3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

При разработке системы используется объектно-ориентированный и функциональный подход программирования. Чтобы понимать технические аспекты системы, следует описать составляющие её классы и методы. Диаграмма классов приведена на чертеже ГУИР.400201.089 РР.1.

3.1 Блок обработки запросов клиента

3.1.1 Класс ProductListPageController

Класс `ProductListPageController` представляет из себя точку входа запроса клиента на список товаров, представленных в магазине.

Класс `ProductListPageController` содержит следующие поля:

– `phoneServiceImpl: PhoneService` – экземпляр класса `PhoneService` из блока сервис, отвечающий за работу с мобильными устройствами;

– `cartService : CartService` – экземпляр класса `CartService` из блока сервис, отвечающий за управление корзиной покупателей.

– `httpSession: HttpSession` – экземпляр класса `HttpSession`, который предоставляет инструменты для работы с HTTP-сессией.

Класс `ProductListPageController` содержит метод:

– `showProductList(String, String, String, Long, Model) : String` – принимает запрос от пользователя для отображения списка товаров, управляет сортировкой элементов (по различным полям и направлениям), задаёт параметры для поиска.

3.1.2 Класс AjaxCartController

Класс `AjaxCartController` представляет из себя обработчик запросов клиента на добавление товаров в корзину.

Класс `AjaxCartController` содержит следующие поля:

– `quantityValidator: PhoneDtoValidator` – экземпляр класса `PhoneDtoValidator` из блока валидации запросов, отвечающий за валидацию запроса;

– `quantityValidator: CartService` – экземпляр класса `CartServiceImpl` из блока сервис, отвечающий за управление корзиной покупателей;

– `httpSession: HttpSession` – экземпляр класса `HttpSession`, который предоставляет инструменты для работы с HTTP-сессией.

Класс `AjaxCartController` содержит метод:

– addPhone(PhoneDto, BindingResult) : ResponseEntity<?> – принимает запрос от пользователя для добавления товара в корзину, вызывает функции для валидации, отправляет ответ, сигнализирующий об результате действия.

3.1.3 Класс CartPageController

Класс CartPageController представляет из себя обработчик запросов клиента на получение информации о корзине и запросов на её редактирование.

Класс CartPageController содержит следующие поля:

– REDIRECT_TO_CART_PAGE: String – строка, указывающая URL для перехода на страницу с информацией о корзине;

– phoneServiceImpl: PhoneService – экземпляр класса PhoneService из блока сервис, отвечающий за работу с мобильными устройствами;

– cartService: CartService – экземпляр класса CartService из блока сервис, отвечающий за управление корзиной покупателей;

– httpSession: HttpSession – экземпляр класса HttpSession, который предоставляет инструменты для работы с HTTP-сессией;

– phoneArrayDtoValidator: PhoneArrayDtoValidator – экземпляр класса PhoneArrayDtoValidator из блока валидации, отвечающий за валидацию запросов по изменению корзины.

Класс CartPageController содержит следующие методы:

– getCart(boolean, boolean, boolean, boolean, List<Long>, Model) : String – принимает запрос от пользователя для отображения информации о корзине;

– updateCart(PhoneArrayDto, Model, BindingResult): String – принимает запрос от пользователя для обновления информации в корзине;

– deleteFromCart(Long, Model) : String – принимает запрос от пользователя для удаления продукта из корзины;

– prepareModelForEmptyCart(Cart,Model) : String – служебный метод, для создания ответа, при действиях с пустой корзиной;

– validationFailed(Cart, BindingResult, Model) : String – служебный метод, для создания ответа, при ошибках при валидации запроса.

3.1.4 Класс OrderOverviewPageController

Класс OrderOverviewPageController представляет из себя обработчик запросов администратора на получение информации о заказах и запросов на их редактирование.

Класс OrderOverviewPageController содержит поле:

– orderServiceImpl: OrderService – экземпляр класса OrderServiceImpl из блока сервис, отвечающий за работу с заказами клиентов.

Класс OrderOverviewPageController содержит следующие методы:

– getOrderOverview(Long, Model) : String – принимает запрос от пользователя для отображения информации о заказе;

– changeOrderStatus(Long, OrderStatus): String – принимает запрос от пользователя для обновления статуса заказа;

– handleOutOfStock() : String – обрабатывает возникшие в классе исключения, и отправляет ответ на перенаправление пользователя на нужную страницу.

3.1.5 Интерфейс PhoneshopPages

Интерфейс PhoneshopPages представляет своеобразное хранилище имён веб-страниц, на которые перенаправляются запросы.

Интерфейс PhoneshopPages содержит следующие внутренние интерфейсы:

– AdminPages: содержит имена веб страниц для администраторов (OrdersPage, OrderOverviewPageAdmin);

– UserPages: содержит имена веб страниц для администраторов (HotPricesPage, LoginPage, CartPage, OrderOverviewPage, OrderPage, ProductDetailsPage, ProductListPage, QuickOrderPage).

3.2 Блок моделей

3.2.1 Класс Phone

Класс Phone представляет описание сущности мобильного устройства

Класс Phone содержит следующие поля:

– id: Long – уникальный идентификатор устройства;

– brand: String – брэнд-производитель устройства;

– model: String – модель устройства;

– price: BigDecimal – цифровое значение цены устройства;

– discountPercent: Integer – число, описывающее процент скидки;

– actualPrice: BigDecimal – цена устройства с учётом скидки;

– displaySizeInches: BigDecimal – размер экрана устройства в пикселях;

– weightGr: Integer – вес устройства в граммах;

– lengthMm: BigDecimal – длина устройства;

– widthMm: BigDecimal – ширина устройства;

– heightMm: BigDecimal – высота устройства;
– announced: Date – дата презентации устройства;
– deviceType: String – тип устройства (телефон, планшет);
– os: String – операционная система, установленная на устройстве;
– colors: Set<Colors> – цветовая гамма телефонов;
– displayResolution: String – разрешение экрана устройства;
– pixelDensity: Integer – плотность пикселей устройства;
– displayTechnology: String – технология экрана устройства;
– backCameraMegapixels: BigDecimal – мегапиксели задней камеры;
– frontCameraMegapixels: BigDecimal – мегапиксели передней камеры;
– ramGb: BigDecimal – объём оперативной памяти устройства;
– internalStorageGb: BigDecimal – объём внутренней памяти;
– batteryCapacityMah: Integer – мощность аккумуляторной батареи;
– talkTimeHours: BigDecimal – время разговора;
– standByTimeHours: BigDecimal – время работы в режиме ожидания;
– bluetooth: String – технология bluetooth;
– positioning: String – позиционирование устройства;
– imageUrl: String – ссылка на изображение;
– description: String – описание устройства.

Класс Phone содержит методы для получения и установки значений в объектах класса (геттеры и сеттеры). Также в классе описаны методы equals и hashCode, используемые при сравнении устройств.

3.2.2 Класс CartItem

Класс CartItem описывает один из элементов, находящийся в пользовательской корзине.

Класс CartItem содержит следующие поля:

– phone: Phone – экземпляр класса Phone из блока моделей, представляющий устройство;
– quantity: Long – обозначает количество телефонов в данном элементе;
– price: BigDecimal – обозначает общую цену элемента.

Класс OrderOverviewPageController содержит следующие методы:

– CartItem(Phone, Long, BigDecimal)– конструктор для инициализации полей экземпляра класса;
– getPhone(): Phone – получение значения поля phone;

– getQuantity(): Long – получение значения поля quantity;
– getPrice (): BigDecimal – получение значения поля price;
– setPhone(Phone): void – установка значения в поле phone;
– setQuantity(Long): void – установка значения в поле quantity;
– setPrice(BigDecimal): void – установка значения в поле price;

3.2.3 Перечисление OrderStatus

Перечисление OrderStatus отображает четыре статуса обработки заказа:
– NEW – новый заказ;
– REJECTED – заказ отменен;
– INDELIVERY – заказ доставляется;
– DELIVERED – заказ доставлен.

3.2.4 Перечисление SortField

Перечисление SortField обозначает поля, доступные для сортировки:
– BRAND – сортировка по брэнду;
– MODEL – сортировка по модели;
– DISPLAYSIZEINCHES – сортировка по размеру заказа;
– PRICE – сортировка по цене.

3.2.5 Перечисление SortField

Перечисление SortField обозначает поля, доступные для сортировки:
– ASC – сортировка по возрастанию;
– DESC – сортировка по убыванию.

3.3 Блок работы с СУБД

3.3.1 Класс JdbcColorDao

Класс JdbcColorDao представляет из себя набор утилит для осуществления SQL-запросов для получения информации о цветах телефонов.

Класс JdbcColorDao содержит следующие поля:

– jdbcTemplate: JdbcTemplate – экземпляр класса JdbcTemplate служащего для отправки SQL-запросов и упаковки ответов в объекты;

– SQL_GET_COLOR: String – SQL-запрос для получения цвета по идентификатору;

– SQL_UPDATE_COLOR: String – SQL-запрос для сохранения цвета в базе данных.

Класс JdbcColorDao содержит методы:

– save(Color) : void – выполняет sql-запрос для сохранения цвета в базе данных;

– get(Long) : Optional<Color> – выполняет sql-запрос для получения информации о цвете из базы данных.

3.3.2 Класс JdbcPhoneDao

Класс JdbcColorDao представляет из себя набор утилит для осуществления SQL-запросов для получения информации о телефонах.

Класс JdbcColorDao содержит следующие поля:

– jdbcTemplate: JdbcTemplate – экземпляр класса JdbcTemplate служащего для отправки SQL-запросов и упаковки ответов в объекты;

– namedParameterjdbcTemplate: JdbcTemplate – экземпляр класса NamedParameterJdbcTemplate служащего для отправки SQL-запросов и упаковки ответов в объекты;

– phoneResultSetExtractor: PhoneResultSetExtractor – экземпляр класса PhoneResultSetExtractor служащий для преобразования ответов в java-объекты;

– jdbcColorDao: JdbcColorDao – экземпляр класса JdbcColorDao служащий для получения информации о цветах телефонов;

– SQL_INSERT_PHONE: String – SQL-запрос для сохранения устройства в базе данных;

– SQL_GET_PHONE: String – SQL-запрос для получения устройства из базы данных;

– SQL_GET_ALL_PHONES: String – SQL-запрос для получения устройств из базы данных;

– SQL_GET_ALL_PHONES: String – SQL-запрос для получения устройств со скидкой из базы данных;

– SQL_GET_COUNT_PHONES: String – SQL-запрос для получения количества устройств из базы данных;

– SQL_WHERE_SEARCH: String – дополнительные условия для запросов описанных выше.

Класс JdbcColorDao содержит методы:

– get(Long) : Optional<Phone> – выполняет sql-запрос для получения устройства из базы данных по уникальному идентификатору;

– get(String) : Optional<Phone> – выполняет sql-запрос для сохранения цвета в базе данных;

– getPhoneAndColors(String) : Optional<Phone> – объединяет результат запросов по получению телефонов и цветов к ним;

– save(Phone) : void – сохраняет устройство в базе данных;

– `findAll(int, int) : List<Phone>` – поиск телефонов со смещением;

– `findAll(ParamsForSearch) : List<Phone>` – поиск телефонов с параметрами поиска;

– `count(ParamsForSearch) : Long` – подсчёт количества телефонов;

– `findMaxDiscountPercentPhones(int, int) : List<Phone>` – поиск телефонов со скидкой, отсортированной по убыванию.

3.3.2 Класс JdbcStockDao

Класс `JdbcStockDao` представляет из себя набор утилит для осуществления SQL-запросов для получения информации о наличии устройст в базе данных.

Класс `JdbcStockDao` содержит следующие поля:

– `jdbcTemplate: JdbcTemplate` – экземпляр класса `JdbcTemplate` служащего для отправки SQL-запросов и упаковки ответов в объекты;

– `SQL_GET_STOCK`: String – SQL-запрос для получения цвета по идентификатору устройства;

– `SQL_UPDATE`: `String` – SQL-запрос для обновления информации об наличии в базе данных.

Класс `JdbcStockDao` содержит методы:

– `update(Long, Long, Long) : void` – выполняет sql-запрос для сохранения наличных устройств в базе данных;

– `get(Long) : Optional<Stock>` – выполняет sql-запрос для получения информации о наличии устройств из базы данных.