SPRINT - 3

caloriestracker

Team Id	PNT2022TMID50493
Date	12.11.22
Project Name	Ai-powered Nurition Analyzer and Fitness Enthusiasts

```
from datetime import datetime
from caloriestracker.objects.meal import MealManager_from_sql
from caloriestracker.objects.company import CompanyPersonal
from caloriestracker.objects.product import ProductPersonal
from caloriestracker.objects.company_product import CompaniesAndProducts
from caloriestracker.text_inputs import  input_decimal, input_string
from caloriestracker.mem import MemConsole
from caloriestracker.npyscreen import MealAddApp, BiometricDataAddApp
from caloriestracker.contribution import generate_contribution_dump,
parse_contribution_dump_generate_files_and_validates_them
from logging import debug
from sys import exit

def main():
    mem=MemConsole()
    mem.run()
    debug(mem.tr("Start mem took {}".format(datetime.now()-mem.inittime)))

    if mem.args.find!=None:
        cp=CompaniesAndProducts(mem)
        cp.find_report(mem.args.find)
        exit(0)

    if mem.args.add_company==True:
        name=input_string("Name of the company: ")
        o=CompanyPersonal(mem)
        o.name=name
```

```
    o.last=datetime.now()
    o.obsolete=False
    o.save()
    mem.con.commit()
    print("CompanySystem added with id={}".format(o.id))
    exit(0)
if mem.args.elaborated!=None:
   elaborated=mem.data.elaboratedproducts.find_by_id(mem.args.elaborated)
   elaborated.needStatus(1)
   elaborated.show_table()
   exit(0)
if mem.args.add_meal==True:
    app = MealAddApp(mem)
    app.run()
    print(app.log)
    exit(0)
if mem.args.add_product==True:
    name=input_string("Add a name: ")
    company=mem.data.companies.find_by_input()
    system_company=None if company==None else company.system_company
    amount=input_decimal("Add the product amount", 100)
    carbohydrate=input_decimal("Add carbohydrate amount",0)
    protein=input_decimal("Add protein amount", 0)
    fat=input_decimal("Add fat amount", 0)
    fiber=input_decimal("Add fiber amount", 0)
    calories=input_decimal("Add calories amount", 0)
    o=ProductPersonal(mem)
    o.mem=name
    o.amount=amount
    o.fat=fat
    o.protein=protein
    o.carbohydrate=carbohydrate
    o.company=company
    o.last=datetime.now()
    o.calories=calories
    o.fiber=fiber
    o.system_company=system_company
```

```
        #Basic insert not all attributes

        o.save()
        mem.con.commit()
        print("Product added with id={}".format(o.id))
        exit(0)

    if mem.args.add_biometrics==True:
        app = BiometricDataAddApp(mem)
        app.run()
        exit(0)

    if mem.args.contribution_dump==True:
        mem.languages.cambiar("en", "caloriestracker")
        generate_contribution_dump(mem)
        exit(0)

    if mem.args.parse_contribution_dump!=None:
        mem.languages.cambiar("en", "caloriestracker")
        dbversion=mem.con.cursor_one_field("select value from globals where
global='Version'")

filenameversion=mem.args.parse_contribution_dump.replace("caloriestracker_collabora
tion_", "").replace(".sql", "")
        if dbversion==filenameversion:
            parse_contribution_dump_generate_files_and_validates_them(mem.con,
mem.args.parse_contribution_dump)
        else:
            print("The dump provided can't be parsed due to is from {} version and developer
version is {}. Tell the user to update app and resend dump.".format(filenameversion,
dbversion, ))
        exit(0)

    if mem.args.update_after_contribution!=None:
        mem.languages.cambiar("en", "caloriestracker")
        dbversion=mem.con.cursor_one_field("select value from globals where
global='Version'")
```

```
        filenameversion=mem.args.update_after_contribution[0:12]
        print(dbversion,  filenameversion)
        if dbversion==filenameversion:
            mem.con.load_script(mem.args.update_after_contribution)
            mem.con.commit()
            print("Personal data you sent to us, have been integrated in the Calories Tracker
system data. Thank you ;)")
        else:
            print("Your Calories Tracker app should be in {} version to parse this script.
Please update to that version because it's {}".format(dbversion, filenameversion))

        exit(0)

    user=mem.data.users.find_by_id(mem.args.users_id)

    meals=MealManager_from_sql(mem, "select * from meals where datetime::date=%s
and users_id=%s", (mem.args.date, user.id))
    meals.order_by_datetime()
    meals.show_table(mem.args.date)
```