Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра электронных вычислительных машин

Факультет компьютерных систем и сетей

**Отчет по лабораторной работе №3**

Тема: «Исследование архитектурного решения»

Выполнили:

студенты группы 790541 Станишевская Е.Ю.

Новик А.H.

Жидаль Е.И.

Свищук П.В.

Мисан Д.А.

Проверил:

к.т.н., доцент Перцев Д.Ю.

Минск

2020

# **Архитектура To be**

Тип приложения: веб-приложение.

Используемые языки: java, js, css, html

Используемый фреймворк: spring boot.

Тип архитектуры: клиент-сервер.

Причина выбора: всем пользователям нужен доступ к одним и тем же данным с разных устройств.

Стратегия развёртывания: нужно распределённое развёртывание, так как фронтэнд-сервер, база данных и бэкэнд-серверы будут находиться на разных устройствах (серверах).

Выбор технологии: библиотеки языка java позволяют работать проводить быстрые вычисления, которые нужны для выполнения запросов пользователей. Языки js, css, html дают возможность написать шаблоны веб-страниц (возможность реализации пользовательского интерфейса).

## **Показатели качества**

### **1 Требования к доступности**

***Объяснение***Приложение должно стремиться к минимальному времени простоя.  
***Обоснование***Чем больше время простоя, тем меньше поисковых запросов можно показать. Чем меньше запросов будет выполнено - тем меньше заказов будет сформировано, тем, в свою очередь, меньше прибыль.  
***Возможный способ измерения***Сэмулировать реальные нагрузки на этапе тестирования и дать группе QA инженеров возможность пользоваться приложением несколько дней.

### **2 Требования к масштабируемости**

***Объяснение***Приложение может горизонтально масштабироваться.  
***Обоснование***Приложение вероятно столкнётся с большим количеством пользователей. Возможность масштабироваться является необходимым условием, чтобы предоставить обслуживание максимальному числу пользователей, что даст максимальную прибыль.  
***Возможный способ измерения*** Увеличить количество серверов, обслуживающих приложение, провести тесты производительности

### **3 Требования к удобству использования**

3.1 ***Объяснение***Количество элементов графического интерфейса должно стремиться к минимуму  
***Обоснование***Чем меньше элементов интерфейса, тем меньше усилий нужно приложить пользователю, чтобы выполнить нужную ему функцию, тем удобнее будет пользоваться приложением. С другой стороны, если не следовать данному требованию, то приложение будет проигрывать конкуренцию более удобным аналогам.  
***Возможный способ измерения*** Поставить задачу QA инженерам найти лишние элементы интерфейса.  
3.2 ***Объяснение***Количество действий для выполнения функции должно стремиться к минимуму

***Обоснование***Аналогично предыдущему требованию.  
***Возможный способ измерения***Поставить задачу QA инженерам посчитать количество действий, необходимых для выполнения функций, чтобы найти далее те места, где можно и стоит уменьшить это количество.  
3.3 ***Объяснение*** Время реакции для всех необходимых операций должно стремиться к минимуму  
***Обоснование***Аналогично предыдущему требованию.  
***Возможный способ измерения***Поставить задачу QA инженерам посчитать среднее время отклика при выполнении операций.  
3.4 ***Объяснение***Приложение должно уведомлять пользователя об успешной аутентификации.  
***Обоснование***Иначе пользователь может оказаться в ситуации, когда не понятно, зашёл ли он в учётную запись успешно или нет. Если у пользователя несколько учётных записей, то без данного требования выяснять, в какой конкретно сейчас он сидит, будет проблематично.

***Возможный способ измерения***Поставить задачу QA инженерам проверить, после успешной аутенфикации на большинстве страниц выводится надпись "В настоящее время Вы зашли как ...".

### **4 Требования к расширяемости**

***Объяснение***Спектр продуктов, предлагаемых приложением, можно расширить. Например, добавить рекомендацию книг, манги, аниме, игр.  
***Обоснование***Чем больше разных видов продуктов приложение может рекомендовать, тем больше количество обслуживаемых пользователей, тем больше прибыль. С другой стороны, выполнение требования позволит приложению адаптироваться к рынку.  
***Возможный способ измерения***Попробовать добавить новый тип продуктов и оценить результаты.

### **5 Требования к безопасности**

5.1 ***Объяснение***Только владелец или администратор может изменять данные, связанные с заказами пользователя.  
***Обоснование***Без этого будет нарушаться конфиденциальность личных данных  
***Возможный способ измерения*** Поставить задачу сотрудникам ИБ найти способы изменить данные, связанные с заказами, которые им не принадлежат.  
5.2 ***Объяснение***Только владелец учётной записи может ей пользоваться.  
***Обоснование*** Невыполнение требования вероятно уменьшит доверие к ресурсу, будет нарушать принцип конфиденциальности.  
***Возможный способ измерения*** Аналогично предыдущему требованию

### **6 Требования к надёжности**

***Объяснение***Данные о заказах можно восстанавливать после сбоев  
***Обоснование***Необходимое условие для сервиса для заказов - наличие данных о пользователях. Нет выполнения условия - нет оформенных заказов. Нет заказов - нет дохода.  
***Возможный способ измерения*** Поставить задачу сотрудникам ИБ попробовать привести приложение в состояние отказа, поставить задачу QA инженерам проверить возможность восстановления.

## **Пути реализации сквозной функциональности**

### **1 Протоколирование**

Будет реализовано с помощью логирования. Это позволит разработчику в случае необходимости быстрее понять, где была совершена ошибка.  
Логированием будет заниматься в основном бэкэнд-сервер.

### **2 Авторизация и аутентификация**

Так как пользователи имеют доступ к одной базе данных, данные каждого пользователя нужно будет хранить вместе с его именем. Чтобы никто не смог изменить данные другого пользователя, они (данные) будут защищены паролем. То есть для входа в систему нужно указать имя и пароль, чтобы потом можно было добавлять продукты, отмечать понравившиеся из них.  
С пользователем будет контактировать фронтэнд-сервер, который будет пересылать запрос бэкэнд-сервер. Тот получит данные из базы и сравнит их с входными данными. Если всё совпало, то фронтэнд-сервер меняет внешний вид веб-страницы (добавляются кнопки), показывая, что аутентификация прошла успешно.

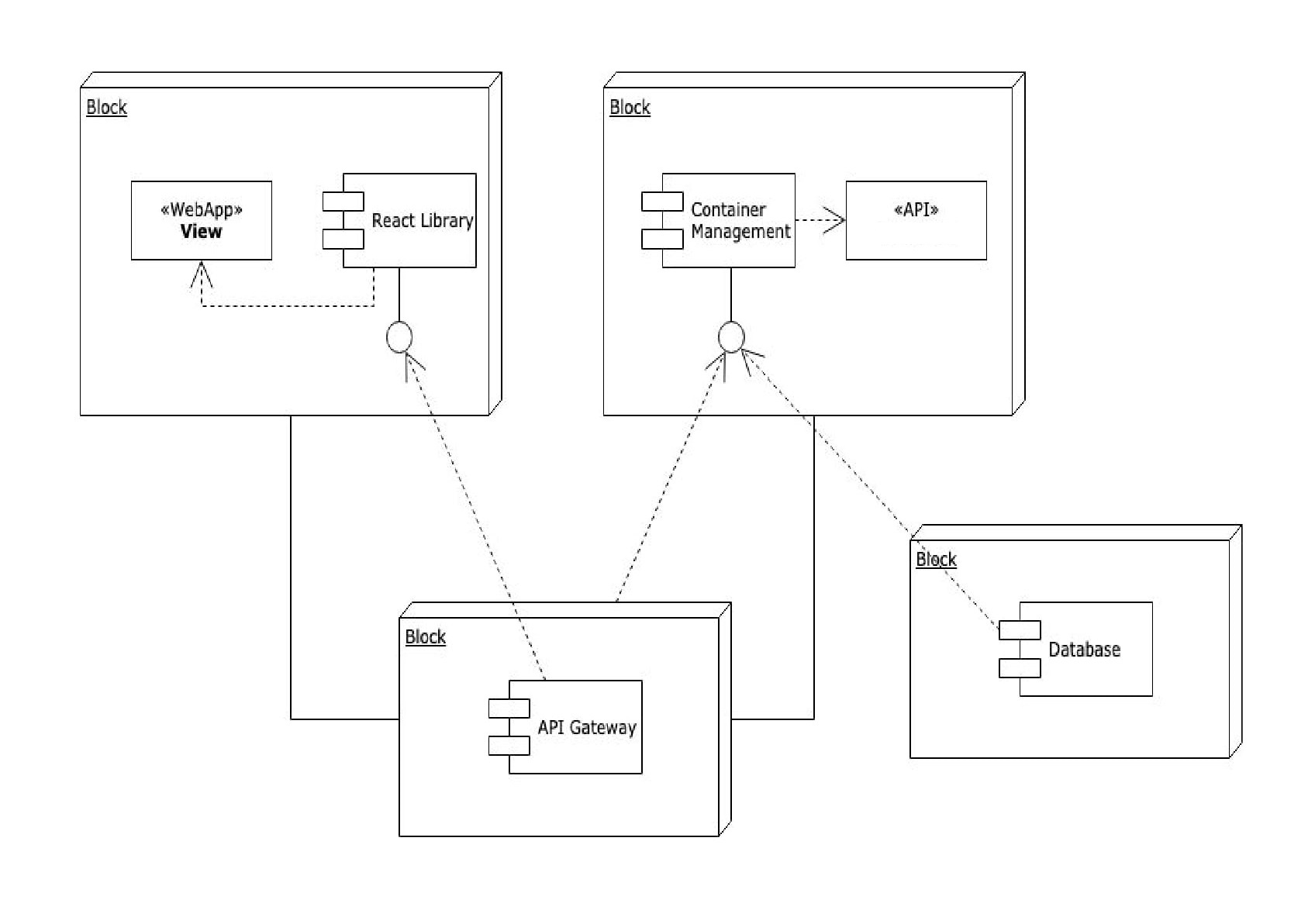
### **3 Обработка исключений**

По мере необходимости вводятся исключения для того, чтобы приложение работало, а не вылетало из-за непредвиденных ошибок.  
В основный код на python будут вставлены обработчики.

### **4 Связь с разработчиком**

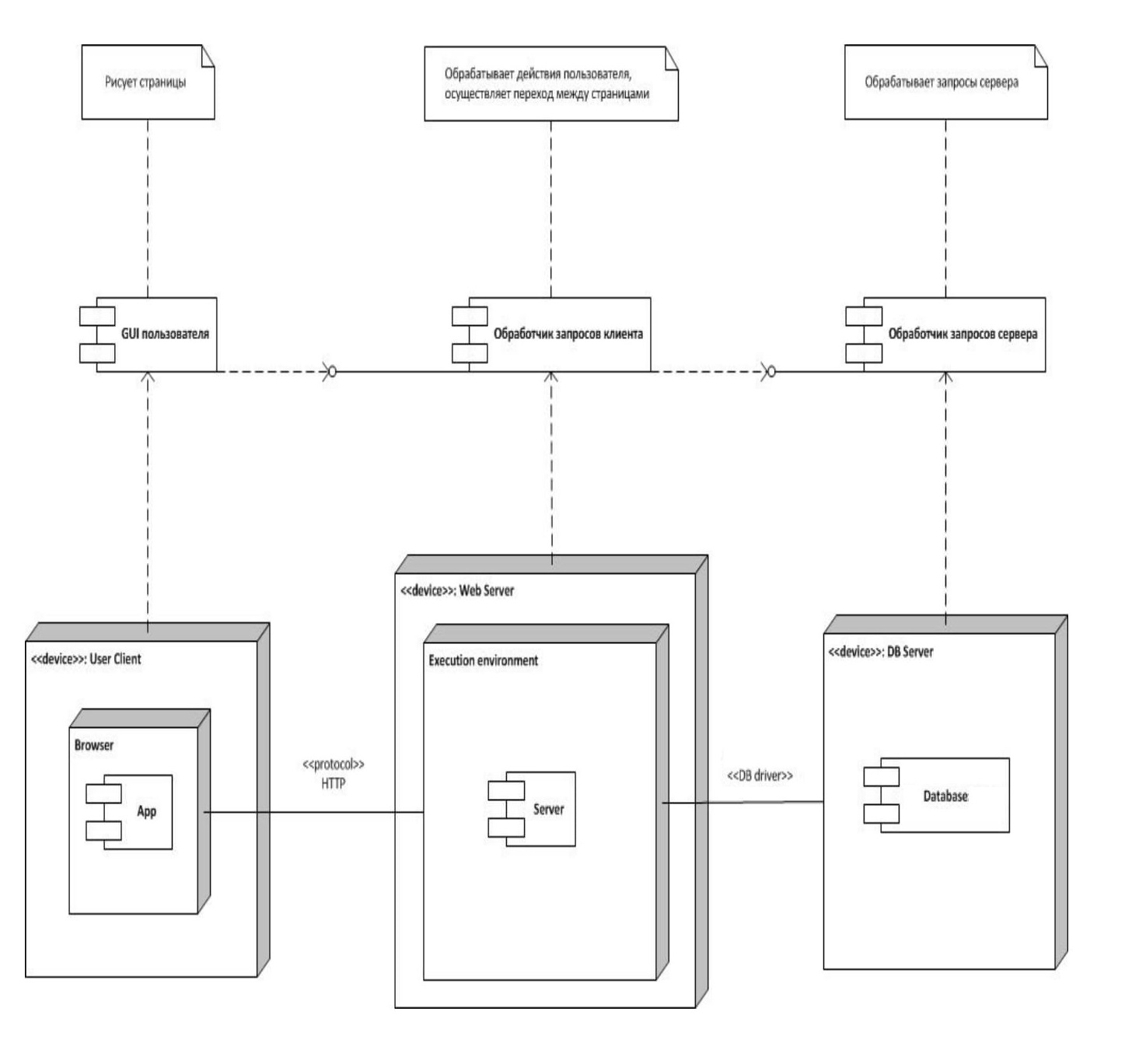
Внизу странизу будет кнопка, которая позволит свзаться с разработчиком. Предположительно, с помощью почты.

## Структурная схема приложения (д**иаграмма компонентов)**

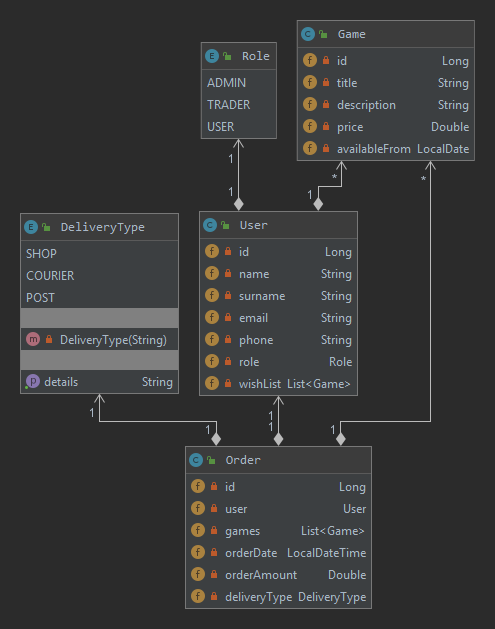


# **Архитектура As is**

**1 Диаграмма развертывания приложения**



## **2 Диаграмма классов**



# **Сравнение**

Обе архитектуры не имеют различий, так как РО реализовал проект, так как нужно.  
Есть различия в запуске. Пока что запускаются полностью все серверы и база данных на одном устройстве. Нужно реализовать запуск каждого компонента на отдельном устройстве.  
Ещё один момент, который нужно сделать - это проверить зависимости между базой данных и бэкэнд-сервером.

# **Возможные улучшения**

Для улучшения можно запустить не один фронтэнд-сервер, а несколько, чтобы можно было разделить нагрузку на каждый сервер. Это позволит ускорить время обслуживания каждого пользователя и улучшить производительность.