УДК 681.3.07

В. В. Войтко
С. В. Бевз
Д. А. Афанасьєв
В. І. Волков
Б. І. Дудченко
П. В. Криворучко

РОЗРОБКА СИСТЕМИ З ВЕБ-ДОДАТКІВ “PICLE”, СПРЯМОВАНИХ НА ЕФЕКТИВНИЙ ЗБІР, КЛАСИФІКАЦІЮ ТА ДОСТАВКУ НОВИН КОРИСТУВАЧУ

Вінницький національний технічний університет

Анотація

Розглянуто основні особливості та ключові рішення, прийняті при розробці системи “Picle”. Описано головний патерн програмування, який ліг в основу веб-системи, а також використані технології: Python Django, Python Celery, Django REST Framework, Python Aiogram, Bootstrap.

Ключові слова: *веб-додаток, екосистема, Python, Django, Celery, Aiogram, Bootstrap.*

Abstract

This article presents the process of building Picle ecosystem, considering some of the problems we faced and choices we made. The article emphasizes the main software design pattern used in the project and other technologies, such as: Python Django, Python Celery, Django REST Framework, Python Aiogram, Bootstrap.

Keywords: *web-application, ecosystem, Python, Django, Celery, Aiogram, Bootstrap.*

Вступ

У свій час Уїнстон Черчіль говорив: «Хто володіє інформацією – той володіє світом» [1]. Актуальність цієї думки завжди була дуже високою. Сьогодні ж, під час особливо стрімкого розвитку як технологій, так і людства в цілому, вона набуває епохального значення.

Сьогодні неозброєним оком видно, що світ взагалі не такий, як був раніше. Просто величезна кількість нових технологій, винаходів, штучний інтелект та інші досягнення в області науки та робототехніки роблять наше життя веселим та справді цікавим. Кожного дня вчені намагаються віднайти у всесвіті нові цивілізації, створюються все нові та нові смартфони, електромобілі на дорогах вже домінують над звичайними, зрештою, активно розробляється та аналізується план колонізації Марсу. На фоні крихітного повсякденного життя звичайних людей, ввесь той об’єм інформації, що виробляється сьогодні, є величезною заплутаною лавиною, дістати з якої потрібні дані іноді буває досить складно.

Тож головним завданням роботи є створення екосистеми з веб-додатків, спрямованих на ефективний збір, класифікацію та доставку новин користувачу з можливістю детального задання фільтрів та критеріїв для вибірки, власне, тих самих новин. Ідеєю такого програмного продукту є надання користувачеві максимально швидкого да зручного доступу до інформації, яка цікавить саме його, а також збереження невеликої історії отриманих новинних сповіщень. Система розробляється для декількох платформ, так що у будь-якого типу користувачів не повинно виникати проблем із користуванням запропонованим сервісом.

Розробка веб-сервісу «Picle»

Picle – це набір систем, який включає в себе такі додатки: веб-сайт, телеграм-бот, Android-програма. Усі вони синхронізуються між собою та обмінюються даними про переглянуті користувачами новини, для того, щоб не показувати одну новину двічі. Таким функціоналом забезпечує спільний HTTP веб-сервер,

що лежить в основі усього ресурсу. Під час побудови плану розробки основної частини проєкту виникало питання, який патерн програмування використовувати. Вибір стояв між таки відомими шаблонами, як: MVP, MVVM, та MVC і MVP (Model-View-presenter) – шаблон проєктування, який зазвичай використовується для побудови інтерфейсу користувача. Елемент Model є поданням бази даних, View – реалізує відображення даних з моделі. Частина Presenter у даному випадку є посередником і відповідає за керування подіями з інтерфейсу користувача [2]. В свою чергу, MVVM (Model-View-ViewModel) замість елементу presenter містить ViewModel. Його можна вважати деякою абстракцією, яка містить модель, пов'язану із поданням, і команди, яким подання View може впливати на Model [3]. Остання з розглянутих концепцій MVC (Model-View-Controller) являє собою досить просту та ефективну схему розділення логіки виконання програми, де елемент Controller обробляє та передає запит користувача клієнтові. [4] Було обрано саме цей шаблон проєктування, а як наслідок – Python фреймворк Django, що його підтримує. Розглянемо патерн MVC детальніше.

Робота усієї програми базується на взаємодії трьох обов'язкових компонентів: бази даних (Model), контролера (Controller) та подання даних (View). Ключовим вузлом у цій системі є контролер: він опрацьовує вхідні дані і на їх основі регулює роботу з базою даних та з поданням інформації. Простими словами, існує спеціальний механізм, який отримує запит від користувача, потім на його основі робить новий запит уже до бази даних на отримання або зберігання інформаційних ресурсів. База даних виконує потрібні операції та повертає результат, який, знову ж таки, перехоплюється контролером. Отримані дані обробляються та надсилаються до подання, яке відповідає за коректне відображення інформації користувачеві (див. рисунок 1).

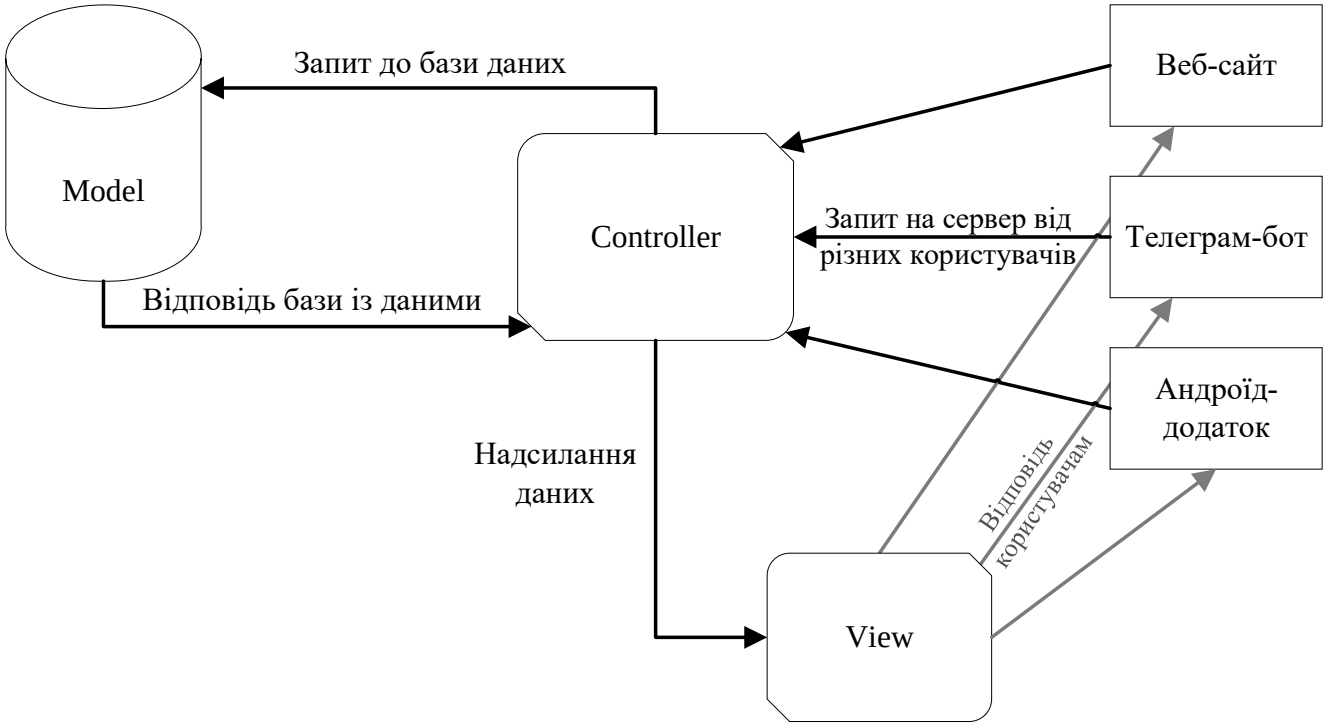

Рис.1. Схема роботи паттерну програмування Model-View-Controller

Використання такого принципу дозволяє чітко розділити роботу над поданням (інтерфейсом програми) і роботу над самою програмою. Також частково вирішується проблема взаємодії з різними типами користувачів, які використовують різні види пристроїв. Можемо інкапсулювати в контролері логіку для обробки запитів як браузера, так і мобільної програми, щоб він в результаті повертав одні і ті ж дані в різному вигляді чи форматі залежно від адресанта. Як наслідок, вся складна робота відбувається в контролері, а на виході маємо чисті перетворені дані. Окрім ізолювання видів логіки, концепція MVC

значно зменшує складність великих програм: код виходить структурованим та зрозумілим інтуїтивно, отже і підтримувати та розширювати програму стає набагато легше.

Розробка системи асинхронної відправки новин

Ще однією вагомою частиною в реалізації поставленої задачі стала розробка і налаштування системи асинхронної доставки новин багатьом користувачам одразу. Задачу ускладнював ще той факт, що кожен користувач сам міг вибирати частоту отримування новин, категорії новин, які хоче мати, і місце, куди хоче отримувати новини: телеграм бот або електронна пошта. Звісно, можна було реалізувати простий перебір новин і розбити їх на потоки для кожного зареєстрованого користувача. Очевидно, що такий підхід надмірно навантажував би систему зі збільшенням кількості користувачів. Тому було вирішено використовувати Celery – асинхронну систему черг із задач з відкритим вихідним кодом, написану на Python [5].

Celery являє собою фреймворк, який легко поєднується з такими фреймворками, як Django та Flask. Для коректної роботи Celery потребує з'єднання з невеличною базою даних (broker), в якій зберігається кеш та інша потрібна для роботи інформація. Для цього була використана система управління базами даних "Redis" [6]. Власне робота системи полягає у тому, що вона отримує із головної програми задачі (tasks), розташовує їх у черги за пріоритетами і за допомогою декількох потоків (workers) виконує задачі. За необхідності результати повертаються у головну програму. Такий підхід ніяк не впливає на роботу головного HTTP серверу, проте витрачає багато ресурсів комп'ютера. Робота Celery зображена на рис. 2.

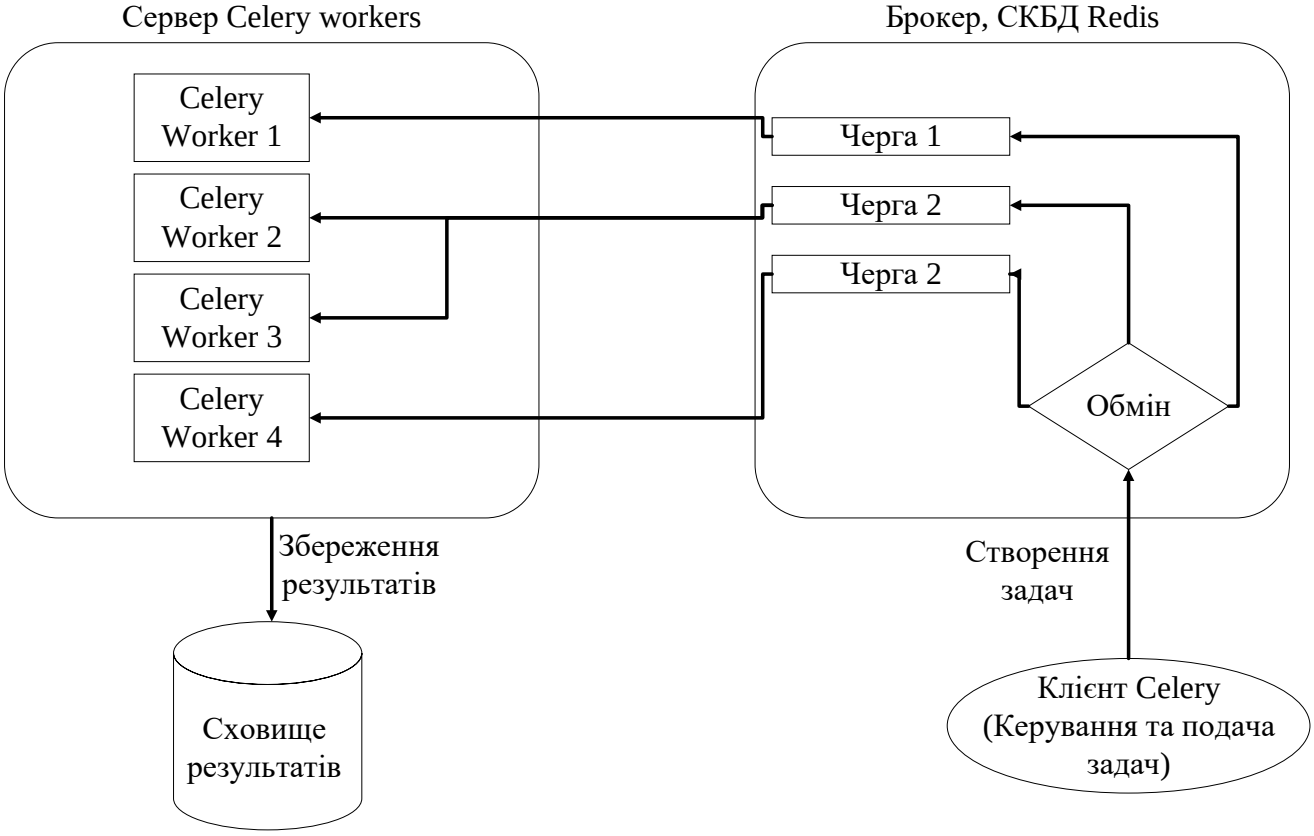

Рис.2. Схема роботи фреймворку Celery

Задачі (tasks), які виконував Celery, були пов'язані з відправленням повідомлень із новинами на пошту та в телеграм. Сервер Celery виконував у наперед заданий час деякий Python код, який обирає потрібну новину та відправляє її користувачу. Завдяки цьому вдалося уникнути проблеми блокування введення-виведення у Python [7].

Інші технології, що використовувалися під час розробки

Для розробки серверної частини використовується Python-фреймворк «Django» останньої стабільної версії, а також його допоміжні бібліотеки: «Django Rest Framework» і «Django-allauth» для реалізації REST API та для системи автентифікації користувачів відповідно.

Використовується система управління реляційними базами даних PostgreSQL [8]. На етапі розробки всі дані користувачів та новин зберігаються на локальному сервері бази даних. Дані зберігаються у вигляді таблиць.

Телеграм бот будується з використанням Telegram API [9], що надає сервіс телеграму, а також python бібліотеки «Aiogram». Її особливість у тому, що побудований код є асинхронним, тобто в моменти, коли потік виконання програми мав би заблокуватися, він не блокується, а переходить до виконання іншої задачі. У нашій ситуації, коли вся взаємодія з Telegram API базується на HTTP запитах до чужого серверу (запити якраз і блокують потік виконання програми), такий підхід до написання коду значно збільшує його ефективність.

Для створення веб-інтерфейсу користувача використовується HTML, CSS та JavaScript бібліотека Bootstrap четвертої версії [10]. З її допомогою будується простий адаптивний дизайн, який гарно виглядатиме як на комп'ютерах, так і на мобільних пристроях. Для підстановки потрібної інформації у HTML-шаблони використовується Django-шаблонізатор.

Мобільна програма створюється середовищем побудови та тестування мобільних застосунків – Android Studio [11] на мові програмування Java. Це дозволяє розробку програмного забезпечення для інтерактивного спостереження та керування новинами, їх фільтрацію та розсилку в зручній для мобільного користувача формі. Під час розробки використовуються такі бібліотеки:

- Java Util, що містить основні рішення примітивних завдань;
- Moshi – для конвертації JSON в Java-об'єкти;
- Glide – для зручного завантаження та використання зображень у програмі;
- ThreeeTen – для роботи з датою та часом.

Програмування веб-серверу ведеться під операційну систему Linux Ubuntu для забезпечення максимально зручного доступу до необхідних ресурсів та інструментів. Активно використовується система управлінням версій Git [12]. З її допомогою відбувається написання коду, його порівняння, злиття та рефакторинг. За допомогою віддаленого репозиторія на GitHub відбувається обмін написаним кодом між членами команди, до того ж, там можна побачити повну історію розробки веб-серверу.

Висновок

Отже, було розроблено екосистему з веб-додатків «Picle», мета якої полягає у реалізації швидкої та ефективної екосистеми із додатків, спрямованих на ефективний збір, класифікацію та доставку новин користувачу. Розробка велася з метою полегшити потенціальним користувачам доступ до інформації у вигляді новин за допомогою пошуку та надсилання потрібних новин на пошту та в телеграм за наперед заданими критеріями. Окрім того, було досліджено декілька потужних сучасних технологій у розробці веб-додатків.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Хто володіє інформацією, той володіє світом [Електронний ресурс] Режим доступу до ресурсу: http://voproshaika.ru/kto-vladeet-informaciej-tot-vladeet-mirom-kto-skazal.html
2. Что такое MVP и как его использовать [Електронний ресурс] Режим доступу до ресурсу: https://skillbox.ru/media/code/chto_takoe_mvp_i_kak_eto_ispolzovat/
3. MVVM: полное понимание (+WPF) Часть 1 [Електронний ресурс] Режим доступу до ресурсу: https://habr.com/ru/post/338518/
4. MVC для веб: проще некуда [Електронний ресурс] Режим доступу до ресурсу: https://habr.com/ru/post/181772/

5. Introduction to Celery [Електронний ресурс] Режим доступу до ресурсу: https://docs.celeryproject.org/en/stable/getting-started/introduction.html
6. Redis documentation [Електронний ресурс] Режим доступу до ресурсу: https://redis.io/documentation
7. GIL Выполняет ли python одновременную работу потоков, когда поток блокируется операцией ввода-вывода? [Електронний ресурс] Режим доступу до ресурсу: https://progi.pro/gil-vipolnyaet-li-python-odnovremennuyu-rabotu-potokov-kogda-potok-blokiruetsya-operaciey-vvoda-vivoda-2034408
8. . PostgreSQL [Електронний ресурс] Режим доступу до ресурсу: https://www.postgresql.org/.
9. Telegram Bot API [Електронний ресурс] Режим доступу до ресурсу: https://core.telegram.org/bots/api.
10. Build fast, responsive sites with Bootstrap [Електронний ресурс] Режим доступу до ресурсу: https://getbootstrap.com/.
11. Android Studio [Електронний ресурс] Режим доступу до ресурсу: https://developer.android.com/studio.
12. Git [Електронний ресурс] Режим доступу до ресурсу: https://git-scm.com/.

Войтко Вікторія Володимирівна – кандидат технічних наук, доцент кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: dekanfki@i.ua.

Бевз Світлана Володимирівна – кандидат технічних наук, доцент кафедри електричних станцій і систем, Вінницький національний технічний університет, м. Вінниця, e-mail: svbevz@i.ua.

Афанасьєв Дмитро Андрійович – студент групи 1ПІ-19б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Україна.

Волков Вадим Ігорович – студент групи 1ПІ-19б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Україна.

Дудченко Богдан Ігорович – студент групи 1ПІ-19б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Україна.

Криворучко Павло Васильович – студент групи 1ПІ-19б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Україна.

Viktoriia Voitko – Ph.D., Associate Professor of Software Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: dekanfki@i.ua.

Svitlana Bevz – Ph.D., Associate Professor, Department of Power Plants and Systems, Vinnitsa National Technical University, Vinnitsa, e-mail: svbevz@i.ua.

Dmytro Afanasiev – student of group 1PI-19b, Faculty for Information Technologies and Computer Engineering, Vinnytsia National Technical University, Ukraine.

Vadim Volkov – student of group 1PI-19b, Faculty for Information Technologies and Computer Engineering, Vinnytsia National Technical University, Ukraine.

Bohdan Dudchenko – student of group 1PI-19b, Faculty for Information Technologies and Computer Engineering, Vinnytsia National Technical University, Ukraine.

Pavlo Krivorichko – student of group 1PI-19b, Faculty for Information Technologies and Computer Engineering, Vinnytsia National Technical University, Ukraine.