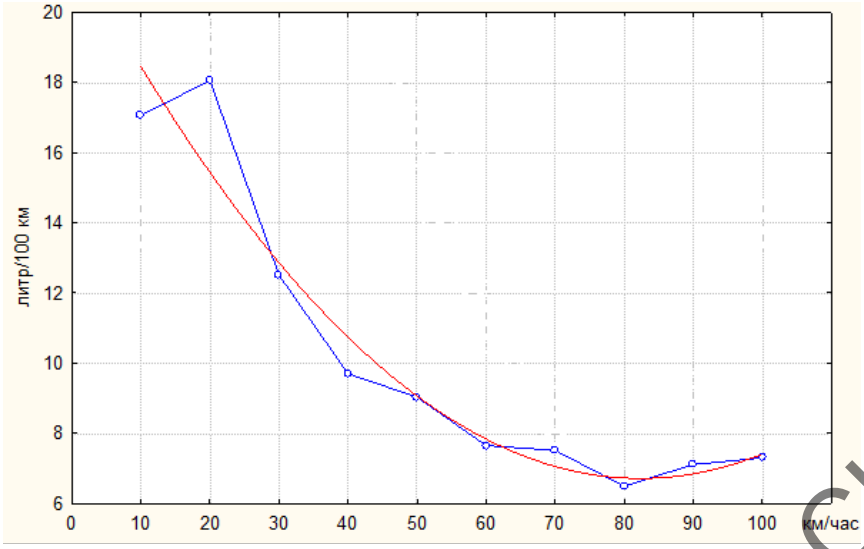

Рисунок 2 – Зависимость расхода топлива от скорости автомобиля

Литература

1. Электрическое и электронное оборудование автомобилей. - М.: Рэн-дал; СПб.: Алфамер Паблишинг, 2008. - 284 с.

2. Электронные системы автомобиля: сайт. - URL: http://awtoel.narod.ru/index.html.

3. Ютт В.Е. Электрооборудование автомобилей: учеб. для студентов вузов. - 2-е изд., перераб. и доп. - М.: Транспорт, 2005. - 304 с.

А. Е. Костерев (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **М. А. Подалов**, ст. преподаватель

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ С ПРИМЕНЕНИЕМ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ

В рамках работы было разработано мобильное игровое приложение для системы Android с API на базе микросервисной архитектуры.

Актуальность работы – подавляющее большинство мобильных приложений до сих пор использует монолитные архитектуры своих API, в данной статье представлено аргументированное альтернативное мнение по этому вопросу и доводы в пользу перехода на микросервисную архитектуру.

Для создания прототипа мобильного игрового приложения была выбрана платформа Unity так как она является лучшим инструментом для разработки игровых мобильных решений на 2020 год. Для разработки сервисов были использованы: Node JS, mongodb, graphql и php.

Каждое современное мобильное приложение использует какой-либо сервис или API для типовых задач, таких как сохранение и обработка данных пользователя и другие.

Чаще всего данные сервисы представляют собой большую монолитную систему в виду простоты разработки на начальном этапе.

Серьёзным недостатком таких систем является их незащищённость по отношению ко многим вопросам, например, в случае возникновения исключительной ситуации в сервисе который использует монолитную архитектуру, сервис теряет доступность и свою работоспособность, как следствие приложение которое использует этот сервис тоже выходит из строя, а отладка сервиса и выявление проблемы займёт больше средств и времени так как он представляет собой одну большую монолитную систему что затруднит его диагностику.

Для повышения надёжности API разработанного мобильного игрового приложения была использована микросервисная архитектура. Данная микросервисная система представляет собой набор сайтов, которые предоставляют интерфейс для обработки бизнес логики приложения. Микросервисы являются слабосвязными системами и их работоспособность не завязана друг от друга, как следствие если произойдёт авария на одном микросервисе приложение всё равно продолжит свою работу. Разбивка монолитного решения на микросервисы позволяет создавать системы которые быстро изменяются и диагностируются в случае проблем, а также легко модифицируются в случае необходимости. В разработанной системе компоненты или же микросервисы выполняют относительно простые функции и взаимодействуют с использованием экономичных сетевых коммуникационных протоколов в стиле REST с использованием JSON.

За счёт повышения гранулярности модулей архитектура нацелена на уменьшение степени зацепления и увеличение связности, что позволяет проще добавлять и изменять функции в системе в любое время. В разработанной системе модули можно легко заменить в любое время: акцент сделан на простоту, независимость развёртывания и обновления каждого из микросервисов, модули организованы вокруг функций: микросервис по возможности выполняет только одну достаточно элементарную функцию.

Разработанное приложение представляет собой игру раскраску по цифрам в стиле pixel-art. Скриншоты приложения представлены на рисунке 1.

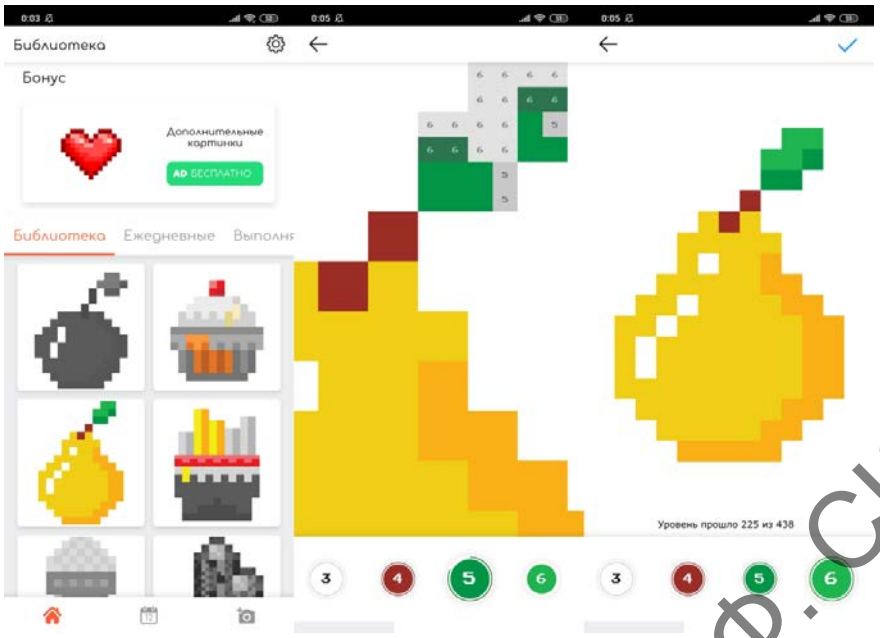

Рисунок 1 – Скриншоты приложения

Для работы приложения были подняты следующие микросервисы, микросервис для сохранения и раздачи статических файлов написан на php и размещён на хостинге 000webhost. Кластер с базой данных на mongodb, размещен на Amazon. Промежуточный сервис для сохранения файлов по определённым правилам безопасности разработан на NodeJS, размещён на heroku. Сервис с бизнес логикой разработан на NodeJS и GraphQL, размещён на heroku и выполняет роль gateway. Сервисы на рисунке 2.

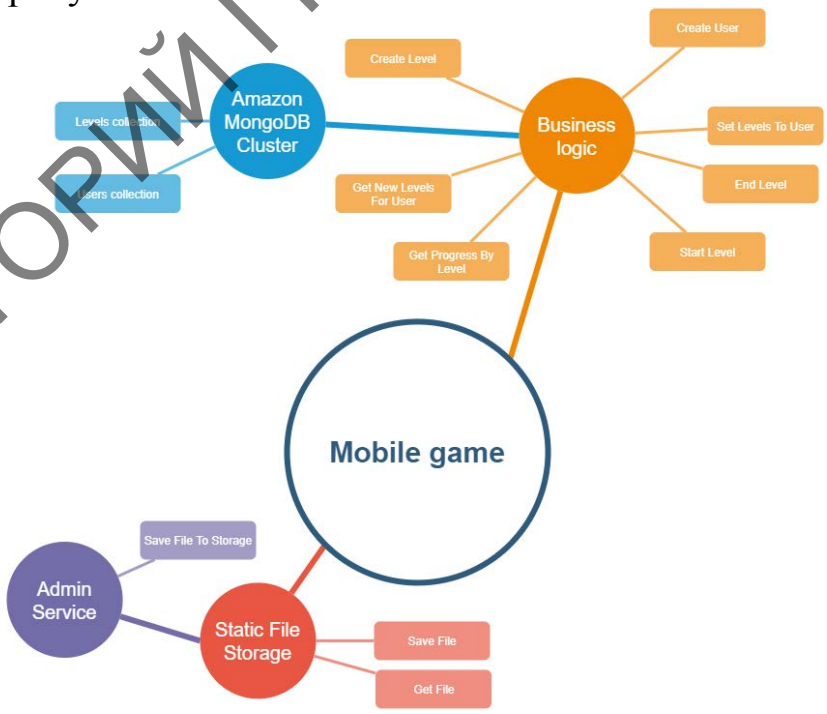

Рисунок 2 – Сервисы

Как вывод данная система является лучшим решением в сравнение с монолитной, имеет ряд преимуществ, пользователи приложения всегда смогут получить положительный опыт использования что является серьёзным аргументом в пользу микросервисов.

Литература

1. Микросервисная архитектура / Wikipedia // [Электронный ресурс]. – 2010. – URL: https:// ru.wikipedia.org/wiki/Микросервисная_архитектура – Дата доступа: 09.03.2020.

2. MongoDB / Wikipedia // [Электронный ресурс]. – 2018. – URL: en.wikipedia.org/wiki/MongoDB – Дата доступа: 8.03.2020.

3. Кластер / Wikipedia // [Электронный ресурс]. – 2011. – URL: https:// ru.wikipedia.org/wiki/Кластер – Дата доступа: 10.03.2020.

4. API / Wikipedia // [Электронный ресурс]. – 2014. – URL: https:// en.wikipedia.org/wiki/api –Дата доступа: 10.03.2020.

А. С. Крылов (ГГТУ имени П.О. Сухого, Гомель)
Науч. рук. **В. И. Токочаков**, канд. техн. наук, доцент

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ЭЛЕКТРИЧЕСКИХ ПОТЕРЬ В МНОГОАБОНЕНТСКИХ СЕТЯХ НАПРЯЖЕНИЕМ 0,4 кВ

Потери электроэнергии в электрических сетях − является важным показателем экономичности их работы, также является очевидным указателем состояния системы учета электричества, производительности взаимодействия работы различных учреждений. Данный указатель все отчетливей говорит о накапливающихся проблемах, которые требуют оперативных решений в формировании, реконструкции и техническом переоснащении электрических сетей, кроме того в модернизации средств и способов их применения и управления, в увеличении точности учета электричества, улучшении эффективности сбора валютных средств за поставленную потребителям электроэнергию и т.п.

Из упомянутого выше следует, собственно, что проблема снижения утрат электричества в электрических сетях не утратила собственной актуальности, а стала одной из ключевых задач обеспечения экономической прочности энергоснабжающих учреждений.