Перш за все, хотілося б зазначити зміни у архітектурі нашої курсової роботи. Замість трьох абстрактних типів даних ми будемо використовувати нові два класи, які я опишу згодом. Ідея з аналізом прикметників у коментарях з метою надання оцінки тим коментарям у яких вона відсутня виявилася не підходящою через занадто довгу тривалість виконання програми. Друга причина – це знайдення нового інтернет магазину “Comphy” з великою кількістю коментарів разом з оцінкою того чи іншого продукту. Також на сайті “Hotline” ми знайшли середню оцінку даному продукту, отже всім коментарям з цього ресурсу ми будемо присвоювати цю оцінку. Без оцінок лишився малий відсоток коментарів з сайту “Rozetka”, ці коментарі не будуть враховуватися під час обчислення загальної оцінки аналізованого товару.

Нові два типі типи даних, які ми будемо використовувати це клас Comment та клас Comments. Клас Comment має два поля – текст коментаря та оцінку продукту, клас Comments буде мати одне два поля, а саме список коментарів та середню оцінку продукту, а також метод determine\_average\_mark, який буде повертати середню оцінку продукту. Незважаючи на всю простоту даних класів, вони є оптимальними для вирішення поставленої проблеми.

Також, ми зіткнулися з проблемою довгого видобутку коментарів з сайтів. Для покращення часу роботи ми використали базу даних, у якій зберігаємо назву продукту, всі коментарі з трьох інтернет ресурсів та середню оцінку. Отже, після першого пошуку продукту знаходяться всі його, зберігаються у базу даних та повертаються користувачеві. Під час будь-якого іншого пошуку того ж продукту повернення його даних буде моментальним через використання напередодні збереженої інформації у базі даних. Назву продукту у базі даних ми зберігали у нижньому регістрі та без пробілів з метою обробки ситуації повторного збереження одного і того самого продукту в базу даних.

Інша зміна, котру ми внесли у нашу курсову роботу це зміна веб фреймворку Flask на Django. Головна причина чому ми це зробили – це легкість в роботі з базами даних у Django та можливість на практиці ознайомитися з цим Python інструментом.

Незважаючи на швидкість отримання інформація для продукту, який вже запитувався раніше, пошук інформації про продукт котрого немає у базі даних займає занадто багато часу. Для вирішення цієї проблеми, ми будемо використовувати JavaScript. Після запиту користувачу буде відправлена тільки одна третя інформації, яка може бути знайдена найшвидше. Після повернення користувачу сторінки на ній ще будуть виконуватися дві JavaScript функції, які отримають інформацію з двох інших інтернет ресурсів та візуалізують її на даній сторінці. Це дозволить менше часу чекати користувачу в очікуванні переходу на наступну сторінку.

Так як всі ресурси з яких ми черпаємо інформацію є ресурсами, котрі займаються електронікою (“Rozetka” - продаж, “Comphy” - продаж, “Hotline” - аналіз) ми вирішили обмежити спектр продуктів для яких може бути здійснений аналіз до суто електронних пристроїв.

Ось реалізація двох використовуваних класів.

# File: comment.py  
# Module has a Comment and Comments classes.  
  
  
class Comment:  
 *""" Represents a comment object. """* def \_\_init\_\_(self, text, mark=None):  
 *""" Initialize a text of comment and mark. """* self.text = text  
 self.mark = mark  
  
  
class Comments:  
 *""" Represent a conatainer of comments. """* def \_\_init\_\_(self):  
 *""" Initialize an empty list. """* self.comments = []  
 self.average\_mark = None  
  
 def determine\_average\_mark(self):  
 *""" Initialize an average mark among all comments. """* total, divide = 0, 0  
 for comment in self.comments:  
 if comment.mark is not None:  
 total += comment.mark  
 divide += 1  
 return total // divide

В залежності від шуканого товару кількість видобувних коментарів може змінюватися. Для деяких, до прикладу телефонів кількість коментарів дорівнює нулю, для інших ж, більш популярних ця цифра може підніматися до двадцяти. Модулі, котрі здобувають інформацію (коментарі) працюють таким чином, що поле text класу Comment може бути ініціалізоване не пустою стрічкою, яка не містить знаку питання у кінці (це зроблено для мінімізації некорисною для користувача інформації на сайті), поле mark може бути ініціалізоване або об’єктом типу int, або об’єктом типу float, або об’єктом None. З цього випливає, що виконання методу determine\_average\_mark буде працювати безпомилково та повертати коректну відповідь після будь-якого виклику цього методу, важливо зазначити, що у поле comments, яке є списком, будуть добавлятися лише об’єкти класу Comment.

Наявність оцінки буде у 90 % всіх об’єктів, котрі знаходяться у списку comments, а отже похибка буде незначною.

Для отримання коментарів та оцінок для кожного інтернет ресурсу буде використовуватися окремий модуль, котрий буде містити функцію insert\_comments, вхідними параметрами цього методу буде назва продукту та об’єкт класу Comments. Ці функції нічого не повертають, а лише додають до поля comments класу Comment об’єкти класу Comment.