УДК 004.4

О.В Верещагіна
Н.П. Бабюк

ОСОБЛИВОСТІ ОРГАНІЗАЦІЇ КОДУ У РОЗРОБЦІ СУЧАСНИХ ВЕБ-ДОДАТКІВ

Вінницький національний технічний університет

Анотація: *В даній роботі розглянуто важливість розуміння, а також варіанти організації коду у розробці сучасних веб-додатків. Наведені два принципи організації коду, їх переваги і недоліки.*
Ключові слова: репозиторій, монорепо, мультирепо, організація коду.

Abstract: In this work the importance of understanding, as well as options for code organization in the development of modern web applications. Two principles of code organization, their advantages and disadvantages are presented.
Keywords: repository, monorepo, multirepo, code organization.

Вступ

Організація коду є важливим аспектом розробки програмного забезпечення, який може значно вплинути на зручність обслуговування та масштабованість проєкту.

Починаючи новий програмний проєкт, потрібно прийняти кілька важливих рішень. Вибір структури керування вихідним кодом, мови програмування, фреймворку має важливе значення та впливає на ведення проєкту протягом багатьох років. Для підприємств і організацій, які пропонують будь-який цифровий продукт, якість і безпека коду мають кардинальне значення. Таким чином, архітектура коду проєкту – те, як він зберігається, отримує доступ і будується – є критично важливим.

На початку розвитку продукту питання керування вихідним кодом, зазвичай, вирішується просто — роблять моноліт архітектуру[1], що означає, що й увесь код буде в одному репозиторії. Далі продукт починає рости та розвиватися, а разом з ним — і команда розробки, що з часом перетворюється на повноцінний R&D[2], де за окремі частини продукту відповідають вже різні команди.

Цілком логічно, що продукт починає розділятися на окремі сервіси. І ось тут постає питання — продовжувати обслуговувати код у вигляді монорепозиторію, чи розділяти його за сервісами на незалежні репозиторії зі своїм циклом розробки/випуску/підтримки?

Оскільки розуміння підходів для організації коду, підбір конкретного способу для певного проєкту, може зберегти величезну кількість часу в майбутньому.

Підходи до організації коду

1. Монорепо, скорочення від «монолітного репозиторію», — це підхід до організації коду, коли весь код проекту зберігається в одному репозиторії [3]. Включаючи всі модулі, бібліотеки та програми проєкту.

Основна перевага монорепо полягає в тому, що воно спрощує спільний доступ до коду та його повторне використання. Розробники можуть легко обмінюватися кодом між модулями, а зміни, зроблені в одному модулі, можна швидко поширювати на інші модулі, які від нього залежать. Це може призвести до швидшого часу розробки.

Ще одна перевага монорепо полягає в тому, що це може спростити застосування стандартів кодування та найкращих практик. З усім кодом в одному місці легше контролювати та забезпечувати узгодженість у всьому проекті.

Однак монорепо також може мати деякі недоліки. По-перше, вони можуть стати дуже великими, і ними важко керувати в міру зростання проєкту. Крім того, залежності між модулями можуть стати складними, і їх важко розплутати. Конфлікти з принципом єдиної відповідальності, якщо з

монорепозиторієм працює багато команд. Код може бути неймовірно базовим і настільки спільним, що виявиться, що ніхто конкретний за нього не відповідає, а зміни вносять всі.

Мердж-конфлікти за наявності багатьох команд дуже ймовірні. Саме відкриття IDE зі збільшенням кодової бази стає окремим викликом. Час локального проходження модульних тестів значно збільшується.

2. Мультирепо — це підхід, коли кожен модуль або програма зберігається у власному репозиторії. Це можна розглядати як традиційний підхід до організації коду[4]. Головна перевага мультирепо полягає в тому, що це полегшує керування залежностями між модулями. Кожен модуль може мати власний набір залежностей, що полегшує відстеження та керування залежностями в проекті в цілому.

Мультирепо також може спростити керування змінами в проекті. Кожен модуль може мати власний цикл розробки та випуску, що дозволяє командам вносити зміни та розгортати оновлення незалежно одна від одної.

Однак, це також може ускладнити обмін кодом між модулями. Розробникам може знадобитися вручну копіювати код між сховищами, що може призвести до дублювання та неузгодженості. Крім того, може бути складніше забезпечити дотримання стандартів кодування та найкращих практик у проекті в цілому.

Висновки

В результаті, можна зробити висновок, що перш ніж прийняти рішення, важливо ретельно зважити плюси і мінуси кожного підходу. І монорепо, і мультирепо мають свої переваги та недоліки, і вибір між ними залежатиме від конкретних потреб і вимог проєкту.

Якщо проєкт відносно невеликий і простий, то гарним вибором може стати створення монорепо. Це може спростити обмін кодом і покращити узгодженість у проекті. Однак, якщо проєкт великий і складний, мультирепо може краще підійти. Це може спростити керування залежностями та змінами в проекті в цілому.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Microservices vs. monolithic architecture – [Електронний ресурс]. – Режим доступу до матеріалу: https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith
2. Що таке — R&D-центр в Україні – [Електронний ресурс]. – Режим доступу до матеріалу: https://dou.ua/forums/topic/36259/
3. Mono vs Multi Repos - Pick Your Poison – [Електронний ресурс]. – Режим доступу до матеріалу: https://www.raftt.io/post/development-challenges-of-working-with-monorepos-and-multirepos
4. Мульти- та монорепозиторії в SOA. Організація коду – [Електронний ресурс]. – Режим доступу до матеріалу: https://dou.ua/forums/topic/34363/

Верещагіна Олександра Валеріївна – студентка групи 3ПІ-19Б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: alexxa.vereshchagina@gmail.com

Науковий керівник – ***Бабюк Наталія Петрівна***, кандидат технічних наук, доцент кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: babiuk@vntu.edu.ua

Vereshchahina Oleksandra Valeriyvna — Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: alexxa.vereshchagina@gmail.com

Supervisor – ***Babuk Natalia Petrivna***, PhD, Associate Professor of Software Engineering Department, Vinnytsia National Technical University, Vinnytsia, e-mail: babiuk@vntu.edu.ua