**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В. И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра информационных систем**

**КУРСОВАЯ РАБОТА**

**по дисциплине «Архитектура информационных систем»**

**Тема: Отдел питания санатория**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 1374 |  | Дюков Н. В.  Зырянов В. М.  Чистополов С. А. |
| Преподаватель |  | Водяхо А. И. |

Санкт-Петербург

2023

**ЗАДАНИЕ**

**НА КУРСОВУЮ РАБОТУ**

**Студенты:** Дюков Н. В., Зырянов В.М., Чистополов С. А.

**Группа:** 1374

**Тема работы:** проектирование сервиса отдела питания санатория.

**Исходные данные:** средствами ПО Enterprise Architect спроектировать сервис для отдела питания санатория. Сформировать технические требования, архитектурное описание и список тестов для проекта.

**Содержание пояснительной записки:** введение, требования, архитектурное описание, архитектурное обоснование, модели, UML описание, use case, классы, активности, развертка, тестирование, заключение, список использованных источников.

**Предполагаемый объем пояснительной записки:** не менее 25 страниц

**Дата выдачи задания:** 14.02.2023

**Дата сдачи отчёта:**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 1374 |  | Дюков Н. В.  Зырянов В. М.  Чистополов С. А. |
| Преподаватель |  | Водяхо А. И. |

**АННОТАЦИЯ**

Содержание курсовой работы заключается в проектировании сервиса отдела питания санатория ПО Enterprise Architect. Были сформированы технические требования, архитектурное описание и список тестов для проекта.

**SUMMARY**

The content of the course work consists in the implementation of the website of an online clothing store by means of the Enterprise Architect software. Technical requirements, an architectural description, and a list of tests for the project were formed.

**содержание**

|  |  |  |
| --- | --- | --- |
|  | Введение | 5 |
| 1. | Требования | 6 |
| 1.1. | Глоссарий | 6 |
| 1.2. | Выделение и фиксация бизнес-требования функциональности системы | 8 |
| 1.3  1.4  1.5 | Пользовательский требования  Системные требования  Функциональные требования | 8  9  9 |
| 2. | Архитектурное описание | 10 |
| 2.1. | Архитектурное обоснование | 10 |
| 2.2. | Модели | 11 |
| 2.3. | UML описание | 12 |
|  | 2.3.1 Use case | 12 |
|  | 2.3.2 Диаграмма классов | 24 |
|  | 2.3.3 Диаграмма активности | 27 |
|  | 2.3.4 Диаграмма последовательности | 29 |
| 3. | Тестирование | 31 |
|  | Заключение | 32 |
|  | Список используемых источников | 33 |
|  |
|  |

**введение**

**Цель работы**: проектирование сервиса для отдела питания санатория.

**Форма выполнения работы:** курсовая работа.

**Задание:** База данных предназначена для отдела питания санатория. В ней хранится информация о блюдах, продуктах, из которых блюда состоят, наличии продуктов на складе, а также о меню санатория на каждый день (план питания). Для каждого блюда хранится: наименование, содержание жиров, белков и углеводов, выход (вес порции), калорийность, набор продуктов, которые входят в состав, и необходимое их количество. По каждому продукту фиксируется следующая информация: наименование, единица измерения, цена. Меню составляется на один день и состоит из списка блюд и количества порций каждого блюда. В отделе питания санатория может быть несколько складов. Поступление продуктов на склад фиксируется в приходных накладных. Каждая приходная накладная включает следующие атрибуты: номер документа, дата документа, склад на который были завезены продукты, контрагент, наименование продуктов, количество каждого продукта, цена продукта и сумма по каждому продукту и общая.

**1.Требования**

**1.1. Глоссарий**

Термины в глоссарии указаны в алфавитном порядке для удобства.

**База данных** - набор постоянно хранимых данных, используемых прикладными программными системами.

**Белки** — это главный строительный материал для человеческого организма. Они находятся в мышцах, костях, коже, тканях и даже волосах.

**Блюдо** — комбинация доведённых до готовности пищевых продуктов, которая кулинарно оформлена в виде порции и готова к употреблению.

**Диета** – система питания, способствующая лечению и профилактики различных заболеваний.

**Жиры** – это органическое вещество, нерастворимое в воде, выполняющее в живом организме энергетическую и структурную функции. В жирах сконцентрировано большое количество энергии, по сравнению с белками и углеводами. Человеческое тело должно обладать разумными запасами жира, иначе без этого будут разрушаться мышцы и ткани организма, не будет необходимого энергетического запаса для построения новых клеток.

**Калорийность** – количество энергии, получаемое живым организмом за счет потребления одного грамма какого-либо продукта, выраженное в калориях.

**Комплексный обед** – обед, меню которого уже составлено заранее, включающее в себя салат-закуску, первое(суп), второе и третье(например компот или чай).

**Контрагент** – это физическое или юридическое лицо, выступающее одной из сторон сделки. Под контрагентами понимаются различные лица, предприятия и учреждения, с которыми организация вступает в торговые, финансовые, гражданско-правовые и другого рода отношения.

**Кухонный персонал** – это человек, работающий на кухне, но не в качестве повара, а в качестве его помощника. При этом такой работник должен обладать такими же навыками и познаниями, что и сам повар.

**Накладная** - документ, в соответствии с которым происходит передача товара, материальных ценностей из одних рук в другие. В накладной указывается вид товара, его количество, цена, общая стоимость. Накладная скрепляется подписями передающей и принимающей сторон и представляет документ бухгалтерского учета.

**Обслуживающий персонал** – это категория работников, которые выполняют определенные функции в сфере обслуживания. Сотрудник может обслуживать рабочие помещения, здания или клиентов предприятия.

**План питания** – это рациональное распределение продуктов и блюд в течение дня.

**Пищевой продукт** - продукты животного, растительного, микробиологического, минерального, искусственного или биотехнологического происхождения в натуральном, обработанном или переработанном виде, которые предназначены для употребления человеком в пищу, в том числе специализированная пищевая продукция.

**Санаторий** - лечебно–профилактическое учреждение, в котором для лечения и профилактики заболеваний используют главным образом природные факторы (климат, минеральные воды, лечебные грязи, морские купания и т. п.) в сочетании с лечебной физкультурой, физиотерапией и рациональным питанием (диетой) при соблюдении определённого режима лечения и отдыха.

**Система управления базами данных (СУБД)** - программный комплекс, обеспечивающий централизированное хранения данных и предоставляющий

приложениям услуги по обработке данных.

**Склад** - специальное помещение для хранения товара, материалов, сырья, оборудования и т. п.

**Сущность** - реальный или представляемый объект, информация о котором должна сохраняться и быть доступна.

**Трапеза** – Любой прием пищи, еда. (завтрак, обед, ужин)

**Углеводы** – главный источник энергии для организма человека. Эти органические вещества бывают простыми (сахар) и сложными (крахмал, клетчатка).

**Фреймворк -** Программное обеспечение, облегчающее разработку и объединение разных модулей программного проекта.

**Use case -** Описание поведения системы, когда она взаимодействует с кем-то (или чем-то) из внешней среды.

**1.2 Бизнес-требования**

**БТ1:** Система должна позволять осуществлять управление поставками продуктов на склад.

**БТ2:** Система должна позволять контролировать наличие продуктов на складе.

**БТ3:** Система должна автоматически формировать меню на каждый день, на основе имеющихся продуктов.

**БТ4:** Система должна позволять рассчитывать калорийность и пищевую ценность меню.

**БТ5:** База данных помогает контролировать состояние складов и продуктов на них, снижает риски непредвиденных недостатков продуктов.

**1.3 Пользовательские требования**

**ПТ1:** Приложение должно иметь простой интуитивный интерфейс.

**ПТ2:** разрешить доступ к базе данных только авторизованным пользователям.

**ПТ3:** Доступ к базе данных должен быть организован через сеть, чтобы иметь доступ с любого места.

**ПТ4:** Возможность изменять и удалять данные.

**1.4 Системные требования**

**СТ1:** База данных должна быть доступна для работы в сети;

**СТ2**: Доступ к базе данных должен быть ограничен для разных уровней пользователей.

**СТ3**: Система должна иметь функционал по созданию, редактированию и удалению данных.

**СТ4**: Система должна обеспечивать сохранность данных при сбое.

**СТ5:** Использование облачных технологий для быстрого доступа к данным.

**СТ6:** Выделенный физический сервер.

**1.5 Функциональные требования**

**ФТ1:** Ведение базы данных блюд, продуктов и складов санатория.

**ФТ2:** Автоматическое формирование меню на каждый день.

**ФТ3:** Расчет калорийности и пищевой ценности меню.

**ФТ4:** Внесение данных о поставке продуктов на склад.

**ФТ5:** Контроль наличия продуктов на складе и выдача предупреждений о нехватке продуктов.

**ФТ6:** Возможность авторизации администратора

**ФТ7:** Ведение базы данных о поставщиках продуктов.

**ФТ8:** Создание приходных накладных на поставку продуктов на склад.

**ФТ9:** Возможность регистрации новых пользователей.  
 **ФТ10:** Возможность обратиться при сбое в работе базы данных в службу технической поддержки

**2.Архитектурное описание**

**2.1 Архитектурное обоснование**

Для создания данного веб-приложения планируется использовать язык программирования Python и фреймворк Qt. Python был выбран за удобство в разработке, поддержке и кроссплатформенности. Фреймворк Qt позволяет создавать десктопные приложения высокого качества и так же ускоряет время разработки.

Для хранения и обработки информации будет использоваться реляционная система управления базами данных Microsoft SQL Server, поскольку в Python существует тесная интеграция с ним. Для взаимодействия веб-сервера и сервера базы данных подходит технология ОDBC, предоставляющая программный интерфейс доступа к базам данных для взаимодействия приложений с СУБД MS SQL Server

Для безопасности и надежности проведения платежей был выбран платежный шлюз системы ЮКасса, при совершении сделок.

Для создания приложения будет использоваться клиент-серверная архитектура. Такой подход позволяет разделить логику пользовательского интерфейса и модели, создающей управляющие события. Серверная часть будет использоваться для моделирования событий и обработки данных.

Такой подход позволяет обрабатывать большое количество данных и своевременно принимать решения по их оптимизации. Также этот подход разделяет устройства отображения информации и устройство её обработки. Это улучшает пользовательский опыт ввиду снижения нагрузки на рабочую систему пользователя.

**2.2 Модели**

В таблице 1. представлены данные и методы работы с ними.

Таблица 1. Данные и методы

|  |  |  |
| --- | --- | --- |
| **Объект** | **Методы** | **Свойства** |
| User | Авторизация, регистрация, изменение информации о пользователе, получение информации о состоянии склада, получение меню, составленное системой, просмотр накладных на складе | ID, ФИО, паспортные данные, телефон, логин и пароль, состояние склада, состав меню (список блюд и кол-во порция каждого блюда). |
| Dish | Расчет калорийности и пищевой ценности и просматривать используемый продукты | Наименование, содержание жиров, содержание белков и углеводов, выход, набор продуктов (наименование, единица измерения, цена), необходимое количество |
| Warehouse | Внесение данных о поставке, контроль наличия продуктов на складе. Создания приходных накладных, содержание информации о поставщиках | ID склада, список продуктов и их количество, ID поставщика с контактными данными, ID прикладной (номер документа, дата, склад, контрагент, список продуктов и их количество, цена продукта) |

**2.3 UML описание**

**2.3.1 Use case**

В данном проекте есть три вида пользователей:

1) Неавторизованные пользователи могут: авторизоваться или зарегистрироваться, обратиться в тех. поддержку.

2) Авторизованные пользователи могут: вести БД блюд, продуктов и складов санатория, внесение данных о поставке продуктов на склад, оставлять отзывы о качестве доставляемой продукции, а также обращаться в техническую поддержку;

3) Администраторы могут: имеет возможность авторизоваться в системе в качестве администратора, получать доступ к истории сообщений пользователей со службы поддержки, просмотр приходных накладных, создание приходных накладных на поставку продуктов на склад, контролировать наличие продуктов на складе и получать предупреждения о нехватке продуктов.

4) Техподдержка: имеет возможность отвечать пользователям на полученные сообщения через форму обратной связи.

Диаграмма вариантов использования (use case) приведена на рисунке 1.

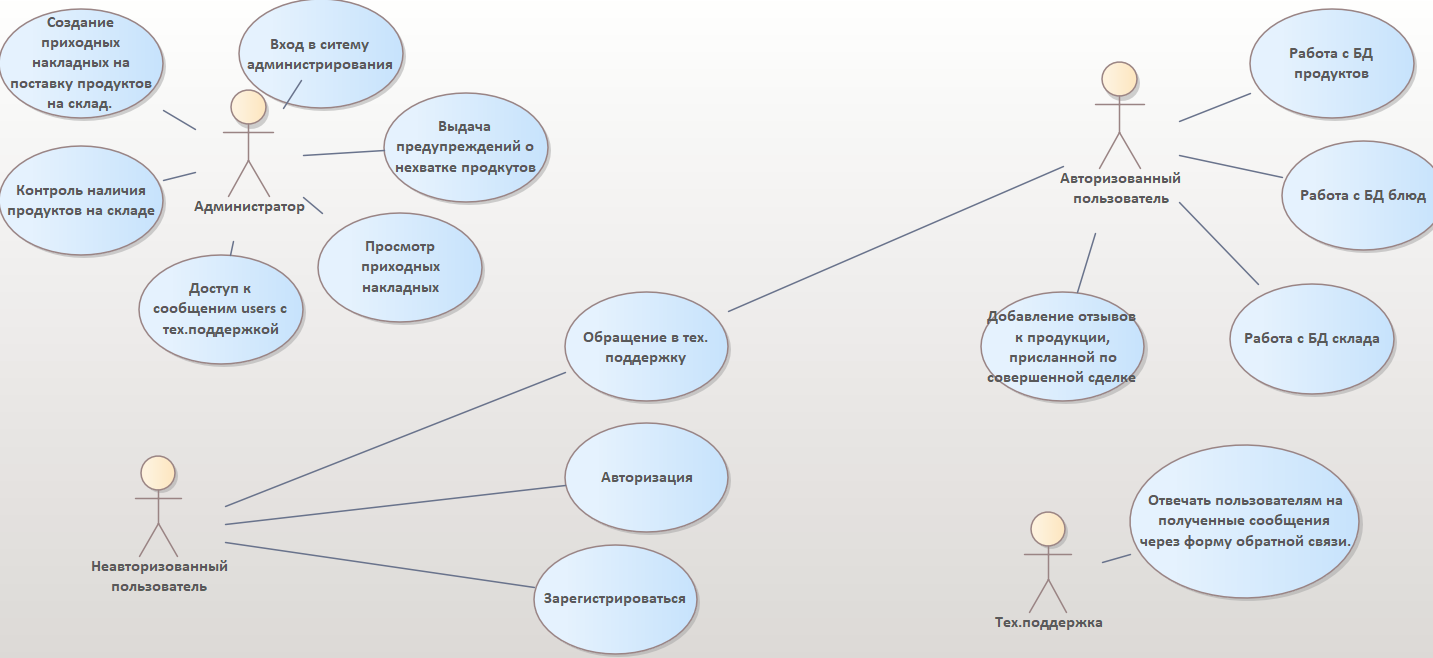


Рисунок 1. Use case диаграмма проекта

**Описание основных действий Use case диаграммы:**

Описание сценария выполнения варианта использования «Создание приходных накладных на поставку продуктов на склад» приведено в таблице 2.3.1.1.

Таблица 2.3.1.1. *Создание приходных накладных на поставку продуктов на склад*

|  |  |
| --- | --- |
| **Вариант использования** | Создание приходных накладных на поставку продуктов на склад |
| Актеры | Администратор |
| Цель | Создание приходных накладных на поставку продуктов на склад, при нехватке продуктов |
| Краткое описание | Незарегистрированный пользователь регистрируется в системе, система, в свою очередь, проверяет корректность введенных данных и добавляет пользователя в базу данных |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствуют |

В таблице 2.3.1.2. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Создание приходных накладных на поставку продуктов на склад.

Таблица 2.3.1.2. *Ход действий для создания приходных накладных на поставку продуктов на склад*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Администратор открывает страницу создания приходной накладной | 2. Система открывает форму создания приходной накладной |
| 3. Администратор заполняет все обязательные поля | 4. Система автоматически рассчитывает цену каждого продукта на основе цены, указанной в БД, и количества поставленного продукта, автоматически рассчитывает сумму общую сумму по приходной накладной и спрашивает, желает ли пользователь сохранить данные |
| 5. Администратор сохраняет приходную накладную | 6. Система сохраняет приходную накладную в БД |

Описание сценария выполнения варианта использования «Авторизация пользователя» приведено в таблице 2.3.1.3.

Таблица 2.3.1.3. *Авторизация пользователя*

|  |  |
| --- | --- |
| **Вариант использования** | Авторизация пользователя |
| Актеры | Неавторизированный пользователь |
| Цель | Авторизация неавторизованных пользователей |
| Краткое описание | Неавторизованный пользователь авторизируется в системе, система, в свою очередь, проверяет корректность введенных данных и сравнивает с базой данных |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствуют |

В таблице 2.3.1.4. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Авторизации пользователя.

Таблица 2.3.1.4. *Ход действий для авторизации* *пользователя*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Пользователь вводит данные, необходимые для авторизации. | 2. Система проверяет, корректны ли введенные данные. Данные сопоставляются с данными в базе. При совпадении, система отправляет код авторизации на email |
| 3. Пользователь вводит код авторизации | 4. Система проверяет корректность введенного пользователем кода. В случае успешной проверки, система авторизует пользователя. |

Описание сценария выполнения варианта использования «Ведение БД блюд, продуктов и складов санатория» приведено в таблице 2.3.1.5.

Таблица 2.3.1.5. *Ведение БД блюд, продуктов и складов санатория*

|  |  |
| --- | --- |
| **Вариант использования** | Ведение БД блюд, продуктов и складов санатория |
| Актеры | Авторизированный пользователь |
| Цель | Заполнение подробной информации о блюде, о продукте или о складе, сохраняя все введенные данные в БД |
| Краткое описание | Заполнение информации о блюде, о продукте или о складе |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствуют |

В таблице 2.3.1.6. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Ведение БД блюд, продуктов и складов санатория.

Таблица 2.3.1.6. *Ход действий для ведения БД блюд, продуктов и складов санатория*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Пользователь выбирает опцию «Ведение БД» в главном меню | 2. Система отображает исходную страницу сервиса и дает выбрать тип данных: «Блюда» «Продукты» «Склады» |
| 3. Пользователь выбирает один из типов данных | 4. Система отображает информацию по выбранному типу данных |
| 5. Пользователь добавляет или изменяет информацию в выбранном типе данных и сохраняет | 6. Система сохраняет измененные данные в БД |

Описание сценария выполнения варианта использования «автоматическое формирование меню» приведено в таблице 2.3.1.7.

Таблица 2.3.1.7. *Автоматическое формирование меню*

|  |  |
| --- | --- |
| **Вариант использования** | Автоматическое формирование меню |
| Актеры | Администратор |
| Цель | Формировать меню на каждый день |
| Краткое описание | Администратор с помощью опции автоматического составлению меню, создает меню на каждый день |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствуют |

В таблице 2.3.1.8. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Автоматическое формирование меню.

Таблица 2.3.1.8. *Ход действий для автоматического формирования меню*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Администратор выбирает опцию «Формирование меню» в главном меню | 2. Система узнаем. На какую дату желает администратор составить меню |
| 3. Администратор выбирает и вводит дату | 4. Система на основе данных о блюдах и продуктах, автоматически формирует меню для выбранной даты |
| 5. Администратор проверяет корректность заявки и подтверждает ее правильность | 6. Система сохраняет меню в БД |

Описание сценария выполнения варианта использования «Авторизация администратора» приведено в таблице 2.3.1.9.

Таблица 2.3.1.9. *Авторизация администратора*

|  |  |
| --- | --- |
| **Вариант использования** | Авторизация администратора |
| Актеры | Неавторизованный администратор |
| Цель | Авторизация неавторизованных администраторов |
| Краткое описание | Неавторизованный администратор авторизируется в системе, система, в свою очередь, проверяет корректность введенных данных и сравнивает с базой данных |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствуют |

В таблице 2.3.1.10. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Авторизации администратора.

Таблица 2.3.1.10. *Ход действий для* авторизации *администратора*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Администратор вводит данные, необходимые для авторизации. | 2. Система проверяет, корректны ли введенные данные. Данные сопоставляются с данными в базе. При совпадении, система отправляет код авторизации на email |
| 3. Администратор вводит код авторизации | 4. Система проверяет корректность введенного администратором кода. В случае успешной проверки, система авторизует администратора. |

Описание сценария выполнения варианта использования «Расчет калорийности и пищевой ценности меню» приведено в таблице 2.3.1.11.

Таблица 2.3.1.11. *Расчет калорийности и пищевой ценности меню.*

|  |  |
| --- | --- |
| **Вариант использования** | Расчет калорийности и пищевой ценности меню |
| Актеры | Авторизированный пользователь |
| Цель | Рассчитать калорийность и пищевую ценность меню с помощью нужной опции пользователю |
| Краткое описание | Рассчитать калорийность и пищевую ценность меню |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствует |

В таблице 2.3.1.12. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Расчет калорийности и пищевой ценности меню

Таблица 2.3.1.12*. Ход действий для расчета калорийности и пищевой ценности меню*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Авторизованный пользователь открывает страницу и выбирает опцию «Расчет калорийности и пищевой ценности» в главном меню | 2. Система просит выбрать дату меню, для которой нужно совершить данную опцию |
| 3. Пользователь выбирает дату, для которой нужно произвести расчет. | 4. Система на основе данных о блюдах и продуктах рассчитывает калорийность и пищевую ценность меню по выбранной даты |

Описание сценария выполнения варианта использования «Внесение данных о поставке продуктов на склад» приведено в таблице 2.3.1.13

2.3.1.13. *Внесение данных о поставке продуктов на склад*

|  |  |
| --- | --- |
| **Вариант использования** | Внесение данных о поставке продуктов на склад |
| Актеры | Администратор |
| Цель | Заполнение и сохранение данных о поставках продуктов в БД |
| Краткое описание | Внесение администратором данных о совершенных поставках продуктов на склады в БД |
| Тип | Базовый |
| Ссылки на другие варианты  использования | Отсутствуют |

В таблице 2.3.1.14 описывается последовательность действий, приводящая к успешному выполнению варианта использования – Внесение данных о поставке продуктов на склад.

Таблица 2.3.1.14. *Ход действий для внесения данных о поставке продуктов на склад*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Администратор через личный кабинет выбирает опцию «Поставка продуктов на склад» | 2. Система спрашивает какой склад интересует администратора |
| 3. Администратор выбирает склад, на который будут доставлены продукты | 4. Система открывает страницу с выбранным складом |
| 5. Администратор вводит данные о поставке продуктов | 6. Система сохраняет данные о поставке продуктов в БД |

Описание сценария выполнения варианта использования «Контроль наличия продуктов на складе и выдача предупреждений о нехватке продуктов» приведено в таблице 2.3.1.15.

Таблица 2.3.1.15. *Контроль наличия продуктов на складе и выдача предупреждений о нехватке продуктов*

|  |  |
| --- | --- |
| **Вариант использования** | Контроль наличия продуктов на складе и выдача предупреждений о нехватке продуктов |
| Актеры | Администратор |
| Цель | Просматривает с помощью опции контроль наличия продуктов на складе или получает уведомление |
| Краткое описание | Пользователь контролирует наличие продуктов на складе и получает предупреждение |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствует |

В таблице 2.3.1.16. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Контроль наличия продуктов на складе и выдача предупреждений о нехватке продуктов.

Таблица 2.3.1.16*. Ход действий для контроля наличия продуктов на складе и выдача предупреждений о нехватке продуктов*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Администратор открывает страницу с контролем продуктов на складе | 2. Система открывает страницу с автоматическим анализом данных о продуктах на складах и их количество, необходимым для приготовления каждого блюда. |
| 3. Администратор выбирают опцию «Нехватка продуктов» | 4. Система выводит предупреждение о нехватке продуктов, если количество какого-то продукта на складе меньше, чем количество, необходимое для приготовления блюда, в хорошем случае пишет, что всех продуктов хватает |
| 5. Администратор при нехватке продуктов просит автоматически создать заказ на поставку продуктов | 6. Система создает заказ на поставку продуктов и посылает его поставщику |

Описание сценария выполнения варианта использования «Ведение БД о поставщиках продуктов» приведено в таблице 2.3.1.17.

Таблица 2.3.1.17. *Ведение БД о поставщиках продуктов*

|  |  |
| --- | --- |
| **Вариант использования** | Ведение БД о поставщиках продуктов |
| Актеры | Администратор |
| Цель | Администратор ведет БД о поставщиков продуктов и вносит информацию о них |
| Краткое описание | Вести БД о поставщиков продуктов на склады |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствует |

В таблице 2.3.1.18. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Ведение БД о поставщиках продуктов

Таблица 2.3.1.18*. Ход действий для ведения БД о поставщиках продуктов*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Администратор открывает страницу с поставщиками | 2. Система отображается страницу с поставщиками продуктов |
| 3. Администратор выбирает функцию «Добавить» поставщика | 4. Система запрашивает у администратора информацию о поставщике (наименование, адрес, контактные данные, и другие данные, которые необходимы для контроля и связи с поставщиком) |
| 5. Администратор вводит информацию в соответствующие поля и нажимает «Добавить» | 6. Система проверят заполнены ли все поля, и если все в порядке сохраняет данные о поставщике |

Описание сценария выполнения варианта использования «Обращение в службу технической поддержки» приведено в таблице 2.3.1.19.

Таблица 2.3.1.19. *Обращение в службу технической поддержки*

|  |  |
| --- | --- |
| **Вариант использования** | Обращение в службу технической поддержки |
| Актеры | Авторизованные пользователи |
| Цель | Обращение пользователя в службу технической поддержки и дальнейшая обработка запроса службой тех. поддержки |
| Краткое описание | Пользователь отправляет обращение в техническую поддержку с целью сообщить об ошибках, предложить улучшения и т.д. |
| Тип | Базовый |
| Ссылки на другие варианты использования | Отсутствует |

В таблице 2.3.1.20. описывается последовательность действий, приводящая к успешному выполнению варианта использования – Обращение в службу технической поддержки

Таблица 2.3.1.20. *Ход действий при обращении в службу технической поддержки*

|  |  |
| --- | --- |
| **Действия актеров** | **Отклик системы** |
| 1. Пользователь открывает страницу с формой отправки сообщения в службу тех. поддержки | 2. Система отображает страницу с формой отправки сообщения в службу тех. поддержки |
| 3. Пользователь набирает текст обращение в поле для текста | 4. Система отображает набранный пользователем текст |
| 5. Пользователь отправляет сообщение в службу технической поддержки | 6. Система проверяет сообщение на наличие бан-слов и, в случае их отсутствия, направляет текст сообщения и данные пользователя в службу технической поддержки |

**2.3.2 Диаграмма классов**

Описание диаграммы классов в данном проекте основными классами являются классы Dihs, Menu, Warehouse, Product, Purchaselnvoise.

Класс **Dish** описывает каждое блюдо, занесенные в Базу Данных. За каждым блюдом закреплены его названия (Name), содержание калорий в блюде (Calories), содержание в блюде жиров, углеводов и белков (БЖУ) (contentOfBJU), содержит поле, которое содержит информацию из каких продуктов сделаны блюда (productSet), Необходимое количество каждого продукта (RequiredAmountedEachProduct) и выход порции (вес) (weight). Для данного класса доступны следующие методы: получение списка блюд, хранящихся в базе данных getDish(), вывод информации о блюде, которая есть в базе данных OutputlnfDish(), метод для изменения информации о блюде setDish(string).

Класс **Menu** описывает каждое меню, которое автоматически составляет его при желании администратора. За каждым меню закреплены список блюд, которые автоматически выбираются системой (ListDish), и разбиение дня на трапезы (завтрак, обед, ужин) (NumberOfServingsEachDish). Для данного класса доступны следующие методы: получение списка блюд, хранящихся в базе данных, которые попали в меню GetListDish(), изменение списка блюд, которая есть в базе данных (SetListDish).

Класс **Product** описывает каждый продукт, занесенные в Базу Данных и хранящийся на каком-то складе. За каждым продуктом закреплена его стоимость за одну единицу (cost), хранит информацию в какой единице счисления хранится продукт (Unit), хранится информация о названии продукта (Name). Для данного класса доступны следующие методы: получение стоимости продуктов, хранящихся в базе данных getCost(),метод для изменения информации о стоимости продуктов setCost(string).

Класс **Warehouse** описывает каждый склад, имеющийся в отделе питания санатория и занесенные в Базу Данных. За каждым складом закреплены его id (IDwarehouse). Для данного класса доступны следующие методы: получение списка id складов, хранящихся в базе данных GetIDwarehouse (int), метод для изменения информации о id склада SetIDwarehouse (integer).

Класс **Purchaselnvoice** описывает приходные накладные, занесенные в Базу Данных. За каждой приходной накладной закреплены сумма по каждому отдельному продукту (amountForEachProduct), содержание стоимости продукта за единицу (costProduct), дата оформления накладной (dateDoc), содержит поле, которое содержит информацию о контрагенте (IDcounterparty), номер документа (IDdoc), склад, на который были завезены продукты (IDwarehouse), содержит информацию о наименовании продутов (NameOfProducts), и количество пришедшего продукта (quantityProducts), общая сумма за продукты (totalAmount). Для данного класса доступны следующие методы: получение списка контрагентов, хранящихся в базе данных GetIDcounterparty(), вывод информации о накладных, которая есть в базе данных GetIDdoc(int), метод для получения информации о складах, на которых хранятся накладные GetIDwarehouse(), просмотр количества пришедших на склад продуктов GetquantilyProducts(), изменение (изменение/добавление/удаление) списка контрагентов, хранящихся в базе данных SetIDcounterparty(), изменение списка накладных SetIDdoc(int), метод для изменения информации о складах, на которых хранятся накладные SetIDwarehouse(), изменение информации о количестве поступивших продуктов SetquantilyProducts().

Между классами Dish и Menu отношение агрегация 1 ко многим, так как одно меню содержит множество разных блюд, а удаление одного блюда не влияет на состояние меню.

Между классами Dish и Product отношение агрегации 1 к 1..\*, так как одно блюдо состоит из некоторого количества продуктов, и удаление какого-то продукта не влияет на блюдо. Между классами Warehouse и Product отношение агрегации 0..\* к 1..\*, так как несколько складов содержит минимум от 0 до n-го количества продуктов, и удаление какого-то продукта не влияет на склад.

Между классами Warehouse и Purchaselnvoice отношение агрегации 1 к 1, так как одна накладная может хранится лишь на одном складе, и удаление какой-то накладной не влияет на склад.

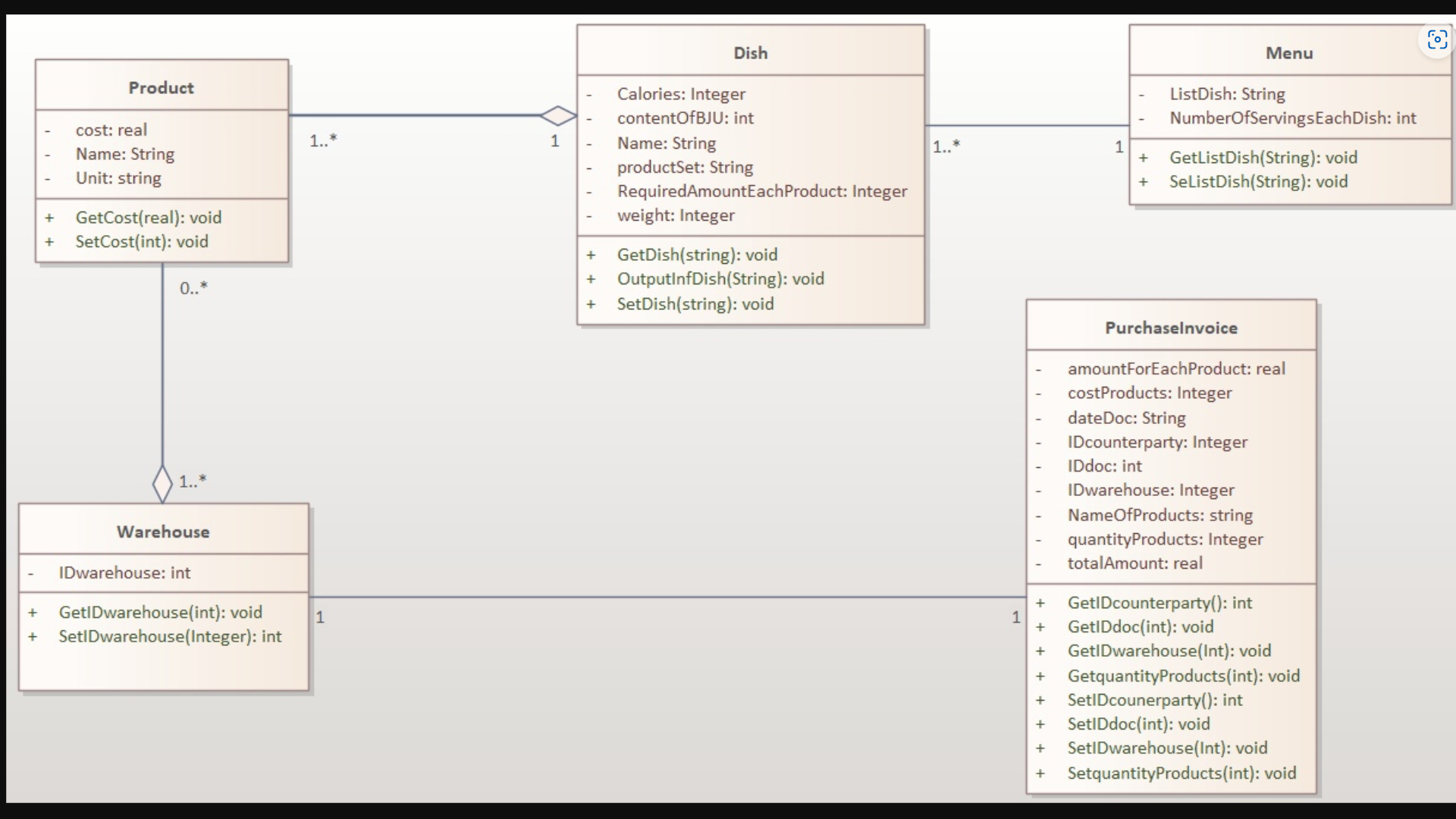
****

Рисунок 2. Диаграмма классов

**2.3.3 Диаграмма активности**.

Описание диаграммы:

Диаграмма активности описывает процессы, которые могут проходить в системе управления продуктами и складами санатория. Она начинается с процесса авторизации или регистрации для неавторизованных пользователей.

Если пользователь регистрируется в системе, он должен ввести данные о себе, включая логин, пароль и адрес электронной почты. После успешной регистрации пользователь может авторизоваться в системе и получить доступ к функциям, предоставляемым системой.

Авторизованные пользователи имеют возможность вести БД блюд, продуктов и складов санатория, оставлять заявки на поставку продуктов, вносить данные о поставке продуктов на склад, оставлять отзывы о качестве доставляемой продукции, а также обращаться в техническую поддержку.

В процессе ведения БД блюд, продуктов и складов пользователи могут создавать новые записи, редактировать и удалять существующие записи.

Пользователи также могут вносить данные о поставке продуктов на склад. Данные должны включать информацию о количестве и типе продукта, дате поставки и ответственном работнике.

Отзывы пользователей о качестве доставляемой продукции могут быть добавлены в систему через специальную форму.

Администратор имеет возможность просмотреть сообщения от пользователей через форму обратной связи и ответить на них. Администраторы могут просматривать приходные накладные и создавать новые приходные накладные на поставку продуктов на склад.

После успешного завершения любой из этих функций пользователь может выйти из системы  
 На рисунке 3 представлена диаграмма активности для нашего проекта.

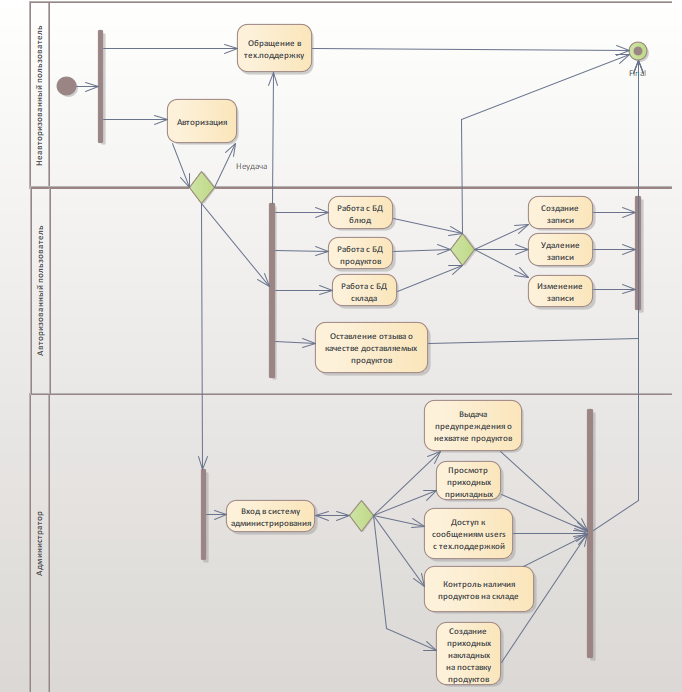
****

Рисунок 3. Диаграмма активности

**2.3.4 Диаграмма последовательностей**

Первым этапом работы над проектом является построение UML диаграммы последовательности, которая поможет понять, как взаимодействуют компоненты системы при выполнении определенных действий или операций.

На диаграмме последовательности можно увидеть два актера: отдел питания и склад, которые взаимодействуют между собой, а также базу данных, которая хранит информацию о блюдах, продуктах и меню.

Предполагается, что сотрудник отдела питания может автоматически сформировать меню на каждый день, выбрав нужные блюда и указав количество порций. Для этого необходимо иметь доступ к базе данных, где хранятся все блюда и их свойства, в том числе калорийность и пищевая ценность.

Также на диаграмме присутствует операция по поставке продуктов на склад, которая фиксируется с помощью приходной накладной. Для этого сотрудник класса Склад должен иметь доступ к базе данных и иметь возможность добавить информацию о поставке.

Для контроля наличия продуктов на складе можно добавить операцию, которая проверяет наличие необходимых продуктов и выдает предупреждение о нехватке продуктов, если таковые имеются. Эта операция может оповещать сотрудника отдела питания о необходимости заказать дополнительные продукты.

Для возможности обращения за помощью при сбое в работе базы данных можно добавить операцию, которая связывает пользователя с службой технической поддержки.

Таким образом, UML диаграмма последовательности помогает описать взаимодействие компонентов системы, их возможные операции и действия. Она также помогает описать функциональные требования к системе и понять, как они могут быть реализованы.

В ходе выполнения работы была построена диаграмма последовательности, описывающая работу отдела питания санатория. Продолжено знакомство с программной средой Enterprise Architect на примере построения данного вида UML диаграмм.  
 На рисунке 4 представлен класс последовательности для нашего проекта.

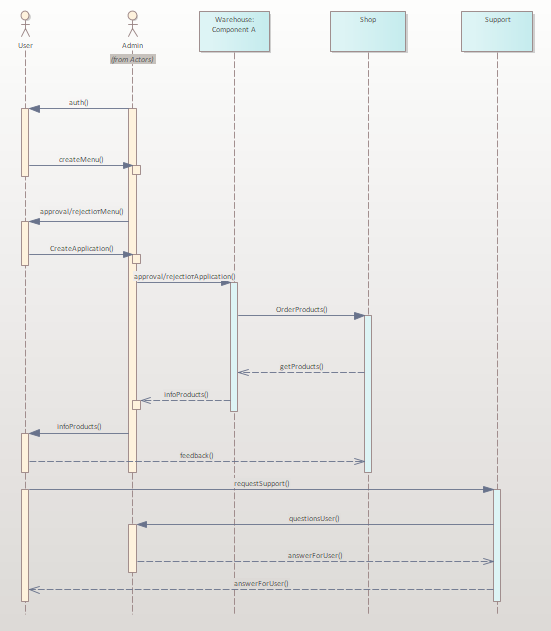
****

Рисунок 4. Диаграмма последовательности

**3. Тестирование**

Функциональность готового сервера должна быть проверена с помощью тестирования следующих компонентов системы:

1. Проверка БД на ведении данных о блюдах, продуктов и складов санатория
2. Работоспособность автоматического формирования меню на день
3. Правильность расчета калорийности и пищевой ценности меню
4. Проверка правильности и возможности внесения данных о поставках продуктов на склад
5. Работоспособность контроля наличия продуктов на складе и работы выдачи предупреждений о нехватке продуктов
6. Правильность работы ведения БД о поставщиках продуктов
7. Возможность пользователей связываться со службой технической поддержки
8. Возможность регистрации и авторизации пользователя
9. Корректность создания приходных накладных на поставку продуктов на склад
10. Возможность авторизации неавторизированного администратора

# ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы была спроектирован сервер для отдела питания санатория. Были решены следующие задачи: изучены современные технологии разработки серверов для отделов предприятий на примере различных CMS; было принято решение придерживаться архитектуры MVC, что позволило создать гибкое веб-приложение; сформированы бизнес-требования, пользовательские, системные и функциональные требования; сформулированы критерии тестирования сервера, описаны возможности пользователей и администраторов, построены диаграммы и тесты, который нужны для определения качества и эффективности работы сервера.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Дари К., Баланеску Э. PHP и MySQL. Создание интернет-магазина. – М.: Издательский дом «Вильямс», 2010. – 640 с.
2. Водяхо А.И., Выговский Л. С, Дубенецкий В.А. Цехановский В.В.,

Архитектурные решения информационных систем. - СПб.: Издательство «Лань», 20 с.

1. Статистика использования PHP на веб-сайтах. [Электронный ресурс] URL: <http://itmozg.ru>
2. Кораблев Ю. А., Сирота Д. Д., Архитектура информационных систем. – СПб: Издательство СПбГЭТУ «ЛЭТИ», 2016. – 58 с.
3. Учебник для вузов по направлению подготовки 230400 «Информационные системы и технологии». – Москва: Издательство «Академия», 2012. – 283 с.