticals Outlook, 2022; Global Medical Devices Outlook, 2022; Global Digital Health Outlook, 2022; Next-generation Diagnostics Outlook, 2022; Global Medical Imaging and Informatics Outlook, 2022, 2022.3

(출처 : 뉴스 더보이스 헬스케어)

헬스케어에 대한 시장 관심도 및 성장속도 급상승

기존 PT 시스템에서 효율성 개선의 여지

② 프로젝트 개발 환경

- Python 버전 : 3.9.x
- OS (운영체제) 버전 : Windows 11, Windows 10
- IDLE (개발툴) : conda 4.14.0, Jupyter Notebook
- 시스템 : RAM = 20GB

③ 프로젝트 분담 및 팀원 역할

- 최창규 : 팀장, 코딩 총괄

코드 : (1) 회원등록, (2) 칼로리 처방전

(4) 하루 총칼로리 출력

- 박현명 : 프레젠테이션 자료 제작

코드 : (1) 회원등록, (3) 하루 섭취 칼로리 및 하루 운동량

- 진소연 : 기안서 작성 및 발표

코드 : (2) 칼로리 처방전, (3) 하루 섭취 칼로리 및 하루 운동량

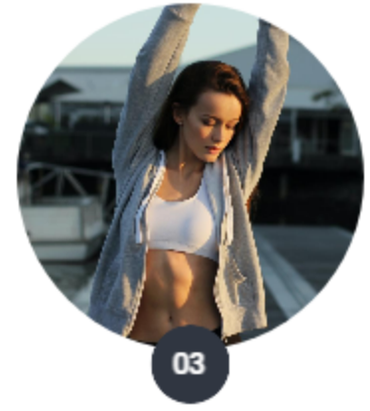
④ 주요 기능 소개

회원등록

닉네임, 체중, 성별, 키,
나이, 목표 체중, 목표 일수 입력시
기초대사량, bmi 지수가
자동으로 계산되어 회원으로 등록

칼로리 처방전

목표 달성 위해 필요한
칼로리 소비량 제시
체질량 지수 및 bmi 안내



하루 섭취 칼로리 및 하루 운동량

회원 정보에
하루 섭취 칼로리, 운동량
추가 저장 (하루 기준 업데이트)

하루 총 칼로리 출력

성공/실패 여부 판단,
실패일 경우
추가 운동량 제안

⑤ 주요 코딩 상세설명

```
def presc(self): # 함수 압축
        self.presc_kcal()
        self.presc_bmi()

    def presc_bmi(self): # 체질량지수 계산 함수
        print('현재 체질량지수는 {:.2f}이고, '.format(int(pt_Info[self._id][6]), end='')) # 체질량 지수를 소수점 2자리 까지 출력
        if int(pt_Info[self._id][6]) < 18.5: # 체질량 지수에 따른 결과 출력문
            print('bmi 결과 저체중입니다.\n')
        elif int(pt_Info[self._id][6]) <= 22.9:
            print('bmi 결과 정상입니다.\n')
        elif int(pt_Info[self._id][6]) <= 24.9:
            print('bmi 결과 과체중입니다.\n')
        elif int(pt_Info[self._id][6]) <= 29.9:
            print('bmi 결과 경도 비만입니다.\n')
        elif int(pt_Info[self._id][6]) <= 39.9:
            print('bmi 결과 중등도 비만입니다.\n')
        elif int(pt_Info[self._id][6]) > 39.9:
            print('bmi 결과 고도 비만입니다.\n')

    def presc_kcal(self): # 하루에 빼야할 칼로리 계산 함수
        while True:
            self._id = input("닉네임 = ")
            print()
            if self._id != "":  # 닉네임이 공백일 경우 예외처리
                if self._id in pt_Info.keys():  # 유효성 검사(pt_Info에 self._id가 있는지 확인)
                    allkcal = 7000 * (int(pt_Info[self._id][0]) - int(pt_Info[self._id][4])) # 7000 * 1kg 은 1kg 당 칼로리
                    # (현재 체중 - 목표체중) * 7000 = 빼야할 칼로리
                    daykcal = allkcal / int(pt_Info[self._id][5]) # 빼야할 칼로리 / 목표일수 = 하루에 빼야할 칼로리
                    print('{}님의 목표 달성을 위해 하루 {:.2f}칼로리 감소 플랜을 제안합니다.\n'.format(self._id, daykcal))
                    return self._id
```

⑤ 주요 코딩 상세설명

```
def kcalkcal(self): # 함수 압축
        self.eat()
        self.workout()

    def eat(self): # 섭취 칼로리
        while True:
            self._id = input("닉네임 = ")
            print()
            if self._id != "": # 공백 예외처리
                if self._id in pt_Info.keys(): # 유효성 검사
                    inventory = input('오늘 하루 섭취한 총 칼로리를 입력하세요. : ')
                    print()
                    if len(inventory) != '':
                        # pt_Info[self._id]리스트 안에 inventory라는 값 추가
                        pt_Info[self._id].append(inventory) # pt_Info[self._id][9]
                        print('{}님이 오늘 하루 섭취한 칼로리는 {}입니다.\n'.format(self._id, inventory))
                        break
                    else:
                        print('칼로리 칸은 비워둘 수 없습니다.')
                else:
                    print('존재하지 않는 닉네임입니다.\n')
            else:
                print('닉네임은 비워둘 수 없습니다.\n')
```

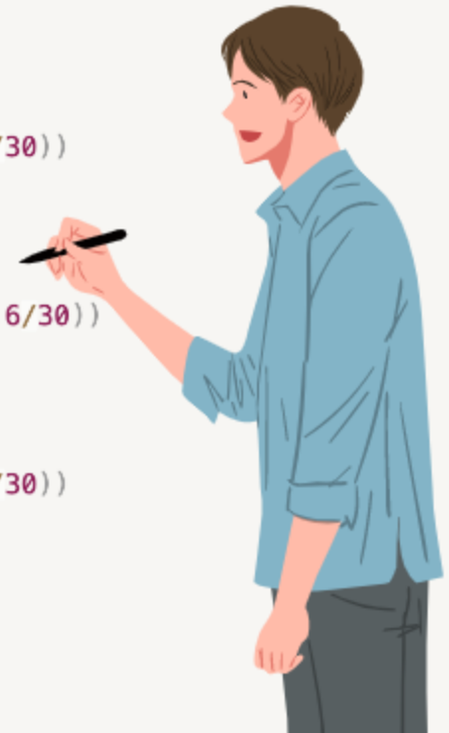
⑤ 주요 코딩 상세설명

```
def workout(self): # 운동 칼로리
    day_kcal = 0 # 초기값 설정
    while True:
        self.wo = input("오늘의 운동 종목('끝' 입력 시 종료) : ")
        print()
        if len(self.wo) != 0:
            if self.wo == '끝': # '끝' 입력시
                print("오늘의 운동이 끝났습니다. 수고하셨어요!\n")
                pt_Info[self._id].append(day_kcal)
                break
            else:
                if self.wo in workout: # 유효성 검사
                    time = int(input("운동 시간(분) : "))
                    print()
                    if self.wo == workout[0]:
                        print("{:.2f}칼로리를 소모하셨습니다.\n".format(float(pt_Info[self._id][0])*time*1.8/30))
                        day_kcal += float(pt_Info[self._id][0]) * time * 1.8 / 30

                    elif self.wo == workout[1]:
                        print("{:.2f}칼로리를 소모하셨습니다.\n".format(float(pt_Info[self._id][0])*time*4/30))
                        day_kcal += float(pt_Info[self._id][0]) * time * 4 / 30

                    elif self.wo == workout[2]:
                        print("{:.2f}칼로리를 소모하셨습니다.\n".format(float(pt_Info[self._id][0])*time*4.6/30))
                        day_kcal += float(pt_Info[self._id][0]) * time * 4.6 / 30

                    elif self.wo == workout[3]:
                        print("{:.2f}칼로리를 소모하셨습니다.\n".format(float(pt_Info[self._id][0])*time*4/30))
                        day_kcal += float(pt_Info[self._id][0]) * time * 4 / 30
                else:
                    print("데이터베이스에 없는 종목입니다.\n")
    else:
        print('운동칸에는 걷기, 달리기, 수영, 자전거 를 입력해주세요.')
```



⑤ 주요 코딩 상세설명

```
def workout(self): # 운동 칼로리
    day_kcal = 0 # 초기값 설정
    while True:
        self.wo = input("오늘의 운동 종목('끝' 입력 시 종료) : ")
        print()
        if self.wo == '끝': # '끝' 입력시
            print("오늘의 운동이 끝났습니다. 수고하셨어요!\n")
            pt_Info[self._id].append(0) # pt_Info[self._id]리스트에 0추가 pt_Info[10 ~ 11]
            break
        else:
            if self.wo in workout: # 유효성 검사
                time = int(input("운동 시간(분) : "))
                print()
                if self.wo == workout[0]:
                    print("{:.2f}칼로리를 소모하셨습니다.\n".format(float(pt_Info[self._id][0])*time*1.8/30))
                    day_kcal += float(pt_Info[self._id][0]) * time * 1.8 / 30
                    pt_Info[self._id].append(day_kcal)

                elif self.wo == workout[1]:
                    print("{:.2f}칼로리를 소모하셨습니다.\n".format(float(pt_Info[self._id][0])*time*4/30))
                    day_kcal += float(pt_Info[self._id][0]) * time * 4 / 30
                    pt_Info[self._id].append(day_kcal)

                elif self.wo == workout[2]:
                    print("{:.2f}칼로리를 소모하셨습니다.\n".format(float(pt_Info[self._id][0])*time*4.6/30))
                    day_kcal += float(pt_Info[self._id][0]) * time * 4.6 / 30
                    pt_Info[self._id].append(day_kcal)

                elif self.wo == workout[3]:
                    print("{:.2f}칼로리를 소모하셨습니다.\n".format(float(pt_Info[self._id][0])*time*4/30))
                    day_kcal += float(pt_Info[self._id][0]) * time * 4 / 30
                    pt_Info[self._id].append(day_kcal)
            else:
                print("데이터베이스에 없는 종목입니다.\n")
```

⑤ 주요 코딩 상세설명

```
def all_kcal(self):  # 하루 총 칼로리 = 하루 섭취 칼로리 - (기초 대사량 + 하루 운동 칼로리)
        while True:
            self._id = input("닉네임 = ")
            print()
            if self._id != "": # 예외처리
                if self._id in pt_Info.keys(): # 유효성 검사
                    all_kcal = int(pt_Info[self._id][9]) - int(pt_Info[self._id][7]) - int(pt_Info[self._id][10])
                    # 하루 총 칼로리 계산
                    daykcal = 7000 * (int(pt_Info[self._id][0]) - int(pt_Info[self._id][4])) / int(pt_Info[self._id][5])
                    # 하루에 소모해야할 칼로리 계산
                    time = 0
                    if all_kcal < 0:
                        time = (daykcal + all_kcal) * 30 / int(pt_Info[self._id][0])
                    else:
                        time = (daykcal - all_kcal) * 30 / int(pt_Info[self._id][0])
                    # 하루에 소모 할 칼로리가 남았을때의 시간 계산
                    if daykcal < - all_kcal: # 하루 총 칼로리가 소모해야 할 칼로리보다 클 때
                        print('{}님은 오늘 {:.2f}칼로리를 소모하셨습니다.\n'.format(self._id, - all_kcal))
                        # 소모한 칼로리는 음수이기때문에 -를 붙여줌
                        print('오늘은 다이어트 성공으로 기록됩니다.\n')
                        pt_Info[self._id][8]['vic'] += 1 # pt_Info[self._id][8]에 있는 'vic'의 값에 1을 더해줌
                        print('성공 횟수 {}회\n'.format(pt_Info[self._id][8]['vic'])) # 더해준 'vic'의 값을 출력
                        print('실패 횟수 {}회\n'.format(pt_Info[self._id][8]['def'])) # 지금까지 실패한 횟수 출력
                        if pt_Info[self._id][5] == 0:
                            print('목표일이 되었습니다.\n')
                            if pt_Info[self._id][8]['def'] == 0:
                                print('목표를 완벽하게 달성하셨습니다.\n')
                                break
                            else:
                                print('목표를 달성하지 못하셨습니다.\n')
                                break
                        else:
                            pt_Info[self._id][5] -= 1
                            print('남은 일수 : {}일\n'.format(pt_Info[self._id][5]))
                            break
                    else:
                        if all_kcal < 0:
                            print('{}님은 오늘 {:.2f}칼로리를 소모하지 못했습니다.\n'.format(self._id, daykcal + all_kcal))
```

⑤ 주요 코딩 상세설명

```
                print('그러므로 추가적인 운동을 제안드립니다.\n')
                print('{0} : {1:.0f}분, {2} : {3:.0f}분'.format(workout[0], time / 1.8 , workout[1], time / 4)) # 운동별 남은 칼로리 소모
                print('{0} : {1:.0f}분, {2} : {3:.0f}분\n'.format(workout[2], time / 4.6, workout[3], time / 4))  # 운동별 남은 칼로리 소모
                print('오늘은 다이어트 실패로 기록됩니다\n.')
                pt_Info[self._id][8]['def'] += 1 # pt_Info[self._id][8]에 있는 'def'의 값에 1을 더해줌
                print('성공 횟수 {}회\n'.format(pt_Info[self._id][8]['vic']))
                print('실패 횟수 {}회\n'.format(pt_Info[self._id][8]['def']))
                if pt_Info[self._id][5] == 0:
                    print('목표일이 되었습니다.\n')
                    print('목표를 달성하지 못하셨습니다.\n')
                    break
                else:
                    pt_Info[self._id][5] -= 1
                    print('남은 일수 : {}일\n'.format(pt_Info[self._id][5]))
                    break
            else:
                print('{}님은 오늘 {:.2f}칼로리를 소모하지 못했습니다.\n'.format(self._id, daykcal - all_kcal))
                print('그러므로 추가적인 운동을 제안드립니다.\n')
                print('{0} : {1:.0f}분, {2} : {3:.0f}분'.format(workout[0], time / 1.8 , workout[1], time / 4)) # 운동별 남은 칼로리 소모
                print('{0} : {1:.0f}분, {2} : {3:.0f}분\n'.format(workout[2], time / 4.6, workout[3], time / 4))  # 운동별 남은 칼로리 소모
                print('오늘은 다이어트 실패로 기록됩니다\n.')
                pt_Info[self._id][8]['def'] += 1 # pt_Info[self._id][8]에 있는 'def'의 값에 1을 더해줌
                print('성공 횟수 {}회\n'.format(pt_Info[self._id][8]['vic']))
                print('실패 횟수 {}회\n'.format(pt_Info[self._id][8]['def']))
                if pt_Info[self._id][5] == 0:
                    print('목표일이 되었습니다.\n')
                    print('목표를 달성하지 못하셨습니다.\n')
                    break
                else:
                    pt_Info[self._id][5] -= 1
                    print('남은 일수 : {}일\n'.format(pt_Info[self._id][5]))
                    break
    try: # 예외처리로 인해 생긴 리스트 삭제
        del pt_Info[self._id][11]
    except: # 실패할 시
        pass # 건너뜀
    del pt_Info[self._id][10] # 하루 운동 칼로리 삭제
    del pt_Info[self._id][9] # 하루 섭취 칼로리 삭제
```


Q & A
질문주세요!