**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра автоматизации процессов управления**

Курсовая РАБОТА

Тема: Создание клиент-серверного приложения на Java

|  |  |  |
| --- | --- | --- |
| Студенты гр. 6371 |  | Кравченко А.Н.  Шутемов А.А. |
| Преподаватель |  | Кондратьева Н.Ю. |

Санкт-Петербург

2018

Содержание

[Интерфейс программы 2](#_Toc533101512)

[Классы пользователей 6](#_Toc533101513)

[Java классы 6](#_Toc533101514)

[База данных 10](#_Toc533101515)

[Используемые сторонние библиотеки 11](#_Toc533101516)

[Примечания 11](#_Toc533101517)

[Словарь 11](#_Toc533101518)

# Интерфейс программы

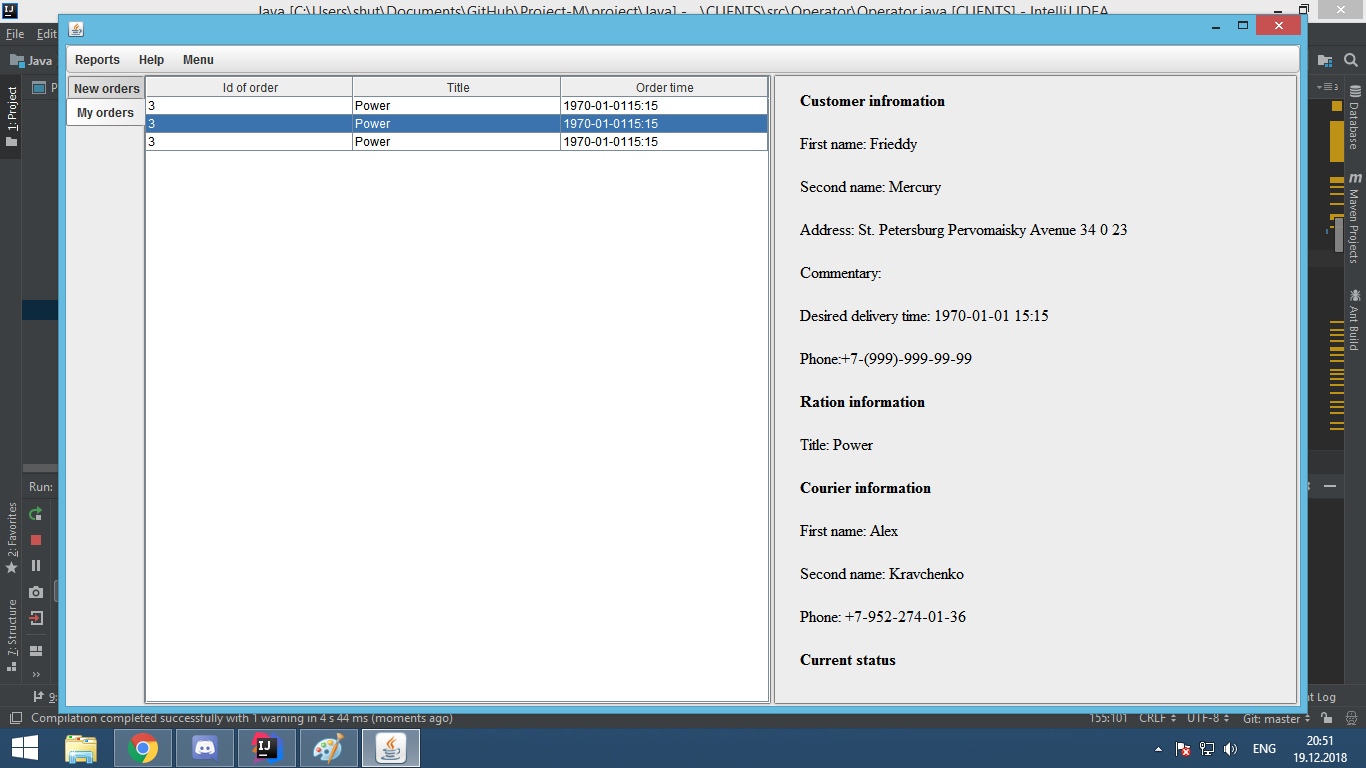
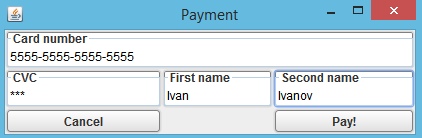
Рисунок 1. Главная страница для класса «Оператор».

Рисунок 2. Страница оплаты заказа.

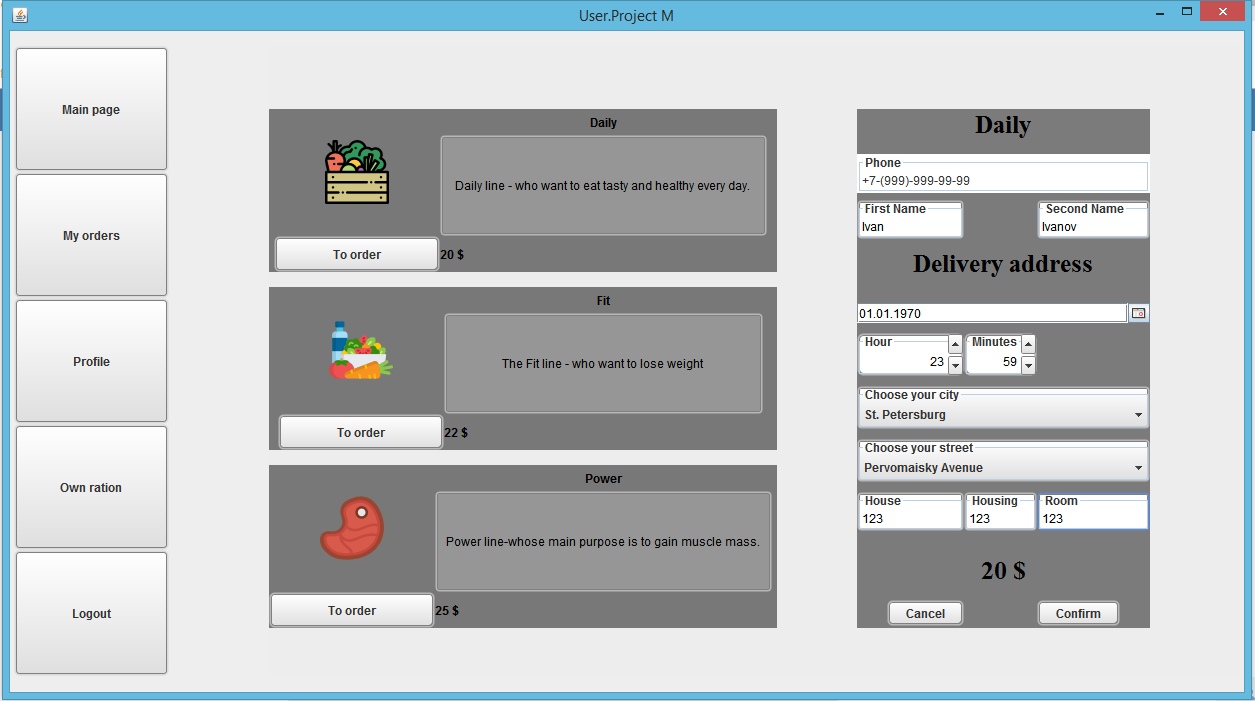
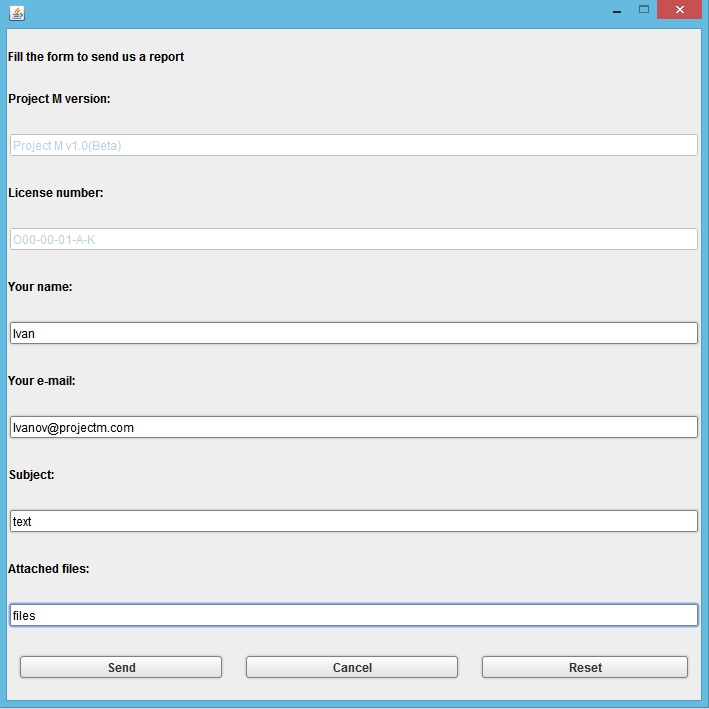
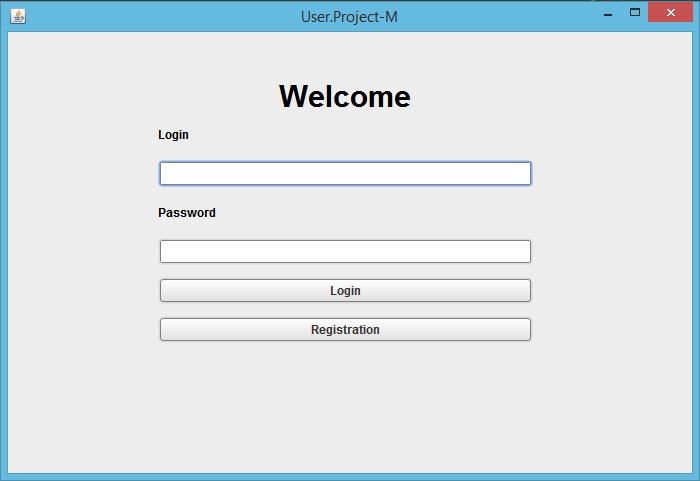
Рисунок 3. Создание отчета об ошибке.

Рисунок 4. Главная страница класса «Пользователь».

Рисунок 5. Страница авторизации.

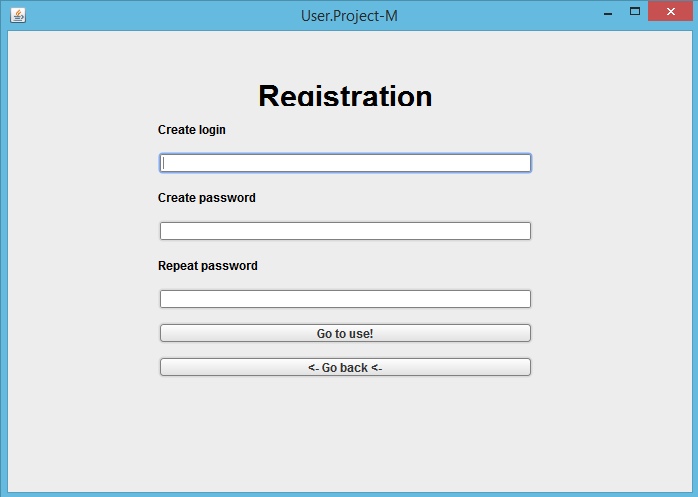


Рисунок 6. Страница регистрации.

# Классы пользователей

1. Пользователь

Класс в системе, который способен зарегистрироваться, авторизоваться и сделать заказ. Класс для запуска – CLIENTS.User.Start. Окно приложения – рисунок 1.

1. Оператор

Класс в системе, который способен авторизоваться, взять заказ и получить информацию о заказе, заказчике и курьере, доставляющем заказ. Класс для запуска – CLIENTS.Operator.Operator. Окно приложения – рисунок 4.

# Java классы

1. Сервер
   1. Abuse\_api

Этот класс реализует интерфейс Requset\_manager\_API (см. пункт 3.4). Класс выполняет запросы в базу данных. В нашем приложении мы используем ‘Select’ и ‘Insert’ запросы. Такие как:

* + 1. insertOrderRequest

Метод имеет тип Boolean. Аргументы: уникальный идентификатор заказа, дата и время, цена заказа, название рациона, полный адрес получателя, имя получателя, фамилия получателя, номер телефона получателя, уникальный логин получателя в системе. Реализует добавление переданной информации о заказе в базу данных.

* + 1. selectIdPositionForOrders

Метод имеет тип Integer. Не имеет аргументов. Реализует получение уникального идентификатора позиции в заказе.

* + 1. insertSimpleUser

Метод имеет тип Boolean. Аргументы: уникальный логин пользователя, пароль пользователя. Реализует первичное создание пользователя в базе данных для авторизации.

* + 1. selectValidationRegistration

Метод имеет тип Boolean. Аргументы: уникальный логин пользователя. Реализует проверку на уникальность логина при регистрации нового пользователя.

* + 1. selectValidationAuthorization

Метод имеет тип Boolean. Аргументы: уникальный логин пользователя, пароль пользователя. Реализует проверку на соответствие логина и пароля пользователя при авторизации.

* + 1. selectInformationRation

Метод имеет тип Ration (см. пункт 3.3). Аргументы: название рациона. Реализует получение информации о заказе из базы данных.

* + 1. selectInformationOrder

Метод имеет тип ArrayList<OrderInfoOperator> (см. пункт 3.1). Аргументов нет. Реализует передачу данных о заказе на клиент (см. пункт 1.1.1).

* + 1. selectInformationCourier

Метод имеет тип ArrayList<CourierInfoOperator> (см. пункт 3.2). Аргументов нет. Реализует передачу данных о курьере на клиент.

* + 1. selectIdPositionCourier

Метод имеет тип integer. Не имеет аргументов. Реализует получение уникального идентификатора курьера.

* + 1. insertCourier

Метод имеет тип void. Аргументы: id, имя, фамилия, телефон. Реализует добавление курьера в БД.

* + 1. insertProfileInformation

Метод имеет тип boolean. Аргументы: логин, имя, фамилия, пол, электронная почта. Реализует добавление информации о пользователе в БД.

* + 1. updateAccountInformation

Метод имеет тип void. Аргументы: логин и новый пароль. Реализует изменение пароля для учетной записи.

* + 1. insertOperator

Метод имеет тип void. Аргументы: id, имя, фамилия, телефон. Реализует добавление оператора в БД.

* + 1. selectInfoForRationPopUpLeft

Метод имеет тип String[]. Аргумент – название рациона. Реализует наполнение дерева в информации о рационе.

* + 1. selectInfoForRationPopRight

Метод имеет тип float[]. Аргумент – название рациона. Реализует получение белков, жиров, углеводов и калорий для конкретного блюда.

* + 1. updateStatus

Метод имеет тип boolean. Аргументы: идентификатор заказа, новый статус. Реализует обновление статуса заказа в БД.

* + 1. insertReports

Метод имеет тип Boolean. Аргументы: идентификатор, отправитель отчета, тема отчета, дата отправления. Реализует добавление отчета в БД.

1. Клиент
   1. User
      1. Cart
         1. CartMarkup

Данный класс включает в себя разметку данных корзины на главной странице пользователя. Создается по конструктору. Если нужны изменения данных, то после нажатия на кнопку «To order» сеттером задаются информационные лейблы. В классе присутствуют геттеры для получения данных о рационе, дате доставки, адресе доставки, имени и фамилии пользователя, номере телефона пользователя.

* + - 1. Intermediate\_Picture

Данный класс реализует вставку картинки для рациона на главной странице пользователя.

* + 1. Pop\_up\_windows
       1. Payment\_Pop\_Up

Данный класс создает всплывающее окно для оплаты заказа при нажатии на кнопку “Confirm” (см. рисунок 4) Присутствующие поля: номер карты, имя и фамилия держателя, срок окончания действия, код CVV/CVC.

* + 1. RationMurkup
       1. RationDataFromDatabase

Данный класс реализует получение и передачу данных о рационе из базы данных.

* + - 1. Rations

Данный класс реализует заполнение информации о рационе на клиенте. В классе «Пользователь».

* + 1. MainFormMurkup

Данный класс реализует создание левого меню навигации пользователя.

* + 1. MainPageContentMurkup

Данный класс реализует методы сет для заполнения главной страницы пользователя из классов Rations (см. пункт 2.1.3.2) и CartMarkup (см. пункт 2.1.1.1).

* + 1. Start

Данный класс реализует создания окна приложения для класса «Пользователь». Содержит в себе авторизацию, регистрацию, главное меню.

* 1. Operator
     1. Content

Данный класс реализует геттеры данных о заказе, такие как: имя и фамилия получателя заказа, телефон, адрес доставки, название рациона, время доставки и статус заказа.

* + 1. InProgress

Данный класс реализует создание всплывающего диалогового окна, который информирует о том, что страница находится в разработке.

* + 1. MenuBar

Данный класс реализует создание меню, расположенного в верхней части окна (см. рисунок 1). Включает в себя такие разделы как: «Отчеты», «Помощь» и «Меню». В разделе «Помощь» пользователь может просмотреть документацию по системе и составить отчет об ошибке (см. пункт 2.2.7). В разделе «Меню» пользователь может закрыть программу (“Exit”) или выйти из учетной записи (“Logout”).

* + 1. Operator

Данный класс реализует создание главного окна для класса «Оператор» (см. рисунок 1). В данном классе выполняются запросы по заполнению таблицы и правой панели.

* + 1. SendReport

Данный класс реализует создание окна для отправки отчета об ошибке (см. рисунок 3).

* + 1. Reports window

Данный класс реализует создание окна для оператора, где видны отправленные им отчеты, и где он может создавать новые.

* 1. Authorization\_and\_Registration
     1. Authorization

Данный класс реализует создание окна авторизации пользователя (см. рисунок 5).

* + 1. Registration

Данный класс реализует создание окна регистрации пользователя (см. рисунок 6).

1. API
   1. CourierInfoOperator

Данный класс реализует создание геттеров и сеттеров для информации об операторе. Используемые поля: уникальный идентификатор оператор, его имя, фамилия и номер телефона.

* 1. OrderInfoOperator

Данный класс реализует создание геттеров и сеттеров для информации о заказе и заказчике. Используемые поля: уникальный идентификатор заказа, дата доставки, адрес доставки, название рациона, уникальный логин пользователя (заказчика), имя и фамилия заказчика, номер телефона заказчика.

* 1. Ration

Данный класс реализует создание геттеров и сеттеров для информации о рационе. Используемые поля: название рациона, описание рациона и его стоимость.

* 1. Request\_manager\_API

Является интерфейсом. Определяет такие методы как insertOrderRequest, selectIdPositionForOrders, insertSimpleUser, selectValidationRegistration, selectValidatioAuthorization, selectInformationRation, selectInformationOrder, selectInformationCourier.

1. AddingOperators
   1. WindowAddingOperator

Данный класс реализует создание окна для добавления нового оператора в БД.

* 1. Operator

Данный класс реализует подключение к серверу для обмена данными.

# Используемые сторонние библиотеки

1. hessian 4.0.51
2. jcalendar 1.4
3. weblaf complete 1.29
4. postgresql 42.2.5
5. servlet-api 2.2

# Примечания

1. Весь код программы доступен по ссылке: <https://github.com/darkeus86/project-m>
2. Структура баз данных представлена в виде скриптов. Путь - \Project-M\project\Java\src\Scripts

# Словарь

1. Геттер – метод в программировании для получения значения поля
2. Сеттер – метод в программировании для установки значения поля
3. Лейбл – элемент в Swing, представляющий из себя текстовое поле.