МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра математического и компьютерного моделирования

Проектирование и разработка информационной системы «Интернет магазин бытовой техники»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студентки 4 курса 441 группы

направления 09.03.03 –Прикладная информатика
код и наименование направления

механико-математического факультета
наименование факультета, института, колледжа

ПАВЛОВОЙ АНАСТАСИИ ИГОРЕВНЫ
фамилия, имя, отчество

Научный руководитель
к.ф.-м.н., доцент _____ Е.Ю. Крылова
должность, уч. степень, уч. звание | подпись, дата | инициалы, фамилия

Зав. кафедрой
зав.каф., д.ф.-м.н., доцент _____ Ю.А. Блинков
должность, уч. степень, уч. звание | подпись, дата | инициалы, фамилия

Саратов 2023

Актуальность работы. Современный рынок бытовой техники является одной из наиболее динамично развивающихся отраслей экономики. Покупатели все чаще предпочитают совершать покупки через интернет магазины, ведь это удобно, быстро и экономично. Однако, создание и развитие интернет магазина бытовой техники требует высокотехнологичного подхода. Для облегчения и ускорения работы предприятия необходима современная автоматизированная система, которая позволит:

- Оптимизировать учет товаров, заказов и клиентов;
- Ускорить процесс обработки заказов;
- Сократить время на обслуживание и ответы на запросы клиентов;
- Обеспечить безопасность данных.

Таким образом, проектирование и разработка информационной системы для интернет магазина бытовой техники является крайне актуальной задачей, которая поможет оптимизировать работу предприятия, улучшить уровень обслуживания клиентов и повысить его конкурентоспособность на рынке.

Цель работы. Целью данной бакалаврской работы является построение информационной системы интернет-магазина.

Задачи работы. В процессе выполнения работы были поставлены следующие задачи:

1. Формирование технического задания;
2. Проектирование ИС средствами IDEF0 и UML нотаций;
3. Проектирование и нормализация баз данных;
4. Разработка программного интерфейса приложения;
5. Разработка пользовательского интерфейса;
6. Реализация средств защиты персональных данных.

Краткая характеристика материалов исследования. Работа основана на источниках, указанных в списке литературы. В работе произведены проектирование и реализация информационной системы при помощи фреймворков Spring Boot, Spring Security, Vue.js, PostgreSQL, IDEF0 и UML нотаций.

Описание структуры. Работа состоит из введения, пяти разделов, заключения, списка использованных источников содержащего 24 наименования и приложения. Каждый раздел содержит 2 подраздела, теорию и практику.

Основное содержание работы. Во введении кратко описывается содержание представленной работы. А так же раскрываются особенности проектирования и разработки информационных систем.

В первом разделе рассматривается построение моделей в нотации IDEF0 для моделирования функциональных процессов компании.

Модель (As is) IDEF0 представляет собой документ в виде блок-схемы, который описывает текущую модель системы, процесса или бизнес-функций, необходимых для определения потенциальных узких мест системы в соответствии с рисунком 1.

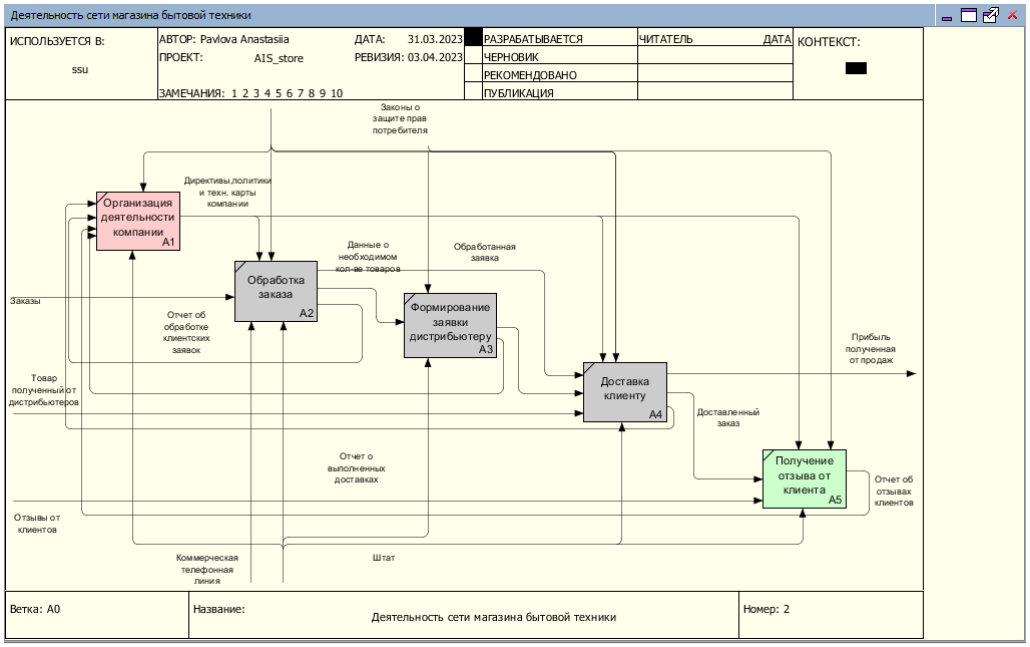

Рисунок 1 — IDEF0 модель предприятия по продаже бытовой техники (As is)

Модель (As to be) IDEF0 это документ в виде блок-схемы, который описывает желаемую модель системы, процессов или бизнес-функций после проведения изменений для улучшения текущей модели (as is). Он используется для проектирования и оптимизации систем, процессов и функций в соответствии с рисунком 2.

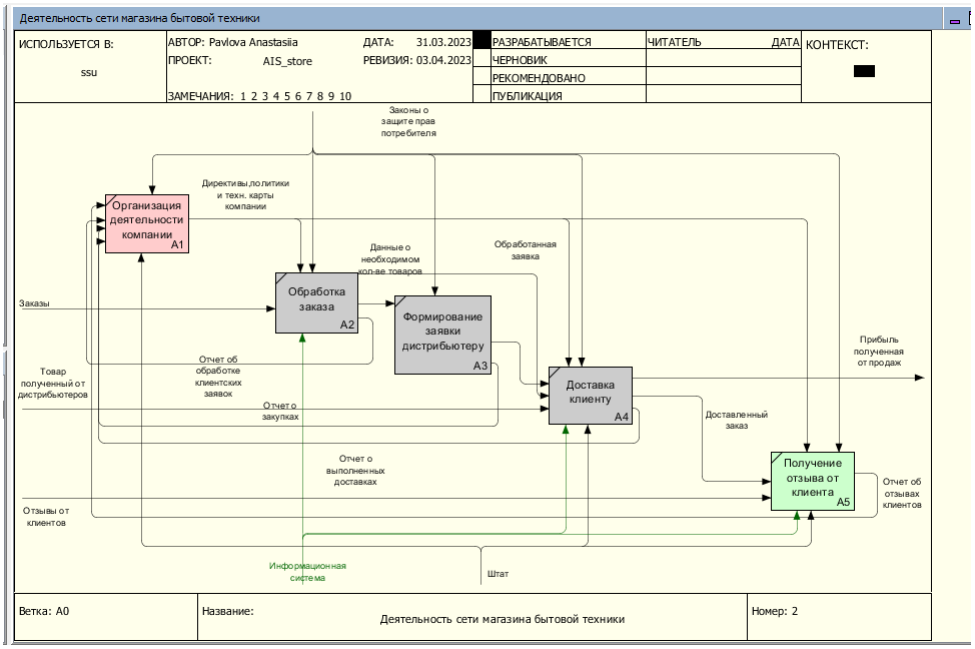

Рисунок 2 — IDEF0 модель предприятия по продаже бытовой техники (As to be)

Второй раздел посвящен проектированию системы при помощи средств UML.

UML позволяет описывать и проектировать различные аспекты приложения, в том числе его структуру, поведение, взаимодействие, а также процессы, которые происходят внутри системы что необходимо для анализа требований при разработке современных информационных систем.

В данном разделе были рассмотрены следующие виды:

- Диаграмма вариантов использования;
- Поток событий;
- Диаграмма активности;
- Диаграмма последовательностей.

Каждая из вышеперечисленных диаграмм и методов рассматриваются как часть процесса проектирования системы и необходима для дальнейшей разработки системы.

Как пример построения упомянутых диаграмм можно привести диаграмму вариантов использование в соответствии с рисунком 3.

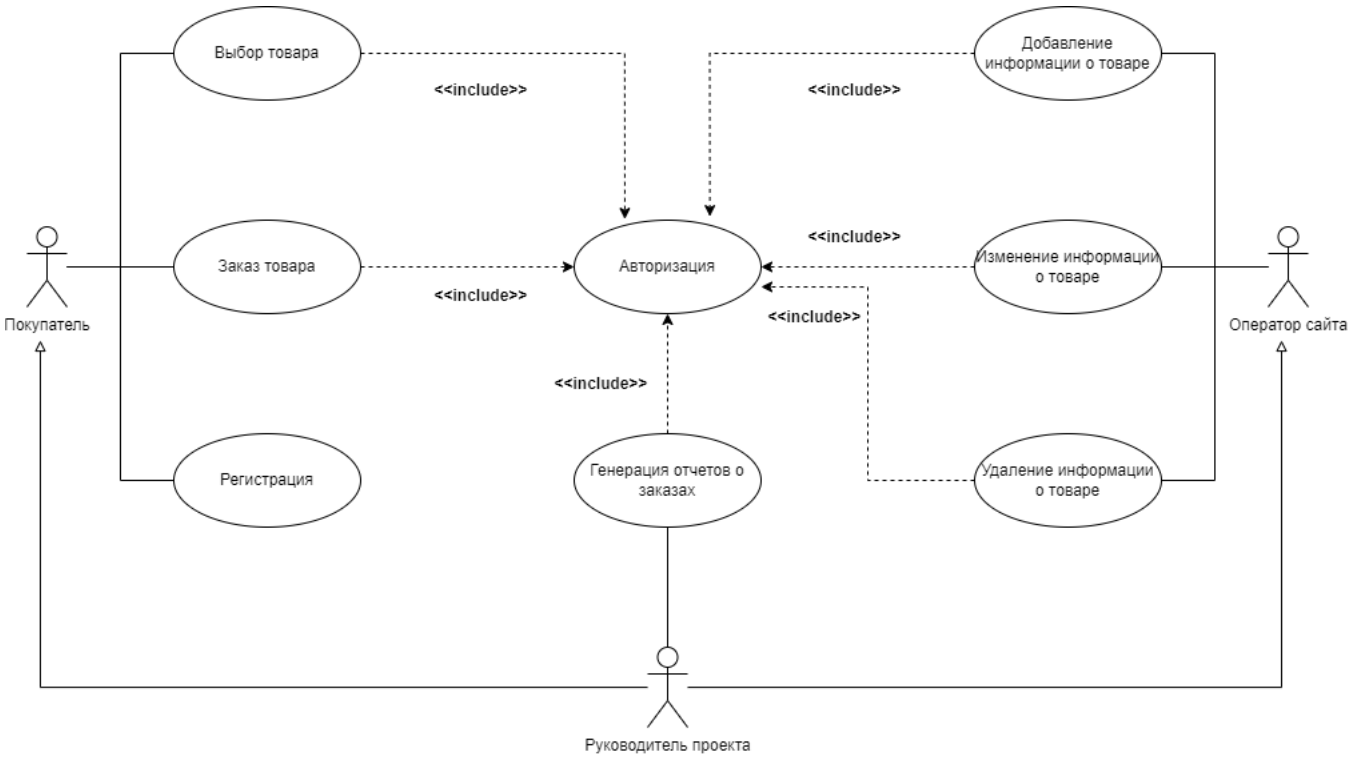

Рисунок 3 — Диаграмма вариантов использования для ИС

В третьем разделе рассмартивается такая важная часть любого проекта, как базы данных. ER-диаграмма представляет собой схему, которая описывает сущности и связи между ними в базе данных. Она состоит из сущностей (объектов), которые отображаются в виде прямоугольников с названием, и связей, которые указывают на связь между этими объектами и отображаются в виде линий или стрелок. ER-диаграммы используют в проектировании баз данных, чтобы определить структуру хранения данных, а также для анализа и моделирования информационных потоков. ER-диаграмма может содержать дополнительную информацию, такую как атрибуты сущностей и виды связей с ними.

Процесс представляется проектирования базы данных при помощи Entity-Relationship диаграммы в соответствии с рисунком 4.

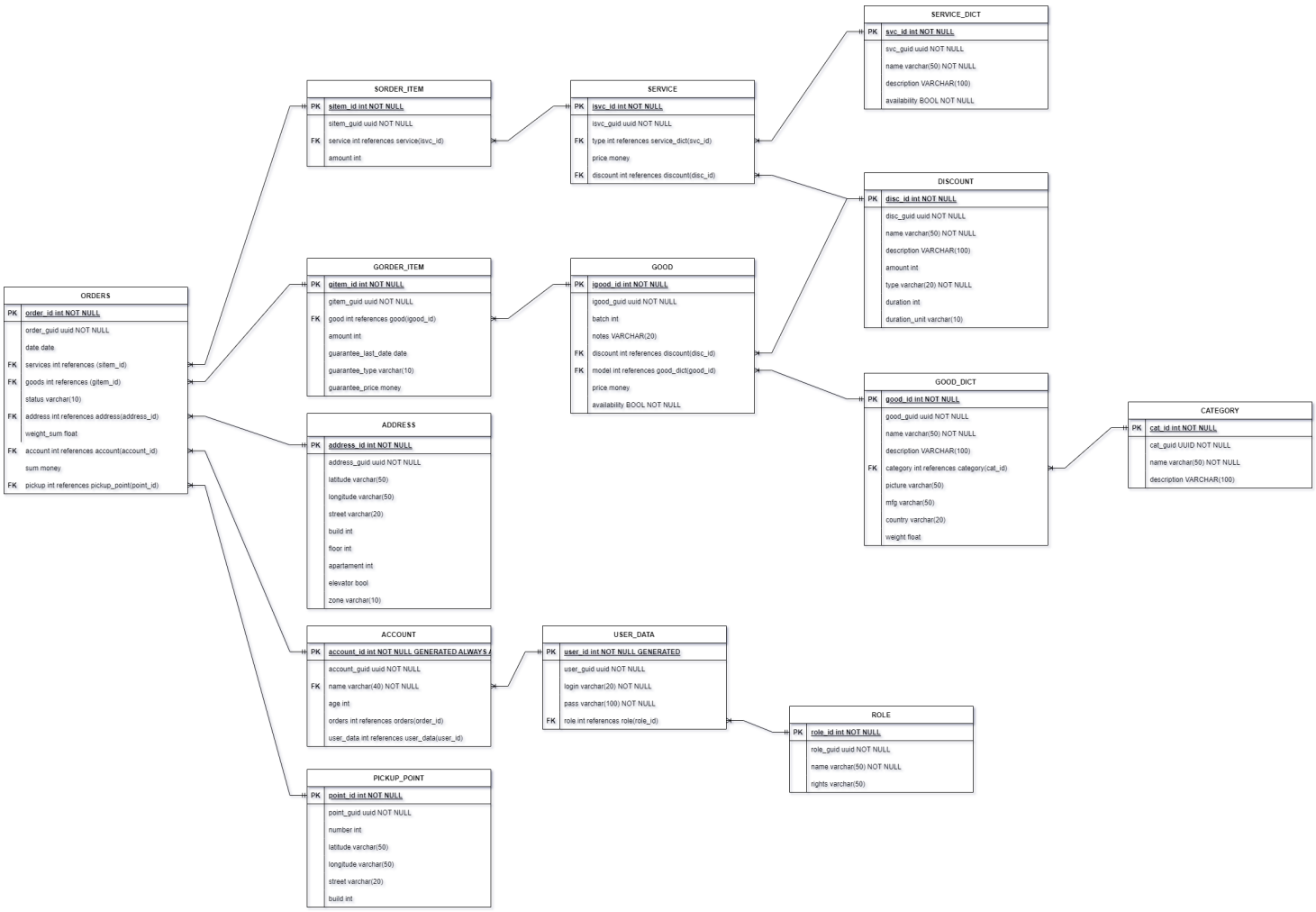

Рисунок 4 — Entity-Relationship диаграмма для ИС

После прохождения этапа проектирования, необходимо нормализовать базу данных для предотвращения избыточности и несогласованности данных. Правильная нормализация позволяет избежать проблем с целостностью данных и упрощает работу с базой данных.

Цель нормализации - разбиение таблиц базы данных на более мелкие, связанные между собой логические единицы, чтобы не хранить повторяющиеся данные и избежать аномалий записи, изменения или удаления данных.

После нормализации таблиц было реализовано создание и заполнение таблиц при помощи средств PostgreSQL, которая является объектно-реляционной системой управления базами данных (СУБД).

Четвертый раздел посвящен разработке программного интерфейса, построению ORM системы и API (Application Programming Interface) средствами Spring Boot и Spring Security.

ORM (Object-Relational Mapping, объектно-реляционное отображение) — это программная технология для преобразования данных, хранимых в реляционных базах данных, в объектно-ориентированное представление. ORM-системы предоставляют программистам объектный интерфейс для работы с базой данных, позволяя использовать объекты и методы из объектно-ориентированного программирования для работы с данными в базе данных. Они обеспечивают автоматическое создание SQL-запросов из кода приложения, а также обеспечивают автоматическое отображение данных из базы данных в объекты приложения. От ORM-систем ожидаются удобство и быстрота в работе с данными, а также снижение вероятности ошибок и увеличение безопасности и целостности данных. Некоторые из наиболее популярных ORM-систем включают в себя Hibernate, Entity Framework, Django ORM, SQLAlchemy.

Для построения подобной системы были предприняты следующие шаги:

- Создана модель данных: определены объекты, которые будут отображать таблицы в базе данных;
- Созданы сущности JPA: каждый объект данных связан с сущностью JPA;
- Опишисаны отношения между объектами данных в модели JPA: Например, отношения могут быть один-ко-многим, многие-ко-многим и т.д.;
- Создан контекст сохранения: это объект, который отслеживает все изменения в объектах данных и сохраняет их в базе данных;
- Настроено подключение к базе данных с использованием файла конфигурации "application.properties";

В пример приведем код файла конфигурации, который настраивает подключение к базе данных.

```
spring.datasource.url=jdbc:postgresql://localhost:5432/diplom
spring.datasource.username=postgres
spring.datasource.password=******
spring.jpa.generate-ddl=true
```

В процессе работы использован Hibernate - фреймворк для языка Java, предназначенный для работы с базами данных. Он реализует объектно-

реляционную модель — технологию, которая «соединяет» программные сущности и соответствующие записи в базе.

Для его использование была внедрена зависимость в pom.xml файл в соответствии с кодом ниже.

```
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>6.1.7.Final</version>
</dependency>
```

Программный интерфейс был реализован как Web REST (Representational State Transfer) API, представляющий собой интерфейс предназначеный для доступа к веб-серверу, использующий протокол http и реализующий crud операции. REST API (Representational State Transfer Application Programming Interface) является стандартом архитектуры программного обеспечения, который используется для создания веб-сервисов и клиент-серверных приложений. REST API позволяет клиентам получать доступ к данным на сервере через интернет и выполнять операции с данными, такие как чтение, изменение, создание и удаление.

Особенности REST API:

1. Независимость от языка программирования. REST API может быть создан на любом языке программирования и может быть использован с любым клиентским приложением, которое может отправлять HTTP-запросы.
2. Использование протокола HTTP для выполнения запросов и передачи данных между клиентом и сервером.
3. Использование стандартных методов HTTP для выполнения операций с данными, например, GET для чтения данных, POST для создания, PUT для обновления данных и DELETE для удаления данных.
4. Возвращение ответов в формате JSON или XML.
5. Кеширование запросов.
6. Отсутствие состояния сеанса между клиентом и сервером, что позволяет масштабировать систему.

Разработка REST API обычно включает в себя проектирование модели данных, разработку API-контроллеров, определение маршрутов запросов и реализацию ответов на запросы. REST API должен иметь документацию, объясняющую ресурсы, которые могут быть запрошены, методы, которые можно использовать, и типы ответов, которые можно получить.

Реализация CRUD операций упрощает процесс управления данными и делает его более эффективным и удобным для пользователей и предприятий.

CRUD операции необходимы для выполнения основных операций с данными в базе данных. Она позволяет пользователям создавать, просматривать, изменять и удалять данные в базе данных.

1. Создание (Create) позволяет создавать новые записи в базе данных, например, создавать новых пользователей или добавлять новые товары в инвентарь;
2. Чтение (Read) позволяет извлекать данные из базы данных, например, просматривать информацию о пользователях или заказах;
3. Обновление (Update) позволяет изменять данные в базе данных, например, обновлять контактную информацию пользователя или изменять количество товаров на складе;
4. Удаление (Delete) позволяет удалять данные из базы данных, например, удалять пользователя или товар из инвентаря.

Пятый раздел отображает процесс разработки пользовательского интерфейса, который необходим для:

- Облегчения работы пользователя при взаимодействии с программным обеспечением или устройством;
- Предоставления информации пользователю в интуитивно понятном и легко воспринимаемом виде;
- Уменьшения количества ошибок и проблем, которые могут возникнуть при работе пользователя с приложением.

Связь между REST API и UI может быть установлена с помощью AJAX-запросов.

Для удобства работы с REST API существуют библиотеки на языке JavaScript, такие как jQuery, AngularJS, React, Vue.js, которые позволяют

легко отправлять AJAX-запросы и обрабатывать ответы. В данной работе использован только фреймворк Vue.js.

Заключение. В результате использования средств IDEF0 и UML для проектирования информационной системы интернет-магазина бытовой техники была создана модель процессов и структуры системы, позволяющая более эффективно управлять заказами, оплатой и доставкой товаров.

Для хранения данных была спроектирована база данных на основе ER-диаграммы, определяющей наличие связей между сущностями и их атрибутами.

Одновременно была построена ORM система, которая позволяет использовать объектно-ориентированный подход к работе с базой данных и облегчает написание кода. В конечном итоге, использование RESTful API позволило удобно и быстро обмениваться данными между клиентской и серверной частями системы, что повышает ее эффективность и удобство использования. В целом, проектирование информационной системы с использованием подхода IDEF0, UML, ER-диаграмм и ORM системы с RESTful API привело к созданию более эффективной и интуитивно понятной системы, которая повышает удобство использования и увеличивает эффективность заказа и доставки товаров.

В ходе работы были реализованы и подробно описаны все стадии разработки информационных систем.

Задачи, поставленные в начале работы полностью выполнены.

Код, прикрепленный в приложении, полностью готов к реализации.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Черемных, С.В. Моделирование и анализ систем: IDEF-технологии. Практикум / С. В. Черемных. – Москва : Финансы и статистика, 2002. - 192 с.

2. Киселев, Д.Ю. Функциональное моделирование на базе стандарта IDEF0: метод. указания / Д.Ю. Киселев, Ю.В. Киселев, А.В. Вавилин. – Самара : Изд-во СГАУ, 2014. - 20 с.

3. Миндалев, И.В. Моделирование бизнес-процессов с помощью IDEF0, DFD, BPMN за 7 дней: учеб. пособие / И.В. Миндалев. – Красноярск : Красноярский государственный аграрный университет, 2016. - 123 с.

4. Функционально-структурное моделирование в системе Ramus Educational [Электронный ресурс] : [сайт]. - URL: https://study.urfu.ru/Aid/Publication/13928/1/Кара-Ушанов_Ramus.pdf (дата обращения 04.04.2023). - Загл. с экрана. - Яз. рус.

5. IDEF0. Знакомство с нотацией и пример использования [Электронный ресурс]: [сайт]. - URL: https://trinion.org/blog/idef0-znakomstvo-s-notaciey-i-primer-ispolzovaniya (дата обращения: 04.04.2023). - Загл. с экрана. - Яз.рус.

6. Буч, Г. Язык UML. Руководство пользователя / Г. Буч, Дж. Рамбо, И. Якобсон. – Москва : ДМК Пресс, 2007. - 496 с.

7. Фаулер, М. UML. Основы / М. Фаулер. – Санкт-Петербург : Символ-Плюс, 2004. -

   181 с.

8. Арлоу, Дж. UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование / Дж. Арлоу, А. Нейштадт . – Санкт-Петербург : Символ-Плюс, 2007. - 39 с.

9. Мюллер, Р.Дж. Базы данных и UML. Проектирование / Р.Дж. Мюллер, А. Нейштадт . – Москва : Лори, 2002. - 402 с.

10. Рамбо, Дж. UML 2.0. Объектно-ориентированное моделирование и разработка / Дж. Рамбо, М. Блаха . – Санкт-Петербург : Питер, 2007. - 544 с.

11. Уорсли, Дж. PostgreSQL. Для профессионалов / Дж. Уорсли, Дж. Дрейк. – Санкт-Петербург : Питер, 2003. - 496 с.

12. Болье, А. Изучаем SQL. Генерация, выборка и обработка данных / А. Болье. – Киев : Диалектика, 2021. - 400 с.

13. Тейлор, А. SQL для чайников / А. Тейлор. – Киев : Диалектика, 2020. - 544 с.

14. Шилдс, У. SQL. Быстрое погружение / У. Шилдс. – Санкт-Петербург : Питер, 2022. - 224 с.

15. Уоллс, К. Spring в действии / К. Уоллс. – Москва : ДМК-пресс, 2022. - 544 с.

16. Гутьеррес, Ф. Spring Boot 2 / Ф. Гутьеррес. – Санкт-Петербург : Питер, 2020. - 464 с.

17. Хеклер, М. Spring Boot по-быстрому / М. Хеклер. – Санкт-Петербург : Питер, 2022. - 352 с.

18. Spilca, L. Spring Security in Action / L. Spilca. – New York : Manning, 2020. - 560 с.

19. Winch, R. Spring Security, 3.1 / R. Winch. – Birmingham : Packt Publishing, 2012. - 419 с.

20. Крейн, Д. Ajax в действии / Д. Крейн. – Киев : Вильямс, 2006. - 640 с.

21. Хэнчетт, Э. Vue.js в действии / Э. Хэнчетт. – Санкт-Петербург : Питер, 2020. - 304 с.

22. Книга рецептов Vue.js [Электронный ресурс]: [сайт]. - URL: `https://ru.vuejs.org/v2/cookbook/` (дата обращения: 4.04.2023). - Загл. с экрана. - Яз.рус.

23. Vue.js для начинающих [Электронный ресурс]: [сайт]. - URL: https://habr.com/ru/company/ruvds/blog/509700/ (дата обращения: 4.04.2023). - Загл. с экрана. - Яз.рус.

24. Документация Vue.js [Электронный ресурс]: [сайт]. - URL: https://vueframework.com/docs/v3/ru/ru/guide/introduction.html (дата обращения: 4.04.2023). - Загл. с экрана. - Яз.рус.