*Министерство образования и науки Российской Федерации*

*Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования*

*«Владимирский государственный университет*

*имени Александра Григорьевича и Николая Григорьевича Столетовых»*

*Кафедра информационных систем и программной инженерии*

***КУРСОВОЙ ПРОЕКТ***

*Прототип программной системы «GAR»*

*Выполнили: студенты гр. ПРИ-115*

*Лапыгина В. Д., Родионова А. В.,*

*Ханова А. А., Крутова А. С.*

*Принял: ст.преп. Тимофеев А.А.*

*Владимир, 2018*

Содержание

[Содержание 1](#_Toc514917105)

[ВВЕДЕНИЕ 3](#_Toc514917106)

[1 ПОСТАНОВКА ЗАДАЧИ 4](#_Toc514917107)

[1.1 Описание предметной области 4](#_Toc514917108)

[1.2 Основные понятия предметной области 4](#_Toc514917109)

[1.3 Функциональные требования к системе 4](#_Toc514917110)

[1.4 Нефункциональные требования к системе 4](#_Toc514917111)

[1.5 Сравнительный анализ аналогов 4](#_Toc514917112)

[2 АНАЛИЗ ЗАДАЧИ И ФУНКЦИОНАЛЬНАЯ ДЕКОМПОЗИЦИЯ СИСТЕМЫ 5](#_Toc514917113)

[2.1 Анализ бизнес-процесса «Название процесса» 5](#_Toc514917114)

[2.2 Сценарий взаимодействия пользователя с системой 5](#_Toc514917115)

[2.3 Аутентификация пользователя в системе 5](#_Toc514917116)

[2.4 Регистрация пользователя в системе 5](#_Toc514917117)

[2.5 Изменение регистрационных данных 5](#_Toc514917118)

[2.6 Просмотр каталога продукции 5](#_Toc514917119)

[2.7 Поиск и фильтрация каталога продукции 5](#_Toc514917120)

[2.8 Добавление элементов каталога в корзину 5](#_Toc514917121)

[2.9 Просмотр корзины и калькуляция заказа 5](#_Toc514917122)

[2.10 Удаление элементов из корзины 6](#_Toc514917123)

[2.11 Размещение заказа 6](#_Toc514917124)

[2.12 Просмотр списка заказов 6](#_Toc514917125)

[2.13 Завершение работы с системой 6](#_Toc514917126)

[3 СТРУКТУРНАЯ ОРГАНИЗАЦИЯ СИСТЕМЫ 7](#_Toc514917127)

[3.1 Общие принципы организации системы 7](#_Toc514917128)

[3.2 Взаимодействие с внешними сервисами/удалённым модулем для оплаты 7](#_Toc514917129)

[3.3 Организация доступа к данным 7](#_Toc514917130)

[3.4 Организация бизнес-логики 7](#_Toc514917131)

[3.5 Организация веб-интерфейса 7](#_Toc514917132)

[3.6 Взаимодействие компонентов системы для … 7](#_Toc514917133)

[3.7 Организация обработки ошибок 7](#_Toc514917134)

[3.8 Организация управления доступом 7](#_Toc514917135)

[4 РЕАЛИЗАЦИЯ КОМПОНЕНТОВ СИСТЕМЫ 8](#_Toc514917136)

[4.1 EJB-компоненты/компоненты бизнес-логики 8](#_Toc514917137)

[4.2 Веб-компоненты 8](#_Toc514917138)

[4.3 Физическая структура баз данных 8](#_Toc514917139)

[4.4 Обеспечение целостности данных 8](#_Toc514917140)

[4.5 Реализация взаимодействия с внешним сервисом 8](#_Toc514917141)

[5 ИНСТРУКЦИЯ ПО УСТАНОВКЕ ПРИЛОЖЕНИЯ 9](#_Toc514917142)

[6 ОРГАНИЗАЦИЯ УПРАВЛЕНИЯ ПРОЕКТОМ 10](#_Toc514917143)

[6.1 Общие принципы взаимодействия в команде 10](#_Toc514917144)

[6.2 Распределение ролей в команде/зоны ответственности 10](#_Toc514917145)

[6.3 Календарный план работ 10](#_Toc514917146)

[7 НАГРУЗОЧНОЕ ТЕСТИРОВАНИЕ 11](#_Toc514917147)

[ЗАКЛЮЧЕНИЕ 12](#_Toc514917148)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 13](#_Toc514917149)

[ПРИЛОЖЕНИЕ А Листинг компонентов системы 14](#_Toc514917150)

[ПРИЛОЖЕНИЕ Б Скрипт создания базы данных 15](#_Toc514917151)

ВВЕДЕНИЕ

Каждый человек самосовершенствуется, приобретает новые навыки, развивает уже имеющиеся умения. Наше приложение создано для того, чтобы помочь людям в этом.

Для того, чтобы добиваться результатов, нужно ставить для себя цели и уверенно идти к их достижению, то есть совершать маленькие шаги, приводящие к положительному итогу.

1 ПОСТАНОВКА ЗАДАЧИ

1.1 Описание предметной области

1.2 Основные понятия предметной области

1.3 Функциональные требования к системе

1.4 Нефункциональные требования к системе

1.5 Сравнительный анализ аналогов

2 АНАЛИЗ ЗАДАЧИ И ФУНКЦИОНАЛЬНАЯ ДЕКОМПОЗИЦИЯ СИСТЕМЫ

2.1 Анализ бизнес-процесса «Название процесса»

2.2 Сценарий взаимодействия пользователя с системой

2.3 Аутентификация пользователя в системе

2.4 Регистрация пользователя в системе

2.5 Изменение регистрационных данных

2.6 Просмотр каталога продукции

2.7 Поиск и фильтрация каталога продукции

2.8 Добавление элементов каталога в корзину

2.9 Просмотр корзины и калькуляция заказа

2.10 Удаление элементов из корзины

2.11 Размещение заказа

2.12 Просмотр списка заказов

2.13 Завершение работы с системой

3 СТРУКТУРНАЯ ОРГАНИЗАЦИЯ СИСТЕМЫ

3.1 Общие принципы организации системы

3.2 Взаимодействие с внешними сервисами/удалённым модулем для оплаты

В разрабатываемом нами приложении осуществляется взаимодействие с сервисом оплаты. Для этого в рамках выполнения курсового проектирования заранее нами был разработан не хитрый REST сервис в среде IntelliJ IDEA 2017.3.4. В данном пункте кратко расскажем о разработке данного сервиса.

Для того чтобы ускорить процесс написания сервиса нами было принято решение реализовать данный Maven Project при помощи Spring Boot. С помощью: <http://start.spring.io/> мы быстро добавили все необходимые нам зависимости и создали пустой рабочий проект Maven на Spring Boot (рисунок …)

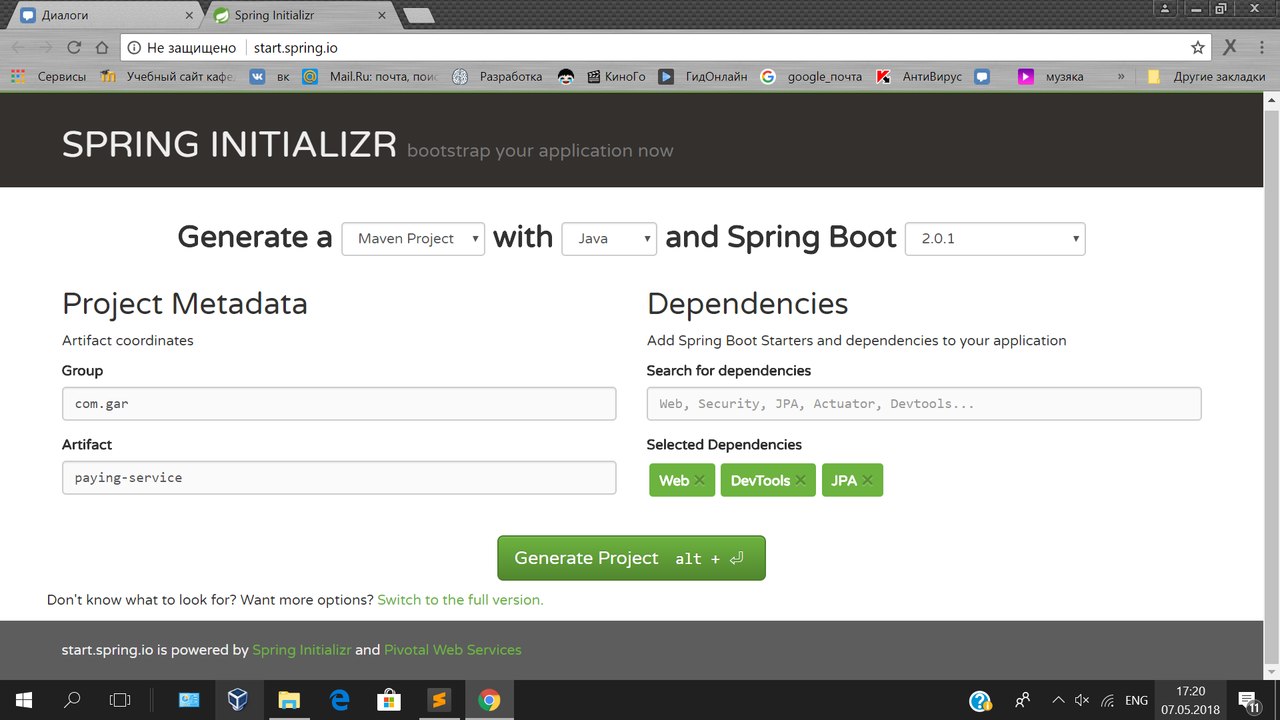


Рисунок 1. Создание Maven Project.

По данному пустому проекте мы буквально за пару часов реализовали наш сервис. Исходные коды представлены в приложении …, а ознакомиться с структурой вы можете на рисунке ….

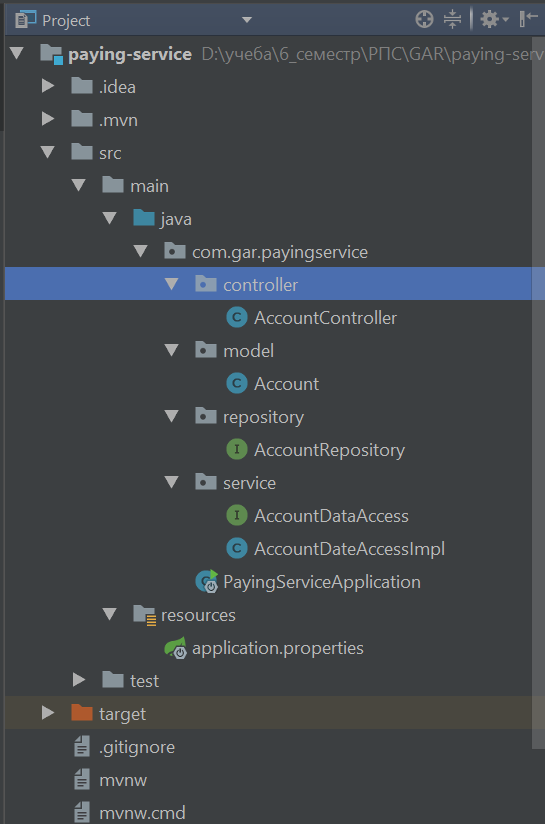


Рисунок 2. Структура проекта.

Вот так быстро нами был заработан простенький веб-сервис с методом оплаты.

3.3 Организация доступа к данным

3.4 Организация бизнес-логики

3.5 Организация веб-интерфейса

3.6 Взаимодействие компонентов системы для …

3.7 Организация обработки ошибок

3.8 Организация управления доступом

4 РЕАЛИЗАЦИЯ КОМПОНЕНТОВ СИСТЕМЫ

4.1 EJB-компоненты/компоненты бизнес-логики

4.1.1 EJB-модуль 1

4.1.2 EJB-модуль 2

4.2 Веб-компоненты

4.2.1 Классы управляемых бинов

4.2.2 JSP-страница login.jsp

4.2.3 JSP-страница register.jsp

4.2.4 Конфигурирование JSF-приложения

4.3 Физическая структура баз данных

По сути дела, физическое проектирование базы данных подразумевает конструирование таблиц в СУБД. И как вы уже знаете из функциональных требований нами была выбрана СУБД MySQL. Причина ее выбора лежит на поверхности это ряд преимуществ: простота использования, гибкость, низкую стоимость владения (относительно платных СУБД), а также масштабируемость и производительность.

В данной СУБД на основании модели классов был написан скрипт создания таблиц, представленный в приложении …. После чего по средствам инструмента для визуального проектирования MySQL Workbench получена физическая модель БД представленная на рисунке … и в приложении ….

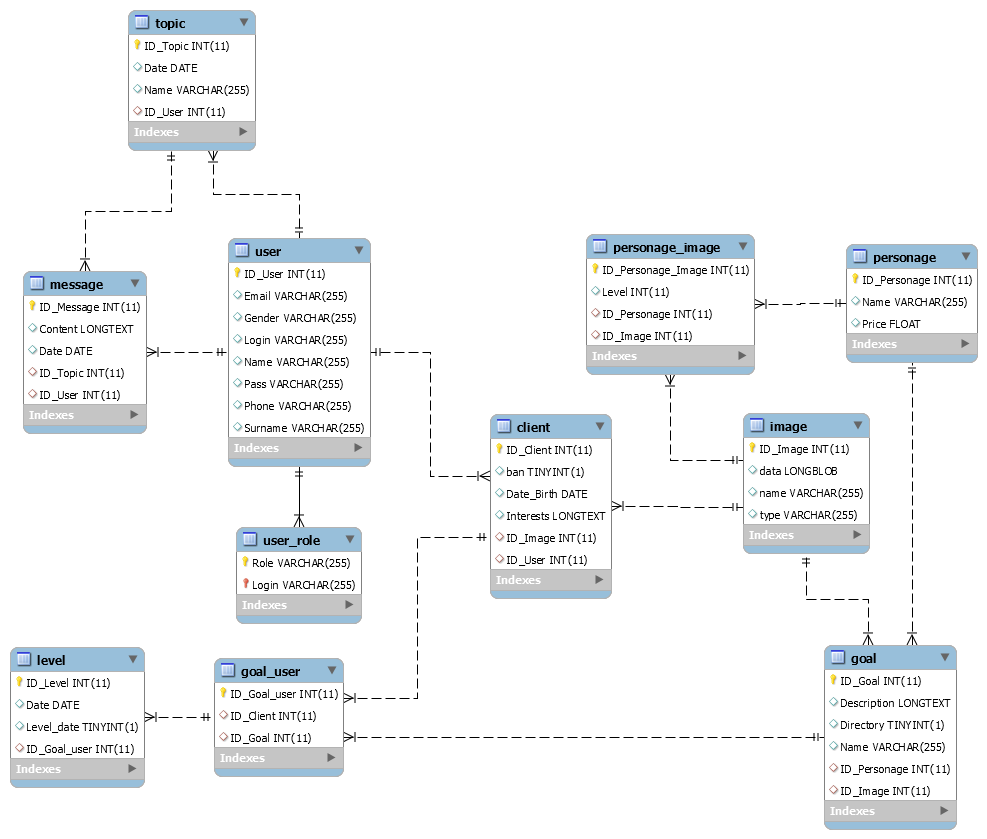


Рисунок 3.Физическая модель БД.

В данном пункте мы разработали физическую модель базы данных необходимую для СУБД MySQL содержащую имена сущностей и полей, а также для каждого атрибута тап данный. Подробнее с проектированием структуры базы банных можно ознакомиться в следующем разделе.

4.4 Обеспечение целостности данных

Для начала разберемся что означает целостность данных. Она означает систему правил, используемых для поддержания связей между записями в связанных таблицах, а также для обеспечения защиты от случайного удаления или изменения связанных данных. В нашей ИС достаточно было обеспечить целостность данных на уровне БД для этого мы воспользовались нормализацией.

Нормализация — это процесс организации данных в базе данных, включающий создание таблиц и установление отношений между ними в соответствии с правилами, которые обеспечивают защиту данных и делают базу данных более гибкой, устраняя избыточность и несогласованные зависимости.

Существует несколько правил нормализации баз данных. Каждое правило называется «нормальной формой». Рассмотрим проведение БД к первой, второй и третей нормальной форме.

1. Первая нормальная форма

* Устраняем повторяющиеся группы в отдельных таблицах.
* Создаем отдельную таблицу для каждого набора связанных данных.
* Идентифицируем каждый набор связанных данных с помощью первичного ключа.

2. Вторая нормальная форма

* Создаем отдельные таблицы для наборов значений, относящихся к нескольким записям.
* Связываем эти таблицы с помощью внешнего ключа.

3. Третья нормальная форма

* Устраняем поля, не зависящие от ключа.

Изучив правила нормализации БД мы предварительно при анализе системы, выявили сущности («User», «Client», «Goal», «Personage», «Image»…). После чего устраняя повторяющиеся группы в таблицах создали отдельную таблицу для каждого набора связанных данных («PersonageImage»…). Идентифицировали каждый набор связанных данных с помощью первичного ключа (ID\_User, ID\_Client …). Создали отдельные таблицы для наборов значений, относящихся к нескольким записям («Image»…). Связали эти таблицы с помощью внешнего ключа. После чего устранили поля, не зависящие от ключа («Topic»…).

В результате приведения БД к 3 нормальной форме мы получили структура БД представленную в пункте 4.3.

4.5 Реализация взаимодействия с внешним сервисом

В пункте 3.2 мы рассказываем о созданном нами веб-сервисе, а в данном пункте мы разберем как же реализовывается взаимодействие.

Как уже говорилось ранее нами использовался архитектурный стиль REST. Он был выбран как быстрое и простое решение для создания сервиса.

Взаимодействие с данным сервисом осуществляется с помощь библиотек:

* httpclient-4.5.5.jar
* httpcore-4.4.9.jar

Для взаимодействия был создан метод:

//оплата персонажа

//исключение обробатываем чуть выше

public boolean payment(String holder, String codeCard, String codeSecurity, String expirationDate, String purchaseValue) throws IOException {

boolean result = false;

HttpClient clientHttp = new DefaultHttpClient();

HttpPost post = new HttpPost("http://localhost:8081/account/purchase");

List nameValuePairs = new ArrayList(1);

nameValuePairs.add(new BasicNameValuePair("name", "value")); //you can as many name value pair as you want in the list.

nameValuePairs.add(new BasicNameValuePair("expirationDate", expirationDate));//2018-05-31

nameValuePairs.add(new BasicNameValuePair("holder", holder));//Lapygina Vasilisa

nameValuePairs.add(new BasicNameValuePair("codeSecurity", codeSecurity));//321

nameValuePairs.add(new BasicNameValuePair("codeCard", codeCard));//1232353424

nameValuePairs.add(new BasicNameValuePair("purchaseValue", purchaseValue));//

post.setEntity(new UrlEncodedFormEntity(nameValuePairs));

HttpResponse response = clientHttp.execute(post);

BufferedReader rd = new BufferedReader(new InputStreamReader(response.getEntity().getContent()));

String line = "";

while ((line = rd.readLine()) != null) {

result = line.equals("true");

}

return result;

}

По средствам данного метода мы отправляем GET запрос на REST сервис и получаем подтверждение если оплата успешно произведена.

В данном пункте мы показали, как в нашем приложении осуществляется взаимодействие с сервисом оплаты.

5 ИНСТРУКЦИЯ ПО УСТАНОВКЕ ПРИЛОЖЕНИЯ

В данном разделе представим пошаговую инструкцию по установке нашего приложения начиная с этапа скачивания исходного кода заканчивая получением полнофункционального приложения с тестовыми данными.

5.1 Установка приложения

Шаг 1: Убедитесь, что у вас установлены обязательные компоненты такие как:

* NetBeans последней версии - <https://netbeans.org/>
* СУБД MySQL (из-за особенностей с кодировкой желательна версия 5.7)
* Проверьте установленный GlassFish Server хотя он и обычно включен в IDE NetBeans. Можете так же выбрать любой удобный вам сервер, но в дальнейшем на нем вам необходимо будет самостоятельно выполнять настройку подключения, а также выполнить настройку домена безопасности.

Шаг 2: Выполните скачивания репозитория архивом (Рисунок …):

<https://github.com/nans1996/GAR>

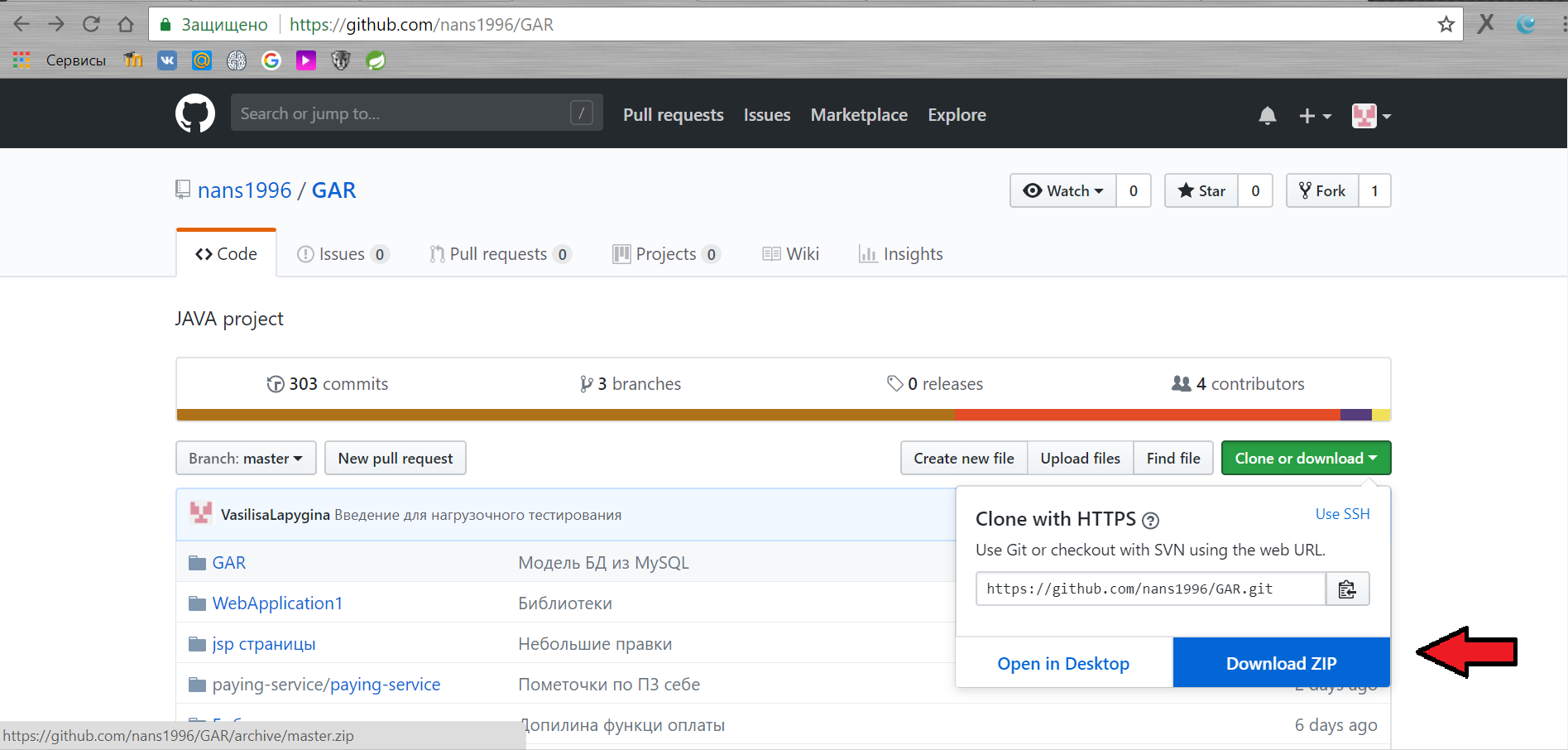


Рисунок 4. Скачивание ZIP архива с проектом.

Шаг 3: Распакуем скачанный архив:

Чтобы извлечь все содержимое сжатой папки, нажмите и удерживайте ее (или щелкните ее правой кнопкой мыши), выберите команду *Извлечь* все, а затем следуйте указаниям.

Шаг 6: Выполним создание базы данных:

* Зайдем в MySQL: mysql –uUSERNAME –pPASSWORD
* Выполним создание БД: create database GAR\_BD CHARACTER SET utf8 COLLATE utf8\_general\_ci;

Шаг 5: Выполним настройку подключения и домена безопасности (рисунок …):

* Откройте терминал (cmd)
* Перейдите в директорию с установленным GlassFish сервером и в папку bin чаще всего это путь: C:\Program Files\glassfish-4.1.1\bin
* Поочередно выполните команды изменив пароль и пользователя на установленные в MySQL:

asadmin create-jdbc-connection-pool --datasourceclassname com.mysql.jdbc.jdbc2.optional.MysqlDataSource --restype javax.sql.DataSource --property portNumber=3306:password=1997:user=root:serverName=localhost:databaseName=GAR\_BD GARPool

asadmin create-jdbc-resource --connectionpoolid GARPool jdbc/GARperson

asadmin create-auth-realm --classname com.sun.enterprise.security.auth.realm.jdbc.JDBCRealm --property datasource-jndi=jdbc/GARperson:user-table=user:user-name-column=Login:password-column=Pass:group-table=User\_Role:group-name-column=Role:jaas-context=jdbcRealm:digest-algorithm=none jdbcRealmGAR

* Если вы совершили ошибки в командах можете выполнить удаление:

asadmin delete-auth-realm jdbcRealmGAR

asadmin delete-jdbc-resource jdbc/GARperson

asadmin delete-jdbc-connection-pool GARPool

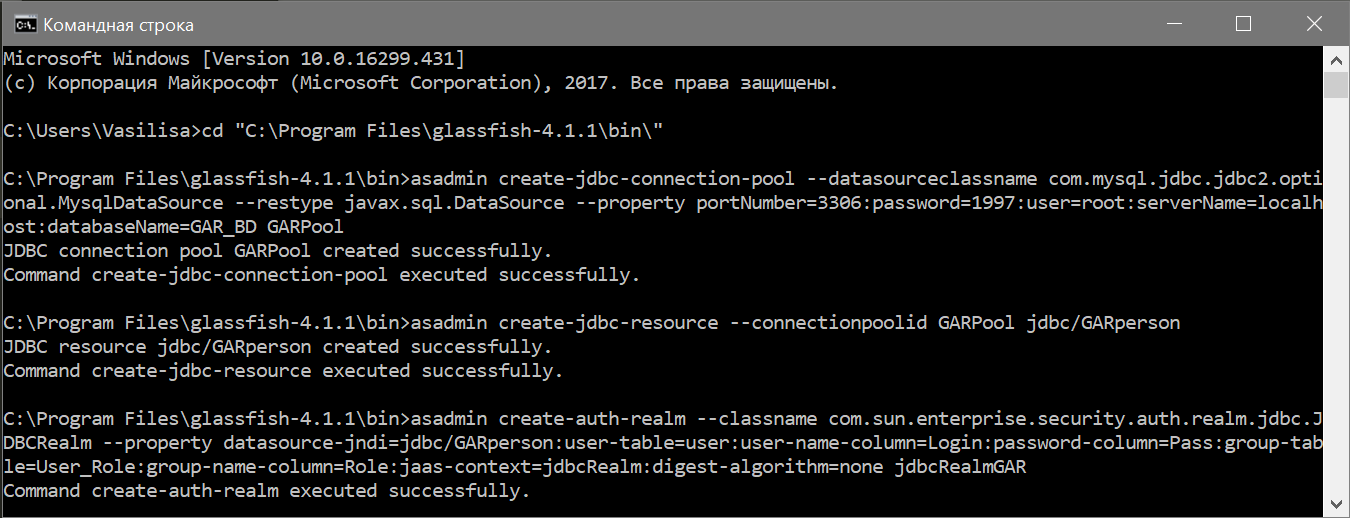


Рисунок 5. Создание подключения и настройка домена безопасности.

* Убедимся в успешности выполненных действий.

Для этого … (рисунок …)

В адресную строку браузера введем: localhost:4848 (рисунок …)

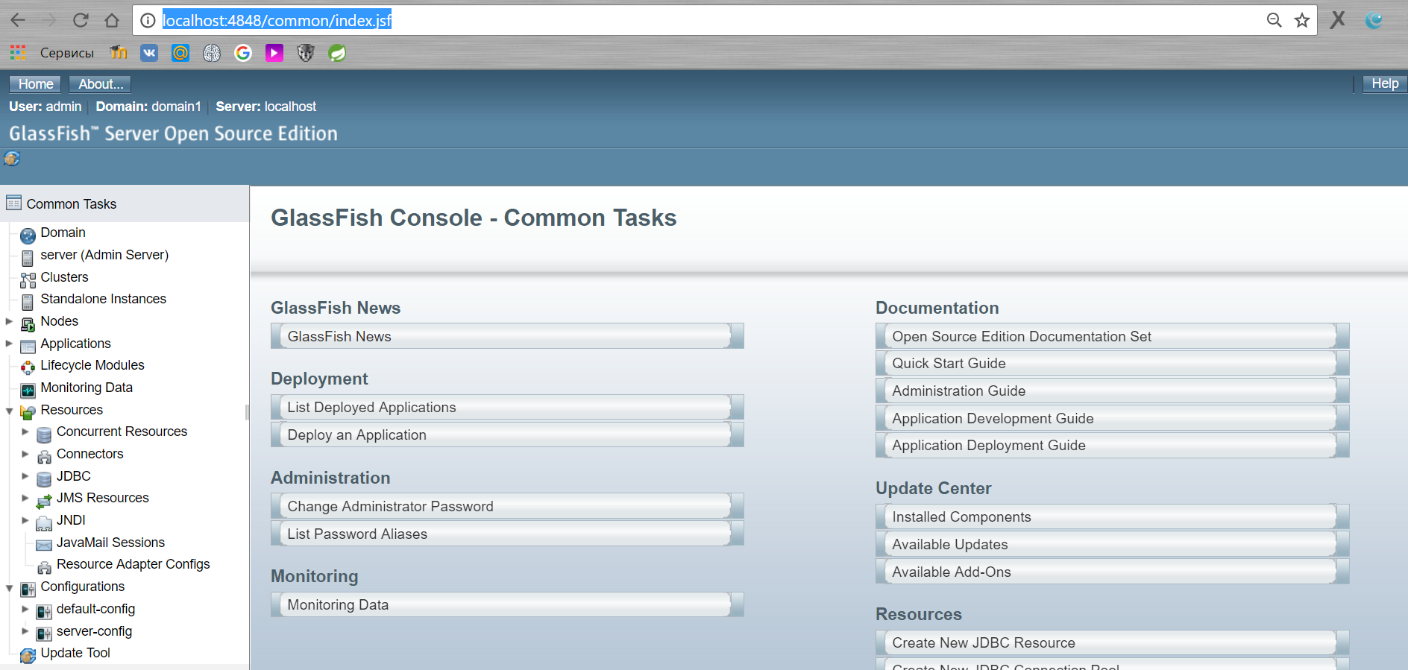


Рисунок 6. Админка сервера.

На вкладке JDBC прейдем в JDBC Connection Pools и зайдем в созданное нами подключение нажмем кнопку Ping. Получим сообщение «Ping Succeeded» (рисунок …)

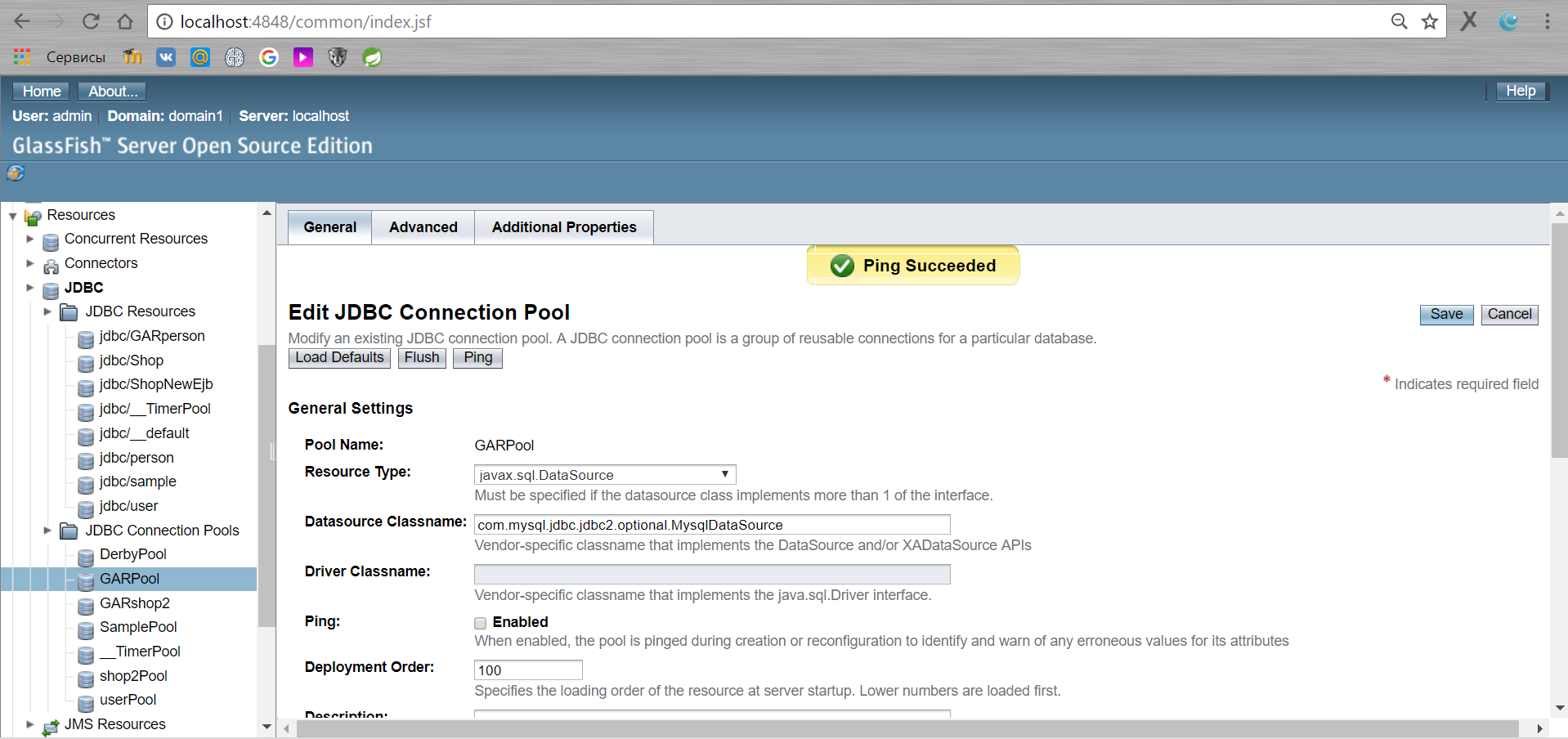


Рисунок 7. Установленное подключение.

Шаг 7: Открытие проекта

* Откроем среду разработки IDE NetBeans
* На панели выберем открыть проект после чего откроем разархивированный в пункте 3 проект (рисунок …)

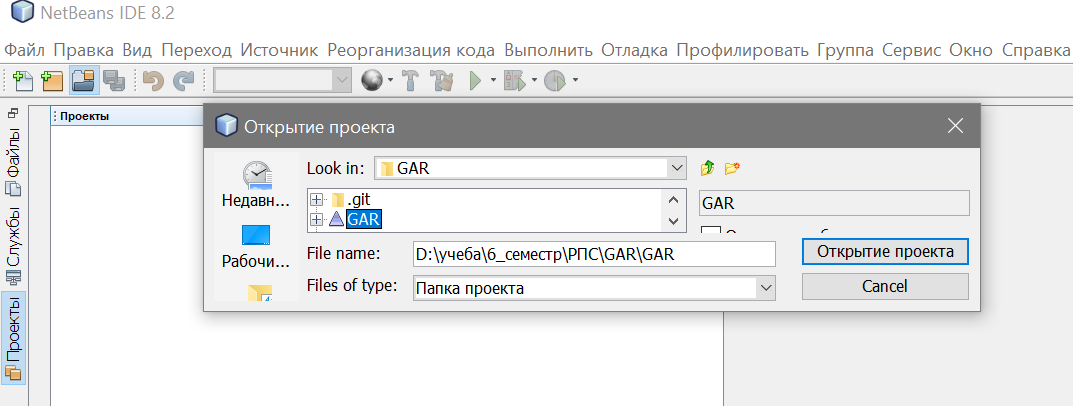


Рисунок 8. Открытие проекта.

* Откроем необходимые модули (рисунок …)

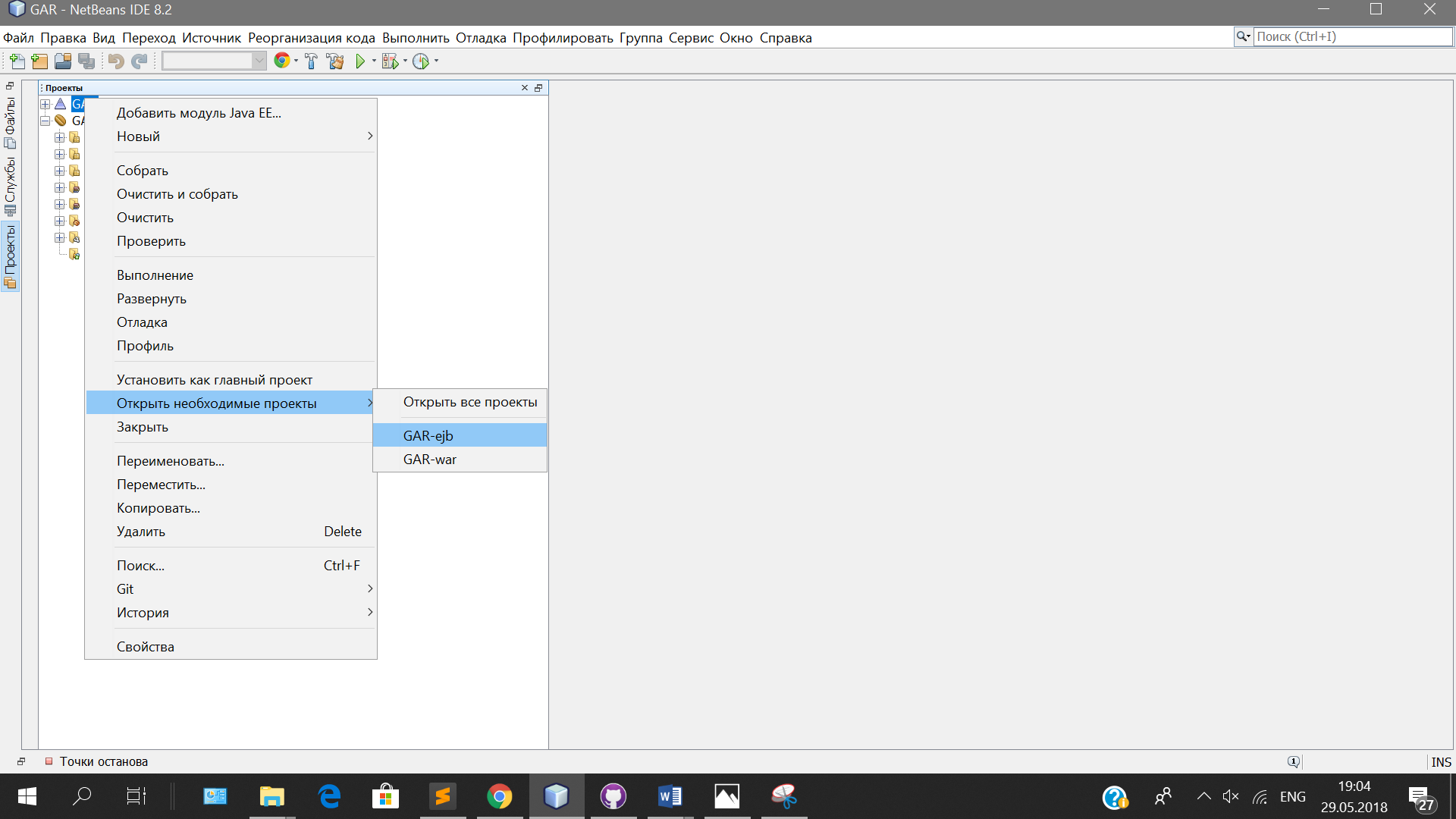


Рисунок 9. Открытие необходимых модулей.

Шаг 8: Деплой проекта

* В модуле persistence.xml укажем стратегию создания таблиц: Create.

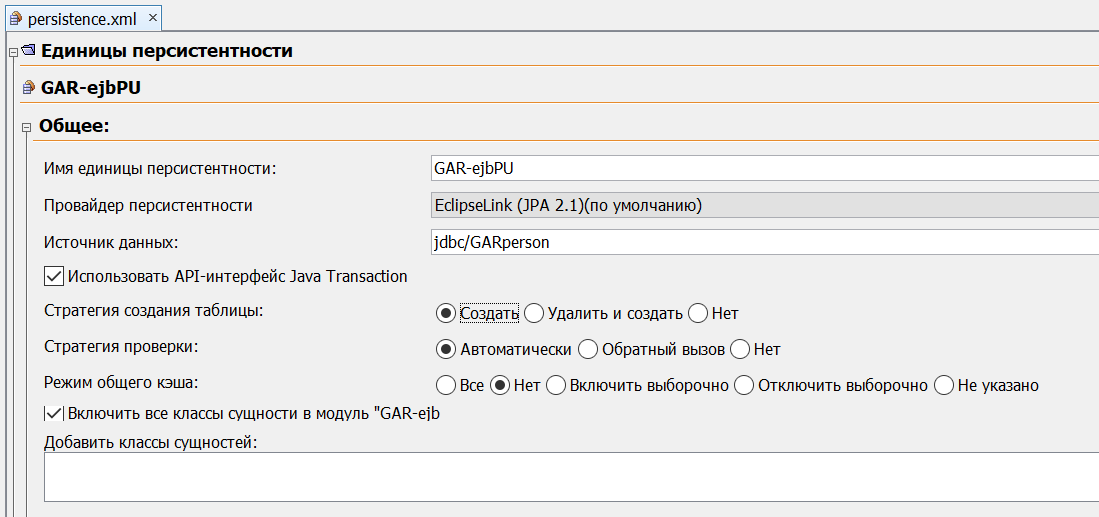


Рисунок 10. Модуль персистенстности.

* Развернём наш проект как показано на рисунке …

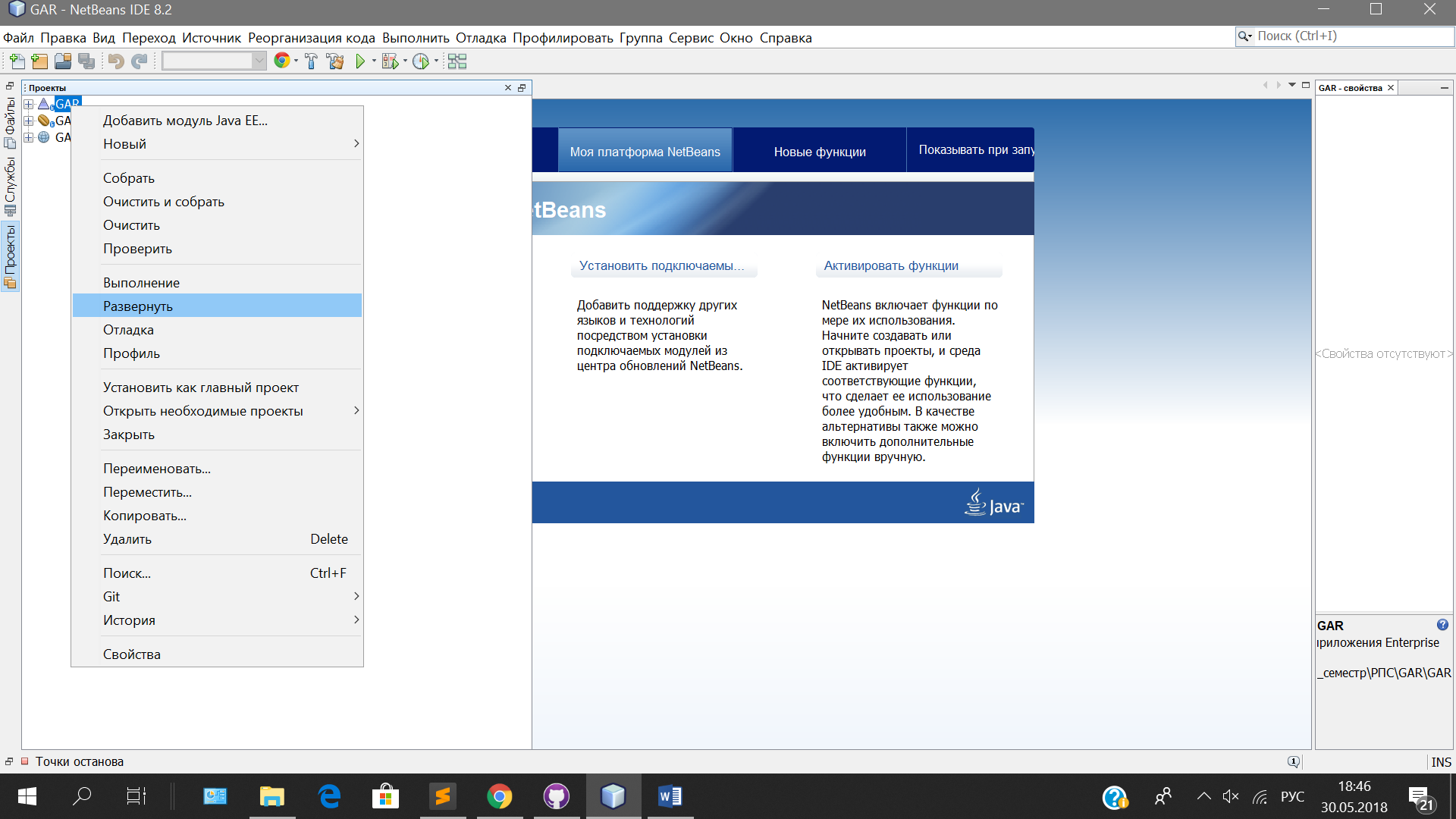


Рисунок 11. Разворачиваем Java EE приложение.

Шаг 9: Открытие проекта (Рисунок …)

* В браузере в адресном сроке введите: http://localhost:8080/GAR-war/

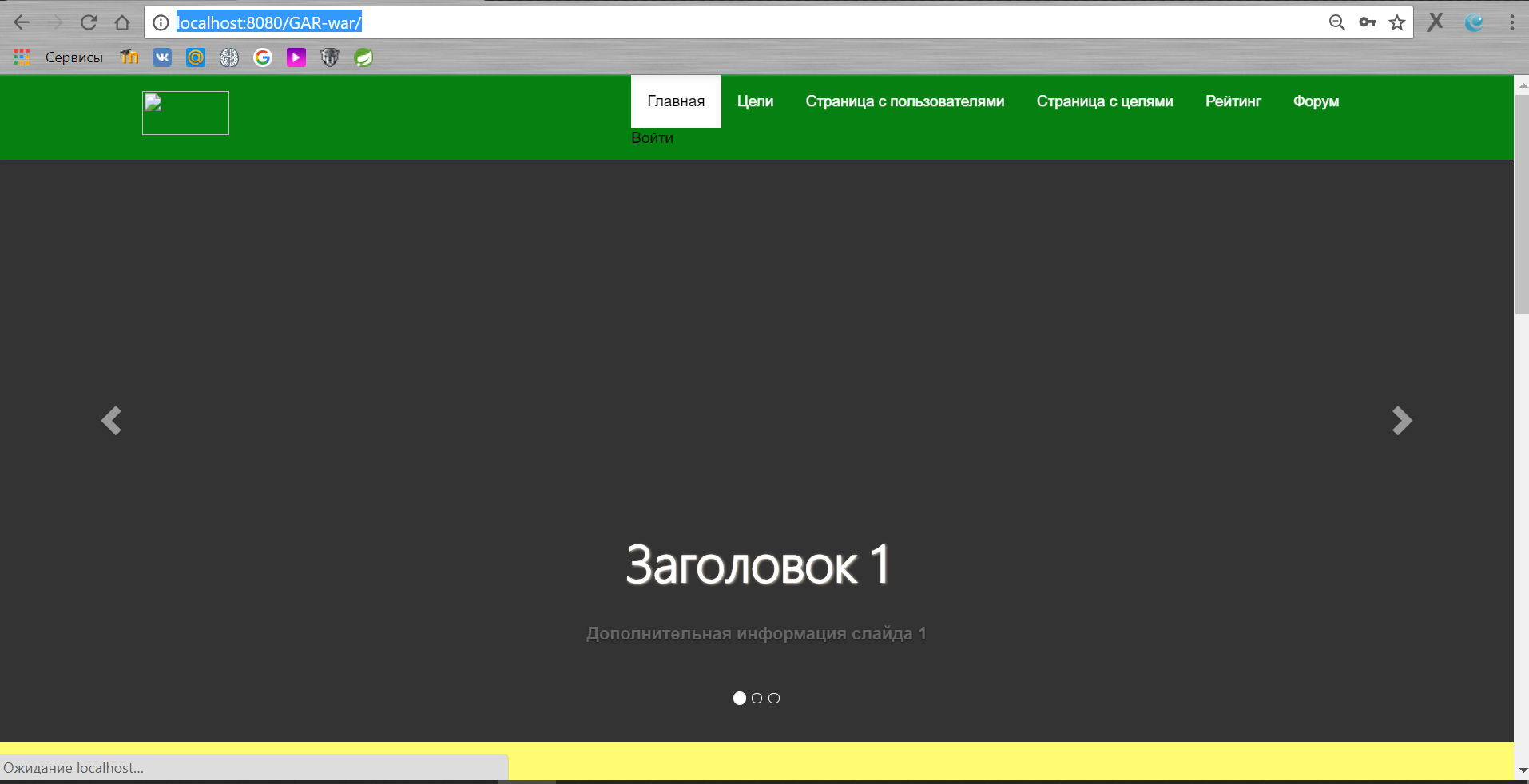


Рисунок 12. Запущенное Web-приложение.

Шаг 10: Заполнение тестовыми данными (данный шаг не обязателен для функционирования приложения)

* Совет для заполнения лучше использовать: SQL Workbench так как, там не только удобный визуальный интерфейс, но есть возможность добавления изображений.
* Так же можете добавить тестовые данные по средствам скрипта:

use GAR\_BD; #открытие бд

#Добавление тестовых пользователей

Insert Into User(Login, Pass, Surname, Name, Phone, Email,Gender) Values('Boss', 'password', 'Boss','Boss','0000000000','adman@inbox.ru','женщина');

Insert Into User(Login, Pass, Surname, Name, Phone, Email,Gender) Values('gambler', 'password', 'Ivanov','Ivan','1111111111','gambler@inbox.ru','мужчина');

#Роли

Insert Into User\_Role(Login, Role) Values('Boss', 'admin');

Insert Into User\_Role(Login, Role) Values('gambler', 'client');

#Проверка

SELECT \* FROM User\_Role,User WHERE User\_Role.Login = User.Login;

#Заполнение дефолтного персонажа

Insert Into Personage(Name, Price) Values('Смурфик', 0);

#Заполнение дефолтного персонажа

Insert Into Personage(Name, Price) Values('Смурфик2', 10);

#Заполнение дефолтной цели 0-это дефолтная цель!!

Insert Into Goal(Name,Directory,Description,ID\_Personage) Values('Спать',0,'Чтобы выглядить красиво нужно высыпаться.',1);

#Дефолтный Игрок

Insert Into Client(Date\_Birth,Interests,ID\_User,ban) Values('1997-07-07','Люблю программировать,рисовать и спать.',1,0);

#Темы для форума

Insert Into Topic(Name,Date,ID\_User) Values('Персонаж','2018-05-04',1);

#Сообщения пользователя

Insert Into Message(Date,Content,ID\_User,ID\_Topic) Values('2018-05-04','Добавте нового персонажа.',1,1);

* Необходимо добавить 1 картинку в БД эта картинка по дефолту будет являться аватаркой регистрируемых пользователей. Этот пункт можно выполнить с помощью SQL Workbench или же в самом приложении.

Если вы без ошибочно выполнили все пункт, поздравляем Web – приложения «GAR» готово к использованию.

5.2 Установка Web-сервиса

Для использования функции покупки персонажа необходимо поднять Web-сервис.

Шаг 1: Убедитесь, что у вас установлены обязательные компоненты такие как:

* IntelliJ IDEA 2017.3.4 x64 и позднее (например учебная версия)
* СУБД MySQL (Можно использовать и другую СУБД)

Шаг 2: Скачайте и разархивируйте репозиторий (если вы еще этого не сделали). Подробная инструкция представлена в пункте 5.1.

Шаг 3: Открытие проекта.

* При запуске IntelliJ IDEA выбираем пункт Import Project (рисунок ….).

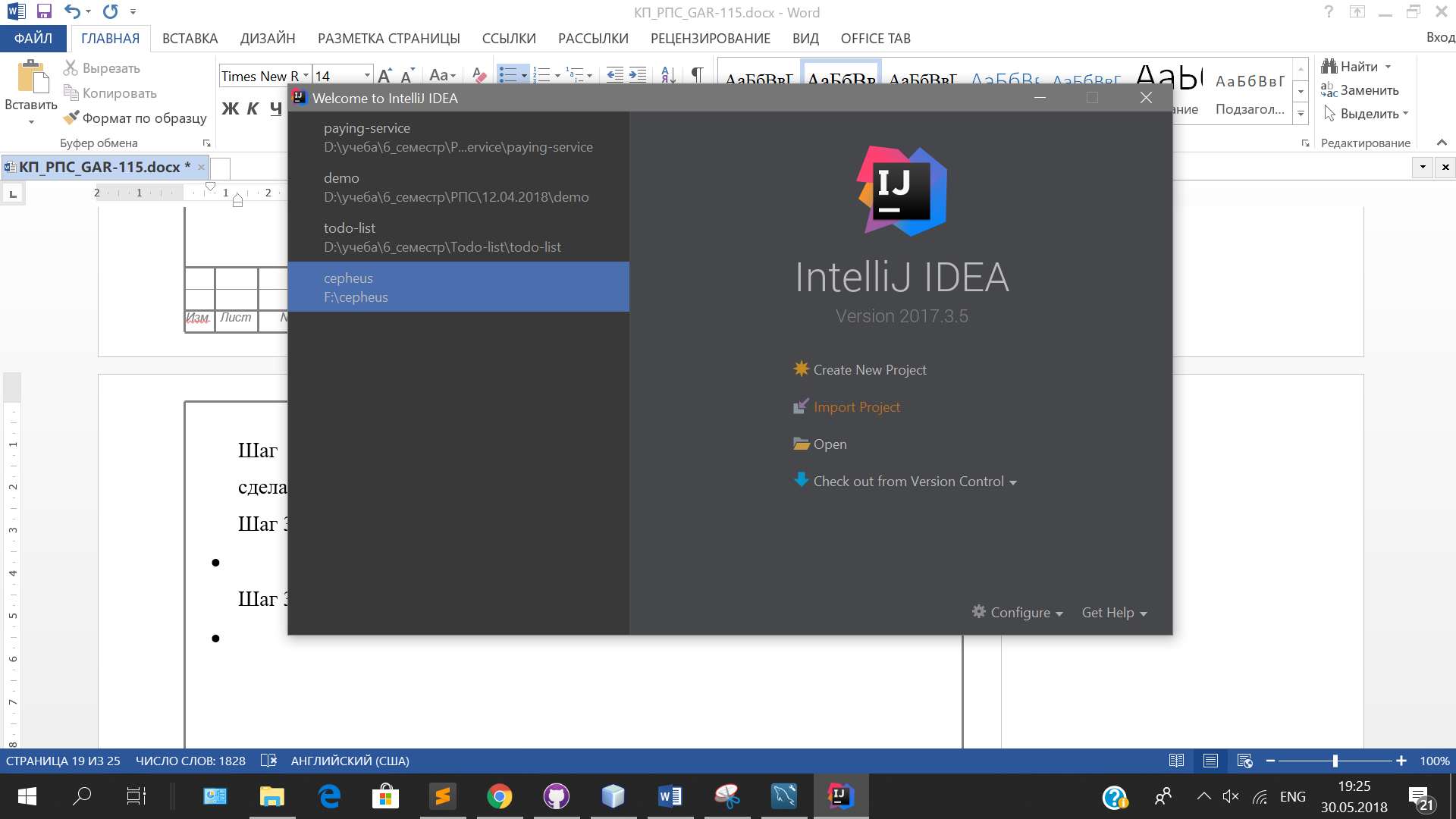


Рисунок 13. Импортируем проект.

* В проекте сервиса выбираем pom.properties ()

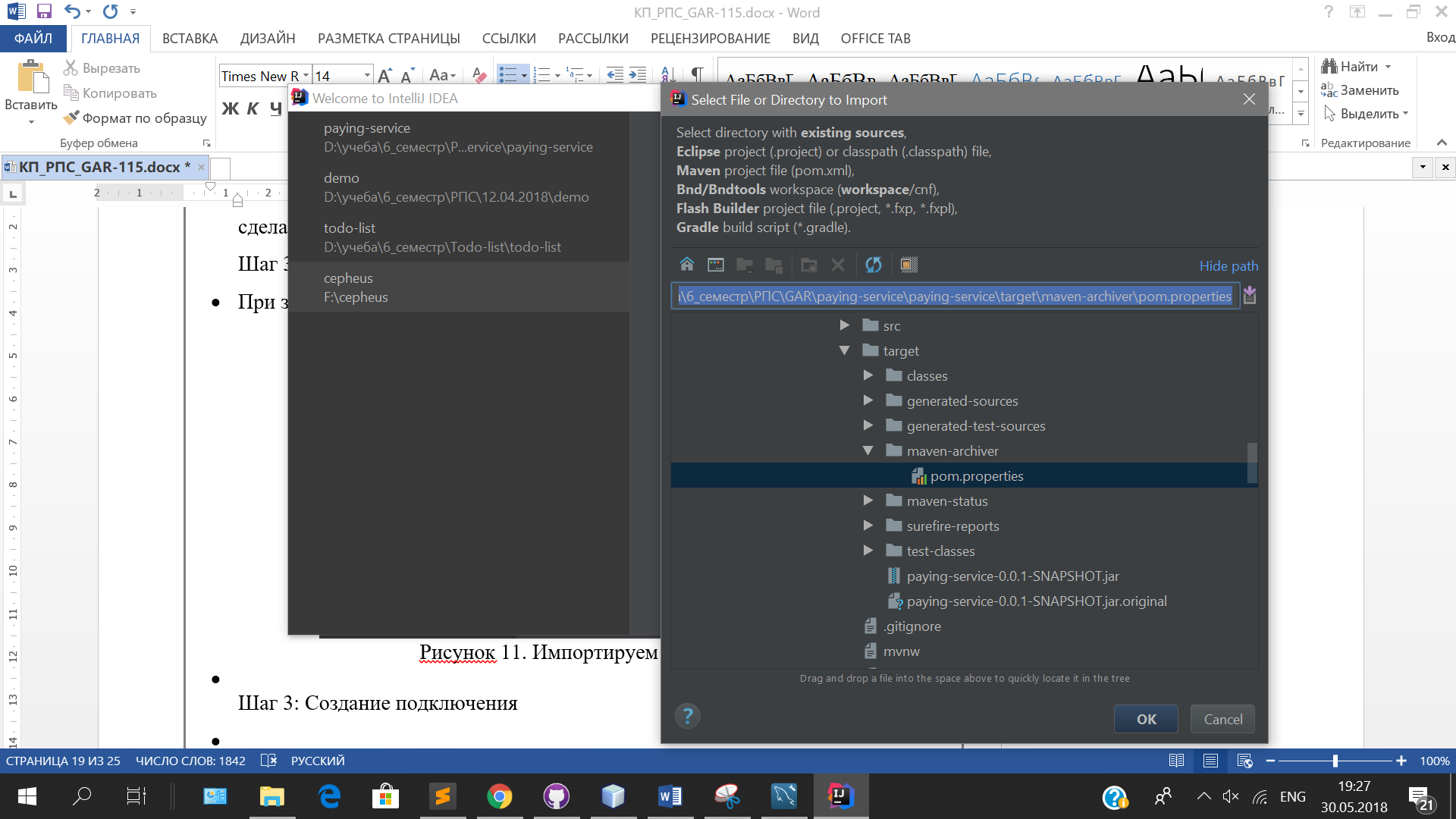


Рисунок 14. Импорт проекта.

* Подгруздка зависимостей. Устанавливаем галочку на против пункта: Import Maven project automaticall

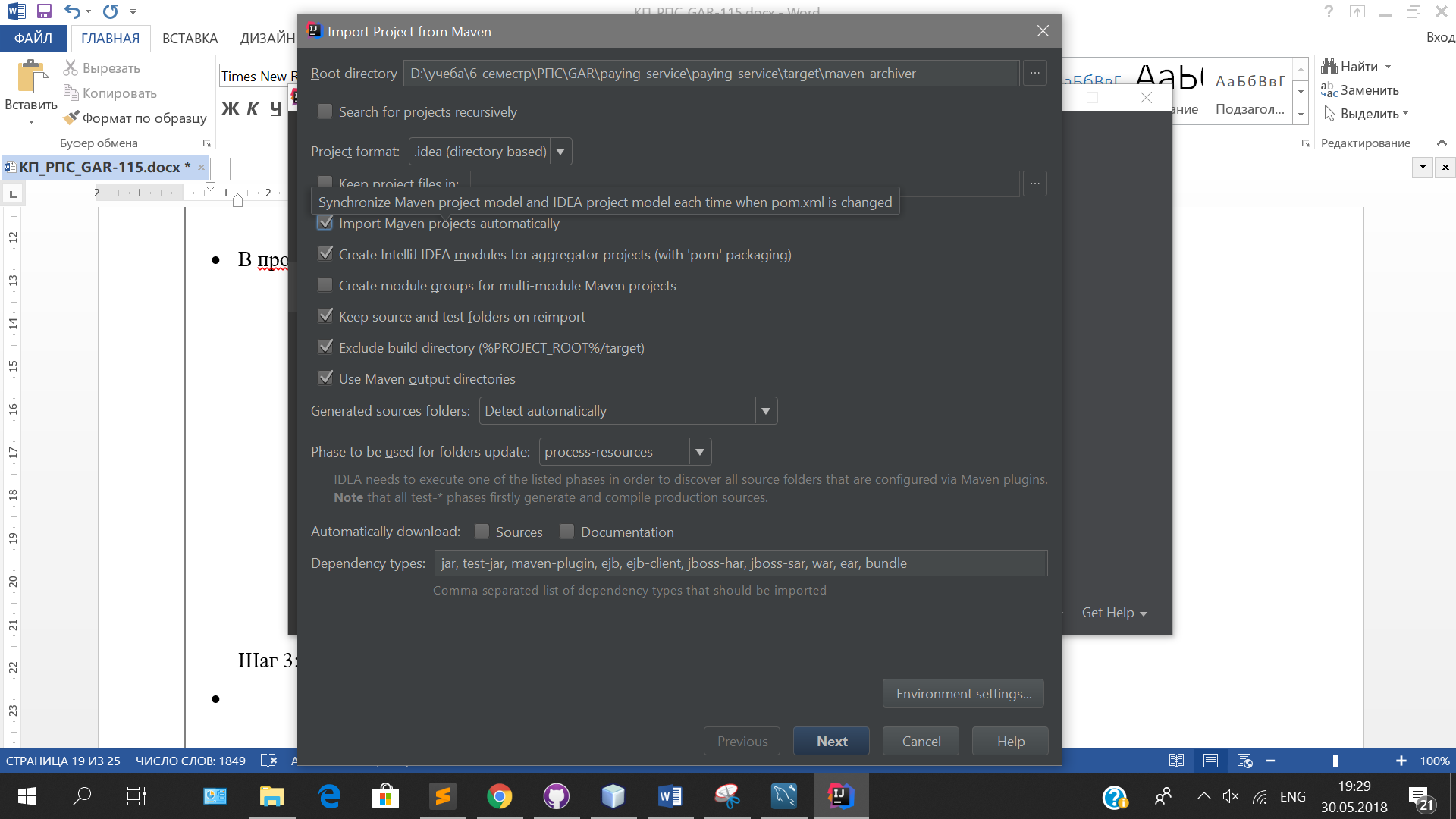


Рисунок 15. Настройка Maven проекта.

* Остальные настройки выполняем по желанию (в последнем пункте желательно добавить понятное вам имя проекта)

Шаг 4: Создание подключения

* Заходим в MySQL и создаем базу данных
* Выбираем вкладку Database после чего выполняем добавление нового подключения (зеленый плюсик) и выберем пункт создать подключение к MySQL.

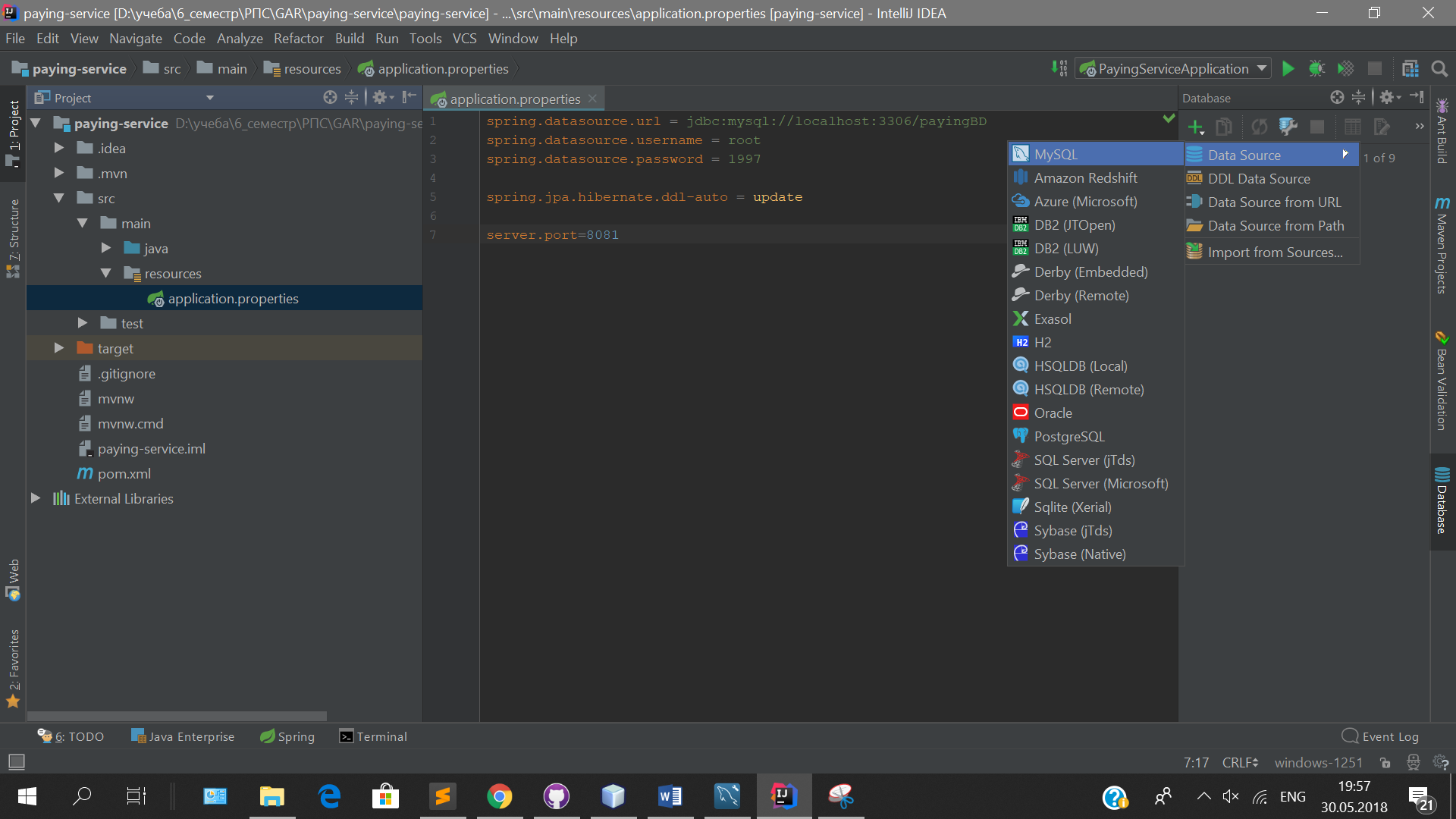


Рисунок 16. Создание подключения.

* Заполним необходимые поля и пропингуем подключение (рисунок …).

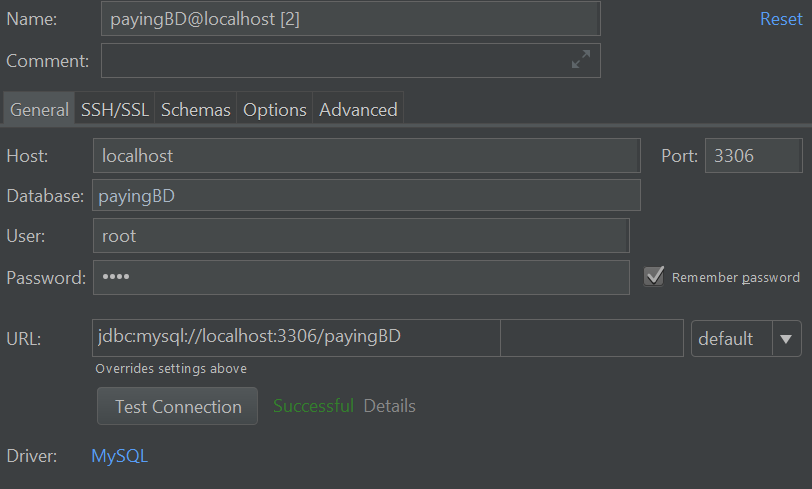


Рисунок 17. Создание подключения.

Шаг 5: Настройка информации о хранилище. Откроем файл application.properties и отредактируем в соответствии с этапами, выполненными ранее (рисунок …)

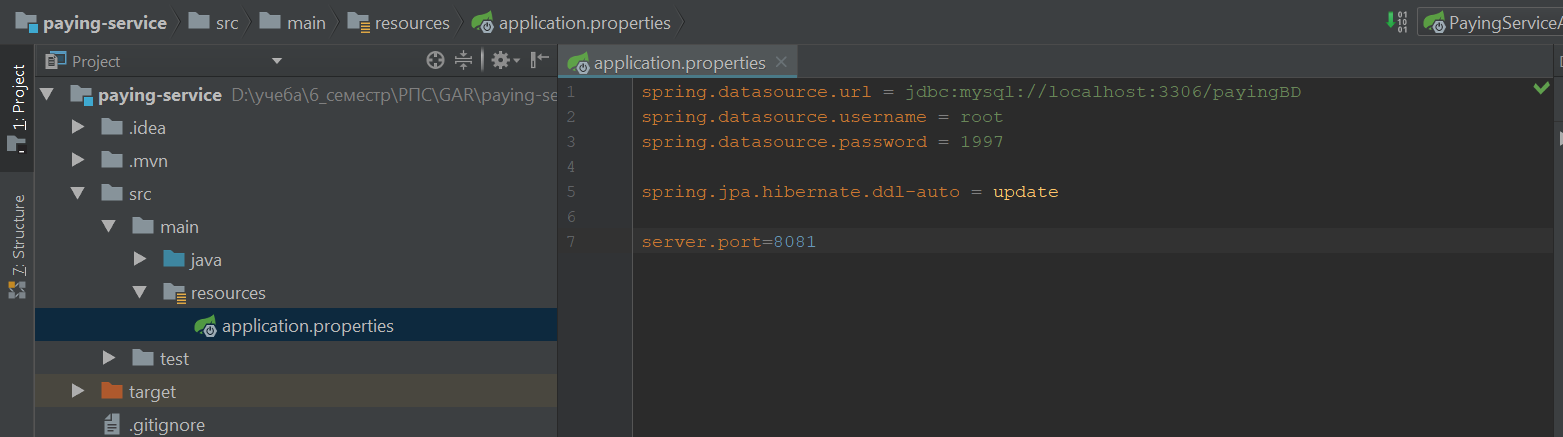


Рисунок 18. Application.properties.

Шаг 6: Сборка и разворачивание прокта

* На вкладке Maven Projects выполним команду собрать. После чего запустим наш проект.

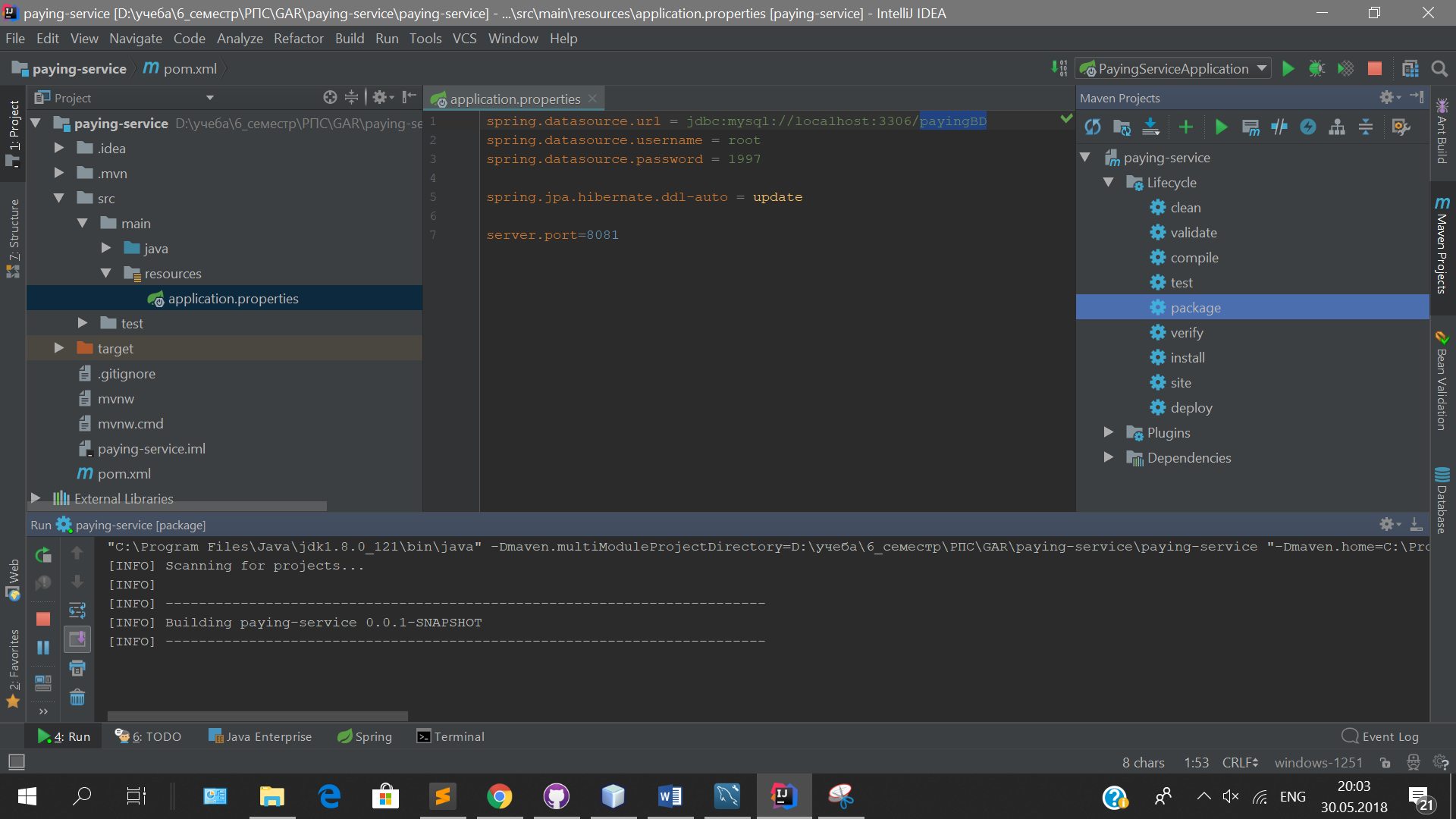


Рисунок 19. Сборка проекта.

Если все пункты выполнены верно проект задиплоется без ошибок и наш сервис готов к использованию.

6 ОРГАНИЗАЦИЯ УПРАВЛЕНИЯ ПРОЕКТОМ

6.1 Общие принципы взаимодействия в команде

6.2 Распределение ролей в команде/зоны ответственности

6.3 Календарный план работ

7 НАГРУЗОЧНОЕ ТЕСТИРОВАНИЕ

Для начала разберем понятие нагрузочного тестирования и расскажем для чего мы его проводим.

Нагрузочное тестирование - вид тестирования производительности, проводимый с целью оценить поведение компонента или системы под увеличивающейся нагрузкой (число одновременно работающих пользователей и/или число транзакций) для определения максимально допустимого уровня нагрузки для исследуемого компонента или системы.  [глоссарий терминов тестирования стр 37]. Из определения становиться понятно, что нашей целью проведения нагрузочного тестирования будет являться оценка поведения системы под увеличивающейся нагрузкой.

Так как наше веб приложение предполагает регулярное (ежедневное) посещение сайта пользователями, то минимальная нагрузка составит для начала 30 человек в день.

В результате нагрузочного тестирования было выявлено …

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Будилов В. А. Интернет-программирование на Java. – СПб.: БХВ-Петербург, 2003

2) Дейтел Х. М., Дейтел П. Дж., Сантри С. И. Технологии программирования на Java 2. – М.: ООО "Бином-Пресс", 2003

3) Иванова Е. Б., Вершинин М. М. Java 2 Enterprise Edition. Технологии проектирования и разработки. – СПб.: БХВ-Петербург, 2003

4) Смирнов Н. Java 2 Enterprise. Основы практической разработки распределенных корпоративных приложений. – М.: Кудиц-образ, 2002

5) ГОСТ 2.105-95 Общие требования к текстовым документам

6) СТП 71.3-04 Дипломное проектирование. Обозначения в документах выпускных квалификационных работ

7) Sun Microsystems, Inc. Enterprise JavaBeans™, Version 3.0: EJB Core Contracts and Requirements, May 2, 2006

8) Sun Microsystems, Inc. Enterprise JavaBeans™, Version 3.0: Java Persistence API, May 2, 2006

9) Sun Microsystems, Inc. Java Platform, Enterprise Edition (Java EE) Specification, v5, April 28, 2006

10) Sun Microsystems, Inc. Java Servlet Specification, v2.5, May 8, 2006

11) Sun Microsystems, Inc. JavaServer Faces Specification, v1.2, May 8, 2006

12) Sun Microsystems, Inc. JavaServer Pages Specification, v2.1, May 8, 2006

ПРИЛОЖЕНИЕ А  
Листинг компонентов системы

ПРИЛОЖЕНИЕ Б  
Скрипт создания базы данных