

УДК

## РАЗВЕРТЫВАНИЕ ПРИЛОЖЕНИЙ С ПОМОЩЬЮ ТЕХНОЛОГИЙ БЕССЕРВЕРНОЙ АРХИТЕКТУРЫ С ПРИМЕНЕНИЕМ ОБЛАЧНОЙ РАЗРАБОТКИ

*Лаппо К.С., студент группы 153503, Владимирцев В. Д., ассистент каф. Информатики*

*Белорусский государственный университет информатики и радиоэлектроники*

*г. Минск, Республика Беларусь*

*Владимирцев В.Д. – ассистент кафедры информатики*

**Аннотация.** Бессерверная архитектура, особенно в области облачной разработки, набрала значительную популярность в последние годы. В данной работе мы предоставим обзор бессерверной архитектуры и ее последствий для облачной разработки. Также мы обсудим преимущества и недостатки бессерверного вычисления и предоставим примеры его использования в облачной разработке.

**Ключевые слова.** Бессерверная архитектура, облачная разработка, облачное вычисление, функции, масштабируемость, гибкость, экономичность, AWS Lambda, Microsoft Azure Functions, Google Cloud Functions.

Традиционная модель облачной разработки включает развертывание приложений на виртуальных машинах или контейнерах. Однако, этот подход имеет несколько ограничений, включая необходимость ручного масштабирования и обслуживания, что может привести к более высоким затратам и меньшей гибкости. Бессерверная архитектура [1] предлагает альтернативный подход к облачной разработке, позволяя разработчикам сосредоточиться на написании кода, а не на управлении инфраструктурой.

Бессерверная архитектура относится к модели облачного вычисления, где провайдер облака управляет инфраструктурой и автоматически выделяет ресурсы по мере необходимости. В этой модели разработчики пишут код в виде функций, которые запускаются при событиях, таких как HTTP-запросы или обновления базы данных. Провайдер облака автоматически масштабирует инфраструктуру для обработки нагрузки, и разработчики платят только за ресурсы, используемые во время выполнения функции.

Бессерверное вычисление [2] предлагает несколько преимуществ по сравнению с традиционной облачной разработкой. Они включают в себя снижение затрат, увеличение масштабируемости и улучшение гибкости. Поскольку разработчики платят только за ресурсы, используемые во время выполнения функции, они могут значительно снизить затраты по сравнению с традиционной облачной разработкой. Автоматическое масштабирование ресурсов также позволяет приложениям обрабатывать большой объем трафика без ручного вмешательства [3]. Кроме того, бессерверное вычисление улучшает гибкость, сокращая время, необходимое для управления инфраструктурой и позволяя разработчикам сосредоточиться на написании кода и доставке ценности пользователям.

Несмотря на преимущества, бессерверное вычисление также имеет некоторые недостатки. Они включают ограниченный контроль над инфраструктурой, увеличенную сложность проектирования приложений и время инициализации. Поскольку провайдер облака управляет инфраструктурой, разработчики имеют ограниченный контроль над аппаратным и программным обеспечением. Это может привести к ограничениям в части настройки и конфигурации. Кроме того, сложность бессерверных приложений может сделать их более трудными в проектировании и тестировании.

Несколько провайдеров облачных услуг предлагают платформы бессерверного вычисления, включая Amazon Web Services (AWS) Lambda [4], Microsoft Azure Functions [5] и Google Cloud Functions [6]. Эти платформы позволяют разработчикам писать функции на различных языках программирования, включая Node.js, Python и Java. Эти функции могут использоваться для различных случаев использования облачной разработки, включая веб-приложения, обработку данных и приложения интернета вещей (IoT).

Разработанный проект RaXiom является веб-приложением для определения раковых опухолей на основе МРТ снимков пациентов, разработанным с использованием бессерверной архитектуры и облачных вычислений. Мы использовали Google Cloud Platform (GCP) и Google App Engine (GAE) [7] для развертывания нашего приложения, что позволяет без затрат на настройку инфраструктуры получить доступ к проекту в сети. В основе бессерверного вычисления находится Firebase с его сервисами Firebase Firestore, NoSQL база данных с полной интеграцией с GCP, и Firebase Functions для настройки обработчиков событий.

Структура Raxiom была настроена так, чтобы минимизировать затраты на инфраструктуру, так как использование облачных сервисов берет плату за отдельные сервисы и их использование, а также максимальную гибкость и безопасность. Так же стоит учесть, что провайдеры облачных услуг часто используют систему скидок и других привилегий, которые влияют на стоимость сервисов. Таким образом общие затраты Raxiom за последний месяц представлены на рисунке 1.

Service	Cost	Discounts	Promotions and others	Subtotal	
Compute Engine	\$0.01	\$0.00	-\$0.01	\$0.00	
Cloud Healthcare	\$0.00	\$0.00	\$0.00	\$0.00	
Cloud Build	\$0.00	\$0.00	\$0.00	\$0.00	
Cloud Storage	\$0.52	\$0.00	-\$0.52	\$0.00	
Vertex AI	\$0.00	\$0.00	\$0.00	\$0.00	
Networking	\$0.56	-\$0.56	\$0.00	\$0.00	
App Engine	\$11.72	\$0.00	-\$11.72	\$0.00	
Cloud Logging	\$0.00	\$0.00	\$0.00	\$0.00	
				Subtotal	\$0.00
				Tax	—
				Total	\$0.00

Рисунок 1 – Общие затраты на инфраструктуру и поддержку Raxiom на Апрель 2023 г.

Для организации обмена данными использовался Apollo GraphQL, который позволяет упростить взаимодействие между клиентом и сервером, подробная иллюстрация технологий для обмена данных указана ниже. Для Frontend части приложения был использован NextJS который предоставляет возможность серверного рендеринга. На рисунке 2 представлена статистика среднего времени выполнения запросов к облачным сервисам и их производительность.

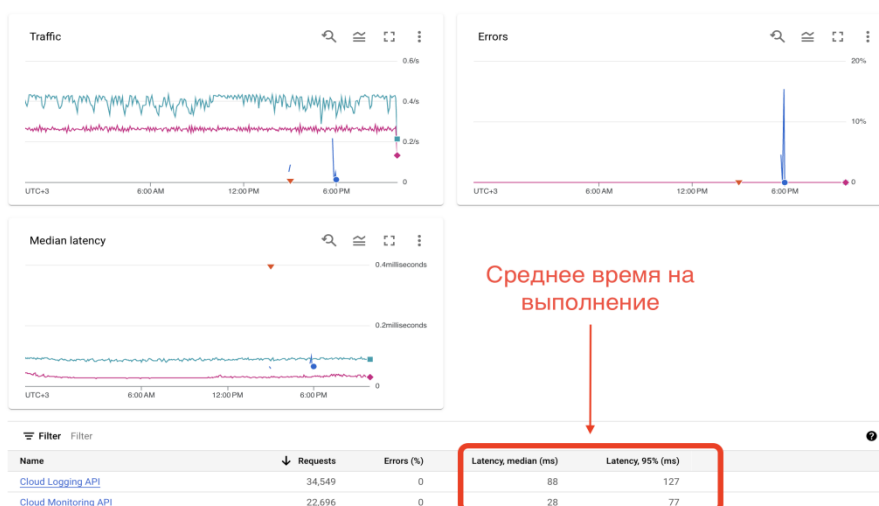


Рисунок 2 – Общие затраты на инфраструктуру и поддержку Raxiom на Апрель 2023 г.

В результате проект показал, что использование бессерверной архитектуры и облачных вычислений может значительно упростить создание и развертывание веб-приложений. Мы смогли создать безопасное и масштабируемое приложение, которое может обрабатывать большой объем трафика без необходимости ручного масштабирования.

**Заключение.** Как итог бессерверное вычисление перевернуло облачную разработку, предоставляя разработчикам более эффективный и экономичный подход к созданию приложений. Несмотря на некоторые ограничения бессерверного вычисления, его преимущества делают его привлекательным вариантом для многих случаев использования облачной разработки. С продолжением роста облачных вычислений, вероятно, что бессерверное вычисление станет еще более распространенным в следующие годы.

#### Список использованных источников:

1. Roberts, J. (2019). *Serverless Architecture: A Comprehensive Guide*. O'Reilly Media, Inc.
2. Kalyanaraman, V. (2020). *Cloud Computing: A Hands-On Approach*. CRC Press.
3. Appleby, S. (2018). *Enterprise Serverless: Designing Applications for the Cloud*. O'Reilly Media, Inc.
4. Amazon Web Services. (2021). *AWS Lambda documentation* – [Электронный ресурс] Электронные данные. – Режим доступа: <https://aws.amazon.com/lambda/>
5. Microsoft Azure. (2021). *Azure Functions. documentation* – [Электронный ресурс] Электронные данные. – Режим

доступна: <https://azure.microsoft.com/en-us/services/functions/>

6. Google Cloud. (2021). *Cloud Functions. documentation* – [Электронный ресурс] Электронные данные. – Режим

доступа: <https://cloud.google.com/functions>

7. Google App Engine. (2021). *documentation* – [Электронный ресурс] Электронные данные. – Режим доступа:

<https://cloud.google.com/appengine>

UDC

## APPLICATION DEPLOYMENT USING SERVERLESS ARCHITECTURE TECHNOLOGIES USING CLOUD DEVELOPMENT

Lappo.K.S.

*Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus*

*Vladymtsev V.D. – Assistant of the Department of Informatics*

**Annotation.** Serverless architecture, especially in the field of cloud development, has gained significant popularity in recent years. In this paper, we will provide an overview of the serverless architecture and its implications for cloud development. We will also discuss the advantages and disadvantages of serverless computing and provide examples of its use in cloud development.

**Keywords.** Serverless, cloud development, cloud computing, functions, scalability, flexibility, cost-effectiveness, AWS Lambda, Microsoft Azure Functions, Google Cloud Functions.