**СОДЕРЖАНИЕ**

[Введение 3](#_Toc106465615)

[1 Описание организации 4](#_Toc106465616)

[2 Описание компьютерной инфраструктуры организации 5](#_Toc106465617)

[2.1 Аппаратная часть 5](#_Toc106465618)

[2.2 Программная часть 5](#_Toc106465619)

[3 Анализ используемых в организации решений 6](#_Toc106465620)

[4 Выполнение практического задания от организации 7](#_Toc106465621)

[4.1 Постановка задачи 7](#_Toc106465622)

[4.2 Проектирование 7](#_Toc106465623)

[4.3 Программирование 8](#_Toc106465624)

[4.3.1 Установка сред разработки 8](#_Toc106465625)

[4.3.2 Проектирование и разработка базы данных 8](#_Toc106465626)

[4.3.3 Разработка серверной части 12](#_Toc106465627)

[4.3.4 Разработка настольного клиента 16](#_Toc106465628)

[4.3.5 Разработка мобильного клиента 21](#_Toc106465629)

[4.4 Тестирование 23](#_Toc106465630)

[4.4.1 Тестирование методом белого ящика 23](#_Toc106465631)

[4.4.2 Тестирование методом черного ящика 26](#_Toc106465632)

[4.5 Инсталляция 30](#_Toc106465633)

[4.5.1 Установка серверной части 30](#_Toc106465634)

[4.5.2 Установка настольной части 32](#_Toc106465635)

[4.5.3 Установка мобильной части 32](#_Toc106465636)

[5 Обзор работы разработанного программного модуля 33](#_Toc106465637)

[Заключение 38](#_Toc106465638)

[Список источников 39](#_Toc106465639)

# Введение

Производственная практика является необходимым этапом формирования у обучающихся требуемых компетенций. Ее ценность заключается в приобретении новых и закреплении уже полученных знаний.

Целью практики является закрепление теоретического материала, приобретение практических навыков.

Достижению поставленной цели способствует решение следующих задач: анализ организации (деятельности, предметной области, структуры, процессов), анализ аппаратной и программной компьютерной инфраструктуры организации, анализ используемых в организации решений, выполнение практического задания от организации, обзор работы разработанного программного модуля.

Практика проходила в МБОУ «Новобокинская СОШ» по адресу Рязанская область, Сараевский район, село Новобокино, улица Кресты, дом 4.

# Описание организации

С момента учреждения и по настоящее время МБОУ «Новобокинская СОШ» занимается образовательной деятельностью.

Согласно ст. 2 ФЗ «Об образовании в Российской Федерации» № 273-ФЗ от 29.12.2012, «образование – единый целенаправленный процесс воспитания и обучения, являющийся общественно значимым благом и осуществляемый в интересах человека, семьи, общества и государства, а также совокупность приобретаемых знаний, умений, навыков, ценностных установок, опыта деятельности и компетенции определённого объёма и сложности в целях интеллектуального, духовно-нравственного, творческого, физического и (или) профессионального развития человека, удовлетворения его образовательных потребностей и интересов». То есть образование – это, в первую очередь, процесс воспитания и обучения.

Структура МБОУ «Новобокинская СОШ» представлена на рисунке 1.



Рисунок 1 – Структура организации

Процесс, связанный с данной производственной практикой, подразумевает информирование всех звеньев организации (сотрудников и обучающихся) о предстоящих событиях, новостях и прочей информации, требующей публичное распространение внутри организации.

# Описание компьютерной инфраструктуры организации

## Аппаратная часть

В организации располагаются компьютеры с двумя различными конфигурациями.

Первая конфигурация:

* процессор: Intel Pentium 2117U (1800 МГц);
* объем ОЗУ: от 512 МБ до 1 ГБ;
* операционная система: Microsoft Windows XP;
* объем долговременной памяти: 80 ГБ HDD.

Вторая конфигурация:

* процессор: Intel Core i5-10400F;
* объем ОЗУ: 8 ГБ;
* операционная система: Microsoft Windows 10;
* объем долговременной памяти: 512 ГБ SSD и 1 ТБ HDD.

Машины с первой конфигурацией установлены в кабинете информатики и используются обучающимися для выполнения практических заданий по дисциплине «Информатика и ИКТ».

Машины со второй конфигурацией используются во всех остальных местах: кабинет директора, учебные классы, места работы заведующих и социального педагога. Также машины со второй конфигурацией имеют принтеры модели HP M28w. Машины, располагающиеся в учебных классах в добавок, имеют проектор для демонстрации изображения учащимся.

## Программная часть

Машины первой конфигурации имеют следующий набор программного обеспечения:

* веб-браузеры: Mozilla Firefox, Yandex Браузер с Алисой;
* офисные приложения: Microsoft Office 2019, LibreOffice;
* графические редакторы: GIMP, TuxPaint;
* среды разработки: PASCAL.NET.

Машины второй конфигурации имеют следующий набор программного обеспечения:

* веб-браузеры: Mozilla Firefox, Yandex Браузер с Алисой;
* офисные приложения: Microsoft Office 2019, LibreOffice;
* приложения для просмотра файлов: SumatraPDF, VLC;
* почтовый клиент: Mozilla Thunderbird;

# Анализ используемых в организации решений

В МБОУ «Новобокинская СОШ» используется система «БАРС.Образование-Электронная Школа».

«БАРС.Образование-Электронная Школа» повышает контроль качества образовательного процесса и обеспечивает его открытость для граждан. В процессе обучения обеспечивается взаимодействие учителей, учащихся и родителей. Каждая организация имеет доступ через web-интерфейс и получает персональную страницу.

«БАРС.Образование- Электронная Школа» позволяет:

* повысить эффективность процесса управления за счет получения оперативной и достоверной информации о состоянии объектов управления и сокращения времени реакции на принятие управленческого решения, постановку задач и контроля исполнения;
* вести электронный журнал, который полностью аналогичен стандартному бумажному варианту;
* осуществлять планирование учебного процесса в виде учебного, календарно-тематического и поурочного планов в электронном виде;
* в зависимости от учебного плана формировать расписание уроков;
* перевести дневники школьников в электронный вид, а также обеспечить к ним доступ родителям. Через дневник информировать об успеваемости, посещаемости, домашних заданиях, собраниях и всевозможных мероприятиях, что повысит степень участия родителей в образовании детей;
* предоставить информацию о порядке проведения ЕГЭ и ГИА, а также вести базу данных с результатами экзаменов.

Данное решение имеет в своем составе модуль социальной сети с возможностью отправки сообщений, а также модуль доски объявлений. Однако, данная функциональность отсутствует в мобильном приложении системы.

Данная функциональность является крайне важной, так как пользователь предпочитает получать информацию со смартфона, который находится всегда «под рукой» и предоставляет более простой интерфейс, нежели в web-версии для ПК.

# Выполнение практического задания от организации

## Постановка задачи

Требуется разработать систему для оповещения сотрудников и учеников. Данная система должна состоять из трёх компонентов:

* настольный клиент для отправки сообщений сотрудниками и управления администратором;
* мобильный клиент для показа сообщений;
* сервер для обработки запросов от клиентов и обращения к базе данных.

Система должна иметь следующую функциональность:

* авторизация пользователей;
* показ сообщений в мобильном клиенте;
* отправка сообщений в настольном клиенте;
* управление пользователями и сообщениями в настольном клиенте;

## Проектирование

Перечислим технологии, используемые в разработке.

C# — это универсальный многопарадигмальный язык программирования. C# является частью платформы .NET, благодаря этому с помощью C# можно легко создавать программное обеспечение для операционной системы Windows.

Java — это объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems в 1995 году. Главном принципом языка Java является принцип «написано однажды — работает везде». Это означает, что программа, написанная на языка Java, будет работать на любой операционной системе, где присутствует виртуальная машина Java.

Eclipse Jetty — это веб-сервер Java и контейнер Java-сервлетов. В то время как веб-серверы обычно используются для предоставления документов пользователям, Jetty в настоящее время часто используется для межмашинных коммуникаций, обычно в рамках более крупных программных платформ. Jetty разрабатывается как бесплатный проект с открытым исходным кодом в рамках Eclipse Foundation.

JUnit — Фреймворк для модульного тестирования программного обеспечения на языке Java.

Maven — фреймворк для автоматизации сборки проектов

JSON (JavaScript Object Notation) — текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми. Формат JSON был разработан Дугласом Крокфордом. Несмотря на происхождение от JavaScript, формат считается независимым от языка и может использоваться практически с любым языком программирования.

SQL – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

MariaDB — свободная реляционная система управления базами данных.

PhpMyAdmin — это Веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL (MariaDB в том числе).

Git — самая распространённая распределённая система управления версиями.

## Программирование

### Установка сред разработки

Загрузим установочные файлы средств разработки.

1. Загрузка Visual Studio производится с сайта visualstudio.microsoft.com, при установке выбирается пункт «Разработка классических приложений .NET» (рис. 2).

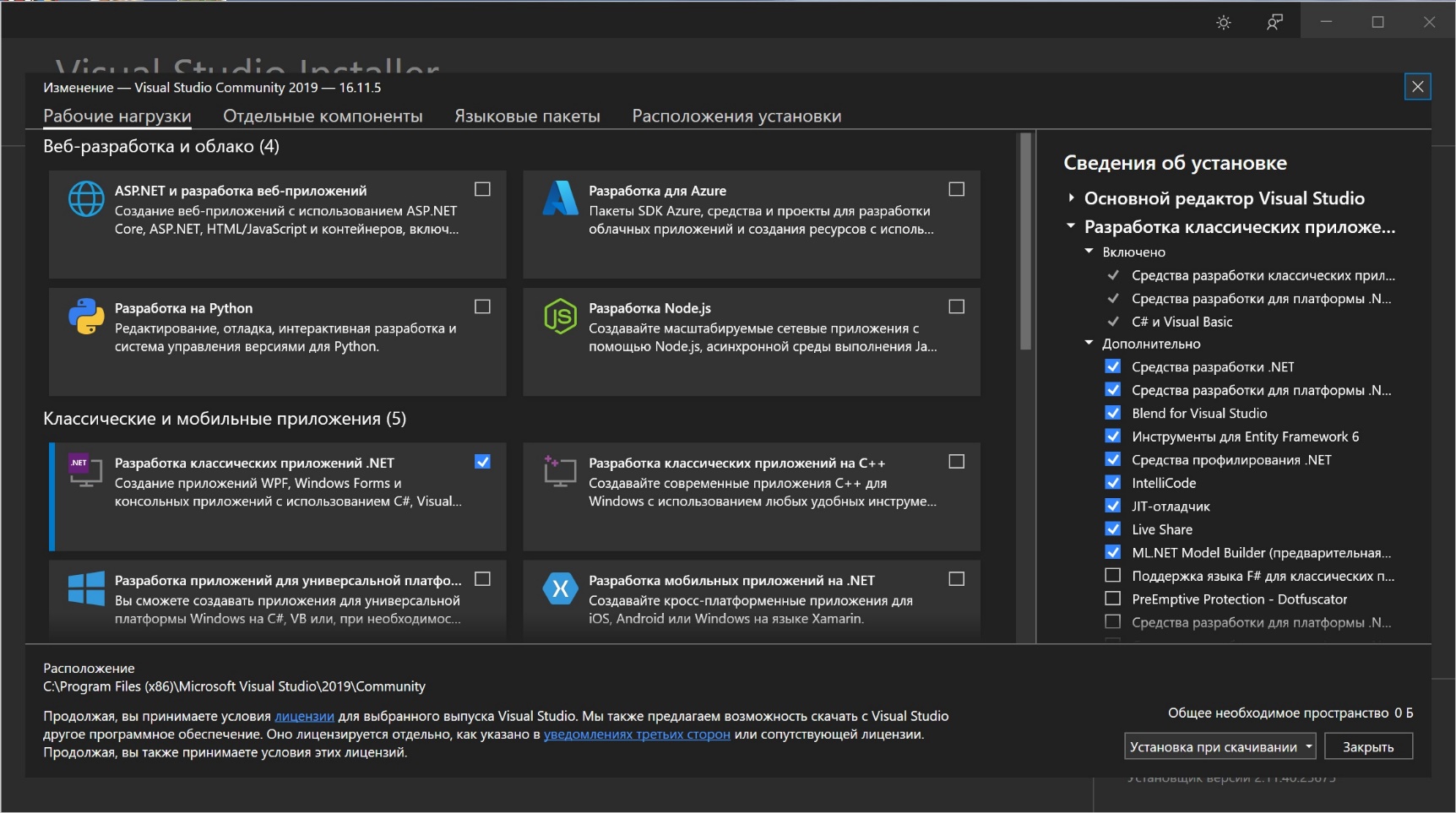


Рисунок 2 ­— Установщик Visual Studio

1. Загрузка Android Studio производится с сайта developer.android.com/studio.
2. Загрузка IntelliJ Idea производится с сайта компании JetBrains.
3. Git загружается с git-scm.com/downloads.

### Проектирование и разработка базы данных

Процесс проектирования БД на основе принципов нормализации представляет собой последовательность переходов от неформального словесного описания информационной структуры предметной области к формализованному описанию объектов предметной области в терминах некоторой модели.

В общем случае можно выделить следующие этапы проектирования:

1. Системный анализ и словесное описание информационных объектов предметной области.
2. Проектирование инфологической модели предметной области – частично формализованное описание объектов предметной области в терминах некоторой семантической модели, например, в терминах ER-модели.
3. Дата логическое или логическое проектирование БД, то есть описание БД в терминах, принятой дата логической модели данных.
4. Физическое проектирование БД, то есть выбор эффективного размещения БД на внешних носителях для обеспечения наиболее эффективной работы приложения.

Структура таблиц определяет эффективность программ, обрабатывающих эти таблицы, и всего приложения в целом. Реляционная модель базы данных основывается на математических принципах теории реляционных наборов. Для простоты манипулирования данными при создании таких наборов рекомендуется нормализовать эти данные.

Нормализация – это процесс исключения избыточной информации: сложные данные разбиваются на отдельные таблицы, между которыми могут быть установлены отношения. Для определения структуры каждой таблицы необходимо выполнить анализ функциональных зависимостей. В результате количество необходимых таблиц определяется числом функциональных зависимостей. Формально нормализация данных обеспечена, если набор таблиц удовлетворяет первым трем правилам, которые называют нормальными формами. Для приведения модели базы данных к требуемому уровню нормальной формы, а это является основой построения реляционной базы данных, процесс проектирования должен пройти несколько этапов. Для построения реляционной базы данных необходимо выделить сущности и связи между ними, определить атрибуты сущностей, задать первичные и внешние ключи, привести модель к требуемому уровню нормальной формы.

Для построения реляционной базы данных необходимо выделить сущности и связи между ними, определить атрибуты сущностей, задать первичные и внешние ключи, привести модель к требуемому уровню нормальной формы. Сущность – это объект, информация о котором хранится в базе данных.

При разработке базы данных выделяем следующие этапы:

#### Определение сущностей

В результате анализа поставленной задачи можно составить концептуальную модель данных и получить следующие сущности: сообщение, пользователь.

Приведем диаграмму «Сущность-связь», которая представлена на рисунке 3.



Рисунок 3 — Диаграмма «Сущность-связь»

В данной модели используется следующее отношение: между сущностями «Пользователь» и «Сообщение» используется отношение «один-ко-многим», так один пользователь может написать несколько сообщений.

#### Приведение модели базы данных к первой нормальной форме

Отношение находится в первой нормальной форме, если все его атрибуты являются простыми (имеют единичное значение). Применим к этим сущностям условия первой нормальной формы: должны отсутствовать повторяющиеся записи, должны отсутствовать повторяющиеся атрибуты, каждый атрибут (поле) должен быть неделимым. Задаем первичные и альтернативные ключи. Для каждой сущности определяем атрибуты, которые будем хранить в БД. Приведем таблицу атрибутов и первичных ключей сущностей информации модели и получим отношение модели в первой нормальной форме.

Условия первой нормальной формы:

* должны отсутствовать повторяющиеся записи;
* должны отсутствовать повторяющиеся атрибуты;
* каждый атрибут должен быть неделим.

Для каждой сущности определим атрибуты, которые будут храниться в базе данных. Определим для каждой сущности атрибуты:

Сущность «Пользователь» имеет следующие атрибуты:

* код;
* имя;
* хэш;
* роль.

Сущность «Сообщение» имеет следующие атрибуты:

* код;
* текст сообщения;
* пользователь.

#### Приведение модели базы данных ко второй нормальной форме

Отношение находится во второй нормальной форме, если оно удовлетворяет следующим условиям:

⦁ выполняется условие первой нормальной формы;

⦁ первичный ключ однозначно определяет запись;

⦁ все поля записи зависят от первичного ключа;

⦁ первичный ключ имеет минимальную форму (отсутствует избыточность).

В соответствии с этим приведем таблицу отношений атрибутов и первичных ключей (табл. 1).

Таблица 1 — Приведение БД ко второй нормальной форме

|  |  |  |
| --- | --- | --- |
| **СУЩНОСТЬ** | **ПЕРВИЧНЫЙ КЛЮЧ** | **АТРИБУТЫ** |
| Пользователь | Код\_пользователя | Имя\_пользователя;  Хэш;  Код\_роли.. |
| Сообщение | Код\_сообщения | Текст\_сообщения;  Код\_пользователя |
| Роль | Код\_роли | Наименовании\_роли |

Так база данных во второй нормальной форме не содержит транзитивных связей, база данных приведена к третьей форме.

#### Физическое описание базы данных

Пятый этап состоит в физическом описании модели. На этом этапе создаются проекты таблиц (структуры), которые будут в дальнейшем реализовываться в конкретной системе управления базами данных на машинных носителях информации.

База данных состоит из 3 таблиц. Структура базы данных приведена ниже.

Таблица 2 — Сущность «Пользователь»

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип поля** | **Размер поля** | **Примечание** |
| User\_id | Int | 11 | Идентификатор пользователя |
| Username | Varchar | 50 | Имя пользователя |
| Hash | Varchar | 64 | Хэш |
| Role\_id | Int | 11 | Роль пользователя |

Таблица 3 — Сущность «Сообщение»

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип поля** | **Размер поля** | **Примечание** |
| Message\_id | Int | 11 | Идентификатор сообщения |
| Text | Text |  | Текст сообщений |
| User\_id | Int | 11 | Автор сообщения |

Таблица 4 — Сущность «Роль»

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип поля** | **Размер поля** | **Примечание** |
| Role\_id | Int | 11 | Идентификатор роли |
| role\_name | Varchar | 50 | Наименование роли |

### Разработка серверной части

Для разработки серверной части используется язык Java с библиотекой Jetty для создания веб-сервера. В качестве среды разработки используется IntelliJ Idea.

Первым шагом является создание проекта. Необходимо выбрать проект с поддержкой Maven (рис. 4).

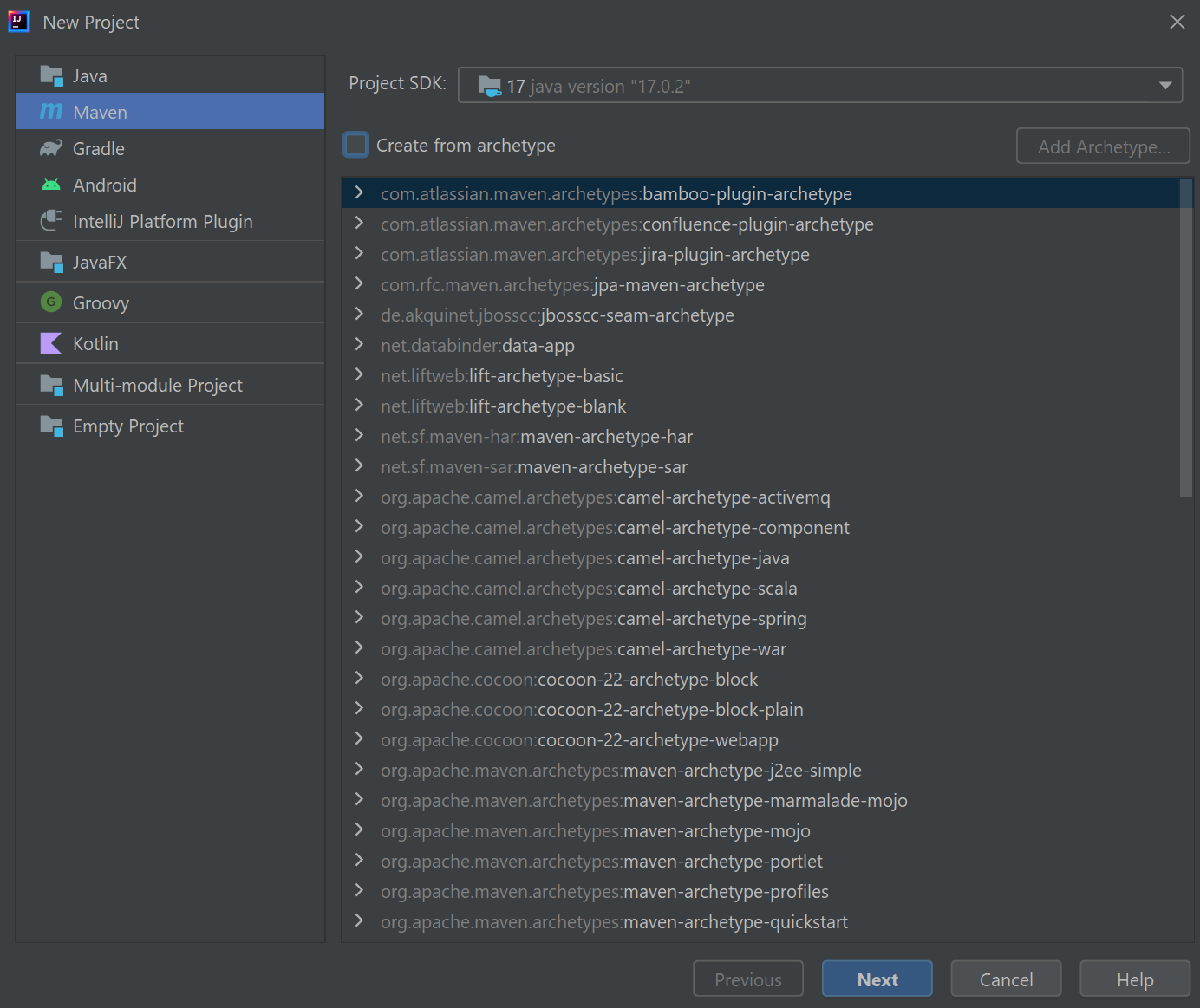


Рисунок 4 — Создание проекта в Intellij Idea

После необходимо добавить зависимости в проект. Для этого нужно найти файл pom.xml и добавить в него следующие строки:

<dependencies>

<dependency>

<groupId>org.eclipse.jetty</groupId>

<artifactId>jetty-server</artifactId>

<version>11.0.9</version>

</dependency>

<dependency>

<groupId>org.eclipse.jetty</groupId>

<artifactId>jetty-webapp</artifactId>

<version>11.0.9</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.29</version>

</dependency>

<dependency>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

<version>2.9.0</version>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter</artifactId>

<version>5.8.2</version>

</dependency>

</dependencies>

После этого необходимо загрузить изменения (рис. 5).

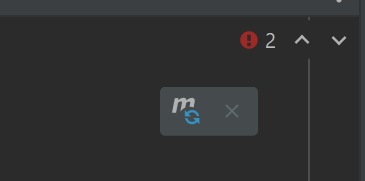


Рисунок 5 — Обновление изменений Maven

После этого создаётся класс Main с методом main. Это точка входа в программу. В методе main происходит создание обработчика сервлетов, инициализация самих сервлетов и создание и запуск сервера. В методе также запускается поток, контролирующий сессии, о которых будет написано далее.

Код метода main:

try {

// Создание обработчика сервлетов и добавление сервлетов

ServletContextHandler contextHandler = new ServletContextHandler(ServletContextHandler.SESSIONS);

contextHandler.addServlet(new ServletHolder(new IndexServlet()), "/index.html");

contextHandler.addServlet(new ServletHolder(new MainServlet(args[0], args[1], args[2])), "/request");

// Создание сервера

Server server = new Server(4242);

server.setHandler(contextHandler);

// Поток для уменьшения длительности сессий и удаления истекших сессий

new Thread(() -> {

try {

while(true) {

SessionController.getInstance().deleteExpiredSessions();

SessionController.getInstance().subOneMinute();

SessionController.getInstance().printSessions();

Thread.sleep(60000);

}

} catch (InterruptedException e) {

e.printStackTrace();

}

}).start();

// Запуск сервера

server.start();

server.join();

} catch (Exception e){ e.printStackTrace();}

MainServelet – это главный сервлет, обрабатывающий запросы от пользователей.

HashUtils – это класс, содержащий статические классы для генерации хэша, используемый при аутентификации пользователей.

SessionController ­ это singleton-класс, используемый для хранения и работы с сессиями.

Возможности данного сервера:

* обработка запросов от пользователей;
* аутентификация пользователей с выдачей сессии;
* работа с базой данных.

Алгоритм работы сервера с клиентом (позитивный сценарий):

1. на сервер поступает запрос на авторизацию, клиент передаёт логин и пароль;
2. сервер вычисляет хэш пароля, находит пользователя в БД и выдает id сессии (что, по сути, тоже является хэш-суммой), изначально длительность сессии равна 5 минутам;
3. теперь для подтверждения личности клиент присылает свой id сессии, по id сессии находится информация о пользователе и о его роли, при каждом запросе от клиента время сессии увеличивается на 1 минуту;
4. каждую минуту время всех сессий уменьшается, данный процесс реализован в классе Main, который описан ранее.

Вид запроса клиент задаёт в переменной action.

Таблица 5 — Виды запросов

|  |  |  |  |
| --- | --- | --- | --- |
| **Запрос** | **Входные параметры** | **Выходные параметры** | **Доступен** |
| Пустой запрос (act=null) | Отсутствуют | Html-страница с сообщениями | Без авторизации |
| Login | Логин, пароль | Id сессии, Объект-пользователь, время сессии | Всем пользователям |
| getMessages | Id сессии | Массив с сообщениями | Всем пользователям |
| getUsers | Id сессии | Массив с пользователями | Администратору |
| getRoles | Id сессии | Массив с ролями пользователей | Администратору |
| sendMessage | Id сессии, текст сообщения | Строка success в случае успеха, строка error в случае ошибки | Всем, кроме учеников |
| delUser | Id сессии, id сообщения | Строка success в случае успеха, строка error в случае ошибки | Администратору |

### Разработка настольного клиента

Настольный клиент обязан иметь следующие функции:

* отправка сообщений;
* управление пользователями и сообщениями (добавление, удаление).

Схема пользовательского взаимодействия представлена на рисунке 6.



Рисунок 6 — Схема пользовательского взаимодействия

Авторизация станет пользовательской точкой входа в приложение. Окно авторизации будет отправлять вводимые данные и получать id сессии.

Макет страницу авторизации представлен на рисунке 7.

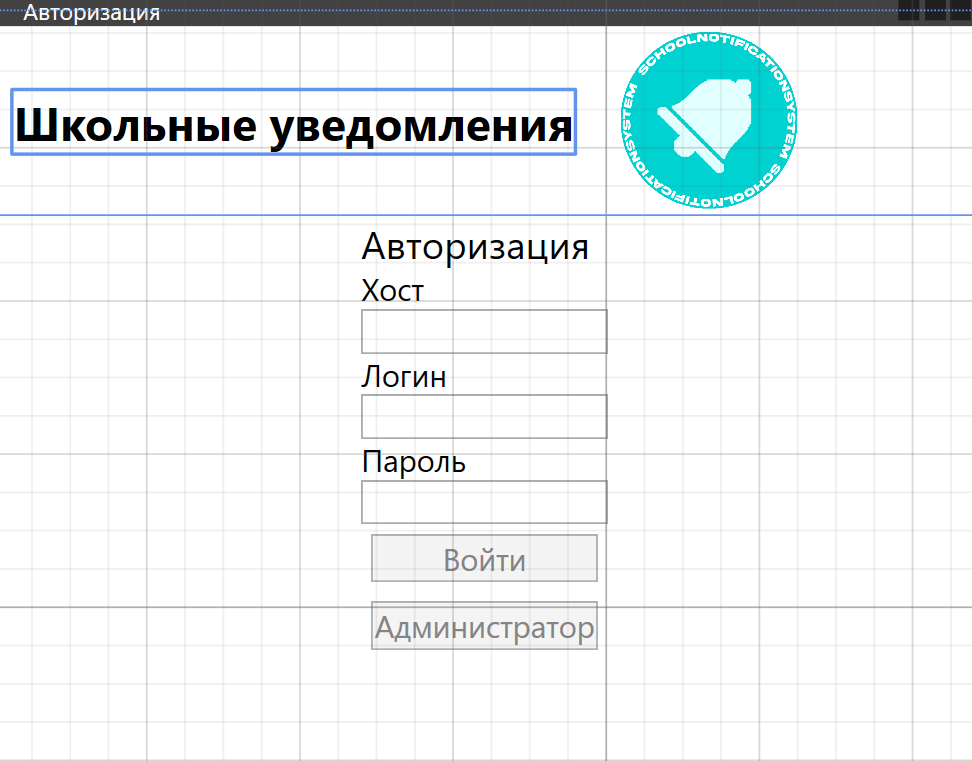


Рисунок 7 — Макет окна авторизации

Метод авторизации:

private static async Task LoginTask(HttpClient client, string host, string login, string password,

Button btnL, Button btnAdm, LoginRole role, Window currentWindow)

{

string stringTask = "";

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(

new MediaTypeWithQualityHeaderValue("application/json"));

client.DefaultRequestHeaders.Add("User-Agent", "SchoolNotificationSystem desktop");

try

{

Uri uri = new Uri($"http://{host}/request?action=login&login={login}&password={password}");

stringTask = await client.GetStringAsync(uri);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка соединения", "Ошибка");

return;

}

finally

{

btnL.IsEnabled = true;

btnAdm.IsEnabled = true;

}

JObject jObject = JObject.Parse(stringTask);

JToken tokenId = jObject["id"];

if (tokenId == null)

{

JToken tokenError = jObject["error"];

string errorMsg = tokenError.ToObject<string>();

MessageBox.Show(errorMsg, "Ошибка");

return;

}

string sessionId = tokenId.ToObject<string>();

if (role == LoginRole.Admin)

{

JToken tokenUser = jObject["user"];

User user = tokenUser.ToObject<User>();

if (!user.roleName.Equals("Администратор"))

{

MessageBox.Show("Пользователь не является администратором", "Ошибка");

return;

}

}

if (role == LoginRole.Admin)

{

var win = new AdminWindow(host, sessionId);

win.Show();

currentWindow.Close();

}

else if (role == LoginRole.User)

{

var win = new UserWindow(host, sessionId);

win.Show();

currentWindow.Close();

}

}

Окно отправки сообщений выглядит следующим образом (рис. 8).

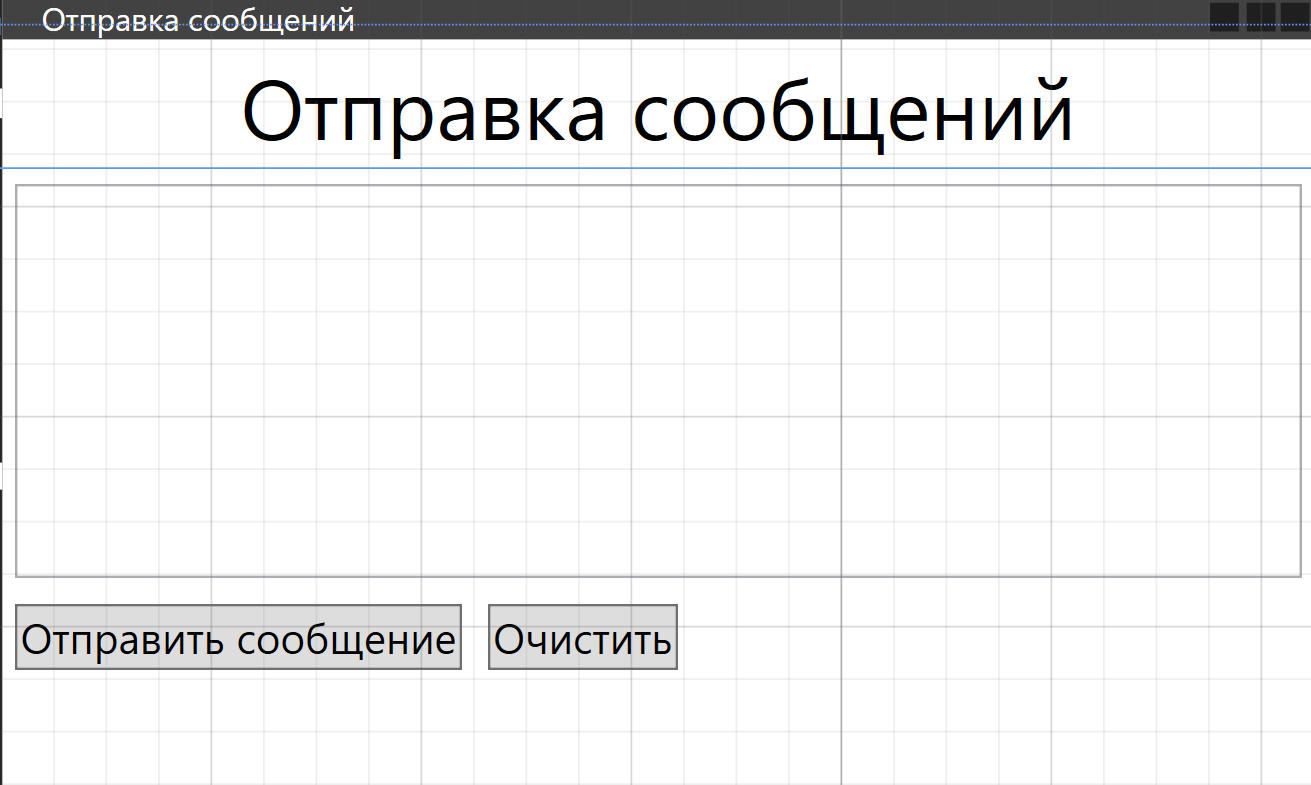


Рисунок 8 — Окно отправки сообщений

Метод отправки сообщения:

private async void btnSend\_Click(object sender, RoutedEventArgs e)

{

string message = tbMessage.Text;

if (String.IsNullOrWhiteSpace(message))

{

MessageBox.Show("Введите сообщение", "Ошибка");

return;

}

client.DefaultRequestHeaders.Accept.Clear();

client.DefaultRequestHeaders.Accept.Add(

new MediaTypeWithQualityHeaderValue("application/json"));

client.DefaultRequestHeaders.Add("User-Agent", "SchoolNotificationSystem desktop");

string result = "";

try

{

Uri uri = new Uri($"http://{host}/request?action=sendMessage&session\_id={sessionId}&text={message}");

result = await client.GetStringAsync(uri);

} catch (Exception ex)

{

MessageBox.Show("Ошибка соединения", "Ошибка");

return;

}

JObject jObject = JObject.Parse(result);

JToken tokenSuccess = jObject["success"];

JToken tokenError = jObject["error"];

string msg = "Неизвестная ошибка";

if (tokenSuccess != null)

{

msg = tokenSuccess.ToObject<string>();

}

else if (tokenError != null)

{

msg = tokenError.ToObject<string>();

}

MessageBox.Show(msg, "Ответ");

tbMessage.Text = "";

}

Окно администратора содержит две страницы: для управления сотрудниками и сообщениями (рис. 9, 10).

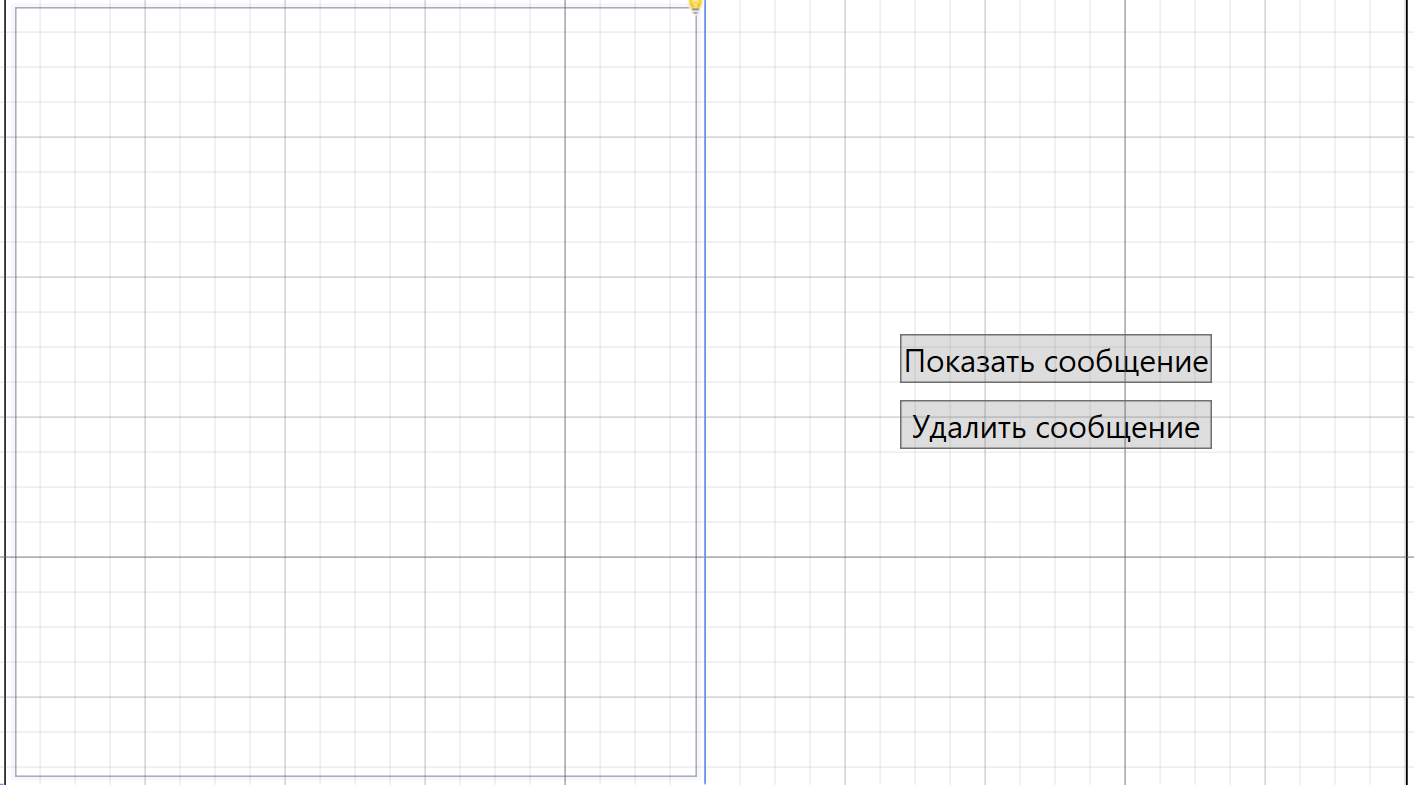


Рисунок 9 — Окно сообщений

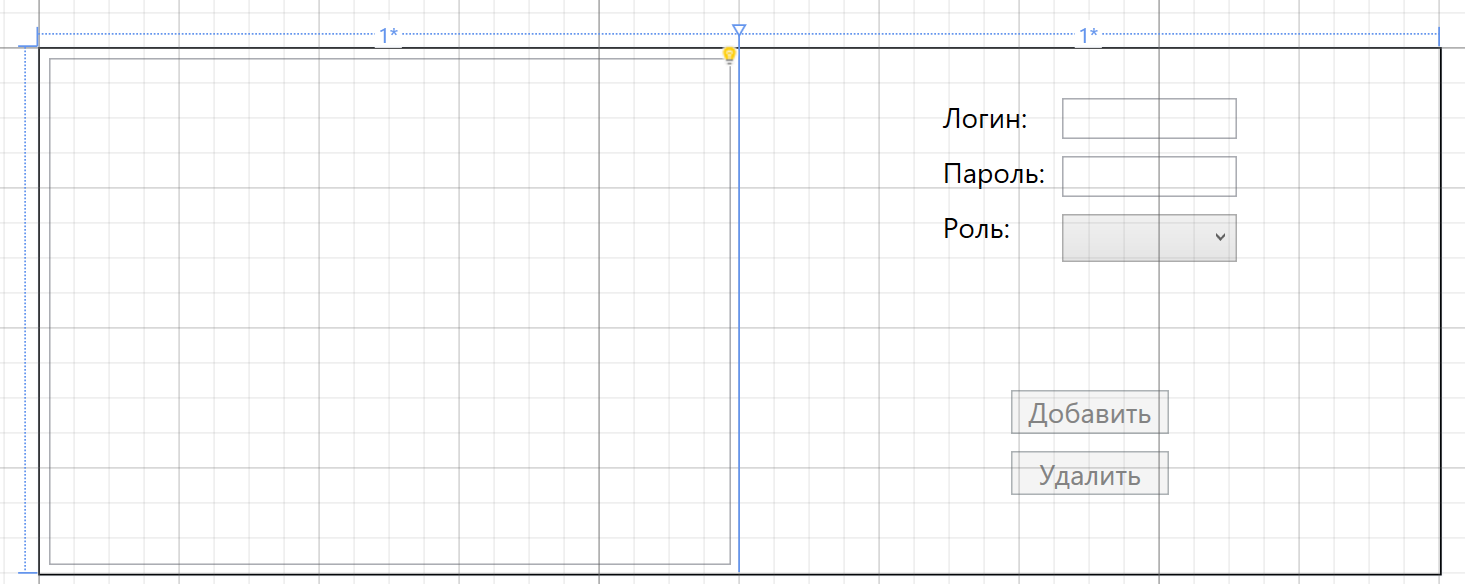


Рисунок 10 — Окно управления сотрудниками

### Разработка мобильного клиента

Мобильный клиент должен показывать список сообщений после авторизации и запоминать логин с паролем во внутренней памяти.

При запуске приложения появляется LoginActivity, в которой пользователь должен авторизоваться, чтобы просмотреть список сообщений.

Алгоритм обработчика кнопки «войти»:

1. записать данные полей в переменные;
2. обратится в SharedPreferences, чтобы записать данные на устройство;
3. отправить запрос на сервер;
4. если получен id сессии, перейти на ListActivity, иначе вывести ошибку.

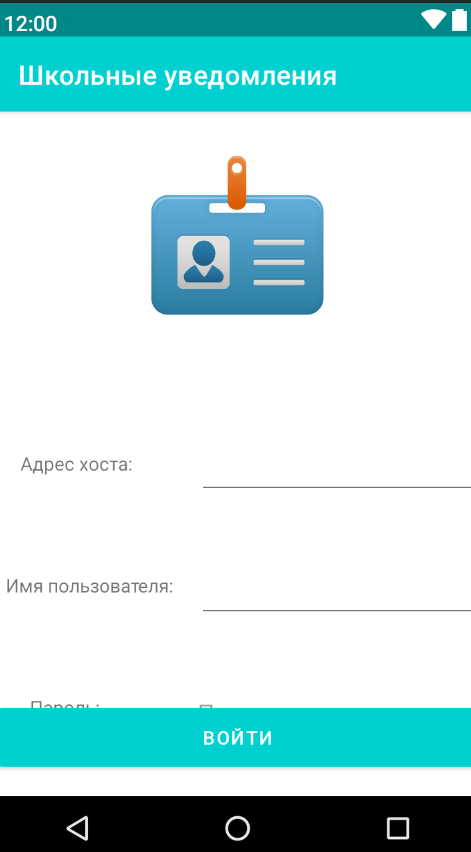


Рисунок 11 — Разметка LoginActivity

Для вывода сообщений об ошибках используется класс ErrorDialog.

public class ErrorDialog extends AppCompatDialogFragment {

private String message;

public ErrorDialog(String message){

this.message = message;

}

@Override

public Dialog onCreateDialog(Bundle savedBundle){

AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

builder.setTitle("Ошибка!")

.setMessage(message)

.setPositiveButton("Понятно", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

dialog.cancel();

}}); return builder.create()}}

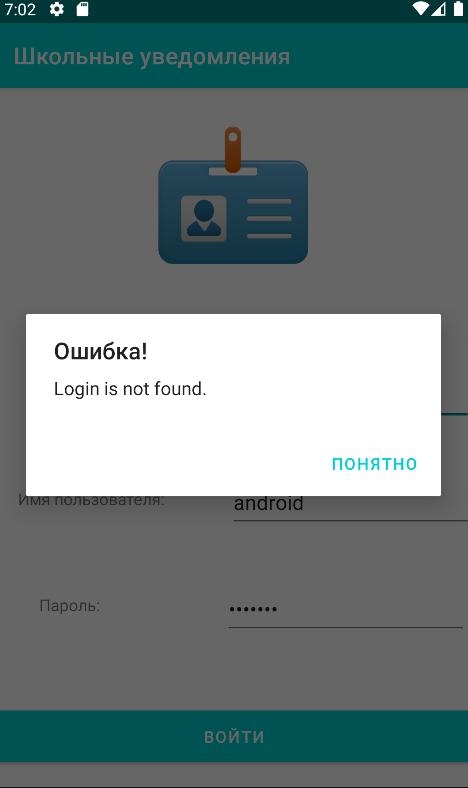


Рисунок 12 — Сообщение об ошибке

В ListActivity находится элемент ListView, в котором отображаются сообщения.

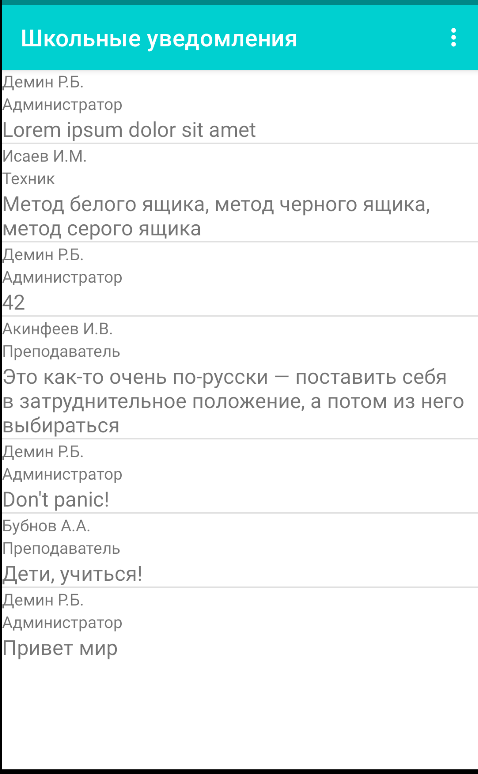


Рисунок — ListActivity

## Тестирование

### Тестирование методом белого ящика

Тестирование методом белого ящика было применено при разработке сервера для тестирования методов кэширования. Для тестирования была использована библиотека JUnit.

Тест-кейсы представлены в таблице 6.

Таблица 6 — Тестирование методом белого ящика

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Сигнатура метода | Тип теста | Краткое описание | Тестовые данные |
| 1 | String getHash(String text) | Позитивный | Проверяется метод получение хэша из строки | text=”test” |
| 2 | boolean isValid(String password, String hash) | Позитивный | Проверяется, является вторая строка, хэшем первой строки | password = "test"  String hash = "9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08" |
| 3 | boolean isValid(String password, String hash) | Негативный | Проверяется, является вторая строка, хэшем первой строки | password = "42";  String hash = "9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08" |
| 4 | boolean isValid(String password, String hash) | Позитивный | Проверяется одновременно метод getHash() и isValid | password = "test" |

Для запуска тестирования необходима создать в IntelliJ Idea новую конфигурацию и запустить её. Для этого необходимо проделать следующие шаги:

1. зайти в настройщик конфигурация (рис. 14);
2. создать конфигурацию типа «JUnit» (рис. 15);
3. выбрать класс, содержащий юнит-тесты (рис. 16);
4. сохранить изменения;
5. выбрать конфигурацию и запустить (рис. 17).

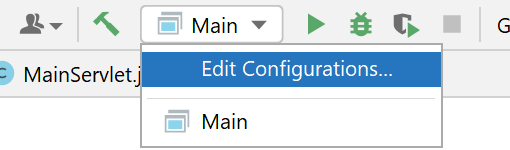


Рисунок — Вход в настройщик конфигураций

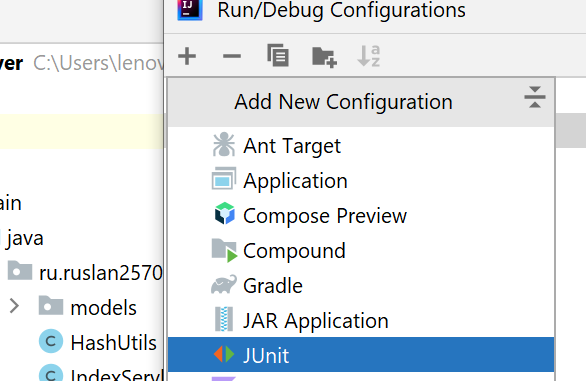


Рисунок — Создание новой конфигурации

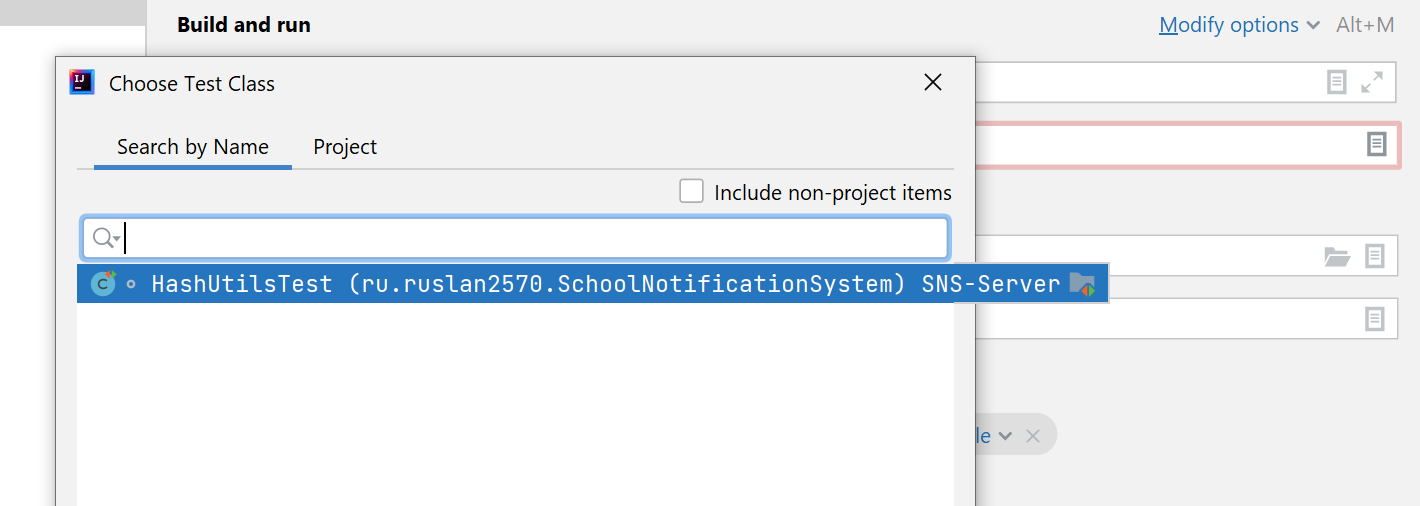


Рисунок — Выбор класса

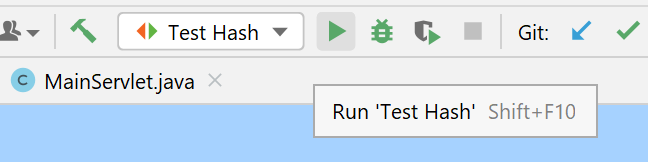


Рисунок — Запуск конфигурации

После сборки и запуска проекта появится вкладка с результатами теста (рис. 18).

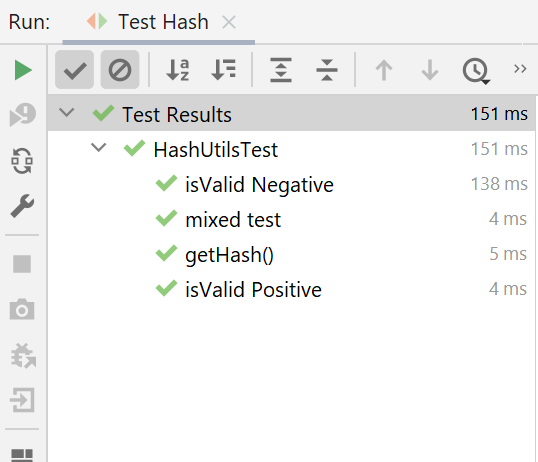


Рисунок — Результаты тестов

### Тестирование методом черного ящика

Тестирование методом черного ящика производится для настольной и мобильной части системы. Для проведения тестирования сервер должен быть уже развернут.

Таблица 7 — Тестирование настольного приложения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Идентификатор | Модуль | Подмодуль | Заглавие тест-кейса | Ожидаемый результат |
| Desktop-auth-0 | SNS-Desktop | Авторизация | Авторизация с валидными данными   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Демин Р.Б.», «42» соответственно; 2. нажать кнопку «Войти». | 1. Откроется окно авторизации |
| Desktop-auth-1 | SNS-Desktop | Авторизация | Вход в окно администрирования   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Демин Р.Б.», «42» соответственно; 2. нажать на кнопку «Администратор». | 1. Откроется окно администрирования |
| Desktop-auth-2 | SNS-Desktop | Авторизация | Авторизация с невалидными данными   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Тест», «тест» соответственно; 2. нажать кнопку «Войти». | 1. Получить ошибку «login is not found» |
| Desktop-auth-3 | SNS-Desktop | Авторизация, отправка сообщений | Проверка отправки сообщения.   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Демин Р.Б.», «42» соответственно; 2. нажать кнопку «Войти»; 3. в поле ввода текста написать «Тестирование методом черного ящика»; 4. нажать кнопку «Отправить сообщение»; 5. проверить наличие сообщения на <http://new-bokino.ru:49205/request> | 1. После отправки появится сообщение «message was send»; 2. сообщение появится на http://new-bokino.ru:49205/request. |
| Desktop-auth-4 | SNS-Desktop | Отправка сообщений | Проверка отправки сообщения пользователем, являющимся учеником   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Логвенчев П.С.», «pasha» соответственно; 2. нажать кнопку «Войти»; 3. в поле ввода текста написать «Отправка сообщения учеником»; 4. нажать кнопку «Отправить сообщение» 5. проверить наличие сообщения на <http://new-bokino.ru:49205/request>. | 1. После отправки появится ошибка «Access denied»; 2. сообщение не появится на http://new-bokino.ru:49205/request |
| Desktop-auth-5 | SNS-Desktop | Администрирование | Удаление сообщения администратором   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Демин Р.Б.», «42» соответственно; 2. нажать на кнопку «Администратор»; 3. нажать на кнопку «Сообщения»; 4. выбрать сообщения с текстом «Тестирование методом черного ящика»; 5. нажать кнопку «Удалить сообщение»; 6. проверить наличие сообщения на <http://new-bokino.ru:49205/request>. | 1. После удаления сообщения появится сообщение «Message was deleted»; 2. сообщение будет отсутствовать на <http://new-bokino.ru:49205/request>. |
| Desktop-auth-6 | SNS-Desktop | Администрирование | Добавление нового пользователя   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Демин Р.Б.», «42» соответственно; 2. нажать на кнопку «Администратор»; 3. нажать на кнопку «Пользователи»; 4. в поля логин и пароль ввести «тест», «тест»; в списке «роль» выбрать произвольный вариант; 5. нажать на кнопку «Добавить». | 1. В списке пользователей появится новый пользователь. |
| Desktop-auth-7 |  |  | Удаление пользователя   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Демин Р.Б.», «42» соответственно; 2. нажать на кнопку «Администратор»; 3. нажать на кнопку «Пользователи»; 4. выбрать пользователя с именем «тест»; 5. нажать на кнопку «Удалить». | 1. Пользователь будет удален. |

Таблица 8 — Тестирование мобильного приложения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Идентификатор | Модуль | Подмодуль | Заглавие тест-кейса | Ожидаемый результат |
| SNS-mobile-0 | SNS-mobile | Авторизация, ListActivity | Авторизация с валидными данными   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Демин Р.Б.», «42» соответственно; 2. Нажать кнопку «Войти». | 1. Откроется ListActivity, в котором будут отображаться сообщения. |
| SNS-mobile-1 | SNS-mobile | Авторизация | Авторизация с невалидными данными   1. Ввести в поля хоста, логина и пароля «new-bokino.ru:49205», «Тест», «тест» соответственно; 2. нажать кнопку «Войти». | 1. Получение ошибки «login is not found» |

## Инсталляция

### Установка серверной части

Перед установкой необходимо создать базу данных на сервере. Для этого необходимо перейти в панель PhpMyAdmin, которая располагается по адресу <https://new-bokino.ru/phpmyadmin/>. После авторизации необходимо создать базу данных «SchoolNS» для этого необходимо нажать на кнопку «Создать БД», в открывшийся странице ввести название БД и нажать на кнопку создать (рис. 19).

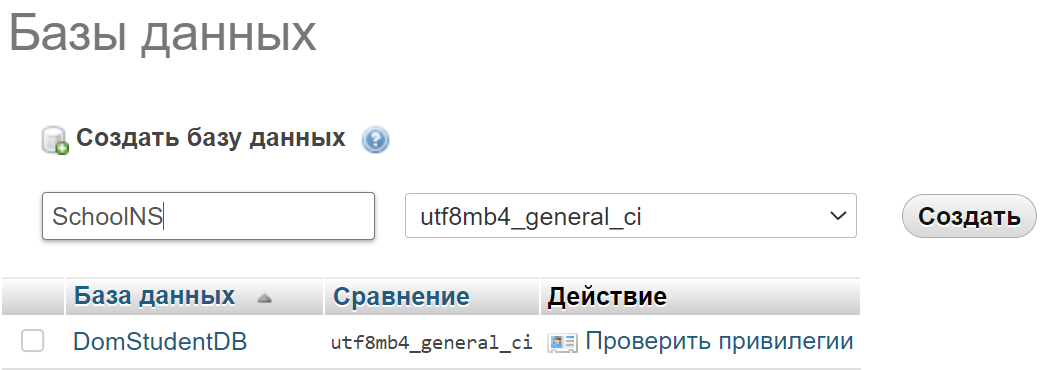


Рисунок — Создание базы данных

После этого нужно создать структуру БД, для этого нажать на кнопку «Импорт», выбрать файл, идущий в пакете, нажать на кнопку «Вперед».

После создания базы данных необходимо подключится к серверу по протоколу SSH. Для этого необходимо на компьютере открыть консоль, ввести команду ssh -p 49213 [ruslan@new-bokino.ru](mailto:ruslan@new-bokino.ru), при первом подключении ввести yes, ввести пароль.

Для получения файла .jar, необходимо ввести следующую команду:

git clone https://github.com/ruslan2570/SchoolNotificationSystem; mv SchoolNotificationSystem/App/SNS-Server.jar ./; rm -Rf SchoolNotificationSystem

Для запуска программы в фоновом режиме будет использоваться программа screen, запустим её с помощью одноименной команды.

После запустим сервер командой java -jar SNS-Server.jar jdbc:mysql://localhost/SchoolNS {Логин} {Пароль}, где {Логин} – это логин пользователя mysql, а {Пароль} – это его пароль.

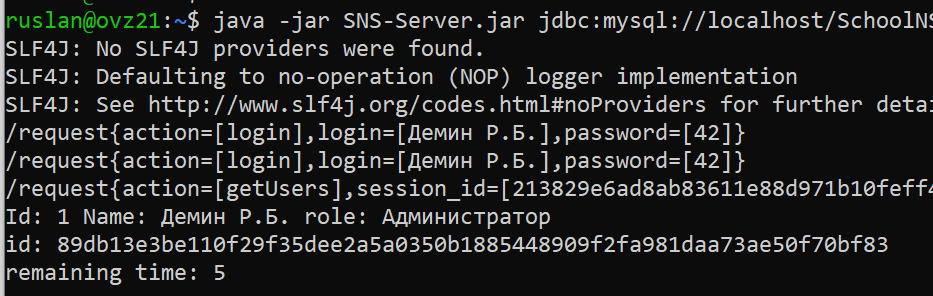


Рисунок — Запущенный сервер

### Установка настольной части

Для установки настольной части необходимо взять zip-архив desktop.zip из пакета и распаковать в любом удобном месте.

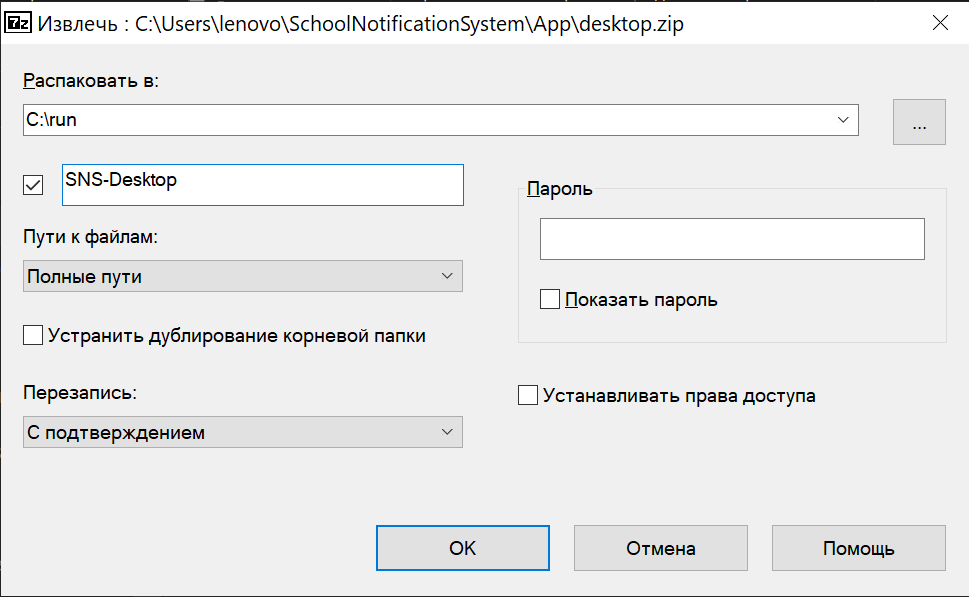


Рисунок — Распаковка программы на примере программы 7-zip

### Установка мобильной части

Для установки мобильного приложения требуется смартфон на базе ОС Android версии не младше 7.0.

Apk-файл, находящийся в пакете, копируется на устройство и устанавливается стандартными средствами системы Android.



Рисунок — Установка приложения на Android

# Обзор работы разработанного программного модуля

Система позволяет оповещать сотрудников и учеников. Данная система состоит из трёх компонентов:

* настольный клиент для отправки сообщений сотрудниками и управления администратором;
* мобильный клиент для показа сообщений;
* сервер для обработки запросов от клиентов и обращения к базе данных.

Система имеет следующую функциональность:

* авторизация пользователей;
* показ сообщений в мобильном клиенте;
* отправка сообщений в настольном клиенте;
* управление пользователями и сообщениями в настольном клиенте;

Продемонстрируем работу системы. Сервер на данный момент уже запущен.

Запускаем приложение для настольной системы и авторизуемся (рис. 23).

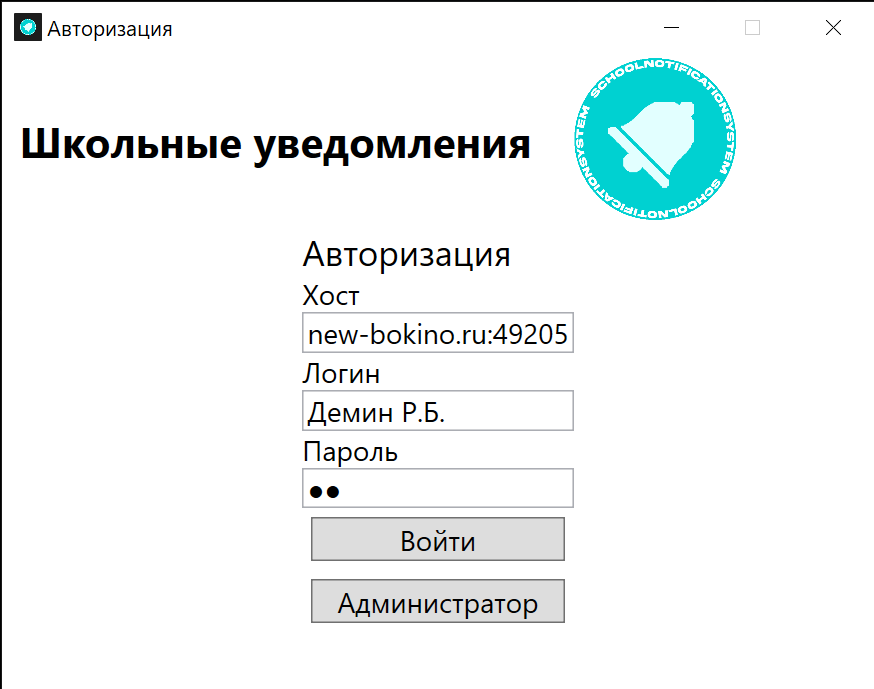


Рисунок — Авторизация в настольном приложении

Входим и отправляем сообщение (рис. 24)

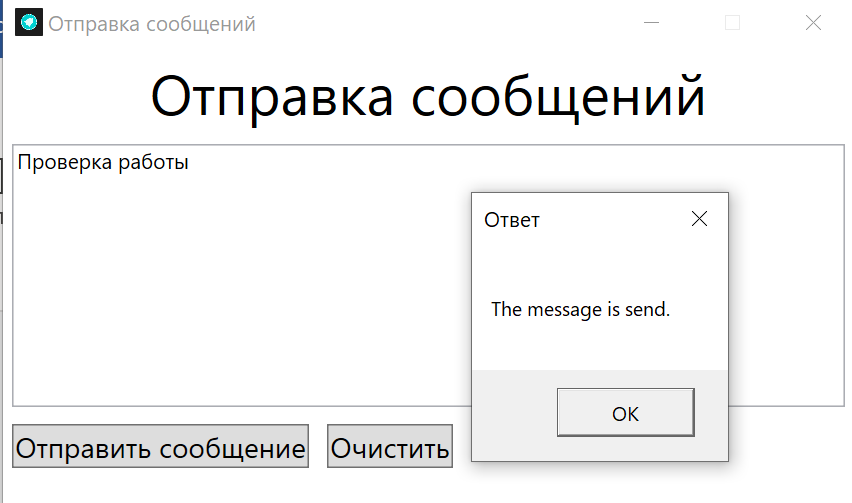


Рисунок — Отправка сообщения

Закрываем окно отправки сообщений, автоматически откроется окно авторизации с данными, которые мы ввели ранее. Теперь нажимаем на кнопку «Авторизация». Проверяем, есть ли наше сообщение в списке (рис. 25).

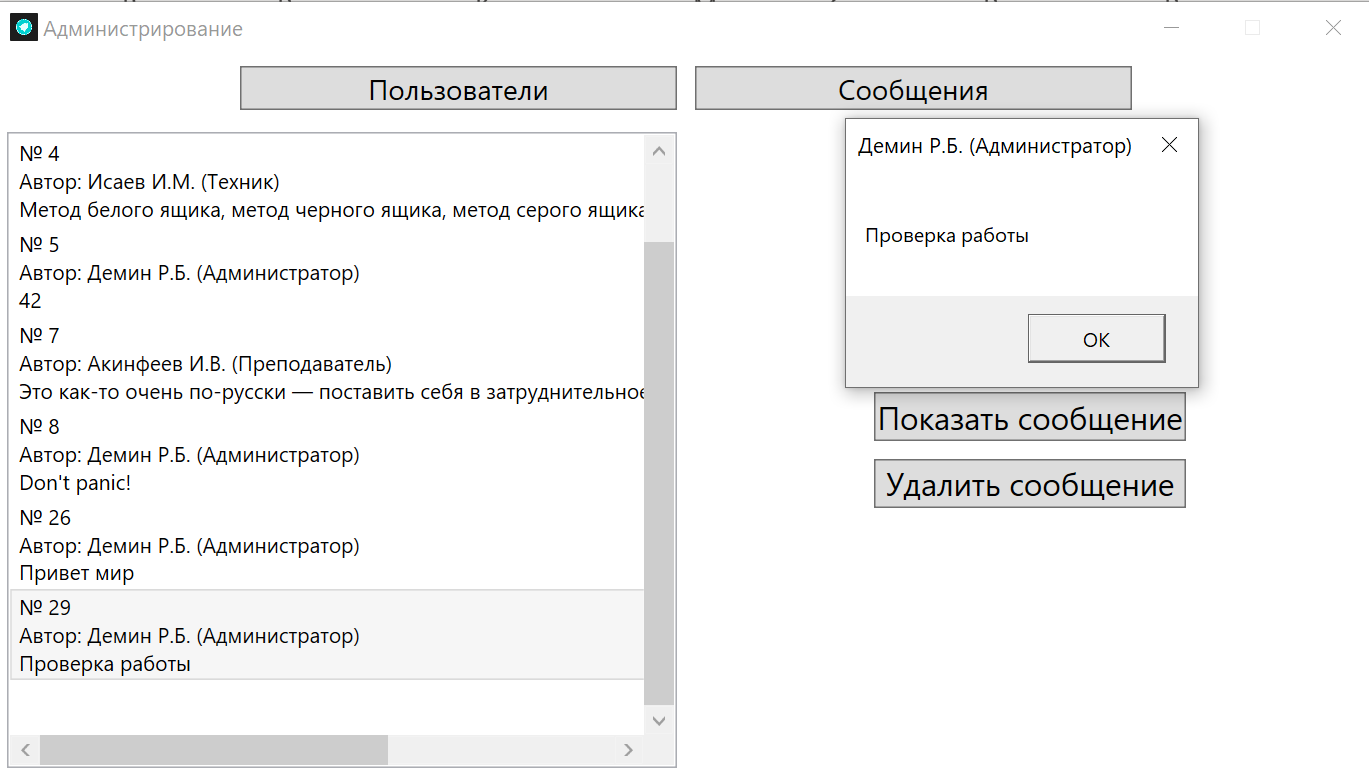


Рисунок — Окно управления сообщениями

Проверяем список пользователей (рис. 26).

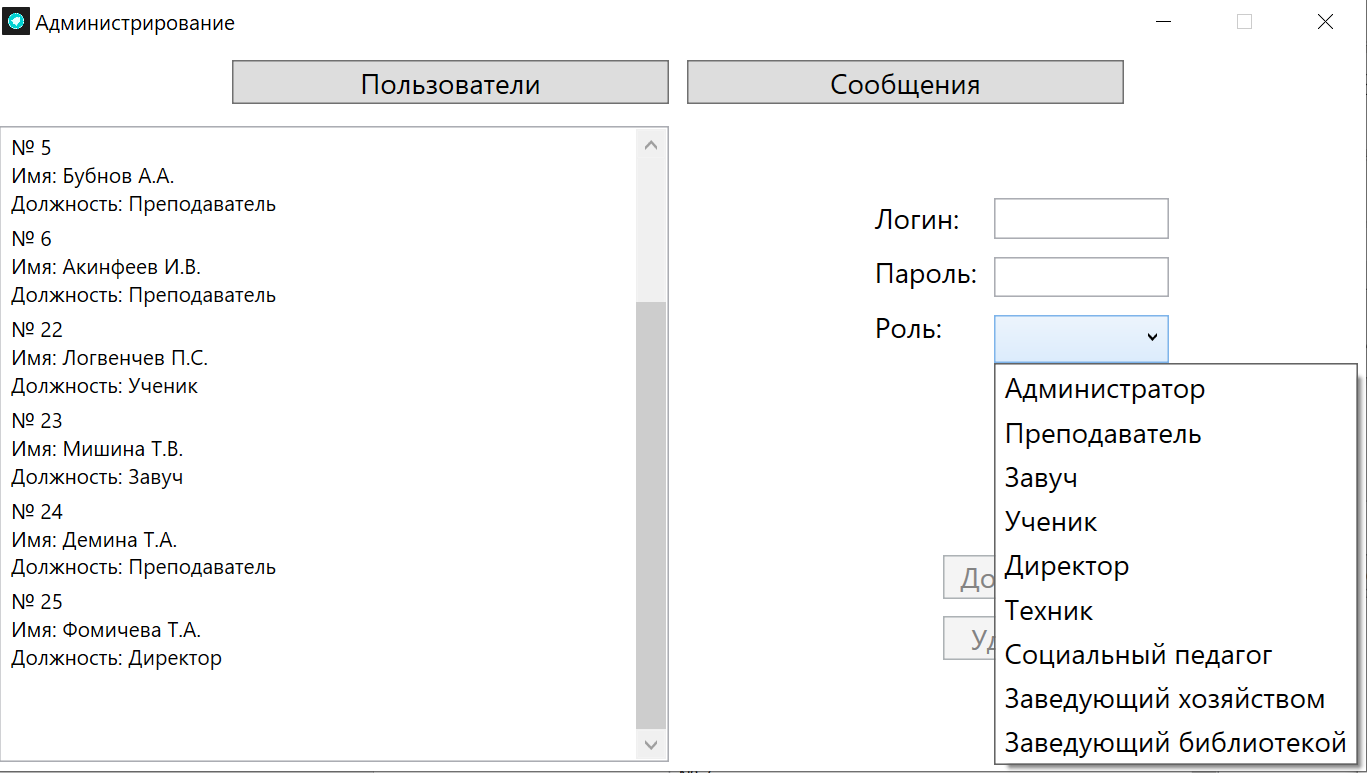


Рисунок — окно управления пользователями

Запускаем мобильное приложение и авторизуемся в нём (рис. 27)

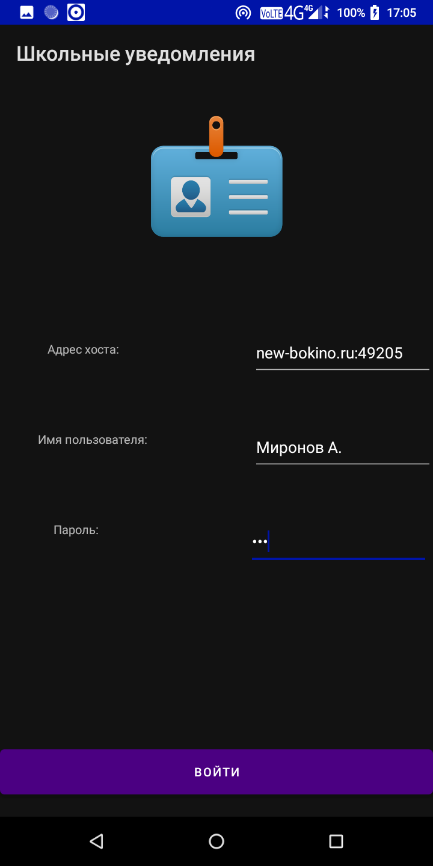


Рисунок — Авторизация в мобильном приложении

После авторизации появляется список сообщений (рис. 28).

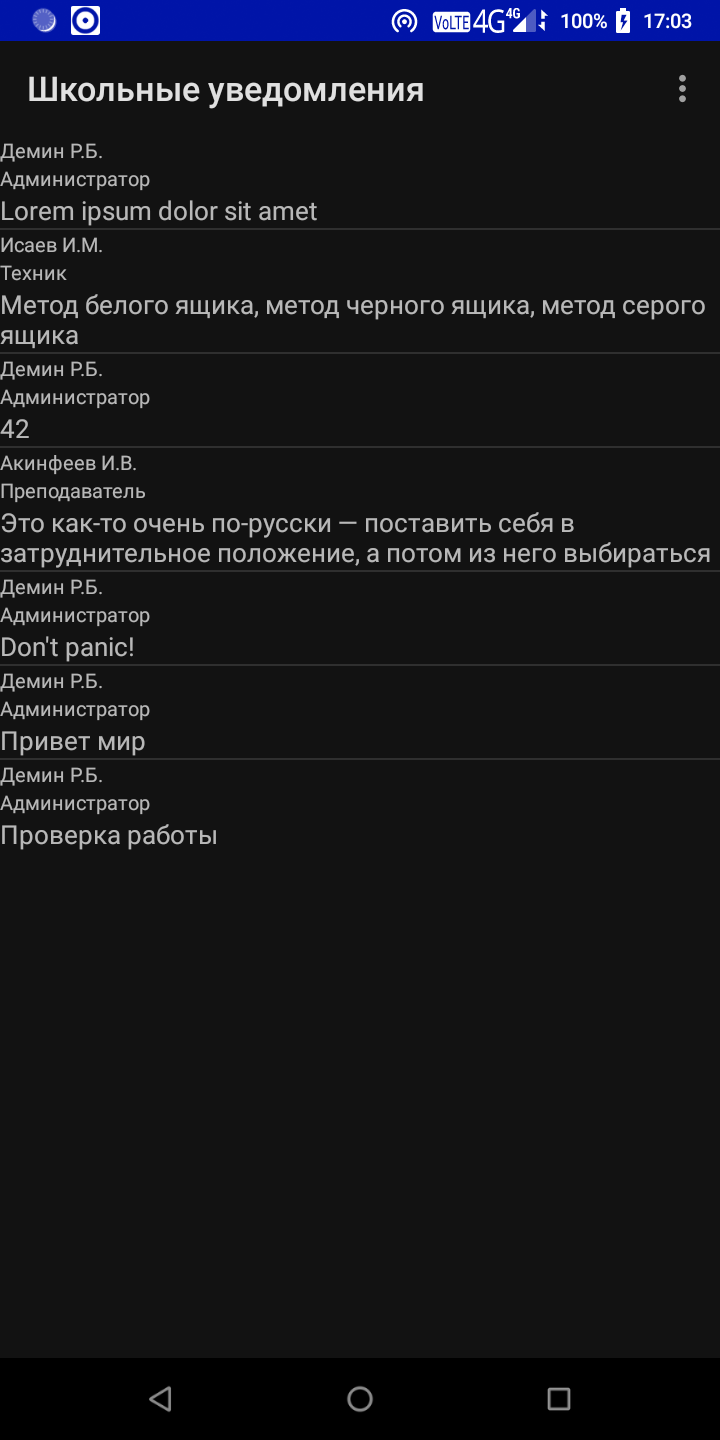


Рисунок — Список сообщений

В списке сообщений присутствует кнопка меню, в которой располагаются кнопки обновления списка сообщений и кнопка выхода в активность авторизации. (рис. 29)

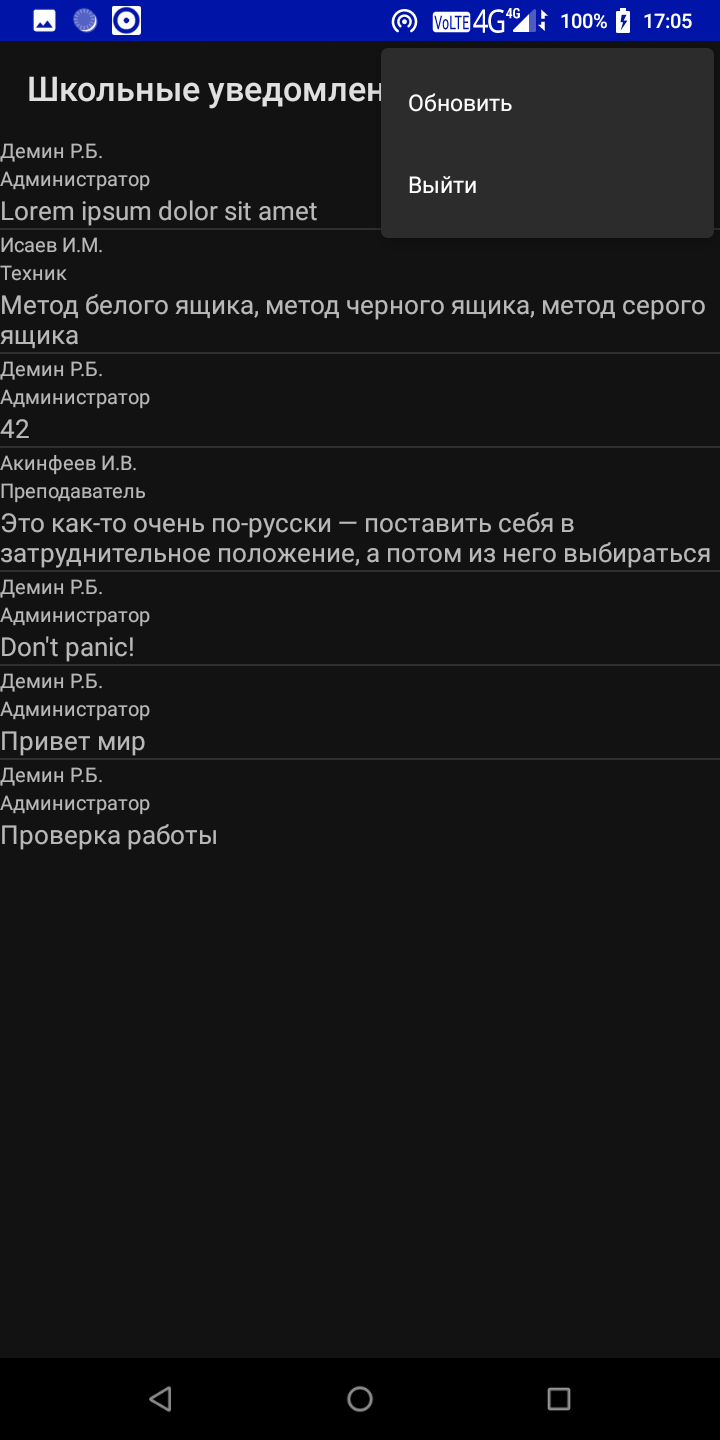


Рисунок — Меню

Тем временем на сервере логируются все запросы и каждую минуту выдаётся список активных сессий (рис. 30).

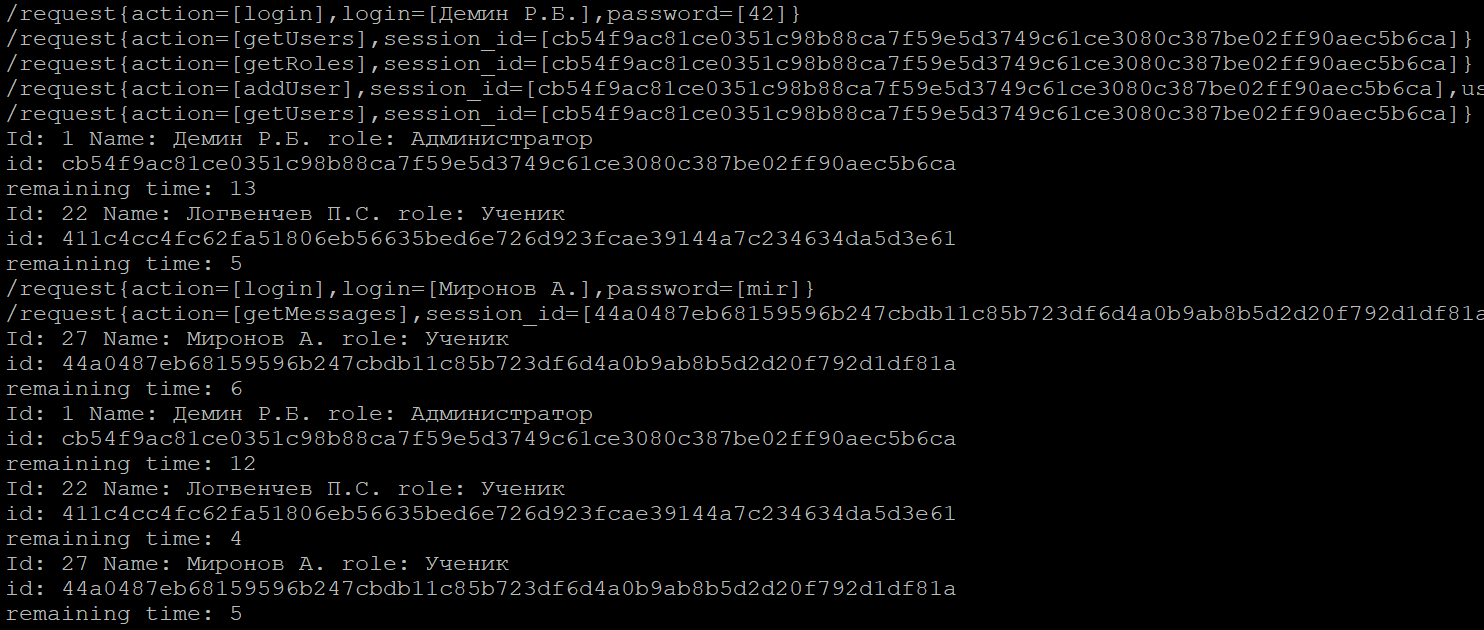


Рисунок — Логи сервера

# Заключение

За время прохождения практики я научился:

* писать серверные приложения на Java;
* работать с СУБД MariaDB через командную строку и панель PhpMyAdmin;
* работать с системой контроля версий git;
* работать с многопоточностью для совершения веб-запросов;
* сериализовывать и десериализовывать JSON.

# Список источников

1. Programming Guide // https://www.eclipse.org URL: https://www.eclipse.org/jetty/documentation/jetty-11/programming-guide/index.html#pg-eclipse-jetty-programming-guide (дата обращения: 01.06.2022);
2. Об образовании в Российской Федерации: ФЗ № 273-ФЗ от 29.12.2012 [сайт]. URL: http://минобрнауки.рф/документы/2974 (дата обращения: 28.08.2014);
3. Работа с библиотекой Newtonsoft.Json на реальном примере. Часть 1 // habr.com URL: https://habr.com/ru/post/481514/ (дата обращения: 01.06.2022).;
4. Гербер Шилдт Java. Полное руководство. - 10-e изд. - СПБ.: Диалектика, 2018. - 1488 с.;
5. Джошуа Блох Java: эффективное программирование. - 3-e изд. - СПБ.: Диалектика, 2019. - 464 с.;
6. Командная строка Linux. Полное руководство. — СПб.: Питер, 2017. — 480 с.
7. С# 4.0: полное руководство. : Пер. с англ. — М. : ООО "И.Д. Вильяме", 2011. —1056 с.;
8. Объектно-ориентированное мышление. — СПб.: Питер, 2014. — 304 с..
9. Что такое Git? // git-scm.com URL: https://git-scm.com/book/ru/v2/Введение-Что-такое-Git%3F (дата обращения: 01.06.2022).