УДК 004.4

Д. Ю. Станішевський
О. В. Войцеховська

Аналіз розвитку підходів до розробки веб-додатків та зростання впливу розподіленої архітектури

Вінницький національний технічний університет

Анотація

У роботі проведений аналіз розвитку тенденцій у сфері розробки веб-сайтів та веб-додатків. Розглянуті сучасні технології для розробки клієнтської частини, методології та принципи для розробки серверної частини, нові типи баз даних для застосування у спеціальних випадках. Також проведений аналіз зростання важливості ролі архітектора на проекті та етапу розробки архітектури в цілому.

Ключові слова: мікросервісна архітектура, веб-додаток, розгортання, CI/CD, хостинг, розподілена система, обернені зв'язки, бази даних.

Abstract

The paper analyzes the development of trends in the field of website and web applications development. Modern technologies for client part development, methodologies and principles for server part development, new types of databases for use in special cases are considered. An analyzed of the growing importance of the role of the architect in the project and the stage of development of architecture in general.

Keywords: microservice architecture, web application, deploy, CI/CD, hosting, distributed system, dependency inversion, database.

Вступ

Ще десять років назад в сфері розробки серверних додатків панувала одна прийнятна архітектура. Існувало три компоненти: база даних, сервер та клієнтська частина.

База даних в панівній кількості проектів використовувалась SQL: MSSQL для проектів що розгортаються на операційних системах Windows Server, PostgreSQL для великих і навантажених проектів, Oracle для фінансового сектору, та MySQL в усіх інших випадках. Тобто існувала універсальна схема вибору бази даних для проекту і окремого процесу вибору та розробки не було. Вся варіативність в цьому питанні збігалась до вибору – де саме буде розгортатись ця база, на одному сервері чи на окремому відносно сервера.

Щодо сервера варіативність зберігалась на рівні вибору мови: існували мови для розробки сайтів (PHP, NodeJS) та для корпоративних систем (Java та C#). При цьому зазвичай вибір мови обмежувався наявністю відповідних спеціалістів або швидкості розробки. Технології API застосовувались не часто, а їхньою сферою застосування були мобільні додатки да програми для персонального комп'ютера. Основним способом взаємодії між фронтендом та бекендом були так звані «шаблонізатори» – технології, що дозволяли фактично будувати веб-сторінки на стороні сервера та повертати їх у готовому вигляді. Це створювало жорсткі зв'язки між клієнтською та серверною частиною веб-додатка. Також це вимагало постійної взаємодії між розробниками різного стеку. Часто розробка взагалі виконувалась одним розробником. Як хостинг застосовувались або фізичні сервери або орендовані вітруальні машини на хостинг сервісах.

Клієнтська частина також мала типовий набір технологій, а саме JQuery для найпопулярніших задач та Bootstrap для адаптивної верстки.

Але за останнє десятиріччя в сфері веб-розробки відбулись докорінні зміни, які покладенні побороти проблеми розробки.

Результати досліджень

На сьогодні набрала популярності мікросервісна архітектура, де замість одного сервера, створюється лінійка спеціалізованих під свої задачі. Це дозволяє краще масштабувати їх та балансувати, дозволяє створювати інкапсуляцію ресурсів та обов'язків, а також полегшує розробку та розуміння[1]. Іноді практикується розробка різних сервісів на різних мовах програмування. Також важливою зміною стала часткова або повна відмова від використання шаблонізаторів. Таким

чином серверна частина розробляється з мінімальною прив'язкою до того, що він буде використовуватись саме веб-сторінкою (так звані оберненні залежності). Замість цього застосовується технологія API – обмін між сервером та клієнтом не сторінками, а лише даними [2].

Зміни відбулись і в сфері баз даних. SQL мав недоліки для багатьох проектів: такі бази є дуже важкими та надлишковими для простих проектів, мають погану придатність до збереження файлів, а жорстка структура таблиць не підходить для даних з непостійною структурою. Тому набрали популярності інші типи баз даних, наприклад документні (MongoDB, ArangoDB, RavenDB, DynamoDB), графові (ArangoDB), ключ-значення (Redis) та файлові (S3, GoogleStorage) [3].

В фронтенд частині також відбулись суттєві зміни. З'явились нові технології для розробки реактивних веб-додатків, які можуть проводити обмін даними з сервером та змінювати власну структуру без перезавантаження сторінки. До найпопулярніших таких технологій можна віднести Angular, React, Vue.

Також з новинок технологій – препроцесори стилів, що дозволяють привносити елементи програмування в побудову CSS.

На другий план відійшли бувші лідери розробки клієнтської частини JQuery та Bootstrap. Перший став не потрібний через власну неоптимізованість та появу більш сучасних бібліотек і зараз використовується лише на старих проектах. Для другого ж ситуація не настільки погана – він досі використовується, але все частіше для прототипування сторінок а не розробки, оскільки його функціональність тепер є в самому HTML та CSS у вигляді Grid та FlexBox.

Найдокоріниші зміни відбулись саме в сфері розгортання та хостингу. Розгорнути сучасний складний проект в ручному режимі є надважкою задачею на декілька днів. Це стало великою проблемою оскільки таке розгортання в процесі розробки необхідне майже щодня. Тому з'явились технології для автоматичного та напівавтоматичного розгортання проектів, та ціла професія яка відповідає за розробку та впровадження таких технологій – DevOps, а подібний процес, відповідно, CI/CD.

Зміни зачепили і хостинги для серверів. З'явились хмарні технології, які беруть значну частину задач по обслуговуванні сервера на себе, а також автоматизацію багатьох інших процесів, таких як логінг та моніторинг.

Отже в результаті таких змін ситуація, щодо розробки нових проектів стала не такою очевидною та однозначною. В наслідок цього зросла роль проектувальників інформаційних систем. Таких людей називають архітекторами рішень та архітекторами систем, залежно від рівня на якому відбувається проектування.

Процес проектування починається задовго до початку розробки та зазвичай навіть до формування команди, оскільки потреба в кількості та якості спеціалістів стане зрозумілою вже коли буде попередня схема додатку. Тому проектування архітектури не відірване від управління командою, і архітектор повинен мати навички менеджменту персоналом.

Гарна архітектура повинна враховувати велику кількість вимог до майбутньої інформаційної системи. В першу чергу вона повинна чудово справлятись з поставленими завданнями та мати мінімальну кількість «технічних обмежень». Система повинна бути готовою до зміни або розширення вимог, тобто вона повинна відповідати параметрам «розширюваності». Також, залежно від специфіки проекту, повинні бути прийняті до уваги безпека, відмовостійкість, стійкість до навантаження та гнучкість. І, як не дивно, – дружність до розробників, оскільки зазвичай чим складніша інформаційна система, тим з більшою вірогідністю вона буде розроблятись протягом тривалого часу та розвиватись. А отже, будуть з'являтись нові працівники, яким треба буде знайомитись з вже написаною кодовою базою.

Якщо при розробці архітектури будуть допущенні критичні проблеми, це може призвести до значних наслідків та фінансових, часових і репутаційних втрат для власників продукту. Зазвичай помилки на рівні архітектури вартують фінансових втрат на декілька порядків вище ніж помилки при розробці[1].

Висновки

Отже, оскільки тренди в розробці веб-додатків сильно ускладнились за останнє десятиріччя, то відповідно зросла роль етапу проектування та планування системи, та призвело до популяризації професії архітектора програмного забезпечення.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Р. Мартін, Чиста архітектура. Харків, Україна : Фабула, 2021, 368 с. ISBN: 978-617-09-5286-8.
2. REST documentation [Електроний ресурс] – Режим доступу до ресурсу: https://docs.github.com/en/rest.
3. Most popular database management system [Електроний ресурс] – Режим доступу до ресурсу: https://www.stackscale.com/blog/popular-database-management-systems/.

Станішевський Дмитро Юрійович – студент групи 2КІ-18б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: stanishevskiy.dmitro@gmail.com

Войцеховська Олена Валеріївна – кандидат технічних наук, доцент кафедри обчислювальної техніки, Вінницький національний технічний університет, Вінниця

Stanishevskiy Dmytro Y. – students, 2KI-18b, Faculity of information Technologies and Computer Engeneering, Vinnytsa National Technical University, email: stanishevskiy.dmitro@gmail.com

Voytsekhovska Olena V. — PhD, Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical 0University