**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ РАДИОФИЗИКИ И ЦИФРОВЫХ МЕДИАТЕХНОЛОГИЙ**

**Кафедра радиофизики и цифровых медиатехнологий**

ЧЁРНЫЙ

Алексей Витальевич

**Разработка и сравнительный анализ ВЕБ-Серверов микросевисной и монолитной архитектуры на базе платформы JAVA**

Курсовая работа

Научный руководитель:

доцент

И. Э. Хейдоров

Минск, 2017

Оглавление

1. Введение.
2. Архитектуры веб-приложений.
   1. Монолитная
   2. Микросервисная
   3. Сравнение
3. Реализация
   1. Монолитной архитектуры
   2. Микросервисной архитктуры
4. Сравнительный анализ
   1. Анализ мнолитной
   2. Анализ микросервисной
   3. Сравнение
5. Вывод

Введение

Целью данной курсовой работы является сравнение двух архитектур веб-серверов, монолитной и микросервисной, для небольшого приложения.

Задачи, лежащие перед данной курсовой работой заключаются в следующем:

1. Изучить и предоставить сведения по соответствующим архитектурам.
2. Разработать сервера-испытуемые на основе предложенных архитектур.
3. Провести сравнительный анализ испытуемых по следующим показателям: быстродействие и затрачиваемы мощности.
4. Сделать вывод о выгодности архитектур для небольших серверов.

С каждым годом использование глобальной паутины только растёт. Всё больше и больше процессов переходят в интернет. В связи с чем растёт сложность вычислений на стороне серверов и повышается важность их стабильности.

Всё большему числу компаний становится выгодно переносить вычисления на сторону сервера, где множество клиентов с веб-интерфейса могут получить доступ к продукту, вместо установки своего ПО каждому клиенту в зависимости от его платформы.

Для обеспечения стабильности сервера и удовлетворения нагрузок используются различные архитектуры веб-приложений. В ходе данной курсовой работы будет рассмотрено 2 из них.

//главав про архитектуры

Сами же сервера изначально строились только по принципу монолитности – всё приложение находилось на одном сервере, а в случае недостатка мощностей запускалась ещё одна копия сервера, в связи с чем общая пропускная способность в идеале повышалась в 2 раза. Но такой способ ведёт к переизбыткам ресурсов, в виду того, что если перенагружено только конкретное место сервера, всё ещё поднимается его копия.

Так как данный способ не решал проблему недостатка мощностей в конкретном узле приложения, была придумана микросервисная архитектура, которая решала проблему путём разбиения единого приложения на множество приложений, каждое из которых запускалось на своём сервере.

//