Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

ОТЧЕТ

о преддипломной (производственной) практике

наименование вида и типа практики

на (в) Юго-Западном государственном университете

наименование предприятия, организации, учреждения

Студента 4курса, группы ПО-92б

курса, группы

Иванова Ивана Ивановича

фамилия, имя, отчество

Руководитель практики от предприятия, организации, учреждения

директор

должность, звание, степень

Куркина А. В.

фамилия и. о.

Оценка _____

_____

подпись, дата

Руководитель практики от университета

к.т.н. доцент

должность, звание, степень

Чаплыгин А. А.

фамилия и. о.

Оценка _____

_____

подпись, дата

Члены комиссии

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

Курск 2024 г.

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

БД – база данных.

ИС – информационная система.

ИТ – информационные технологии.

UI - User Interface, пользовательский интерфейс.

ООП - объектно-ориентированное программирование.

ПО – программное обеспечение.

РП – рабочий проект.

СУБД – система управления базами данных.

ТЗ – техническое задание.

ТП – технический проект.

CRUD - Create, Read, Update, Delete, основные операции для работы с данными.

ER (Entity-Relationship model) – модель данных, позволяющая описывать концептуальные схемы предметной области. ER-модель используется при высокоуровневом (концептуальном) проектировании баз данных.

СКУД – система контроля и управления доступом

УД – учётные данные

ЧС – чрезвычайная ситуация

PkID(Primary key Identifier, первичный ключ) – в реляционной модели данных один из потенциальных ключей отношения, выбранный в качестве основного ключа (или ключа по умолчанию).

UID(Unique identifier, уникальный идентификатор) – идентификатор, который используют для однозначного определения объекта, однако признак уникальности не заложен в ID.

FKID(Foreign key identifier, внешний ключ) – идентификатор, который применяется для принудительного установления связи между данными в двух таблицах с целью контроля данных, которые могут храниться в таблице внешнего ключа.

TreeView – элемент графического интерфейса для иерархического отображения информации. Представляет собой совокупность связанных отношениями структуры пиктограмм в иерархическом древе.

1 Анализ предметной области

1.1 Характеристика СКУД и её назначение

История контроля доступа гораздо интереснее, чем вы думаете. На протяжении всей истории человечества мы наблюдаем за тем, как в нем используются элементы безопасности. Механические деревянные замки были обнаружены на территории современного Ирака еще в 4000 году до н. э., а гробница фараона Тутанхамона была заперта с помощью веревочного узла. Исторически контроль доступа включал в себя нечто большее, чем просто замки. У стражников королевства были сложные смены, которые позволяли им находиться в определенных местах только в определенное время, чтобы обеспечить круглосуточную охрану. Удивительно, что такие устаревшие технологии, как рвы, разводные мосты и сторожевые башни, когда-то были самыми современными инженерными решениями, используемыми для обеспечения защиты периметра и контролируемого доступа.

Сегодня под термином СКУД (Система контроля и управления доступом) понимается вид физической безопасности, который управляет точками входа в ваш бизнес или внутренние помещения здания. Простой пример СКУД – ворота, физически не допускающие неавторизованных пользователей и позволяющие войти только авторизованным пользователям.

В наше время контроль доступа является сложным процессом, применяющимся для защиты мест, имущества, данных или граждан от личных до огромных корпоративных масштабов. Современные достижения в области технологий позволили создать более надежные и эффективные способы управления и защиты, и современные решения по контролю доступа выходят далеко за рамки стандартных ключей и карт доступа.

Удаленное управление остается важнейшей задачей с начала 2020 года, позволяя предприятиям обеспечивать безопасность своих зданий, даже когда там никого нет, и прокладывая путь к продуктивным гибридным моделям работы. Удаленный доступ и управление безопасностью позволяют как корпоративным, так и небольшим организациям оставаться гибкими, позволяя

командам выполнять повседневные задачи без необходимости физического присутствия на объекте. Такие функции, как удаленное отпирание дверей, полезны для того, чтобы впустить в здание поставщиков, подрядчиков и сотрудников, которые забыли или потеряли свои учетные данные. Благодаря доступу к отчетам о деятельности в режиме реального времени удаленное управление также позволяет организациям гибко перестраиваться на ходу.

Три лидера отрасли, каждый из которых разработал инновационные системы контроля доступа – Keyscan, HID и RBH Access Technologies разрабатывают передовые системы безопасности, которые являются универсальными, масштабируемыми и гибкими, оставаясь при этом высокофункциональными.

1.2 Ключевые элементы системы контроля доступа и принцип их работы

1.2.1 Учётные данные

Учетные данные контроля доступа используют RFID (радиочастотной идентификации) для передачи сигналов на панель контроля доступа. Каждая метка имеет уникальный зашифрованный идентификационный номер. Вы можете выдать всем сотрудникам метки одного типа, но при этом настроить одну метку на разрешение входа, а другую - на запрет входа в определенные зоны здания.

УД бывают следующих видов:

- Приложения для смартфонов
- Пароль или пин-код
- Биометрия
- Брелок или карта доступа

1.2.1.1 Приложения для смартфонов

Для контроля доступа с помощью мобильных устройств используются смартфоны, планшеты и носимые электронные устройства, которые слу-

жат удостоверением личности пользователя для входа в офис или другие деловые помещения. Поскольку все больше работодателей поощряют тенденцию Bring Your Own Device (BYOD), контроль доступа с помощью приложений становится отличным инструментом для обеспечения дополнительного уровня безопасности в любой организации. В настоящее время электронные устройства даже позволяют осуществлять биометрическую аутентификацию без необходимости инвестировать в дорогостоящие биометрические считыватели.

1.2.1.2 Пароль или ПИН-код

Преимущество парольных систем по сравнению с системами дискреционного контроля доступа на основе матрицы доступа заключается в том, что в них нет объекта, связанного с монитором безопасности, который хранит информацию о контроле доступа к конкретным объектам.Кроме того, парольные системы обеспечивают безопасность даже в том случае, если посторонние лица имеют неограниченный или технически возможный доступ к носителям, на которых записаны и хранятся зашифрованные объекты.Эти преимущества парольных систем управления доступом делают их чрезвычайно широко распространенными в документальных информационных системах.

1.2.1.3 Биометрия

Объекты со строгими требованиями к соблюдению норм и правил, такие как больницы и производственные предприятия, требуют особенно надежных систем контроля доступа и безопасности. Биометрические технологии используют измерения тела и физические характеристики для поиска уникальных идентификационных признаков человека (обычно отпечатков пальцев, сканирования сетчатки глаза или лица), чтобы сделать идентификацию положительной.

1.2.1.4 Брелок или карта доступа

Системы контроля доступа на основе карт играют важную роль в защите вашей собственности. Системы доступа без ключа - это эффективный и доступный способ обеспечить безопасность людей, помещений и данных. Карточные системы доступа могут стать отличным выбором, независимо от того, охраняете ли вы склад, общежитие или коммерческое здание.

Простейшая современная СКУД состоит из:

- Считывателя
- Контроллера
- Замка

1.2.2 Считыватель

Считыватель меток устанавливается на одной или обеих сторонах двери - на одной стороне двери, если система контролирует только вход, или на обеих сторонах, если система контроля доступа контролирует вход и выход. Считыватель содержит антенну, которая подключается к панели контроля доступа и получает от нее питание. Когда человек входит в здание со своей меткой контроля доступа, на антенну считывателя поступает его зашифрованный идентификационный номер.

1.2.3 Контроллер

Контроллер - это ядро системы. В нем хранится информация об авторизации, которая настраивается администратором системы. Контроллер получает зашифрованный номер метки от считывателя, расшифровывает его, затем сравнивает ID-номер с ID-номерами, которые были загружены в систему. Если номера совпадают, и пользователь имеет право доступа к двери, дверь разблокируется.

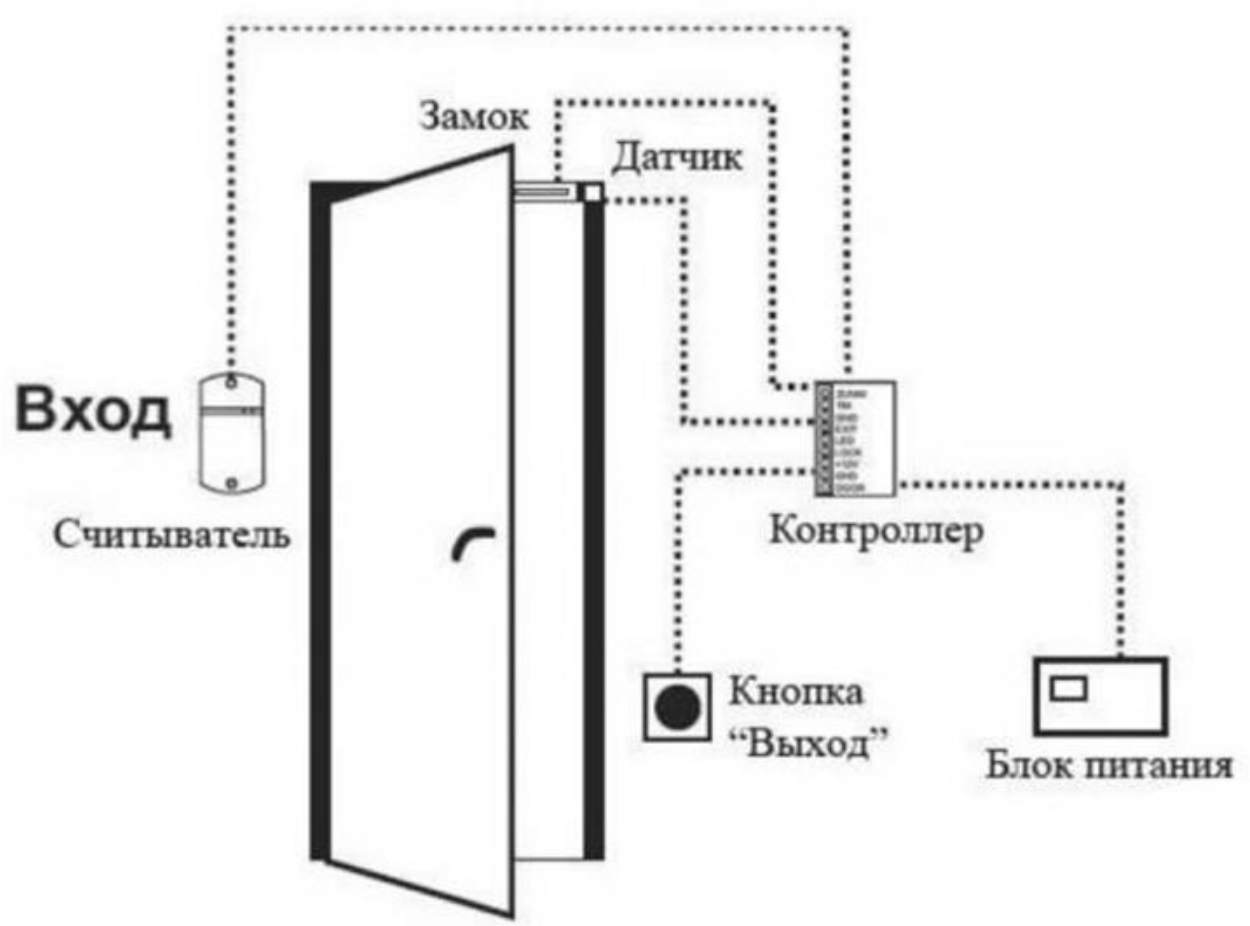

Рисунок 1.1 – Схема устройства современной СКУД

1.2.4 Замок

Панель управления доступом, или контроллер, управляет электрическим замком двери. Если пользователю разрешено войти в здание, дверь автоматически разблокируется и может быть открыта. Общая схема устройства системы изображена на рисунке 1.1.

1.3 Анализ аудитории пользователей

Система контроля и управления доступом - важный аспект безопасности. Она обеспечивает дополнительный уровень защиты, позволяя контролировать и отслеживать, кто имеет доступ к объекту защиты.

СКУД применима во многих отраслях - в здравоохранении, корпоративном, образовательном и государственном секторах. Сегодня они также при-

меняются на различных объектах, таких как стационарные, долговременные и жилые учреждения, больницы, муниципалитеты, кондоминиумы, распределительные комплексы, малые и крупные предприятия, а также колледжи и университеты.

Среди аудитории пользователей программного обеспечения СКУД – сисадмины и специалисты по информационной безопасности, нанятые организацией специально для контроля правил системы, устранения неполадок, проведения аудитов безопасности, поддержания работоспособности, внесения изменений.

В нашем случае, на круизном лайнере, аудиторией приложения является человек из экипажа, занимающийся внесением данных пассажиров в базу, наложением штрафов, созданием правил для входа в те или иные сектора, манипуляцией дверьми во время ЧС и т.д.

1.4 Перспективы развития

В перспективе развития СКУД на круизном лайнере ключевым является масштабирование её возможностей. Для самой системы это может быть увеличение быстродействия, расширение пула правил и ограничений, добавление умных систем обнаружения (камер) для фиксации нарушений и автоматического наложения штрафов, внедрение технической поддержки в реальном времени для нерядовых случаев.

В перспективе развития ПО для СКУД ключевым является наиболее точное отражение функций системы для воздействия на них со стороны специалиста.

В будущем можно рассмотреть интеграцию бэкенда в виде набора новых бизнес-правил, дополнительного уровня защиты базы данных от атак злоумышленников, таких как SQL-инъекции, а фронтенда – как более широкую реализацию библиотеки "tkinter ", что обеспечит более гибкое управление ресурсами и улучшит масштабируемость приложения.

От веревок и деревянных замков до облачных систем и биометрии - безопасность прошла долгий путь сквозь века. Аутентификация повсюду -

от самых больших и надежных объектов до телефона в вашем кармане. При взгляде на будущие тенденции в области контроля доступа можно с уверенностью сказать одно: инновации, интеграция и адаптивность необходимы, но не в ущерб безопасности. Будущее контроля доступа, особенно для предприятий, выходит за рамки защиты данных и ограничения доступа и позволяет делать все это быстрее и с большей надежностью.

1.5 Сценарий проекта. Неформальное представление предметной области

У нас небольшая фирма, занимающаяся разработкой СКУД. Мы выиграли контракт на разработку СКУД для небольшого круизного лайнера AIDABlu.

В основные функции СКУД входит: идентификация лиц и объектов (транспортных средств), имеющих право доступа на объект, регистрация входа-выхода (въезда-выезда), управление уровнями доступа для персонала. В СКУД входят все сущности, так или иначе связанные с предоставлением доступа. В данные сущности входят: Пассажиры, Двери, Комнаты, Штрафы. У каждой двери есть свод правил, определяющих, открыть ли её пассажиру или нет.

В обязанности СКУД входит: Контроль доступа (ДОСТУПЫ) в обычном режиме и в случае ЧС(ДОСТУПЫ-ЧС). Доступ каждого пассажира(ПАССАЖИРЫ) к каждой двери(ДВЕРИ) предоставляется/не предоставляется на основе данных ограничений пассажира и двери. Дверь может вести в комнату(КОМНАТЫ), также имеющую данные ограничений для предоставления доступа. В процессе поездки пассажир может получить штраф(ШТРАФЫ), что является отдельным ограничением, закреплённым за пассажиром. Так как не все пассажиры могут являться совершеннолетними(ДЕТИ), за ними должен быть закреплён взрослый сопровождающий, несущий за него ответственность на время поездки.

1.6 Бизнес-правила

На основе анализа неформального описания предметной области были сформулированы бизнес-правила, представленные в таблице 1.1.

Таблица 1.1 – Бизнес-правила

	Бизнес-правило	Ограничение
1.	Каждый человек на борту имеет постоянный доступ к жизненно-необходимым дверям:(коридорные и лестничные, туалет, собственный номер, столовая, выход на площадку, медпункт, детская комната, "Дьюти-фри")	Помещение переполнено, шторм(для площадки), отсутствие несовершеннолетнего спутника(дет.комната), пол (каждому – свой туалет). В этом случае доступ не предоставляется.
2.	Доступ к комнатам идёт далее по иерархической системе. Так, пользователи «комфорта» получают доступ к: (бару, аквапарку, кинотеатру, кальянной, бильярдной, банкетному залу, тиру)	Возраст(бар/кальянная), мед. Ограничения (аквапарк), время(банкетный зал), судимости (тир),штрафы, накладываемые после проишествий.
3.	Кроме вышеперечисленного, пользователям «Все включено» открывается доступ к: (SPA, пентхаусу, рыболовной площадке, комнате для погружений, VIP-кинотеатру)	Мед.ограничения(пассажир может посетить SPA, если того требует здоровье и не может посещать комнату для погружений).
4.	Персонал имеет доступ ко всем дверям	Персонал не имеет доступа к личным номерам пассажиров

Продолжение таблицы 1.1

	Бизнес-правило	Ограничение
5.	Пассажир может менять тариф в процессе путешествия	Это происходит в случае ЧС или доплаты персоналу. А в случае персонала – при отстранении от полномочий на время поездки(Тариф меняется на «эконом»).
6.	Штраф пассажира может быть снят	Только если штраф относится к снимаемым, прошли 1 сутки с момента наложения и пассажир заплатил выкуп за снятие штрафа.
7.	Каждый человек на борту должен иметь свой пропуск	Количество пропусков на каждого человека не должно превышать 1
8.	Несовершеннолетние также имеют свой пропуск	Для разблокировки пропуска, за ним должен быть закреплён сопровождающий
9.	Ограничения пассажира могут быть как постоянными, так и наоборот	К ограничениям, которые не меняются и не оспариваются во время поездки относятся: Судимости и мед.ограничения
10.	У каждого пассажира должна быть своя комната	Пассажир может войти только в ту жилую комнату, что закреплена за ним
11.	В случае возникновения ЧС, СКУД начинает работать в другом режиме	В случае шторма все двери наружу закрываются (состояние), двери лестниц/коридоров всегда открыты, дабы не создавать давки и паники.

2 Техническое задание

2.1 Основание для разработки

Полное наименование системы: «Программное обеспечение для системы контроля и управления доступом на круизном судне». Основанием для разработки программы является приказ ректора ЮЗГУ от «15» апреля 2024 г. №1779-с «Об утверждении тем выпускных квалификационных работ».

2.2 Цель и назначение разработки

Целью данной работы является проектирование базы данных круизного лайнера и разработка приложения для карт доступа пассажиров.

В настоящее время перед СКУД стоят задачи: четкое и быстрое регулирование правил доступа во время поездки и обеспечение безопасной эвакуации пассажиров в случае ЧС. Для выполнения этих задач требуется систематизация данных.

Основными задачами при проектировании и разработке базы данных и приложения являются:

- исследование предметной области;
- проектирование базы данных;
- создание базы данных;
- заполнение базы данных информацией;
- разработка интерфейса;
- реализация приложения.

2.3 Требования к интерфейсу

Приложение должно включать в себя:

- навигацию по таблицам;
- отдельные интерфейсы для каждой таблицы с полями;
- отображение текущей таблицы в реальном времени(TreeView);
- возможность так или иначе воздействовать на внесённые данные базы внутри приложения;

- понятную навигацию и легкий поиск среди элементов конкретной таблицы;

- отображение локального времени во избежание ошибок;

2.4 Требования к программной системе приложения «СКУД на круизном лайнере»

2.4.1 Требования к данным программной системы

Система должна уметь эффективно обрабатывать данные пассажиров, включая личную информацию, данные о тарифе, информацию о штрафах и спутниках. Необходимо обеспечить конфиденциальность и безопасность этих данных.

Потенциальные объекты:

- ДОСТУПЫ;
- ДОСТУПЫ-ЧС;
- ПАССАЖИРЫ;
- ДВЕРИ;
- КОМНАТЫ;
- ШТРАФЫ;
- ДЕТИ;

2.5 Построение ER-модели данных

На основе анализа неформального описания предметной области были определены наборы объектов и связей.

2.5.1 Определение объектов

Потенциальные объекты и атрибуты, включая необязательные: Пассажир

- (Первичный)ID
- *Имя
- *Фамилия

- *Возраст
- *Пол
- *Тариф
- ○Медицинские заметки
- ○Судимости
- *Комната
- ○Образование
- ○Квалификация

Ребёнок

- (Первичный)ID
- *Имя
- *Фамилия
- *Возраст
- *Пол
- *Тариф
- *Спутник
- ○Медицинские заметки
- ○Судимости
- *Комната

Дверь

- (Первичный)ID
- *Наименование
- *Комната,в которую ведёт
- ○Системное время
- ○Номер
- *Статус
- *Скорость закрытия
- ○Ограничитель

Комната

- (Первичный)ID
- *Наименование

- *Кол-во дверей
- *Тип
- ○Ограничение по времени (ближайшее мероприятие)
- ○Ограничение по полу
- ○Ограничение по медицинской карте
- ○Ограничение по судимости
- ○Ограничение по штрафу

Штраф

- (Первичный)ID
- ○Время наложения
- *Действителен ли
- *Возможно ли снять
- ○Сумма выкупа
- *Вид

2.5.2 Определение связей

Слева направо: Каждый пассажир (ПАССАЖИР) может сопровождать ребёнка(РЕБЁНОК).

Справа налево: каждый ребёнок(РЕБЁНОК) должен быть сопровождаем Пассажиром(ПАССАЖИР).

Слева направо: Каждый пассажир(ПАССАЖИР) или ребёнок(РЕБЁНОК) имеет свою личную комнату(КОМНАТА).

Справа налево: Каждая жилая комната(КОМНАТА) может принадлежать пассажиру(ПАССАЖИР) или ребёнку(РЕБЁНОК).

Слева направо: Каждый пассажир(ПАССАЖИР) может иметь штрафы(ШТРАФ).

Справа налево: каждый штраф(ШТРАФ) должен быть закреплён за пассажиром(ПАССАЖИР)-нарушителем.

Слева направо: каждая дверь(ДВЕРЬ) может вести в комнату(КОМНАТА).

Справа налево: Каждая комната(КОМНАТА) имеет ведущие в неё двери (ДВЕРЬ).

Слева направо: каждый пассажир(ПАССАЖИР) или ребёнок(РЕБЁНОК) имеют доступы к дверям(ДВЕРЬ).

Справа налево: Каждая дверь(ДВЕРЬ) предоставляет доступы пассажирам(ПАССАЖИР) или детям(РЕБЁНОК).

2.5.3 Определение кардинальности связей

Слева направо: Каждый пассажир (ПАССАЖИР) может сопровождать одного и более ребёнка(РЕБЁНОК).

Справа налево: каждый ребёнок(РЕБЁНОК) должен быть сопровождаем одним Пассажиром(ПАССАЖИР).

Слева направо: Каждый пассажир(ПАССАЖИР) или ребёнок(РЕБЁНОК) имеет одну личную комнату(КОМНАТА).

Справа налево: Каждая жилая комната(КОМНАТА) может принадлежать одному или многим пассажирам(ПАССАЖИР) или детям(РЕБЁНОК).

Слева направо: Каждый пассажир(ПАССАЖИР) может иметь 1 или несколько штрафов(ШТРАФ).

Справа налево: каждый штраф(ШТРАФ) должен быть закреплён за одним пассажиром(ПАССАЖИР)-нарушителем.

Слева направо: каждая дверь(ДВЕРЬ) может вести в одну комнату(КОМНАТА).

Справа налево: Каждая комната(КОМНАТА) имеет одну или несколько дверей (ДВЕРЬ).

Слева направо: каждый пассажир(ПАССАЖИР) или ребёнок(РЕБЁНОК) имеют доступ к одной или нескольким дверям(ДВЕРЬ).

Справа налево: Каждая дверь(ДВЕРЬ) предоставляет доступы сразу нескольким пассажирам(ПАССАЖИР) или детям(РЕБЁНОК).

2.5.4 Разрешение связи «Многие ко многим»

Описание решения Связи M:M между объектами ПАССАЖИР и ДВЕРИ, РЕБЁНОК И ДВЕРИ необходимо решить с помощью объектов пересечения.

Это объекты пересечения с именами ДОСТУПЫ и ДОСТУПЫ ЧС, в которых хранится информация о предоставленном доступе к дверям в обычной и чрезвычайной ситуациях, исходя из сравнения ограничений с обеих сторон. Это дает нам возможность привязывать пассажиров/детей к дверям.

Идентификаторы PkID для объектов ДОСТУП/ДОСТУП ЧС использует ассоциативную связь для применения UID из обоих объектов ПАССАЖИР(РЕБЁНОК) и ДВЕРЬ в качестве PkID.

После добавления объектов пересечения появились следующие связи:

Слева направо: Каждый Пассажир(ПАССАЖИР) имеет несколько доступов(ДОСТУП) к дверям и доступов к дверям в случае ЧС (ДОСТУП ЧС).

Справа налево: Каждый Доступ(ДОСТУП) или (ДОСТУП ЧС) предоставляется лишь одному пассажиру(ПАССАЖИР).

Слева направо: Каждый Ребёнок имеет несколько доступов(ДОСТУП) к дверям и доступов к дверям в случае ЧС (ДОСТУП ЧС).

Справа налево: Каждый Доступ(ДОСТУП) или (ДОСТУП ЧС) предоставляется лишь одному ребёнку(РЕБЁНОК).

Слева направо: Каждая дверь(ДВЕРЬ) предоставляет несколько доступов(ДОСТУП)/(ДОСТУП ЧС).

Справа налево: каждый доступ (ДОСТУП)/(ДОСТУП ЧС) запрашивается только от одной двери(ДВЕРЬ)

2.5.5 Извлечение вторичных ключей из простых объектов с помощью внешних ключей

Связь 1:M между объектами ПАССАЖИР и ШТРАФ необходимо решить с помощью внешнего ключа. Внешний ключ хранится в поле объекта

ШТРАФ(Принадлежность) и является обязательным. Это дает нам возможность привязывать каждый штраф к нарушителю.

Идентификатор FkID для объекта Штрафы ссылается на UID Пассажира. Связь 1:M между объектами ПАССАЖИР и РЕБЁНОК необходимо решить с помощью внешнего ключа. Внешний ключ хранится в поле объекта РЕБЁНОК(Спутник) и является обязательным. Это дает нам возможность привязывать каждого ребёнка к его сопровождающему.

Идентификатор FkID для объекта Ребёнок ссылается на UID Пассажира. Связь M:1 между объектами ПАССАЖИР и КОМНАТА необходимо решить с помощью внешнего ключа. Внешний ключ хранится в поле объекта ПАССАЖИР(Комната) и является обязательным. Это дает нам возможность закреплять каждого пассажира за своей комнатой.

Идентификатор FkID для объекта ПАССАЖИР ссылается на поле номера ДВЕРИ, ведущей в эту КОМНАТУ. Связь M:1 между объектами РЕБЁНОК и КОМНАТА необходимо решить с помощью внешнего ключа. Внешний ключ хранится в поле объекта РЕБЁНОК(Комната) и является обязательным. Это дает нам возможность закреплять каждого пассажира за своей комнатой.

Идентификатор FkID для объекта РЕБЁНОК ссылается на поле номера ДВЕРИ, ведущей в эту КОМНАТУ. Связь M:1 между объектами ДВЕРЬ и КОМНАТА необходимо решить с помощью внешнего ключа. Внешний ключ хранится в поле объекта ДВЕРЬ(Комната,в которую ведёт) и является обязательным. Это дает нам возможность закреплять каждую комнату за ей принадлежащими дверьми.

Идентификатор FkID для объекта ДВЕРЬ ссылается на UID КОМНАТЫ.

В приведенной на рисунке 2.1 ER-диаграмме представлены сущности и атрибуты, которые будут использоваться в программной системе

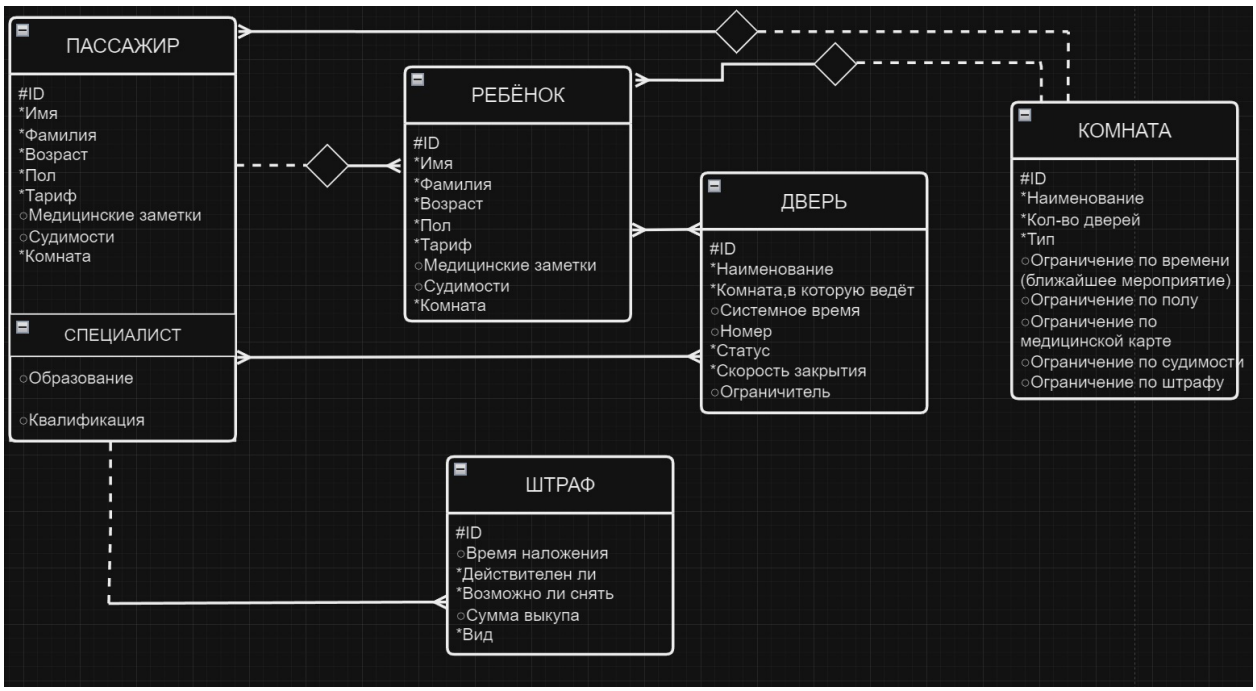

Рисунок 2.1 – ER- диаграмма

2.5.6 Архитектура системы

Приложение разработано на языке Python с использованием библиотеки tkinter. Через библиотеку sqlite3 программа взаимодействует с СУБД SQLite.

2.5.7 Функциональные требования к программной системе

На основании анализа предметной области, разрабатываемая программная система «СКУД на круизном лайнере» должна включать в себя следующие ключевые функциональные возможности:

- Открыть/закрыть таблицу
- Редактировать таблицу
- Перейти к следующему/предыдущему элементу
- Добавить новый элемент – Реализация опции добавления нового элемента с последующей актуализацией этих данных в БД
- Редактировать выбранный элемент – Реализация опции редактирования элемента с последующей актуализацией этих данных в БД
- Найти элемент по идентификатору

- Удалить элемент – Реализация опции редактирования элемента с последующей актуализацией этих данных в БД

2.5.8 Моделирование вариантов использования

На рисунке 2.2 представлена диаграмма прецедентов, которая служит важным средством для систематизации функциональных требований и возможностей системы, выявляя ключевые сценарии использования. Диаграмма прецедентов изображает, как пользователи могут взаимодействовать с системой, а также помогает выявить набор основных функциональных возможностей, доступных пользователям.

При построении диаграммы вариантов использования применяется унифицированный язык визуального моделирования UML.

Каждый прецедент на диаграмме отражает отдельный путь взаимодействия в системе и представляет потенциальные действия, которые могут быть предприняты пользователями в различных ролях. Пользователем или действующим лицом является сущность, взаимодействующая с системой извне (например, человек, техническое устройство).

2.5.9 Вариант использования «Открыть/закрыть таблицу»

Пользователь выбирает интересующую его таблицу из списка предложенных и открывает окно, репрезентирующее требуемый от приложения интерфейс.

2.5.10 Вариант использования «Редактировать таблицу»

Пользователь решает внести какие-либо изменения в таблице. Он может сделать это, добавляя новые элементы, удаляя или редактируя существующие.

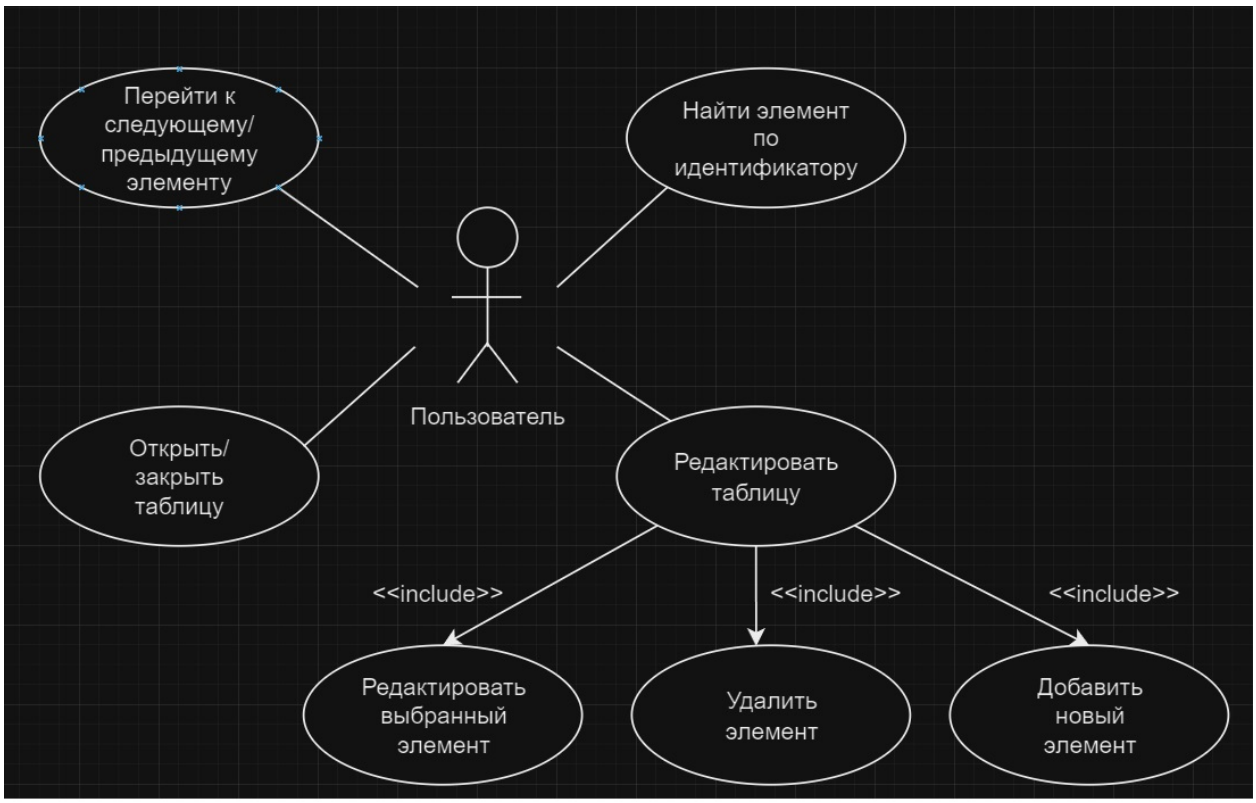

Рисунок 2.2 – Диаграмма прецедентов

2.5.11 Вариант использования «Перейти к следующему/предыдущему элементу»

Пользователь листает элементы в таблице посредством перехода к предыдущему/следующему за текущим.

2.5.12 Вариант использования «Найти элемент по идентификатору»

Пользователь решает не листать элементы по одиночке т.к. не хочет тратить на это время. Он помнит идентификатор искомого элемента, вводит его в предназначенное текстовое поле и переходит к искомому элементу.

2.5.13 Вариант использования «Добавить новый элемент»

Пользователь заполняет поля данных будущего объекта и добавляет его в таблицу по клику.

2.5.14 Вариант использования «Редактировать выбранный элемент»

Пользователь редактирует необходимые ему поля у текущего элемента и вносит эти изменения в таблицу по клику.

2.5.15 Вариант использования «Удалить элемент»

Пользователю больше не нужен выбранный элемент и он стирает его в таблице по клику.

2.6 Нефункциональные требования к программной системе

2.6.1 Требования к надежности

Для обеспечения надёжной работы базы данных, она должна быть приведена в нормализированный по трём формам вид: -Первая нормальная форма (1NF)

Все атрибуты объекта ПАССАЖИР имеют только одно значение.

Все атрибуты объекта РЕБЁНОК имеют только одно значение.

Все атрибуты объекта ДВЕРЬ имеют только одно значение.

Все атрибуты объекта КОМНАТА имеют только одно значение.

Все атрибуты объекта ШТРАФ имеют только одно значение.

Все атрибуты объекта пересечения ДОСТУПЫ имеют только одно значение.

Все атрибуты объекта пересечения ДОСТУПЫ ЧС имеют только одно значение.

-Вторая нормальная форма (2NF)

Атрибуты объекта ПАССАЖИР зависят от полного UID (id пассажира).

Атрибуты объекта РЕБЁНОК зависят от полного UID (id ребёнка).

Атрибуты объекта ДВЕРЬ зависят от полного UID (id двери).

Атрибуты объекта КОМНАТА зависят от полного UID (id Комнаты).

Атрибуты объекта ШТРАФ зависят от полного UID (id штрафа).

Атрибуты объекта пересечения ДОСТУПЫ зависят от двойного UID.

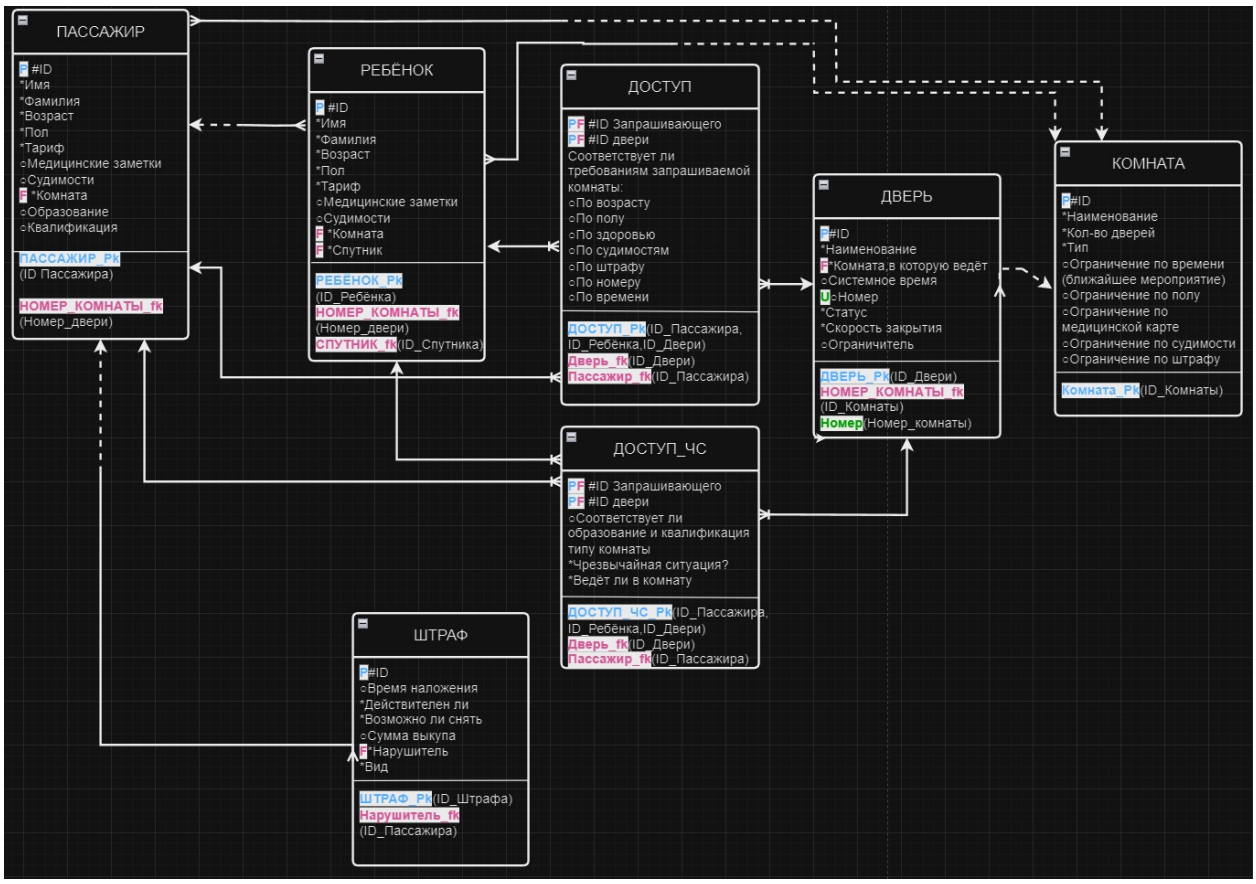

Рисунок 2.3 – Реляционная модель данных

Атрибуты объекта пересечения ДОСТУПЫ ЧС зависят от двойного UID.

-Третья нормальная форма (3NF) При проверке всех объектов не было выявлено независимых от UID атрибутов или атрибутов, зависимых от других атрибутов.

На основе ER-модели данных построена реляционная модель данных, которой должна соответствовать БД. Она показанная на рисунке 2.3

2.6.2 Требования к безопасности

1. Поддержка современных операционных систем, обеспечивающая стабильную и безопасную работу на различных платформах;

2. Реализация защиты от атак типа SQL-инъекций, предотвращающая нежелательное вмешательство в работу приложения;

3. Регулярное обновление компонентов системы для предотвращения уязвимостей, связанных с устаревшим программным обеспечением.

2.6.3 Требования к программному обеспечению

Для реализации программной системы должны быть использованы следующие языки программирования:

Python - основной язык реализации приложения;

SQL - язык структурированных запросов к базе данных.

Для работы и приложения требуется ОС Windows 7 или более поздняя версия/Ubuntu 16.0 или более поздняя версия с установленным интерпретатором языка python и установленной СУБД SQLite.

2.6.4 Требования к аппаратному обеспечению

Системе необходим центральный процессор с количеством ядер от 2 и выше с частотой ядра от 1.4 ГГц. Объем оперативной памяти – 4 Гб. Свободное место на диске: 13МБ

2.7 Требования к оформлению документации

Требования к стадиям разработки программ и программной документации для вычислительных машин, комплексов и систем независимо от их назначения и области применения, этапам и содержанию работ устанавливаются ГОСТ 19.102–77. Программная документация должна включать в себя:

1. Анализ предметной области;
2. Техническое задание;
3. Технический проект;
4. Рабочий проект.

3 Технический проект

3.1 Общая характеристика организации решения задачи

Необходимо спроектировать и разработать приложение, которое обеспечит функционирование СКУД на круизном лайнере AIDABlu.

Приложение представляет собой панель управления, представляющую набор взаимосвязанных таблиц в базе данных системы. Панель содержит текстовую и графическую информацию (TreeView).

Приложение является десктопным, т.е. располагается и запускается внутри лишь одной операционной системы. Каждое окно приложения (за исключением главного) – это панель управления для конкретной таблицы базы данных. Приложение написано на языке Python v3.10. Управление таблицами БД реализовано с помощью библиотеки sqlite3 и SQL – языка структурированных запросов к базе данных.

3.2 Общие сведения о программно-информационной системе

Полное наименование системы: Программное обеспечение для системы контроля и управления доступом на круизном судне.

Краткое обозначение системы: "СКУД на круизном лайнере ".

Описание системы: "СКУД на круизном лайнере "предназначена для корпораций, организующих круизные путешествия, предоставляя им платформу для удобного контроля и управления данными всей бизнес-системы круизного лайнера на время поездки. Система создана для обеспечения комфортной и безопасной поездки каждого пассажира и функционирования даже в форс-мажорных ситуациях.

Условия эксплуатации: "СКУД на круизном лайнере "предназначена для использования как в нормальных, так и в чрезвычайных условиях работы.

Архитектура системы: Программное обеспечение основано на десктопной архитектуре, используя современные технологии разработки, вклю-

чая TKinter для Front-End части и sqlite3 для реализации запросов к БД в Back-End. Система использует базу данных SQLite.

Технологии и инструменты: В разработке использовались tkinter, sqlite3, asyncio, time

3.3 Обоснование выбора технологии проектирования

3.3.1 Описание используемых технологий и языков программирования

В процессе разработки приложения используются программные средства и языки программирования. Каждое программное средство и каждый язык программирования применяется для круга задач, при решении которых они необходимы.

3.3.2 tkinter

Выбор tkinter для Front-End обосновывается его простотой и кроссплатформенной поддержкой, а также минимальными требованиями к интерфейсу в пользу быстродействия.

Пакет tkinter («интерфейс Tk») - это интерфейс Python для создания GUI. Tkinter доступен на большинстве платформ Unix, включая macOS, а также на системах Windows.

Tkinter входит в состав большинства инсталляций Python, что делает его легкодоступным для разработчиков, которые хотят создавать приложения с графическим интерфейсом, не требуя дополнительных инсталляций или библиотек.

3.3.3 SQLite

Выбор SQLite в качестве системы управления базами данных также обосновывается его удобством как на этапе проектирования, так и на этапе реализации.

SQLite - это библиотека на языке C, которая предоставляет легкую дисковую базу данных, не требующую отдельного серверного процесса и позволяющую обращаться к базе данных с помощью нестандартного варианта языка запросов SQL. Некоторые приложения могут использовать SQLite для внутреннего хранения данных. Также можно создать прототип приложения с использованием SQLite, а затем перенести код на более крупную базу данных, такую как PostgreSQL или Oracle.

3.3.4 asyncio и time

Модули asyncio и time были использованы для актуализации работы системы в реальном времени и многопоточном режиме.

3.3.5 Язык структурированных запросов к базе данных SQL

SQL - это стандартизированный язык программирования, который используется для управления реляционными базами данных и выполнения различных операций над данными в них.

SQL используется для следующего:

- изменение структуры таблиц и индексов базы данных;
- добавление, обновление и удаление строк данных;
- извлечение подмножеств информации из реляционных систем управления базами данных (РСУБД).

3.3.6 Язык программирования Python

3.3.6.1 Достоинства языка Python

Python - очень продуктивный язык. Благодаря простоте Python разработчики могут сосредоточиться на решении проблемы. Написание кода экономит время и освобождает его для более ёмкой работы с другими составляющими проекта.

Python поставляется под лицензией OSI с открытым исходным кодом. Это делает его свободным для использования и распространения. Можно за-

гружать исходный код, изменять его и даже распространять свою версию Python. Это полезно для организаций, которые хотят изменить некоторые специфические функции и использовать свою версию для разработки.

Стандартная библиотека Python огромна, в ней можно найти практически все функции, необходимые для решения любой задачи. Таким образом, не придется зависеть от внешних библиотек.

Во многих языках, таких как C/C++, для запуска программы на разных платформах необходимо изменять код. С Python дело обстоит иначе. Вы пишете один раз и запускаете программу в любом месте.

3.3.6.2 Недостатки языка Python

Язык программирования Python использует большой объем памяти. Это может быть недостатком при создании приложений, когда мы предпочитаем оптимизацию памяти.

Python обычно используется для программирования на стороне сервера. Мы не видим Python на стороне клиента или в мобильных приложениях.

Python - динамически типизированный язык, поэтому тип данных переменной может измениться в любой момент. Переменная, содержащая целое число, в будущем может стать строкой, что может привести к ошибкам времени выполнения.

3.4 Проектирование пользовательского интерфейса

На основании требований к пользовательскому интерфейсу, представленных в пункте 2.3 технического задания, был разработан графический интерфейс десктопного приложения с применением python tkinter, ttk и SQLite. Этот процесс подчеркивает важность интуитивно понятного и эффективного взаимодействия с пользователем. Разработанный интерфейс ориентирован на обеспечение легкости в использовании и интуитивного понимания функционала приложения, предоставляя пользователю простое и эффективное взаимодействие с приложением.

1. Навигация по таблицам: Реализация функции навигации на основе полей psr_btn, psr_ua_btn, drs_btn, rms_btn, pns_btn;

2. Отдельные интерфейсы для каждой таблицы с полями: диверсификация интерфейсов происходит на основе метода show_table;

3. Отображение текущей таблицы в реальном времени: актуальность этого графического элемента (дерева) поддерживается за счёт методов TreeRefresh и TreeCreate;

4. Возможность так или иначе воздействовать на внесённые данные базы внутри приложения: Добавление, изменение или удаление элементов реализованы в методах add_element, update_element и delete_element соответственно, внутри которых формирование строк запросов к БД реализовано с помощью класса Utils.

5. Понятная навигация и лёгкий поиск среди элементов конкретной таблицы: Пользователь может переходить по элементам последовательно (метод MoveTo класса main), или же использовать отдельное текстовое поле для метода search_element, перейдя сразу к искомому элементу таблицы.

6. Отображение локального времени: Учитывая специфику проекта – систему для круизного лайнера, учтём, что часовой пояс может измениться во время путешествия. На уровне python и SQL была установлена привязка к локальному времени (localtime), а функционал часов запущен в асинхронном потоке во избежание ошибок и помех работы основной системы.

3.5 Диаграмма компонентов и схема обмена данными между файлами компонента

Диаграмма компонентов описывает особенности физического представления разрабатываемой системы. Она позволяет определить архитектуру системы, установив зависимости между программными компонентами, в роли которых может выступать как исходный, так и исполняемый код. Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы, а также зависимости между ними. На рисунке 3.1 изображена диаграмма компонентов для проектируемой системы. Она включа-

ет в себя сервер с операционной системой, на которой установлена система управления содержимым, включающая в себя базу данных и интерфейс. Помимо этого на диаграмме изображен клиентский компьютер с операционной системой, на которой установлен браузер.

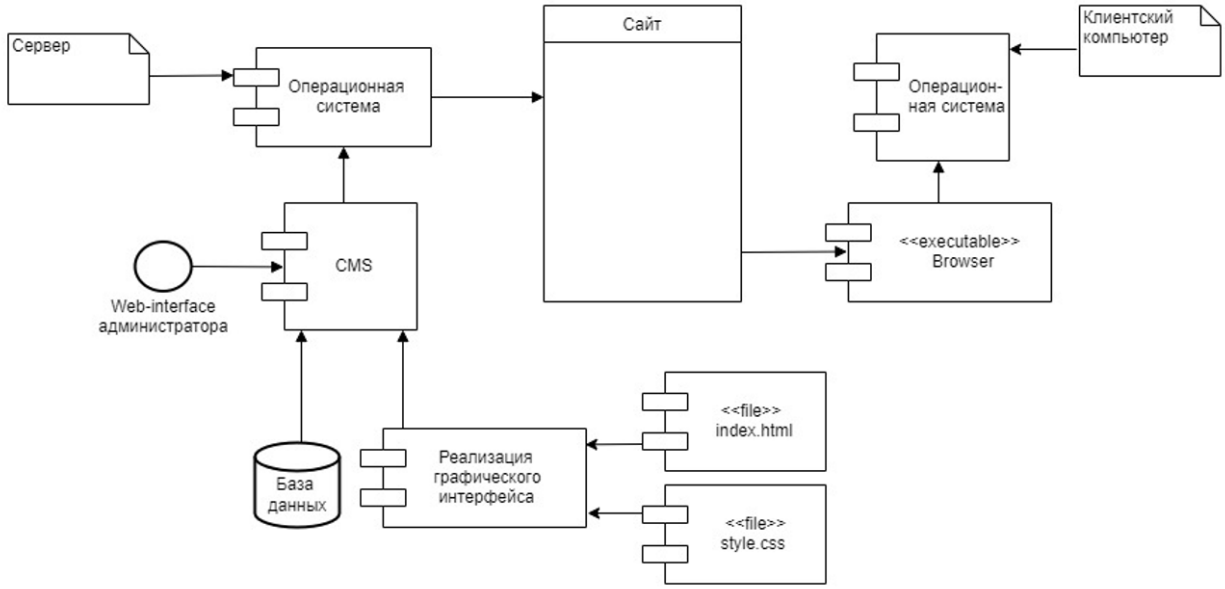

Рисунок 3.1 – Диаграмма компонентов

Любой компонент должен быть вызван в сценарии страницы web-сайта. Web-страница передает данные компоненту в момент вызова последнего.

На рисунке 3.2 представлена схема обмена данными между сценариями компонента при вызове компонента на странице сайта.

При вызове компонента в сценарии web-страницы указываются значения параметров компонента, которые далее посредством массива $arParams передаются в сценарий файла component.php.

В сценарии файла component.php посредством метода IncludeComponentTemplate класса CBitrixComponent происходит вызов одного из шаблонов компонента. Id шаблона также определяется в сценарии страницы web-приложения и неявно для разработчика передается указанный выше метод. Подключается сценарий файла template.php одного из шаблонов, в который передается, возможно, измененный в сценарии

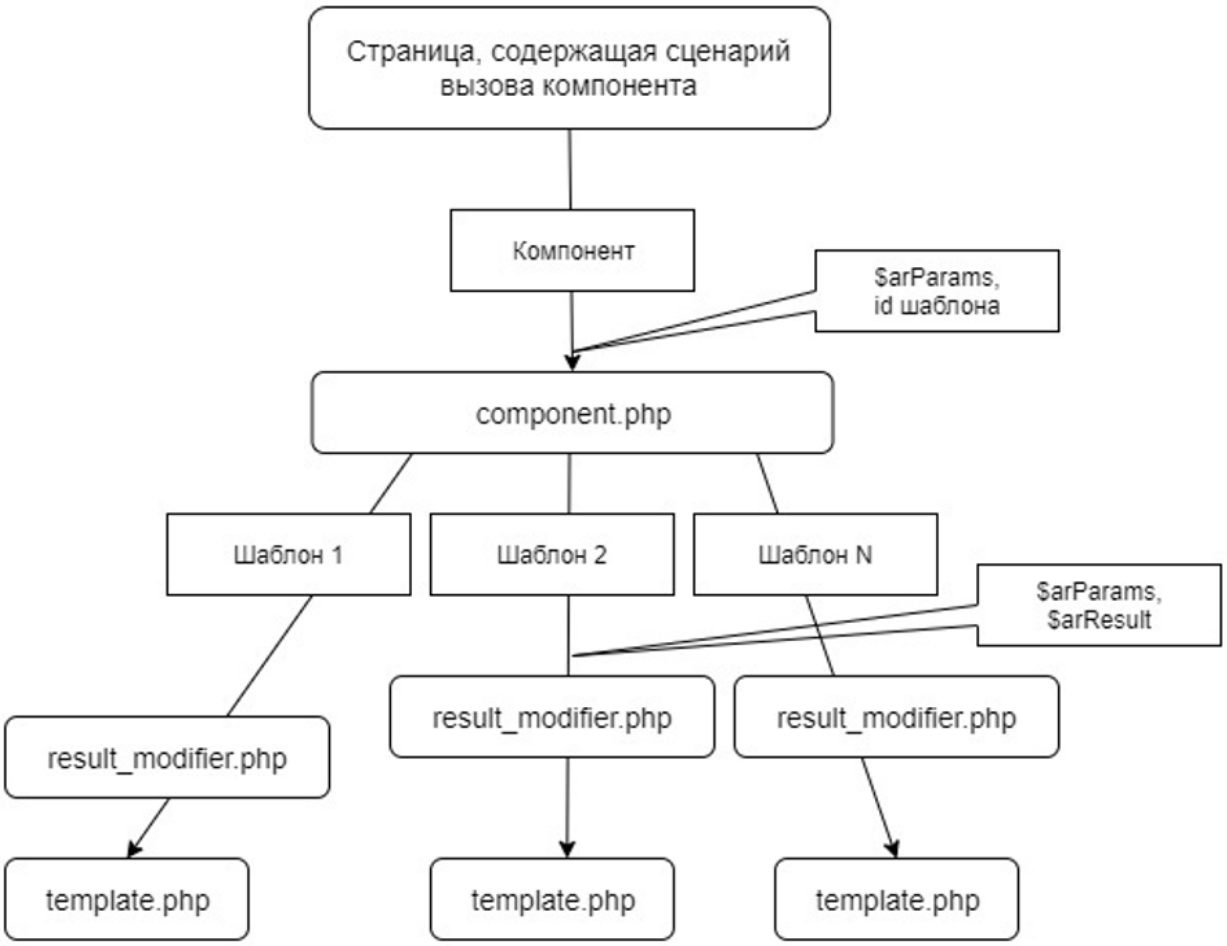

Рисунок 3.2 – Диаграмма компонентов

component.php массив $arParams и, также, сформированный в сценарии component.php массив $arResult. Оба этих массива доступны также и в файле result_modifier.php, который подключается перед подключением файла template.php.

Работа компонента заканчивается в момент завершения работы сценария файла component.php, т.е. возможно выполнить действия уже после подключения шаблона. Однако, если массив $arResult будет изменен в сценарии шаблона, в сценарий файла компонента component.php измененные данные переданы не будут.

3.6 Диаграмма размещения

Диаграмма размещения (рис. 3.3) отражает физические взаимосвязи между программными и аппаратными компонентами системы.

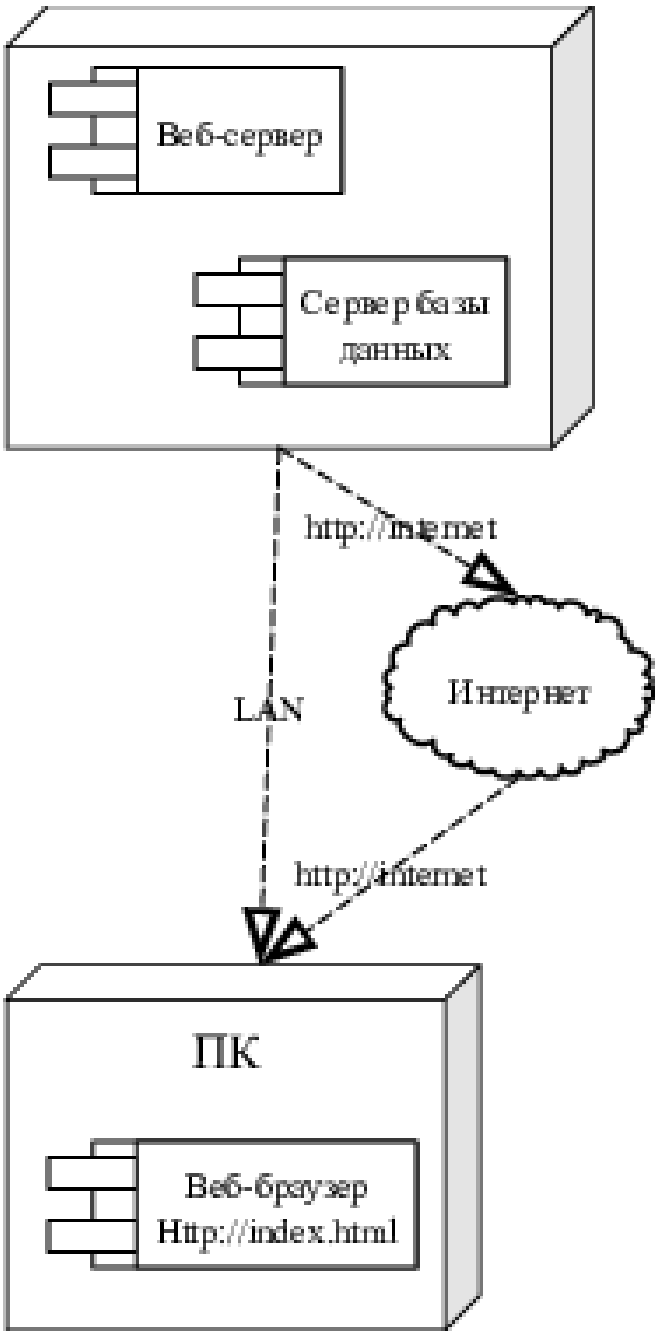

Рисунок 3.3 – Диаграмма размещения. Не помещается на страницу. Очень длинный заголовок

Она является хорошим средством для показа маршрутов перемещения объектов и компонентов в распределенной системе.

В таблице 3.1 приведен пример использования пакета xltabular с автоматическим расчетом ширины столбца.

Таблица 3.1 – Сравнение протоколов SSE и WebSocket

	SSE	WebSocket
Направленность	Однонаправленный, полудуплексный: данные посылает только сервер	Двунаправленный, полнодуплексный: и сервер, и клиент могут обмениваться сообщениями
Соединение	HTTP	WS
Тип данных	Только текст	Бинарные и текстовые данные
Доп. возможности	Встроенный механизм идентификаторов событий и переподключения	Переподключение и идентификация события реализуются на стороне приложения

3.7 Содержание информационных блоков. Основные сущности

Проанализировав требования, можно выделить шесть основных сущностей:

- «Новости»;
- «Продукция»;
- «Услуги».

В состав сущности «Новости» можно включить атрибуты, представленные в таблице 3.3.

Таблица 3.3 – Атрибуты сущности «Новости»

Поле	Тип	Обязательное	Описание
1	2	3	4
_id	ObjectId	true	Уникальный идентификатор
head	String	true	Заголовок новости
short	String	false	Аннотация к новости

Продолжение таблицы 3.3

1	2	3	4
createdAt	Date	true	Время создания новости
author	String	false	Автор новости
content	String	true	Текст новости
views	Integer	true	Количество просмотров новости зарегистрированными пользователями

Пример использования различных типов столбцов представлен в таблице 3.5. Рекомендуется использовать пакет xltabular для создания таблиц.

Таблица 3.5 – Атрибуты сущности «Новости разметки в LaTeX» с использованием различных типов столбцов и многострочным заголовком

Поле	Тип	Обязательное	Описание
1	2	3	4
_id	ObjectId	true	Уникальный идентификатор
head	String	true	Заголовок новости
short	String	false	Аннотация к новости
createdAt	Date	true	Время создания новости
author	String	false	Автор новости
content	String	true	Текст новости
views	Integer	true	Количество просмотров новости зарегистрированными пользователями

В системе предусмотрен внутренний механизм связи между разделами и элементами информационных блоков, поэтому введения дополнительных

идентификаторов при реализации связей между сущностями не предполагается.

Экземпляры сущностей реализуются в информационных блоках посредством элементов, атрибуты сущности – посредством полей и свойств элемента.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Фримен, А. Практикум по программированию на JavaScript / А. Фримен. – Москва : Вильямс, 2013. – 960 с. – ISBN 978-5-8459-1799-7. – Текст : непосредственный.

2. Бретт, М. PHP и MySQL. Исчерпывающее руководство / М. Бретт. – Санкт-Петербург : Питер, 2016. – 544 с. – ISBN 978-5-496-01049-8. – Текст : непосредственный.

3. Веру, Л. Секреты CSS. Идеальные решения ежедневных задач / Л. Веру. – Санкт-Петербург : Питер, 2016. – 336 с. – ISBN 978-5-496-02082-4. – Текст : непосредственный.

4. Гизберт, Д. PHP и MySQL / Д. Гизберт. – Москва : НТ Пресс, 2013. – 320 с. – ISBN 978-5-477-01174-2. – Текст : непосредственный.

5. Голдстайн, А. HTML5 и CSS3 для всех / А. Голдстайн, Л. Лазарис, Э. Уэйл. – Москва : Вильямс, 2012. – 368 с. – ISBN 978-5-699-57580-0. – Текст : непосредственный.

6. Дэкетт, Д. HTML и CSS. Разработка и создание веб-сайтов / Д. Дэкетт. – Москва : Эксмо, 2014. – 480 с. – ISBN 978-5-699-64193-2. – Текст : непосредственный.

7. Макфарланд, Д. Большая книга CSS / Д. Макфарланд. – Санкт-Петербург : Питер, 2012. – 560 с. – ISBN 978-5-496-02080-0. – Текст : непосредственный.

8. Лоусон, Б. Изучаем HTML5. Библиотека специалиста / Б. Лоусон, Р. Шарп. – Санкт-Петербург : Питер, 2013 – 286 с. – ISBN 978-5-459-01156-2. – Текст : непосредственный.

9. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.

10. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.

11. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.

12. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.

13. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.