ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА ИНФОРМАЦИОННЫХ И РОБОТОТЕХНИЧЕСКИХ СИСТЕМ

РАЗРАБОТКА И АВТОМАТИЗАЦИЯ СИСТЕМЫ ОНЛАЙН УЧЁТА И ВЕДЕНИЯ БАЗЫ ДАННЫХ WEB-САЙТА

Выпускная квалификационная работа
обучающегося по направлению подготовки 09.03.02. Информационные системы и технологии
заочной формы обучения 5 курса группы 07001353
Сидорова Александра Андреевича

Научный руководитель
доцент
Щербинина Н.В.

БЕЛГОРОД 2018

РЕФЕРАТ

Разработка и автоматизация системы онлайн учета и ведения базы данных web-сайта. – Сидоров Александр Андреевич, выпускная квалификационная работа бакалавра. Белгород, Белгородский государственный национальный исследовательский университет (НИУ «БелГУ»), количество страниц 75, включая приложения 89, количество рисунков 24, количество таблиц 6, количество использованных источников 38.

КЛЮЧЕВЫЕ СЛОВА: web-ресурс, информационная система, база данных, онлайн бронирование, форма, текстовая информация, бизнес-процесс.

ОБЪЕКТ ИССЛЕДОВАНИЯ: деятельность по оказанию услуг квест комнатой.

ПРЕДМЕТ ИССЛЕДОВАНИЯ: средства автоматизации онлайн-бронирования заявок при помощи web-инструментов.

ЦЕЛЬ РАБОТЫ: совершенствование процесса бронирования услуг путем разработки автоматизированной системы онлайн учета на примере квест комнаты «Кома».

ЗАДАЧИ ИССЛЕДОВАНИЯ: провести анализ деятельности организации «Кома», выявить недостатки в существующих бизнес-процессах бронирования квест комнаты «Кома»; произвести обзор существующих средств онлайн-бронирования организаций на основе web-модулей; обосновать необходимость создания web-модуля квест комнаты «Кома»; спроектировать структуру и содержание web-модуля; выбрать программные средства и CMS платформы для реализации web-модуля; опубликовать и реализовать созданный ресурс в сети Интернет.

МЕТОДЫ ИССЛЕДОВАНИЯ: анализ деятельности организации с целью обоснования необходимости разработки интернет - ресурса; средства графического моделирования бизнес-процессов; технологии web-программирования; методика оценки окупаемости проекта.

ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ: автоматизированная система онлайн-бронирования квест комнаты «Кома».

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Тенденция к упрощению и децентрализации сбора информации, путем решения функциональных задач способом разработки базы данных онлайн ресурсов, влечет за собой распределение обработки информации с применением средств, позволяющих существенно упростить и персонализировать время и способы получения данных. Информационные технологии с каждым годом сильнее трансформируются из систем автоматической переработки входной информации, в средства оценки, накопления, выработки и анализа наиболее эффективных экономических и управленчески обоснованных решений. В этих условиях главный пользователь – специалист, использует внедренные средства автоматизированных систем в форме онлайн учета и обработки полученной информации для более эффективной работы с web-ресурсом и его функционированием в целом.

Средства систем управления базами данных дают практически неограниченные возможности автоматизировать решение информационных задач, формализированных для обеспечения информационной поддержки и организации упорядочевания потоков информации, используемых предприятием для эффективной работы. С учетом своих проффесиональных знаний и навыков, пользователь может избирать метод решения поставленной задачи, а также управлять данными для необходимых вычислений и операций, проводить анализ результатов принятой информации и принимать походящее соответственной ситуации управленческое решение. Данная информация обуславливает актуальность темы выпускной квалификационной работы.

Целью выпускной квалификационной работы является совершенствование процесса бронирования услуг за счет разработки автоматизированной системы онлайн учета на примере квест комнаты «Кома».

Исходя из поставленной цели, были сформулированы задачи:

- анализ деятельности организации «Кома»;
- проведение декомпозиции бизнес-процессов сайта организации и

выявление недостатков в текущей работе;

– обоснование необходимости технологических, технических, программных и информационных проектных решений;

– обоснование выбора программных средств по разработке и проектированию программного средства;

– проведение реинжиниринга бизнес-процессов организации и построение модели «КАК ДОЛЖНО БЫТЬ»;

– описание функциональных требований к разрабатываемой автоматизированной системе онлайн-бронрирования;

– проектирование и разработка системы онлайн-бронирования;

– внедрение и реализация системы;

Предмет исследования – средства автоматизации онлайн-бронирования заявок при помощи web-инструментов.

Объект исследования – деятельность по оказанию услуг квест комнатой.

Поставленные задачи, решались в трех разделах выпускной квалификационной работы.

В первом разделе осуществляется анализ бизнес-процессов оргнаизации квест комнаты «Кома», а также выявлены недостатки в текущем процессе работы квест комнаты «Кома».

Во втором разделе дано обоснование проектных решений по программному, технологическому, техническому и информационному обеспечению. Выбраны подходящие программные продукты для разработки и проектирования системы онлайн бронирования, определены программные средства для моделирования бизнес-процессов.

В третьем разделе представлен процесс проектирования и разработки системы онлайн-бронирования. Также обоснована целесообразность разработки системы онлайн-бронирования с экономической точки зрения, а также представлен расчет экономической эффективности разработки с помощью выбранной методики.

В ходе выполнения выпускной квалификационной работы будет разработана автоматизированная система онлайн-бронирования с функцией доступа к базе данных, которая позволит в результате внедрения в организацию «Кома» сократить время обработки заявок, повысить прибыль организации за счет возможности хранения и обработки данных о клиентах и повысить продуктивность работы администратора-оператора.

Структура выпускной квалификационной работы состоит из введения, трех разделов, заключения, списка использованных источников и приложений.

1 Аналитическая часть

1.1 Технико-экономическая характеристика предметной области

Организация «Кома» зарегистрирована по юридическому адресу город Белгород, улица проспект Славы, дом 43.

Компания «Кома» была поставлена на учет 6 апреля 2014 года в Налоговую инспекцию по городу Белгороду.

Организации присвоен Общероссийский Государственный Регистрационный Номер – 1137746943796.

Полное наименование организации «Кома».

Руководителем является директор Зайцев Максим Олегович, действующий как индивидуальный предприниматель.

Основным видом деятельности является проведение командных интеллектуальных игр, осуществляемых непосредственно при помощи телевидения, радио, телефона и сети Интернет. Код по общероссийскому классификатору видов экономической деятельности (ОКВЭД) – 92.72.

Организация занимается созданием, развитием и управлением квест-комнатами. Квест-комната – это закрытое помещение, из которого команде участников в среднем из 2-4 человек предстоит выйти за определенный отрезок времени, решая головоломки, опираясь на подсказки, которые спрятаны в этой комнате.

Квест-комнаты оснащены тематическими декорациями, соответствующей атрибутикой и реалистической бутафорией. Все окружение дает возможность оказаться в «другом мире», погрузиться в прохождение игры. Дорогостоящие профессионально оборудованные комнаты имеют датчики анализа поведения игроков, спецэффекты. Любая мелочь может предать реалистичность игровой обстановке. Тематика квестов может быть разной, как правило, это сюжеты фильмов, компьютерные стратегии или художественная литература. Часто квест комнаты устроены по индивидуально

составленному проекту. Каждый такой вариант находит своих поклонников, так как не имеет жестких ограничений.

Потенциальными клиентами являются молодые люди возрастом 16-35 лет со средним и высоким уровнем дохода, которые ищут новые развлечения и предпочитают активный отдых. Также часто участниками квеста становятся корпоративные клиенты. В организации «Кома» имеется несколько отличающихся по сценарию квестов, чтобы удовлетворить предпочтения и потребности любых клиентов.

Количество проводимых игр не зависит от дня недели. Поставщиками новых сценариев и реквизита, являются Воронежские компании. Грамотный и интересный сценарий, а также качественное и высокотехнологичное оборудование выполняют ключевую роль в успешной деятельности организации. Возможны негативные моменты, когда полученный сценарий имеет серьезные недоработки, а оборудование прибывает в нерабочем или несоответствующем нормам состоянии.

Помимо основного направления – проведение игр в квест-комнатах – «Кома» занимается проведением интеллектуальных игр выездного характера. Данная сфера работы требует более щепетильного отношения к организации, написании сценария и адаптации к условиям местности и окружения на территории проводимой игры. Важную роль играет то, как организатор ответственный за мероприятие выслушает клиента: пожелания по тематике, по времени игры, по количеству головоломок, по сложности цепочки квеста. Организатор и сценарист, в свою очередь, должны уметь «схватывать на лету» и предлагать клиенту большое количество вариантов реализации запланированного мероприятия. Это направление работы является сезонным и зависит от готовности проведения досуга такого рода от корпоративных клиентов.

Анализ бизнес-процессов организации «Кома»

Фирма по размерам небольшая, тип организационной структуры – линейный.

Прцесс оформления заявки на игру производится в оффлайн режиме, путем звонка на контактный номер телефона организации и заполнения рукописной формы данных о клиенте администратором-оператором. В соответствии с этим данная модель «КАК ЕСТЬ» является длительным процессом, с возможностью допущения ошибок администратором-оператором из-за помех сотовой связи или особенностей произношения клиента.

В организации «Кома» директор принимает стратегические решения, а именно: контролирует работу персонала, применяет меры поощрения отличившихся работников, налагает взыскания на нарушителей производственной и трудовой дисциплины, ведет переговоры и оформляет договоры, связанные с поставками, заказами, реализацией игр.

Также к основным должностным функциям директора относятся:

– обеспечение соблюдения требований, установленных для предприятия в государственных стандартах, санитарных, ветеринарных, противопожарных правилах и других нормативных документах;

– оформление, получение лицензии, соответствующие разрешения и иные документы;

– доводение до сведения покупателей информацию о товарах и иной информации, способствующей правильному выбору товара покупателем;

– доведение до покупателей сведения об организационно-правовой форме юридического лица, фирменное наименование (наименование), юридический адрес, режим работы и пр.;

– обеспечение наличие оборудования, инвентаря в соответствии с требованиями стандартов необходимых для сохранения качества и безопасности проведения игр;

– организация, планирование и координация деятельности

предприятия развлекательного характера, управление текущей деятельности, направленной на развлечение клиента;

– осуществление контроля за рациональным использованием материальных, финансовых и трудовых ресурсов.

Администратор-оператор «Кома» выполняет часть работы директора и операционные задачи, а именно:

– анализирует результаты проведенных игр и качества обслуживания клиентов;

– принимает звонки клиентов, консультируя о видах предостовляемых сценариев;

– принимает заявки, поступающие на контактный номер организации;

– вносит в рукописную форму журнала заявок, информацию о забронированной игре;

– ведет переговоры, связанные с поставками, заказами и реализацией сценариев и оборудования;

– обеспечивает организацию учета товарно-материальных ценностей и представляет отчетность о количестве проведенных игр директору предприятия;

– принимает решения о назначении, перемещении и освобождении от занимаемых должностей дополнительных работников развлекательного предприятия;

– представляет интересы предприятия и действует от его имени.

Директор и администратор-опреатор совещаются между собой и принимают решения вместе.

Также в штате сотрудников числятся два SMM специалиста, один из которых имеет небольшой опыт работы оператором-администратором. В их обязанности входит:

– повышение количества продаж игр потенциальным клиентам;

– составление и оформление рекламных объявлений и их реализация

на рекламных площадках;

– учет отчетности по графикам статистики успешности рекламы, CRM и представление отчетности начальству об объемах приведенных клиентов;

– вести подготовку рекламных объявлений в социальных сетях и поисковых системах, с учетом соответствия целесообразности формата объявления, в зависимости от площадки размещения и формата интеграции;

В приложении А представлена организационная структура управления «Кома».

1.2 Экономическая сущность задачи

Объектом исследования является деятельность по оказанию услуг квест комнатой.

Основываясь на том, что введение модуля автоматизированной системы онлайн бронирования позволит уменшит затраты времени обработки заявок, позволит вести отчетность о проведенных играх и вести клиентскую базу игроков, с экономической точки зрения данная разработка оправдана.

Для изучения данного объекта следует выделить два класса задач:

– анализ деятельности «Кома»;

– мероприятия по анализу и исследованию систем автоматизации онлайн бронирования.

К первому классу задач относятся следующие задачи:

– анализ функций сотрудников;

– анализ организуемых мероприятий;

– анализ нормативных документов.

Ко второму классу задач относятся следующие задачи:

– анализ инструментов для разработки системы онлайн бронирования;

– анализ шаблонов проектирования системы онлайн бронирования;

Информацию для решения класса задач «Анализ деятельности «Кома» получают из следующих первичных документов:

- устав;
- положение об оказании услуги в «Кома».

Информацию для решения класса задач «Мероприятия по анализу и исследованию систем автоматизации онлайн бронирования» получают из первичных документов первого класса задач, специальной технической-технологической литературы по web-программированию и ресурсов Интернет.

1.3 Обоснование необходимости и цели использования вычислительной техники для решения задачи

Чтобы исследовать существующую предметную область необходимо произвести структурно-функциональный анализ деятельности «Кома». На рисунке 1.1 представлена структурно-функциональная диаграмма «Оказание услуги организации квест-проекта Кома».

Рисунок 1.1 – Контекстная диаграмма
«Оказание услуги квест-комнаты «КАК ЕСТЬ»»

Данная бизнес модель описывает основной вид деятельности организации, а именно – процесс оказания услуги по проведению развлекательной интеллектуальной игры.

Для детального проведения анализа необходимо произвести декомпозицию на сущности и связи. На рисунке 1.2 представлена декомпозиция диаграммы «Оказание услуги квест-комнаты «Кома»».

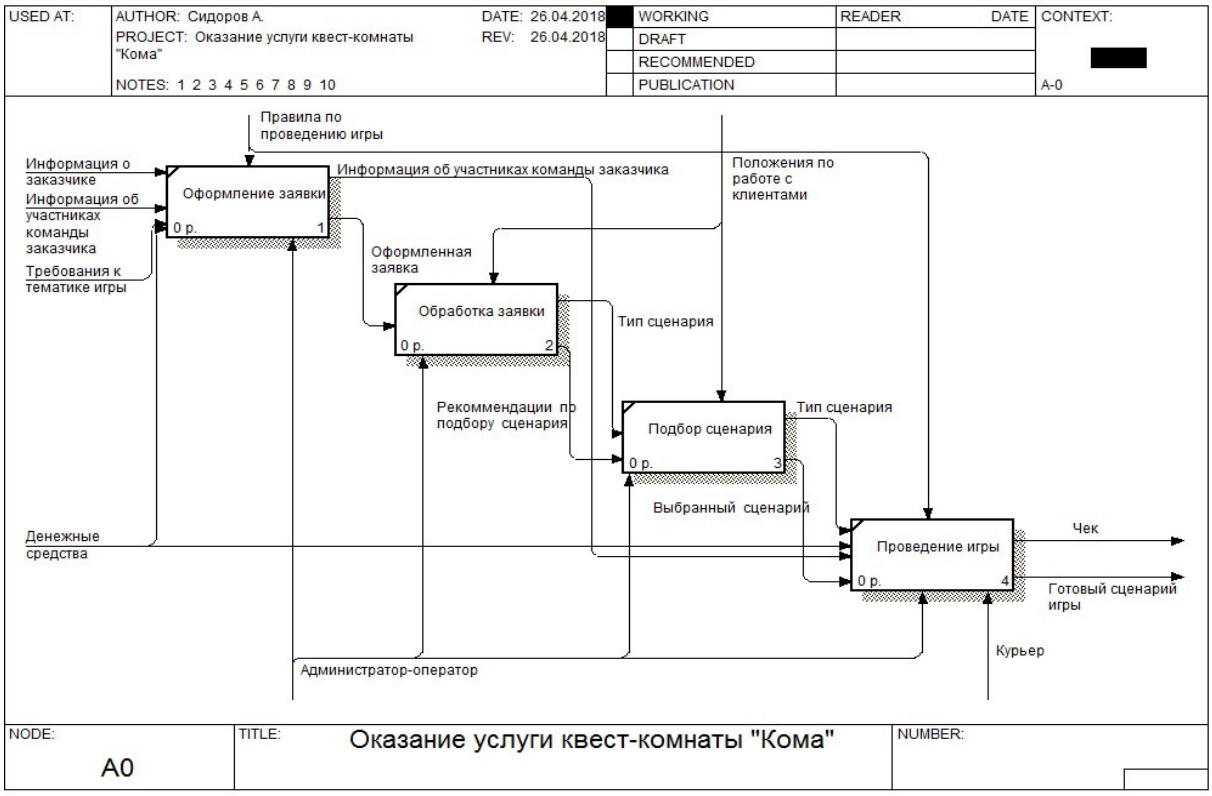

Рисунок 1.2 – Детализация контекстной диаграммы «Оказание услуги квест-комнаты «КАК ЕСТЬ»»

Процесс оказания услуги «Кома» начинается с оформления заявки. Оформленная заявка обрабатывается. После принятия типа сценария, проводится игра с командой заказчика.

На рисунке 1.3 представлена декомпозиция блока «Оформление заявки» в нотации DFD.

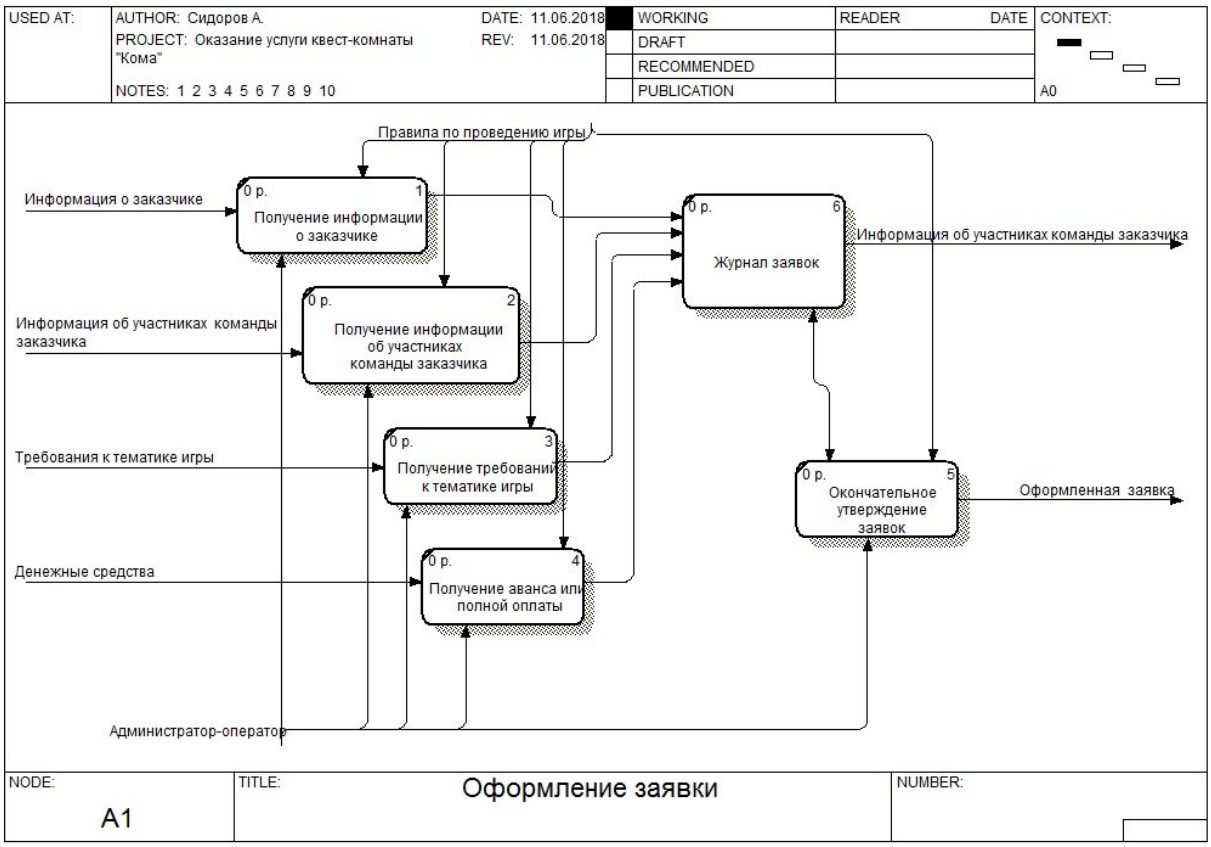

Рисунок 1.3 – Декомпозиция А1 блока
«Оформление заявки» в нотации IDEF0

На данном этапе администратор-оператор получает информацию о заказчике (номер телефона, имя), о команде заказчика (количество людей, возраст, время игры) и требования к заявке (описание игры и тип сценария). Вся информация вносится в журнал заявок для дальнейшего анализа и работы с заявкой.

Предназначение следующего блока заключается в анализе оформленной заявки. Администратор-оператор получает информацию по заявке и проверяет по ней наличие подходящего сценария. Если сценарий есть в наличии, то заявка прорабатывается – продумываются варианты проведения игры. После чего администратор-оператор создает рекомендации по подбору сценария и описывает итоговую заявку.

На рисунке 1.4 представлена декомпозиция блока «Обработка заявки».

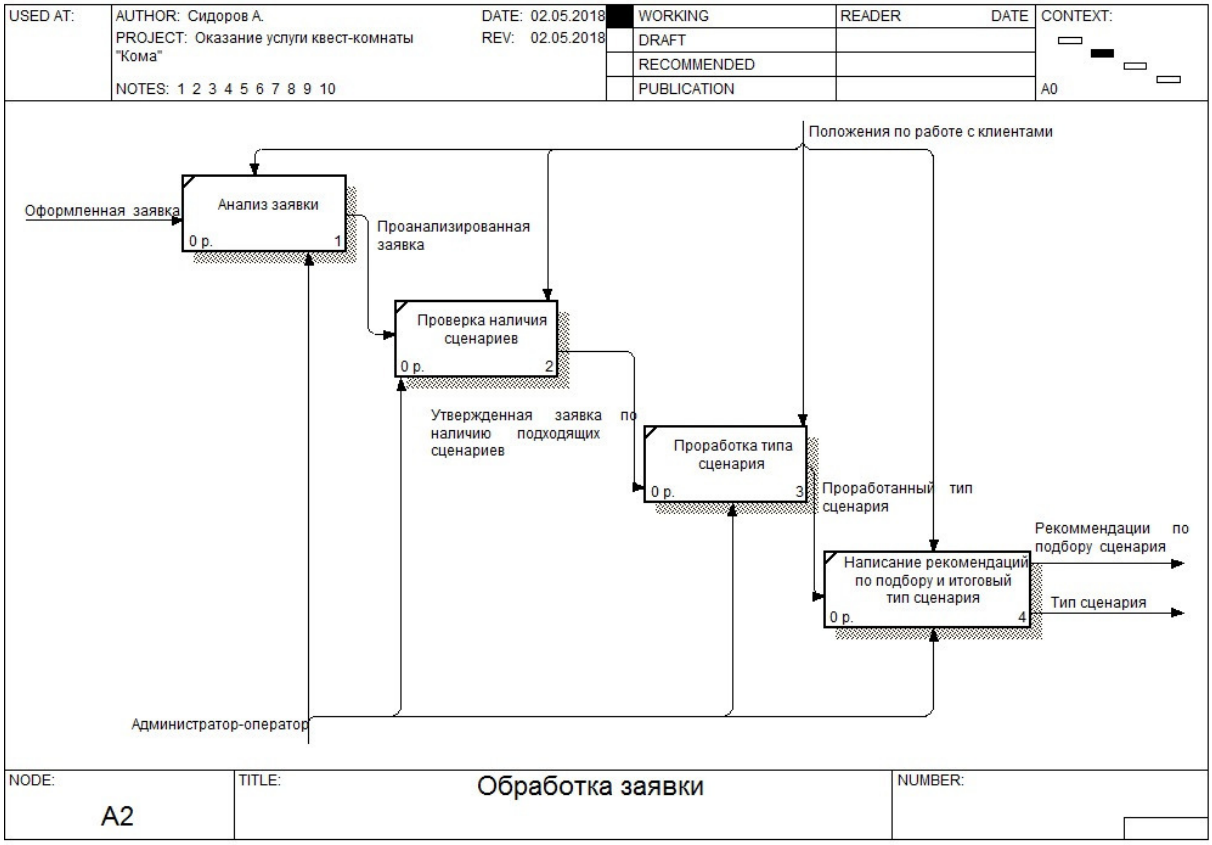

Рисунок 1.4 – Декомпозиция A2 блока «Обработка заявки» в нотации IDEF0

На следующем блоке продемонстрирован бизнес-процесс подбора сценария. Подготовленные рекомендации по подбору сценария анализируются и обрабатываются. На основе их и типа сценария подбирается игра, добавляются подходящие сюжеты, выбирается сценарий и заполняется накладная. Далее происходит подготовка сценария проведения игры по типу и выбранному сценарию.. В связи с этими факторами декомпозиция выполнена в структуре нотации IDEF3.

На рисунке 1.5 представлена детализация блока «Подбор сценария».

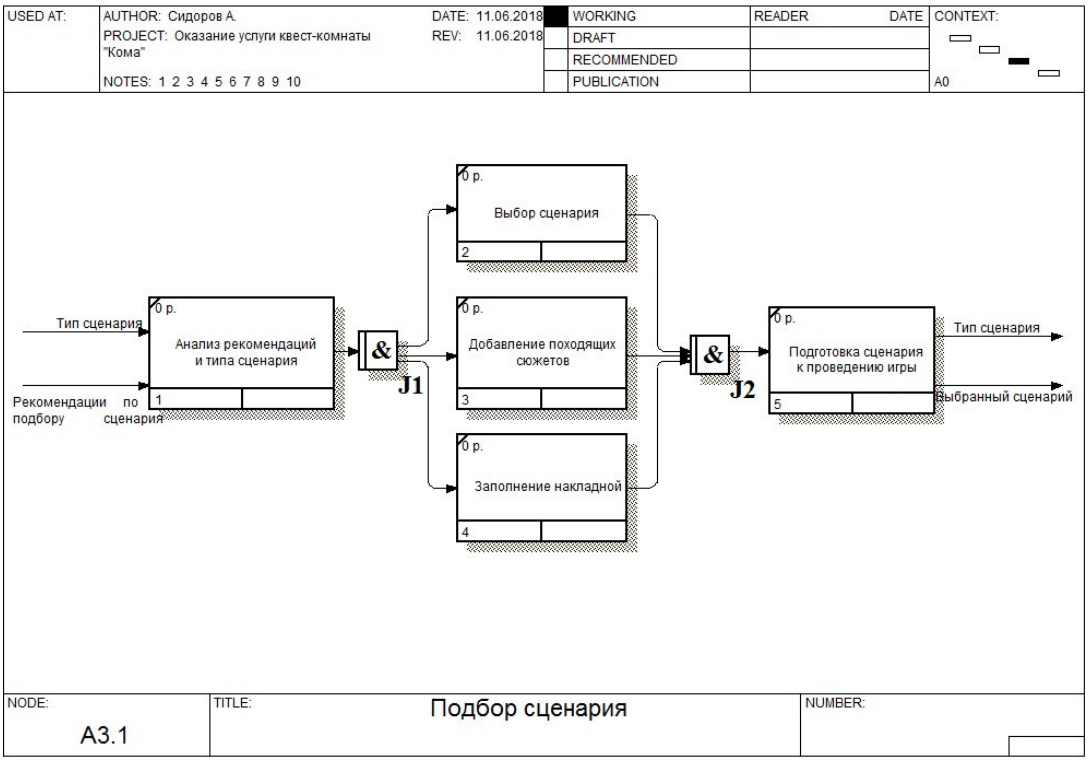

Рисунок 1.5 – Декомпозиция А3.1 блока «Подбор сценария» в нотации IDEF3

На рисунке 1.6 проиллюстрирована детализация блока «Проведение игры».

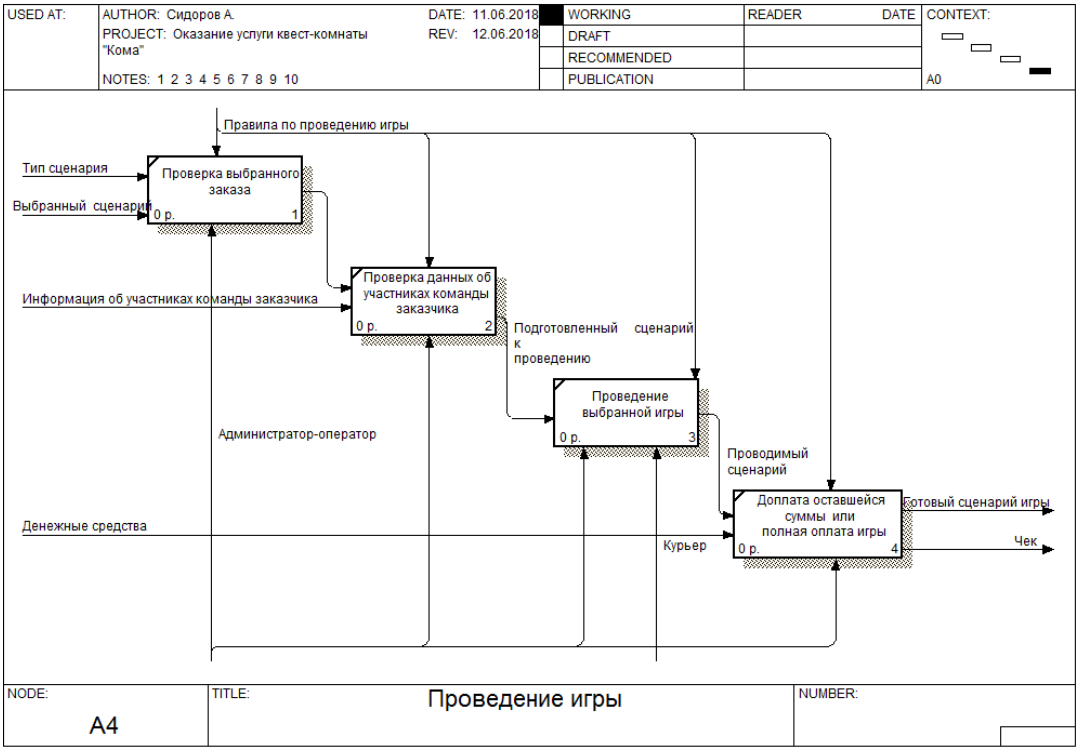

Рисунок 1.6 – Декомпозиция А4 блока «Проведение игры» в нотации IDEF0

На данном этапе происходит проведение игры непосредственно для команды заказчика.

Сначала администратор-оператор проверяет выбранный заказ. Если он не удовлетворяет заявленным требованиям, то заказ переделывается. Если всё подходит по требованиям, то заявка подготавливается к проведению. Игра проводится только для команды заказчика. Если команда заказчика прошла идентификацию, то игра проводится и затем оплачивается.

Основные недостатки, на существующей практике бронирования игры:

– отсутствуют система учета, контроля и анализа заявок и продаж;

– отсутствует система учета, контроля и анализа заказчиков;

– отсутствует система анализа и контроля эффективности работы сотрудников;

– отсутствует система учета, контроля, прогнозирования и анализа онлайн типов сценария;

– отсутствует система учета, контроля, прогнозирования и анализа онлайн доступа клиента к свобоному для проведения игры времени.

Для решения обозначенных проблем необходима вычислительная техника и подходящая автоматизированная система с соответствующим программным средством для повышения эффективности работы сотрудника организации «Кома».

1.4 Постановка задачи

Проанализировав деятельность квест комнаты «Кома» и выявив основные недостатки процесса записи на игры, необходимо совершенствование бронирования услуг для повышения эффективности работы администратора-оператора. Поэтому необходима разработка и внедрение автоматизированной системы онлайн учета и ведния базы данных , которая позволит оптимизировать время работы администратора-опреатора,

путем ускорения процесса приема заявок, упорядочит базу данных клиентов и повысит прибыль организации за счет совершенствования данного бизнес-процесса.

На любом веб-сайте для успешного и эффективного ведения бизнеса должна присутствовать система доступа к данным по бронированию игры, включающая в себя информацию по графику, дню недели, времени и занятости квеста. Данные собранные этой системой позволяют успешно обрабатывать полученную информацию и масштабировать бизнес организации за счет комфорт и доступность клиента-пользователя.

Также организация развлекательного характера, в том числе и квест комната, должна иметь возможность собирать, обрабатывать и хранить информацию о своих клиентах. Это позволит создать лояльность клиентов к компании, за счет тесного контакта с ними. На основе клиентских предпочтений организация может осуществлять sms и e-mail рассылку о своей продукции, новинках и акциях. Данная система позволит повысить имидж и прибыль организации.

Чтобы доступ к полученной информации о бронировании игр был максимально комфортным, система бронирования должна быть эффективно оптимизирована также и для администратора сайта, использующего полученную информацию и управляющего дальнейшими организационными процессами. Если сотруднику будет проще выполнять данные операции, то показатели компании будут расти.

Квест комнаты реализуют бронирование через телефонный звонок, однако в век информационных технологий и развития онлайн пользования ресурсов, все больше людей используют веб-ресурсы для совершения покупок, бронирования или записи. Ввод данной системы упростит работу организации, сократит время обработки заявок, сократит издержки и повысит результативность компании.

1.5 Анализ существующих разработок и обоснование выбора технологии проектирования

Для понимания как должен будет выглядеть готовый продукт после этапов проектирования и разработки, необходимо проанализировать существующие автоматизированные системы онлайн учета и ведения баз данных, выполняющие схожие функции.

Для анализа будут рассмотрены 2 программных средства:

– виджет сайта hesus.ru;
– модуль Kvestbook service;

Hesus.ru – онлайн ресурс предоставляющий функцию создания собственного виджета с последующей интеграцией на свой веб-сайт, позволяющий организовать систему бронирования для квест комнаты как на рисунке 1.7.

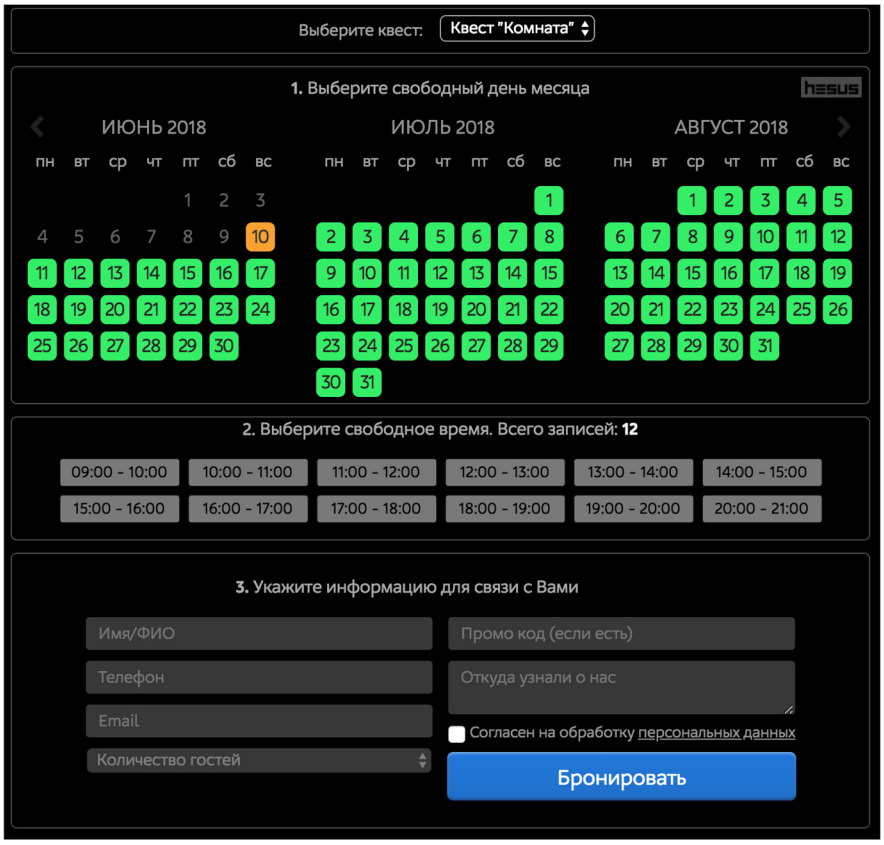

Рисунок 1.7 – Виджет сайта hesus.ru

Главной функцией является простой и понятный интерфейс, который подтолкнет клиентов и посетителей заказать услуги организации. Данный виджет работает только с сайтами написанными на ограниченном числе языков программирования, таких как Perl и Ruby. Стоимость использования данного виджета составляет 600 рублей в месяц. Недостаток данного виджета как системы онлайн учета в том, что его невозможно модернизировать, т.к. он откомпилирован.

Kvestbook service – специализированные системы бронирования для квест-комнат, перформансов и escape-румов, с функциями подключения системы бронирования, системы управления клиентами, подробной статистики, модуля бронирования для сайта, интеграции с агрегаторами, E-mail уведомлений, SMS-уведомлений клиента и администратора, расширенной кастомизации модуля бронирований, назначения администраторов и многое другое. Данное средство разработано с помощью языков web-программирования, что демонстрирует рисунок 1.8.

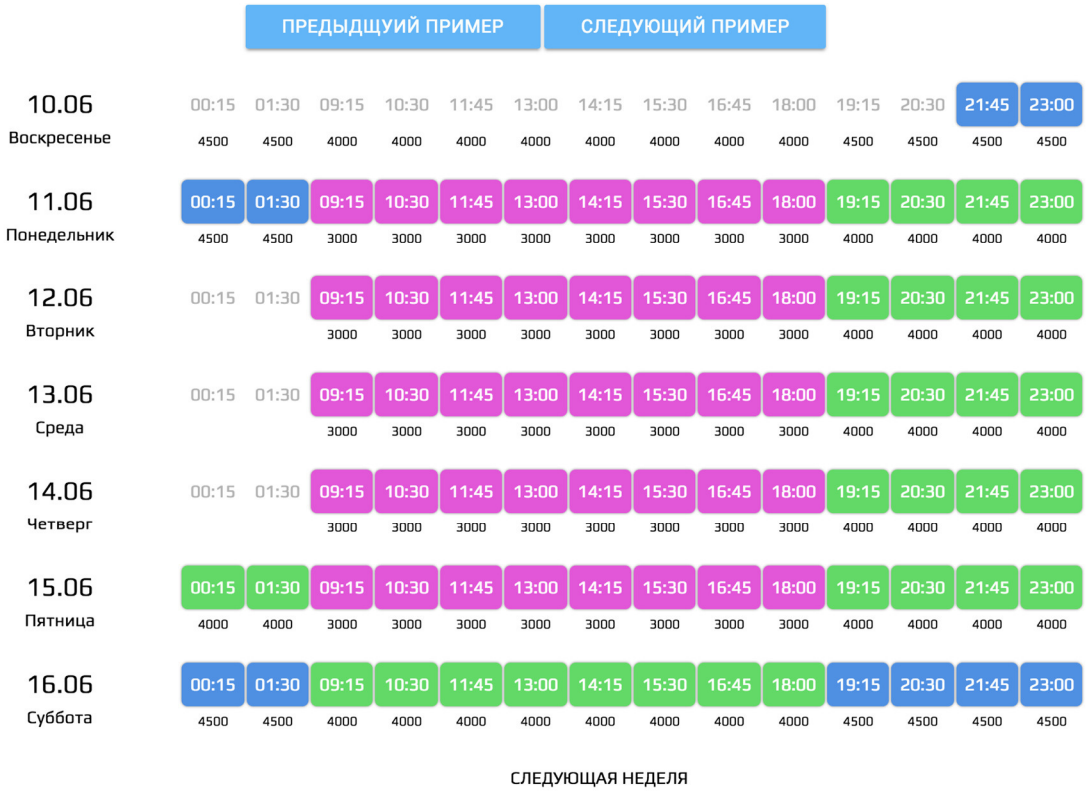

Рисунок 1.8 – Модуль Kvestbook service

Стоимость пользования системой является бесплатной только в тестовом периоде длительностью 7 дней. Дальнейшая оплата составляет 20 или 35 рублей в день, в зависимости от выбранного тарифа Продвинутый или Super. Кроме очень высокой стоимости программного средстванедостатком является наличие бесполезных в работе системы инструментов, которые усложняют навигацию и работу пользователя, а также дополнительные функции требуют определенных ресурсов, но при этом они не используются.

Недостатком программного средства Kvestbook service является его высокая стоимость.

Таким образом, среди перечисленных сторонних программных разработок отсутствует сабалнсированное решение, которое отвечало бы поставленным целям выпускной квалификационной работы.

Выводы по первому разделу.

В первом разделе была изучена и проанализирована деятельность организации «Кома». Изучен процесс бронирования игры, процесс оказания услуги. Также были проанализированы сайты квестовой тематики, а именно: изучена цветовая гамма, эргономичность сайта, функциональность сайта, рассмотрены наличие разных способов оплаты. Обоснована необходимость создания автоматизированной системы онлайн учета и ведения базы данных сайта.

Приложение будет разрабатываться с помощью языков web-программирования;

Были определены задачи выпускной квалификационной работы, для достижения поставленных задач был сформулирован четкий план ведения проекта.

2 Обоснование проектных решений

2.1 Обоснование проектных решений по техническому обеспечению

Техническое обеспечение – это совокупность технической и сетевой инфраструктуры организации, обеспечивающая беспрерывную работу всей деятельности. В составе технической инфраструктуры могут быть: сервера, персональные компьютеры, систем хранения данных, сетевые устройства коммуникации и сетевые кабели.

Техническая инфраструктура организации «Кома» не имеет вычислительного оборудования. Вся финансово-вычислительная деятельность происходит на печатном носителе (журнал учета заявок).

Для работы разработанной системы необходим выделенный персональный компьютер со средними техническими характеристиками, который должен иметь доступ к сети Интернет.

Необходимо приобрести персональный компьютер со следующими техническими характеристиками:

- процессор с тактовой частотой не менее 2x3600 (например, Intel Core i5 2x3600);
- оперативная память не менее 1 Гб;
- сетевая плата с пропускной способностью не менее 1000 Мбит/с (например, Realteck PCI GBE 1000 Мбит/с);
- жесткий диск HDD объемом не менее 250 Гб (например, WTD 512 Гб).

Данные характеристики являются рекомендуемыми. Их выбор основывался на средне-статических характеристиках офисных компьютеров, имеющие схожие исполняемые задачи.

Основные задачи, которые будет решать приобретённый компьютер:

- работа web-сервера;
- запуск личного кабинета администратора «Кома»;

– работа с полученными данными и формирование заявки;

– хранение базы контактного номера клиента.

Рекомендуемая периферия для ПК:

– мышь Defender;

– клавиатура Defender;

– монитор Samsung 17.5”;

– лазерный принтер Canon 1010.

Обосновать данный выбор можно тем, что данного оборудования и внешней периферии с указанными мощностями достаточно для выполнения поставленных задачи перед ПК.

2.2 Обоснование проектных решений по информационному обеспечению

Информационное обеспечение – это совокупность массива данных (выходные, промежуточные, входные), программ для решения задач, систем кодирования и классификации оперативных документов, нормативно-справочной информации.

Технология обработки данных зависит от следующих факторов:

– функционирование в режиме диалога с пользователем;

– наличие накопителей информации;

– исключение бумажных технологий для обработки информации.

В состав технологических операций входят:

– запуск программы;

– ввод первичных данных;

– контроль информации (валидность по заполнению) и возможность корректировки;

– справочно-информационное обслуживание;

– формирование информационных массивов;

– вывод информации.

Первичная информация для заполнения справочников вводится на этапе разработки базы данных. К такой информации относятся статус времени календаря, тип доступа пользователя, один пользователь с правами администратора.

В web-приложении вывод информации осуществляется с помощью пользовательского интерфейса через устройство вывода информации (например, клавиатура, мышь). Оболочка для интерфейса представлена интернет браузером. Все формы для вывода информации представлены в виде удобного функционального интерфейса с подсказками.

Существует несколько способов связей между данными:

– реляционная – модель является простейшей и наиболее привычной формой представления данных в виде таблиц;

– иерархическая – связи могут быть отражены в виде дерева-графа, где запись-потомок имеет в точности одного предка;

– сетевая – связи представлены в виде дерева-графа, где у потомка может иметься любое число предков, возможны связи «всех со всеми».

В настоящее время реляционные системы лучше соответствуют техническим возможностям персональных компьютеров. Скоростные характеристики этих систем управления базами данных поддерживаются специальными средствами ускоренного доступа к информации – индексирование баз данных.

Таким образом, будет разработано web-приложение с практичным интерфейсом для администратора и пользователя.

2.3 Обоснование проектных решений по программному обеспечению

Программное обеспечение – совокупность программ, обеспечивающих функционирование вычислительной системы (системное программное обеспечение), а также программ, предназначенных для решения конкретных

задач пользователя (прикладное программное обеспечение).

К выбираемому программному обеспечению в данном случае относятся операционная система (ОС) и среда программирования.

Все ОС подразделяются на:

– однопользовательские и многопользовательские;

– однозадачные и многозадачные.

Разрабатываемое web-приложения работает на любой операционной системе: Windows, Linux, MacOS , AndroidOS, iOS и др.

Мультиплатформенность разработки связана с тем, что для работы используется веб-браузер, который по умолчанию установлен на любой доступной ОС.

В текущей выпускной квалификационной работе будет использоваться операционная система Windows 10, по причине того, что поставляется сразу в комплекте с приобретаемым ПК.

Далее необходимо описать какая система управления базами данных будет использоваться. В программировании web-приложений управление БД осуществляется при помощи клиент-серверных систем управления базами данных (СУБД), таких как Oracle, MS SQL Server, PostgreSQL, MySQL и др. Клиент-серверные СУБД обрабатывают запросы централизованно, к их достоинствам относят обеспечение высокой надежности баз данных, высокой доступности и высокой безопасности.

MySQL – свободная система управления базами данных, одна из наиболее часто применяемых в программировании сайтов. СУБД MySQL поддерживает большое количество существующих типов таблиц (InnoDB, MyISAM и т. д.), а благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц. Управление базами данных с помощью MySQL очень удобно, что сделало данную систему востребованной и популярной;

Microsoft SQL Server – использует язык запросов Transact-SQL, поддерживается операционными системами семейства Windows

Desktop/Server. В СУБД Microsoft SQL Server присутствует графическое ПО для конструирования и оптимизации запросов (SQL Management Studio и Studio Express);

PostgreSQL – базируется на языке SQL. Среди преимуществ PostgreSQL выделяют поддержку БД практически неограниченного размера, наличие надежных механизмов репликации, легкую расширяемость, поддержку большого набора встроенных типов данных и многое другое;

Oracle Database – работает на Windows, Unix, Linux, MacOS. Oracle Database, в отличие от MySQL, например, имеет более широкую область применения. СУБД Oracle обладает высокой производительностью, широким функционалом, уникальными технологиями (RAC, RAT и т. д.). В программировании сайтов для небольших и средних компаний применяется достаточно редко ввиду своей высокой стоимости. К тому же, довольно сложно найти хостинг с поддержкой данной СУБД.

Обоснование выбора:

– Oracle – является дорогим СУБД;

– Microsoft SQL Server – СУБД имеет мало тонких настроек для эффективной работы web-приложения;

– PostgreSQL – имеет мало справочной и специальной информации в открытом доступе, чтобы работать с данной СУБД;

Таким образом, в качестве системы управления базами данных выбрана MySQL под Windows.

Данная СУБД была выбрана, потому что:

– данная СУБД работает на любой стационарной ОС (Windows, Mac OS, Linux);

– полнофункциональная версия полностью бесплатна, в отличие от MSSQL и других СУБД;

– для работы требует низкое количество системных ресурсов;

– данное СУБД используют все виртуальные-хостинги;

– подходит для высоконагруженных проектов, обладает высоким набором стандартных функций для работы с данными.

Для работы с СУБД MySQL будет использоваться утилита phpMyAdmin – программа, написанная на PHP и предназначенная для управления сервером MySQL через локальную или Интернет сеть. Данная утилита поддерживает широкий набор операций над MySQL. Наиболее часто используемые операции поддерживаются с помощью пользовательского интерфейса (управление базами данных, таблицами, полями, связями, индексами, пользователями, правами, и т. д.), одновременно вы можете напрямую выполнить любой SQL запрос.

Возможности phpMyAdmin:

– интуитивно понятный веб-интерфейс;

– поддержка большинства функций MySQL;

– поддержка импорта данных из CSV и SQL;

– поддержка экспорта в различные форматы CSV, SQL, XML, PDF, ISO/IEC 26400 - OpenDocument текст и таблицы, Word, Excel, LATEX и другие;

– администрирование нескольких серверов;

– генерирование наглядных схем баз данных в виде PDF;

– создание комплексных запросов с помощью функции запрос по шаблону;

– глобальный или частичный поиск в базе данных.

Для работы web-приложения необходим использоваться любой из web-серверов, таких как Apache или IIS.

Apache – это полнофункциональный, расширяемый веб-сервер, полностью поддерживающий протокол HTTP/1.1 и распространяющийся с открытым исходным кодом. Сервер может работать практически на всех распространенных платформах. Существуют готовые исполняемые файлы сервера для Windows NT, Windows 9x, OS/2, Netware 5.x и нескольких UNIX-

систем. При этом он очень прост в установке и конфигурации. Apache настраивается с помощью текстовых конфигурационных файлов. Основные параметры уже настроены "по умолчанию" и будут работать в большинстве случаев. Самая простая функция, которую может выполнять Apache – обслуживать HTML-сайт. При получении запроса на определенную страницу сервер отправляет в ее ответ браузеру.

IIS – это унифицированная веб-платформа, которая совмещает IIS, ASP.NET, службы FTP, PHP и Windows Communication Foundation (WCF). Данный веб-сервере обеспечивается высокий уровень веб-безопасности благодаря сокращению объема сервера и автоматической изоляции приложений, простое развертывание и запуск веб-приложений ASP.NET, Classic ASP и PHP на одном сервере, изоляция приложений путем уникальной идентификации рабочих процессов и их запуска в изолированной среде по умолчанию, еще более сокращающие риски безопасности, простое добавление, удаление и даже замена встроенных компонентов IIS с настраиваемыми модулями, отвечающими потребностям пользователя, повышение скорости работы веб-сайта с помощью встроенного динамического кэширования и расширенного сжатия.

Недостаток IIS в том, что данный web-сервер работает только на ОС Windows, а Apache – многоплатформенный веб-сервер. Поэтому, если в будущем организация планирует переносить код web-приложения на другую ОС, то целесообразно на этапе проектирования использовать популярный web-сервер Apache.

Совместно с web-сервером Apache необходимо использовать интерпретатор PHP – это серверный язык создания сценариев. Конструкции PHP, вставленные в HTML-текст, выполняются сервером при каждом посещении страницы. Результат их обработки вместе с HTML-текстом передается в браузер пользователю.

Выбранные программные обеспечения бесплатны, которое должно быть проинсталлировано на выделенном персональном компьютере.

Кроме вышеперечисленного программного обеспечения для работы с web-приложением персональный компьютер должен быть оснащен программами для просмотра web-сайтов, таких как Google Chrome, Mozilla Firefox, Opera и Internet Explorer

Google Chrome web-браузер, разработанный компанией Google с высокой степенью защиты, огромным количеством плагинов и системой отладки кода;

Mozilla Firefox web-браузер разработанный компанией Mozilla Foundation, с хорошей степенью безопасности, скорости работы, гибкостью и расширяемостью;

Opera web-браузер разработанный компанией Opera Software;

Internet Explorer web-браузер разработанный компанией Microsoft. По умолчанию установлен на компьютерах с операционной системой Windows.

Для работы с web-браузером можно использовать любой web-браузер, версии которых выпущены в 2018 году.

Далее будет обоснован выбор программных средств проектирования и разработки системы онлайн бронирования.

Были рассмотренны несколько программных продуктов.

Adobe Dreamweaver CS6 – программное обеспечение для web-дизайна и создания визуальных проектов. Программа Adobe Dreamweaver помогает разрабатывать реалистичную среду для интерактивного просмотра, управлять проектами на уровне пикселов или с помощью специально созданного кода. Встроенные подсказки в Adobe Dreamweaver позволяют более точно создавать код в HTML, JavaScript, Ajax, Spry, jQuery и Prototype. На рисунке 2.1 представлен интерфейс Adobe Dreamweaver CS6.

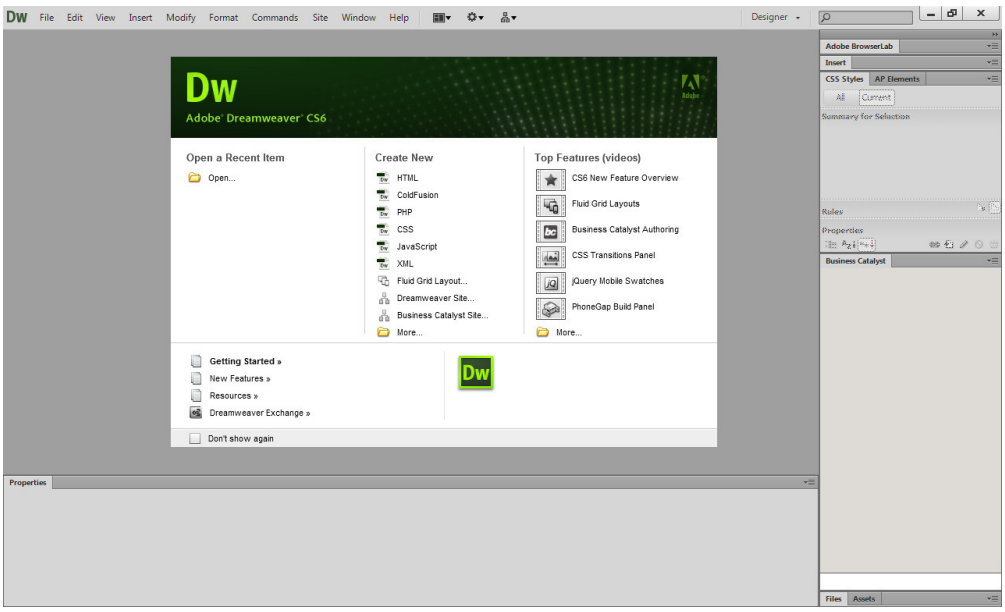

Рисунок 2.1 – Интерфейс Adobe Dreamweaver CS6

Рассмотрим основные функции Adobe Dreamweaver CS6.

Поддержка CSS3/HTML5. Работа со стилями через панель CSS, в которой реализована поддержка CSS3. Создание кода с помощью технологии HTML5 с удобными подсказками по коду и визуализацией в представлении «Дизайн».

Интерактивный просмотр. Проверка страниц перед публикацией при помощи механизма рендера WebKit с поддержкой контента HTML5.

Создание комплексных проектов. Разработка web-сайтов с бесплатными пробными версиями ПО, использование интегрированного решения электронной торговли Adobe Business Catalyst при создании коммерческих онлайн-проектов. Встроенная поддержка CMS-систем, включая WordPress, Joomla! и Drupal. Вывод подсказок по кодированию нестандартизированных файлов в Dreamweaver. Поддержка протоколов FTPS и FTPeS.

Поддержка ведущих технологий. Дизайн и разработка в среде с поддержкой HTML, XHTML, CSS, XML, JavaScript, Ajax, PHP, Adobe ColdFusion и ASP.

Интеграция с Adobe BrowserLab. Тестирование web-контента на различных web-браузерах и операционных системах.

Поддержка передовых возможностей работы. Создание стандартного кода HTML5 и CSS3, в том числе с использованием переходов CSS3, при разработке онлайн-проектов. Внедрение web-шрифтов и современной типографики в проекты. Добавление файлов FLV в любые web-страницы и создание кода на основе стандартов.

Microsoft Office FrontPage 2003 – это визуальный конструктор сайтов. Он подходит для создания простых сайтов, без использования мощных php-фреймворков. При разработке интернет-магазина с помощью данного программного средства могут возникнуть трудности при отладке кода (т.к. нету встроенного отладчика кода), будет затрачено большое время (т.к. нету авдополнения кода и перемещения между функциями), трудности организации архитектуры сайта из большого количества web-страниц. На рисунке 2.2 представлен интерфейс Microsoft Office FrontPage 2003.

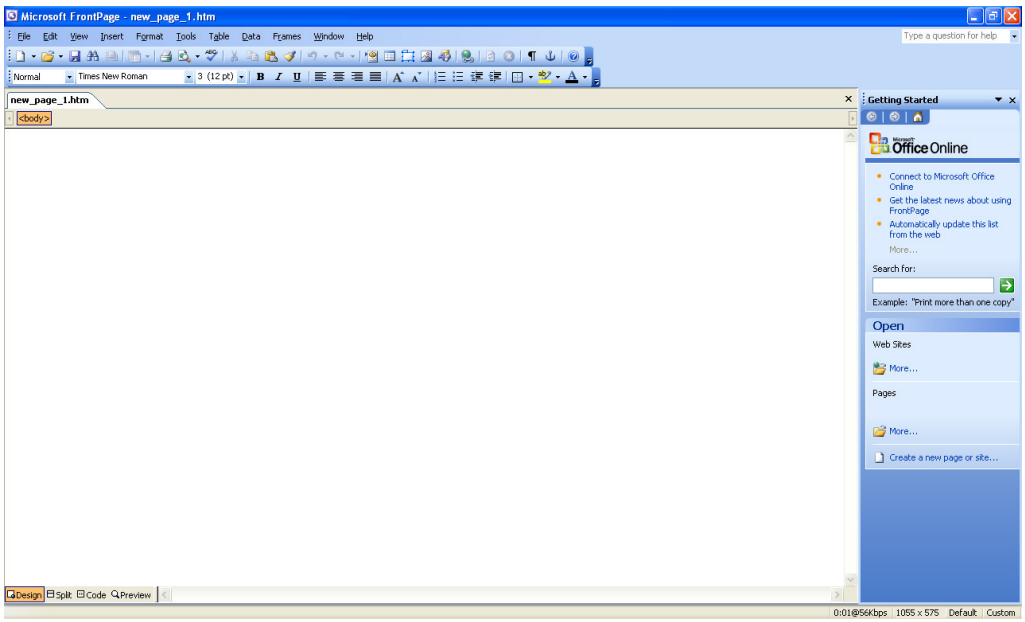

Рисунок 2.2 – Интерфейс Microsoft Office FrontPage 2003

JetBrains PhpStorm 9.0 – это интегрированная среда разработки на PHP с интеллектуальным редактором, которая глубоко понимает код,

поддерживает PHP 7.0, 5.6, 5.5, 5.4 и 5.3 для современных и классических проектов, обеспечивает лучшее автодополнение кода, рефакторинги, предотвращение ошибок налету и поддерживает смешивание языков как на рисунке 2.3.

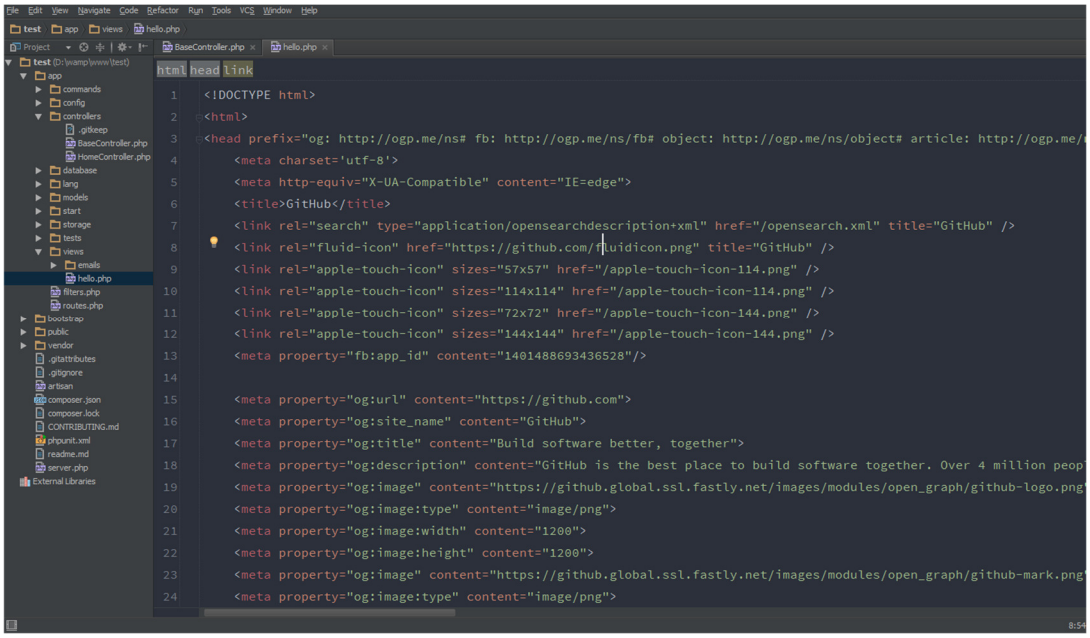

Рисунок 2.3 – Интерфейс JetBrains PhpStorm 9.0

Программный продукт поддерживает технологии веб-разработки, включая HTML5, CSS, Sass, SCSS, Less, Stylus, Compass, CoffeeScript, TypeScript, ECMAScript Harmony, шаблоны Jade, Zen Coding, Emmet, и, конечно же, JavaScript.

PhpStorm включает в себя всю функциональность WebStorm (HTML/CSS редактор, JavaScript редактор) и добавляет полнофункциональную поддержку PHP и баз данных / SQL.

Выделим ключевые возможности.

Интеллектуальный редактор PHP кода с подсветкой синтаксиса, автодополнением кода, расширенными настройками форматирования кода, предотвращением ошибок налету.

Поддерживает PHP 7.0, 5.6, 5.5, 5.4 и 5.3, генераторы, сопрограммы и все синтаксические улучшения.

PHP рефакторинги, code (re)arranger, детектор дублируемого кода.

Поддержка Vagrant, Composer, встроенный REST клиент, Command Line Tools, SSH консоль.

Поддержка фреймворков (MVC view для Symfony2, Yii) и специализированные плагины для ведущих PHP фреймворков (Symfony, Magento, Drupal, Yii, CakePHP и многие другие).

Визуальный отладчик для PHP приложений, валидация конфигурации отладчика, PHPUnit с покрытием кода (поддержка PHPUnit 5), а также интеграция с профилировщиком.

HTML, CSS, JavaScript редактор. Отладка и модульное тестирование для JS. Поддержка HTML5, CSS, Sass, SCSS, Less, Stylus, Compass, CoffeeScript, TypeScript, ECMAScript Harmony, Emmet и других передовых технологий веб-разработки.

Поддержка стилей кода, встроенные стили PSR1/PSR2, Symfony2, Zend, Drupal и другие.

Интеграция с системами управления версиями, включая унифицированный интерфейс. Удаленное развертывание приложений.

Live Edit: изменения в коде можно мгновенно просмотреть в браузере без перезагрузки страницы.

Была выбрана интегрированная среда разработки JetBrains PhpStorm 9.0, потому что данный редактор отлично подойдет как для продвинутых web-разработчиков, так и для программиста с начальным уровнем знаний web-программирования. Возможность автодополнения кода позволяет не запоминать все базовые функции языка php, js. Данная IDE индексирует весь сайт, что позволяет быстро перемещаться между пользовательскими функциями и переменными для работы с ними. Кроме того, встроенный отладчик кода позволит эффективно разрабатывать web-сайт и систему онлайн бронирования.

2.4 Обоснование проектных решений по технологическому обеспечению

Технологическое обеспечение – это разделение информационной системы рассматриваемой организации на подсистемы по технологическим этапам обработки различных видов информации:

– первичной информации (этапы технологического процесса сбора, передачи, накопления, хранения, обработки первичной информации, получения и выдачи результатной информации);

– организационно-распорядительной документации (этапы получения входящей документации, передачи на исполнение, этапы формирования и хранения дел, составления и размножения внутренних документов и отчетов);

– баз данных и знаний (этапы формирования баз данных и знаний, ввода и обработки запросов на поиск решения, выдачи варианта решения и объяснения к нему);

Технологическое обеспечение системы онлайн бронирования «Кома» будет заключаться в создание новой технологии работы, а именно:

– сбора информации от клиентов: контактные данные, дата и время игры;

– занесение информации в базу данных, в виде создания новой заявки и уведомления с помощью подключенного сервиса;

– проверка, уточнение, подтверждение и корректировка информации;

– подготовка и выдача результативной информации;

При внедрении системы онлайн бронирования «Кома» сотрудникам необходимо будет получать уведомления с помощью подключенного сервиса (E-mail или SMS) и редактировать либо отменять бронирование через специальный web-интерфейс. В данном интерфейсе будет доступна вся необходимая информация о заказе, о клиенте, о дате и времени, чтобы сотрудник компании мог быстро произвести необходимые операции.

Внедрение новой технологии обосновывается:

– необходимостью в более оперативном приеме заявок и удобном контроле за графиком работы организации;

– необходимостью в доступе за контролем или регулированием бронирования из любого места с помощью персонального компьютера;

Разработанная система онлайн бронирования позволит:

– повысить скорость обработки заявок;

– повысить возможность удобного использования контактной информации;

– повысить контроль над работой сотрудников, путем отслеживания их манипуляций с бронью;

При работе с системой онлайн бронирования могут возникнуть следующие ошибки:

– ошибки при выводе данных;

– ошибки при отображении времени.

2.5 Обоснование выбора программных средств

Программные средства для решения поставленной цели выпускной квалификационной работы можно разделить на следующие категории:

– Программные средства для моделирования бизнес процессов технологии работы;

– Программные средства для проектирования модели данных.

В основе первых двух категорий лежат CASE-технологии – это технология автоматизированной разработки систем, обеспечивающие с помощью предназначенного для этих целей инструментария (CASE-средств) комплексную поддержку разработки либо поддержку отдельных стадий жизненного цикла сложных программных систем – их специфицирование, проектирование, реализацию, тестирование, сопровождение и развитие.

Для моделирования бизнес процессов технологии работы было выбрано

программное средство AllFusion Process Modeler 7.

Для проектирования модели данных было выбрано программное средство AllFusion Erwin Data Modeler 7.

AllFusion Erwin Data Modeler 7 – средство для проектирования и документирования баз данных, которое позволяет создавать, документировать и сопровождать базы данных, хранилища и витрины данных.

Особенности:

– Синхронизация моделей/баз данных;

– Автоматизированное создание структуры базы данных и обратное проектирование;

– Поддержка нотаций (IDEF1X, IE, Dimensional);

– Перенос структур баз данных из одного типа СУБД в другой.

Выводы по второму разделу.

Со стороны технического обеспечения было принято решение в покупке персонального компьютера с рекомендуемыми техническими характеристиками, т.к. в компании ПК отсутствует.

С точки зрения информационного обеспечения было описаны этапы сбора, обработки и ввода информации. Определена реляционная модель данных.

С точки зрения программного обеспечения выбраны дополнительное прикладное программное обеспечение для работы интернет-магазина: веб-сервер Apache, система управления базами данных MySQL, интерпретатор PHP, утилита управления MySQL – phpMyAdmin. Проектирование и разработка интернет-магазина будет происходить интегрированной среде разработки JetBrains PhpStorm.

Выбор программных средств основывается на CASE-технологиях. Для проектирования информационной модели будет использовано Fusion Erwin Data Modeler 7, а для описания технологии работы – AllFusion Process Modeler 7.

3 Проектная реализация

3.1 Информационное обеспечение автоматизированной системы онлайн бронирования квест комнаты

Информационная модель и ее описание «КАК ДОЛЖНО БЫТЬ»

Новая разработанная информационная модель содержит новый элемент, который выполняет функцию автоматизированной системы онлайн бронирования квест комнаты связанный с базой данных.

На рисунке 3.1 представлена новая информационная модель в нотации IDEF3.

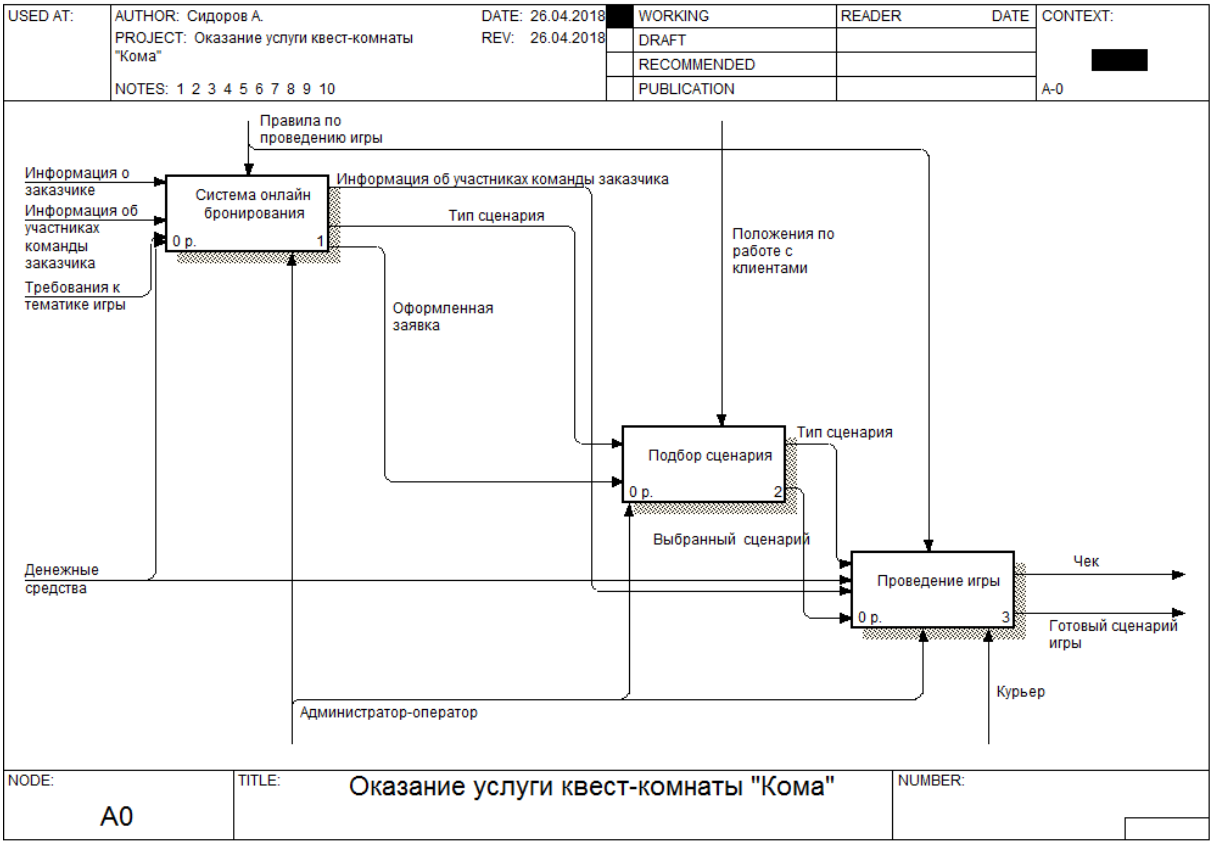

Рисунок 3.1 – Детализация контекстной диаграммы «Оказание услуги квест комнаты «Кома», модель «КАК БУДЕТ» в нотации IDEF3 после реинжиниринга

Особенность данной информационной модели, в том, что она

объединяет блоки «Оформление заявки» и «Обработка заявки» в один. Задачи, выполняемые в данных блоках, будет выполнять блок «Система онлайн бронирования».

Входящей информацией является информация о заказчике, об участниках команды заказчика, требования к тематике игры.

Выходной информацией является тип сценария и информация об участниках команды заказчика.

Механизмами являются администратор-оператор и курьер.

Управляющими данными являются правила по проведению игры.

Декомпозиция блока «Система онлайн бронирования» представлена на рисунке 3.2.

Рисунок 3.2 – Декомпозиция блока «Система онлайн бронирования» в нотации DFD

В текущей декомпозиции данный блок выполняет конкретную задачу, а именно: оформление заявки и добавление его в БД.

Когда клиент обращается на сайт для выбора сценария, модуль предоставляет необходимую информацию. После выбора сецнария и заполнения формы, сайт отправляет собранную информацию в БД – указывает имя и контактные данные.

Также указывается дата и время игры и сумма.

После чего система онлайн бронирования автоматически присылает уведомление об оформленной заявке на привязанный сервис.

Характеристика первичных документов с нормативно-справочной и входной оперативной информацией.

Первичный документ – это документ, включающий исходные сведения, полученные в результате наблюдений, разработок, исследований, результатов и других видов человеческой деятельности. Состав первичных документов для каждого класса задач различается. Необходимо систематизировать первичные документы по первому классу задач.

Для класса задач «Анализ технологии автоматизированной системы онлайн бронирования» первичные документы содержат большой объем числовой и текстовой информации. К этой группе первичной информации относятся следующие документы:

Регистрация клиента.

Данный первичный документ содержит 2 поля, из которых одно числовое, второе текстовое. Объем документа маленький. Для того чтобы записаться на игру на разработанном web – сайте пользователю необходимо пройти стандартную регистрацию, введя персональные данные для дальнейшего их внесения в клиентскую базу, а также использования при записи на игру.

Форма записи на игру.

Первичный документ содержит поля для выбора типа сценария, даты и времени бронирования. Все данные вводятся в интерактивной форме пользователем и вносятся в базу данных.

Все перечисленные документы заносятся в БД через соответствующие

формы разработанного ресурса. Формы имеют унифицированный вид, что упрощает работу с ними. Если клиент подает заявку на услугу по телефону, либо лично в квест комнате, данные формы сайта заполняет администратор-оператор.

Характеристика базы данных

Инфологическая модель базы данных автоматизированной системы онлайн бронирования состоит из 9 таблиц, связанные между собой. Схема представлена на рисунке 3.3.

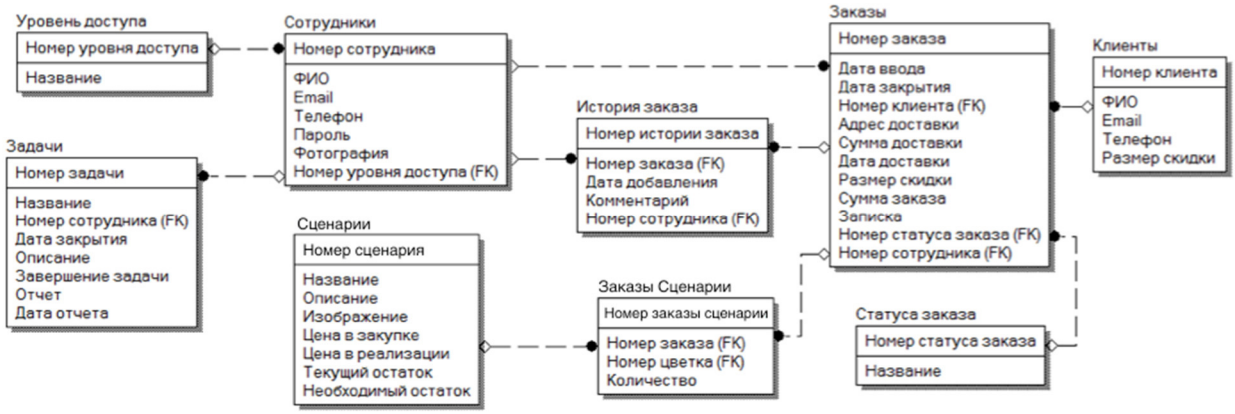

Рисунок 3.3 – Инфологическая модель базы данных

Физические модель данных является образом базы данных на основе логической модели данных. На рисунке 3.4 представлена физическая модель базы данных.

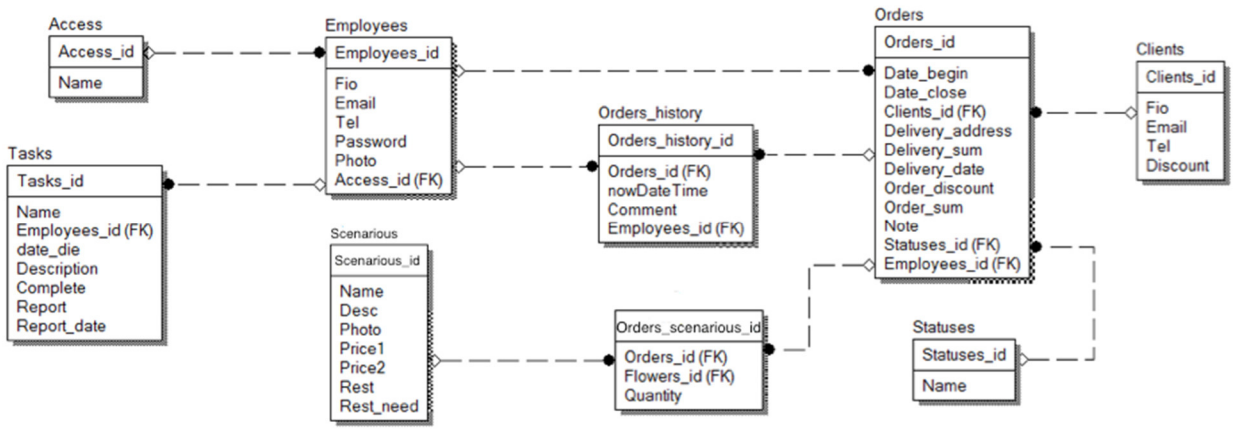

Рисунок 3.4 – Физическая модель базы данных

База данных состоит из 9 таблиц. Все названия таблиц и их атрибуты описаны в таблицы А.1 приложения А. Данная таблица представляет даталогическую модель базы данных.

Все сущности находятся в третьей нормализованной форме. Это означает, что она находится во второй нормальной форме, и отсутствуют транзитивные функциональные зависимости не ключевых атрибутов от ключевых.

Сущности «Уровень доступа» и «Сотрудники» имеют тип связи «один-ко-многим». Данные сущности связаны между собой по ключу «Номер уровня доступа».

Сущности «Сотрудники» и «Задачи» имеют тип связи «один-ко-многим». Данные сущности связаны между собой по ключу «Номер сотрудника».

Сущности «Сотрудники» и «История заявки» имеют тип связи «один-ко-многим». Данные сущности связаны между собой по ключу «Номер сотрудника».

Сущности «Заявки» и «История заказа» имеют тип связи «один-ко-многим». Данные сущности связаны между собой по ключу «Номер заявки».

Сущности «Клиенты» и «Заявки» имеют тип связи «один-ко-многим». Данные сущности связаны между собой по ключу «Номер клиента».

Сущности «Статус заявки» и «Заявки» имеют тип связи «один-ко-многим». Данные сущности связаны между собой по ключу «Номер статуса заявки».

Сущности «Сценарии» и «Заявки» имеют тип связи «многие-ко-многим». Но чтобы не было избыточности данных, создана таблица «Заявки_Сценарии». Теперь «Заявки» и «Заявки_Сценарии», «Заявки» и «Заявки_Сценарии» имеют тип связи один-ко-многим. Данные сущности связаны между собой по ключу «Номер заявки» и «Номер сценария» соответственно.

Характеристика результативной информации

Результативной информацией автоматизированной системы онлайн бронирования будут являться конечные формы, данные с которых пользователь может использовать для различных целей: анализа, сбора, изучения, исследования и других целей.

Ниже перечислена краткая характеристика результативной информации:

– web-страница Мои задачи – на данной странице содержится информация об активных, просроченных задачах менеджера;

– web-страница Продажи – на данной странице отображена сводная информация по всем продажам (заказам). Кроме того, заказы можно отфильтровать по необходимому статусу;

– web-страница Детали заказа – на данной странице содержится вся информация по заказу, по составу заказа, и по его истории;

– web-страница Клиента – содержатся сводная информация по обслуживаемым клиентам.

– web-страница Карточка клиента – содержит всю подробную информацию о клиенте, включая все активные и завершенные заказы.

– web-страница Задачи – содержит информация о всех активных, просроченных и выполненных задачах;

– web-страница Сотрудники – содержит сводную информацию о пользователях и администраторах автоматизированной системы брони;

– web-страница Информация о сотруднике – содержит детальную информацию о выбранном пользователе или администраторе.

3.2 Программное обеспечение автоматизированной системы онлайн бронирования

Общая характеристика разрабатываемой системы

Профессиональные автоматизированные системы – это главный

инструмент общения человека с информационными системами, которые выполняют роль автономных рабочих мест, интеллектуальных терминалов больших ЭВМ, рабочих станций в локальных сетях. Автоматизированные системы имеют открытую архитектуру и легко адаптируются.

Автоматизированные системы имеют проблемно-профессиональную ориентацию на конкретную предметную сферу, и является средством общения специалиста с автоматизированными информационными системами.

Автоматизированная система онлайн бронирования обеспечивает:

- простоту, удобство и дружеское отношение к пользователю;
- простоту адаптации к конкретным функциям пользователя;
- компактность размещения и невысокие требования к условиям эксплуатации;
- высокую надежность и живучесть;
- сравнительно простую организацию технического обслуживания.

Эффективным режимом работы автоматизированной системы онлайн бронирования есть его функционирования в рамках локальной вычислительной сети. Созданные автоматизированные системы онлайн бронирования специалистов дают возможность пользователю работать в диалоговом режиме, оперативно решать текущие задачи, удобно вводить дани, вести контроль, обработку информации, определять достоверность результатной информации, выводить и передавать каналами связи. Информационное обеспечение автоматизированной системой онлайн бронирования ориентируется на конкретную, обычную для пользователя, предметную сферу.

В любой разработке программного обеспечения оно должно в первую очередь выполнять главную поставленную перед ним задачу. В текущей выпускной квалификационной работе перед автоматизированной системой онлайн бронирования стоит задача вести учет принятых заявок, с возможностью оформления заявки по выбранному сценарию. Чтобы решить эти задачи, необходимо описать функции автоматизированной системы

онлайн бронирования.

К основным функциям относятся:

– основные функции – выполняют прямые функции, например ввод первичной информации, её обработки, анализ, расчет, выдача готового результата, ответы на запросы;

– служебные функции – выполняют непрямые, дополнительные функции программного обеспечения (например, проверка пароля, архивация базы данных).

К основным функциям работы автоматизированной системы онлайн бронирования относятся:

– ввод, редактирование, удаление и поиск клиентов;

– ввод, редактирование, удаление и поиск сценариев;

– ввод, редактирование, удаление и поиск заказов;

– фильтрование заказов по нужному статусу;

– ввод, редактирование, удаление и поиск сотрудников;

– ввод, редактирование, удаление и поиск задач;

К служебным функциям CRM-системы можно отнести следующие:

– проверка на правильность ввода данных;

– экранирование html-тегов, sql-инъекций и опасного PHP-кода;

– управление статусами заявок;

– добавление сотрудниками комментариев по заявке.

Структура сайта, состав и типы используемых файлов

Для удобства структура автоматизированной системы онлайн бронирования, представлена в виде каталогов. Таким образом, код всего проекта разбит по папкам в зависимости от выполняемых задач.

Корневой каталог содержит следующие папки и файлы:

– каталог config – содержит файл config.php, который содержит константы и настройки подключения к БД;

– каталог controller – содержит controller.php, который обрабатывает

запросы пользователя;

– каталог function – содержит function.php, который описывает пользовательские служебные функции (например, для экранирования данных, редиректа окна и др);

– каталог model – содержит model.php, который описывает функции отправляющие запросы в БД на получение, вставку, редактирование, удаление записей;

– каталог sql – содержит koma_db.sql, в котором описан запрос на создание таблиц в БД и заполнение их данными;

– каталог userfiles – содержит фотографии товаров и сотрудников;

– каталог views – содержит файлы шаблоны сайта с расширением php, каталог css (каскадные таблицы стилей), js (скрипты), images (изображения), inc (подключает шапку и подвал), bootstrap (css фреймворк).

На рисунке 3.5 изображена структура сайта, в виде иерархии каталогов.

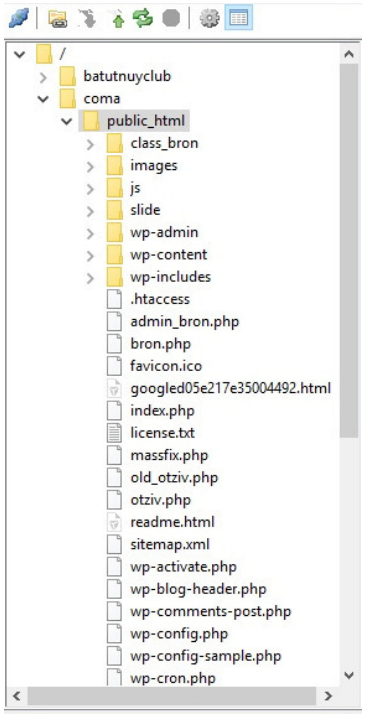

Рисунок 3.5 – Структура автоматизированной системы онлайн бронирования

Описание программных модулей

Программный модуль – это часть программы, имеющая конкретное

назначение, выполняющая определенные функции, характерные для данного программного модуля, полностью или частично зависящая от других модулей.

Рассмотрим различные программные модули.

Головной модуль - управляет запуском программного продукта (существует в единственном числе). Головным модулем автоматизированной системы онлайн бронирования является файл index.php, который автоматически запускается при открытии.

Управляющий модуль – обеспечивает вызов других модулей на обработку. В данной разработке управляющим модулем является controller.php – его задача состоит в получение и обработке запросов пользователя, передача подготовленных данных в сервисный модуль (model.php) для получения необходимых результативных данных, и передача этих данных в рабочий модуль (view/).

Рабочие модули - выполняют функции представления данных. Данным модулем являются шаблоны и макеты web-страниц в формате html и php, которые отображают полученные данные из model.php пользователю.

Сервисные модули и библиотеки, утилиты - осуществляют обслуживающие функции. К таким модулям относится model.php, задача которого состоит в отправке определенных запросов (добавление, редактирование, удаление, выборка информации) к базе данных.

Web-сайт построен по архитектуре программирования MVC.

MVC (Model-view-controller, «Модель-представление-поведение», «Модель-представление-контроллер») – это шаблон проектирования приложений, при котором управляющая логика поделена на три отдельных компонента таким образом, что модифицирование одного из них дает минимальное влияние на остальные.

MVC Модель (Model). Модель предоставляет данные (обычно для View), а также реагирует на запросы (обычно от контроллера), изменяя своё состояние.

MVC Представление (View). Отвечает за отображение информации

(пользовательский интерфейс).

MVC Поведение (Controller). Интерпретирует данные, введённые пользователем, и информирует модель и представление о необходимости соответствующей реакции.

Особенностью при использовании MVC в PHP, является то, что существует одна точка входа в php приложение – index.php

Данные о PHP модели содержаться в ее атрибутах и могут быть изменены только через специальные функции. Модель может содержать в себе несколько представлений. Функции могут добавлять данных в БД, удалять, редактировать и читать. Ниже представлен код получения списка товаров для главной страницы автоматизированной системы онлайн бронирования:

```
<?php
	function runQuerySelect($query){
		$res = mysql_query($query);
		$arr = array();
		while ($row = mysql_fetch_assoc($res)){ $arr[] = $row; }
		return $arr;
	}

	function select_table($table){
		$query = "SELECT * FROM $table";
		return runQuerySelect($query);
	}
?>
```

Функция select_table($table) выполняет запрос на получение всех данных из таблицы, название которой, хранится в переменной $table.

А функция runQuerySelect($query) непорседственно выполняет sql-запрос, который содержится в переменной $query, а полученный результат помещает в массив $arr.

Ниже представлена функция модели, которая добавляет клиента в БД:

```
<?php

function insert_clients(){
   $fio = clear($_POST['fio']);
   $email = clear($_POST['email']);
   $tel = clear($_POST['tel']);
   $discount = clear($_POST['discount']);
   $query = "INSERT INTO `clients`(`fio`, `email`, `tel`, `discount`)
            VALUES ('$fio', '$email', '$tel', $discount)";
```

```
    return runQueryInsert($query);
}

function runQueryInsert($query){
    mysql_query($query);
    if(mysql_affected_rows()>0){
        $_SESSION['message'] = "<div class='alert alert-info'>Запись добавлена.</div>";
    }else{
        $_SESSION['message'] = "<div class='alert alert-danger'>Ошибка при добавлении
записи.</div>";
    };
    return mysql_insert_id();
}
?>
```

Аналогично работают функции на редактирование данных и удаление.

PHP контролеры получают запросы пользователей, которые направляются через index.php, и в соответствии с ними, корректируют работу модели, т.е. автоматизированной системы онлайн бронирования. Ниже представлен пример работы контролера:

```
<?php
// получение динаминой части шаблона
if(empty($_GET['view'])){ $view = 'auth'; }
else { $view = $_GET['view']; }
switch($view){
    case('staff_all'):
    // сотрудники
    verifying_access(); // проверка авторизации
    $employees = select_table('employees'); // получение списка сценариев
    $access = select_table('access');

    // добавление сценария
    if(isset($_POST['add'])){
        insert_employees();
        redirect();
    }
    break;
}
require_once TEMPLATE.'index.php';
?>
```

Данный код можно описать следующим образом:

Если при получении через метод $_GET['view'] параметр view – пустой, то инициализируем переменную $view = ‘auth, иначе если он не пустой, то инициализируем $view значением, которое хранится в данном параметре.

Оператор switch($view) получает данную переменную и проверяет условие. Если есть описание условия, например, для $view = ‘'staff_all'’, то он попадает в блок case(‘'staff_all'’), где выполняются определенные действия, в данном случае получается список сценариев и массив с уровнями доступа.

Далее с помощью функции require_once подключается файл index.php, который подключает шаблон со значением $_GET['view'], например auth.php. Данный шаблон будет являться представлением модели.

Представление отслеживает изменение в модели и создает или меняет интерфейс автоматизированной системы онлайн бронирования.

Ниже представлен пример вывода списка сценариев.

```
<?php foreach($employees as $key => $val): ?>
   <?php if($val['access_id'] == 2 ): ?>
      <div class="col-md-12">
         <a href="?view=staff_one&id=<?=$val['employees_id']?>">
            <div class="col-md-4">
               <img src="/userfiles/staff/<?=$val['photo']?>" class="img-circle">
            </div>
            <div class="col-md-8">
               <h5><?=$val['fio']?></h5>
               <?=$val['email']?><br/>
               <?=$val['tel']?>
            </div>
         </a>
      </div>
   <?php endif; ?>
<?php endforeach; ?>
```

Программный код всех php-страниц представлен в Приложении В.

При обращении пользователем по нужному URL выбирается соответственный контролер, который обращается к представлению и модели, и выводится информация. Другими словами контролер в MVC есть связующим звеном модели и представления.

Для создания базы данных необходимы sql-запросы на создания таблиц и вставки данных. Код sql-запроса на создание таблиц представлен в Приложении В.

MVC можно представить схемой, изображенная на рисунке 3.4.

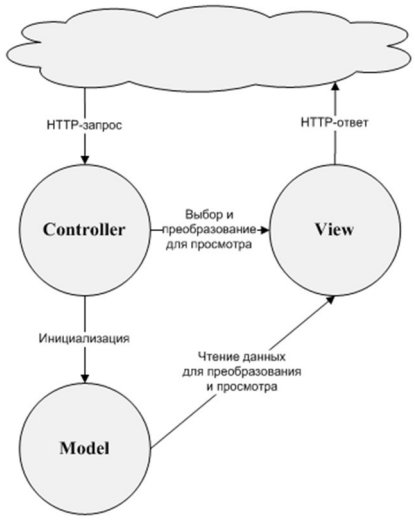

Рисунок 3.4 – Процесс обработка запроса в схеме MVC

Web-страницы имеет унифицированный интерфейс, т.е. все поля имеют удобный, точный и понятные формы с подсказками.

Для проектирования форм использовались html элементы input, textarea, radio и button.

Для вывода данных использовались html-элементы table и div.

3.3 Технологическое обеспечение системы онлайн бронирования сотрудника квест-комнаты с функций ведения статистического учета

Организация технологии сбора, передачи, обработки и выдачи информации состоит в последовательности организованных операций.

Технология сбора информации основывается на считывание данных с материалов, которые представлены в приложении. Пользователь при оформлении заявки вводит в печатную форму персональные данные о заказчике: email, телефон, фамилия, имя, отчество, дата, время, стоимость, указывает сценарий, которые выбран в заявке, размер скидки и после чего итоговая сумма рассчитывается автоматически.

Технология передачи данных заключается в занесении собранных данных в базу данных. Данный процесс происходит, когда пользователь

нажимает кнопку «ЗАБРОНИРОВАТЬ». Вводимые данные заносятся в базу данных для обработки заявки.

Технология обработки данных заключается в анализе и корректировке введенных данных. После занесения данных в БД администратор проверяет корректность введенных данных. Если все данные корректны, то администратор меняет статус заявки на необходимый.

Технология выдачи – это результат совокупности проделанных операций сбора, передачи, обработки данных. Результатом работы является лист данных. В данном листе данных отражены все данные по заявке.

3.4 Описание контрольного примера реализации проекта

Для демонстрации работы автоматизированной системы онлайн-бронирования квест-комнаты «Кома» с функцией дальнейшей обработки будет рассмотрен процесс оформления заявки на проведение игры по сценарию «Во Все Тяжкие». Дата проведения игры 15.06.2018. Время проведения игры 13:00. Также необходимо занести заявку клиента в БД. После совершения бронирования нужно проверить статусы бронирования будующих дат и времени, для избежания ошибок с оффлайн системой журнального учета поступающих телефонных заявок.

Для запуска системы онлайн-бронирования клиенту необходимо запустить браузер с персонального компьютера или любого мобильного устройства с данной поддержкой. Далее необходимо в адресной строке ввести ссылку на искомый ресурс системы онлайн-бронирования в формате http://квест-кома.рф/бронирование/.

В результате этих действий откроется страница бронирования как на рисунке 3.5.

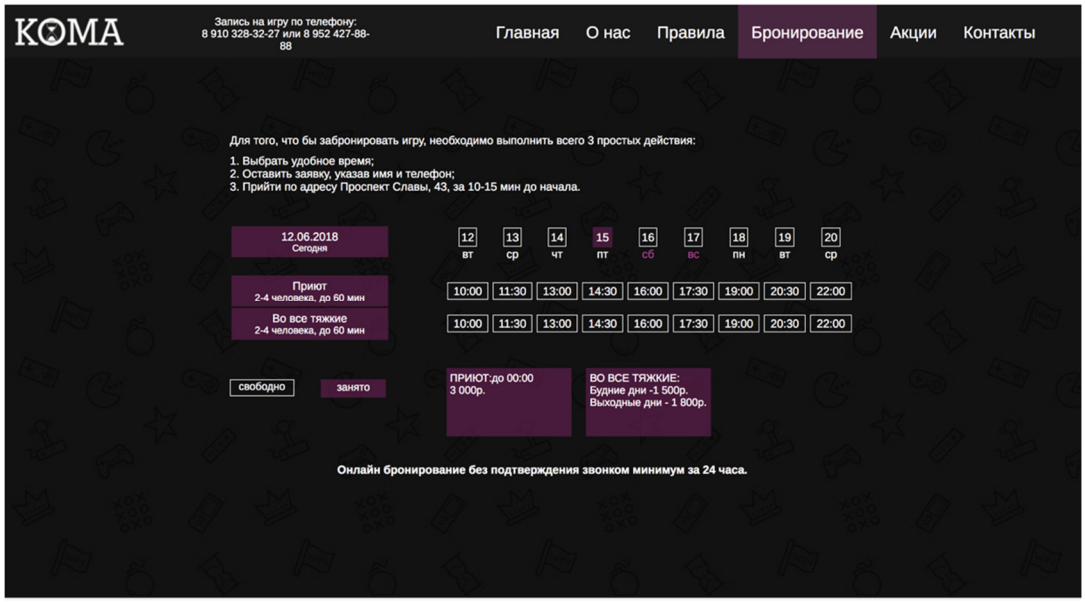

Рисунок 3.5 – Страница бронирования

Для бронирования игры необходимо выбрать нужную дату и далее необходимое время расположенное горизонтально по отношению к сценарию игры. При нажатии клиентом на выбранную ячейку откроется дополнительное окно ввода данных клиента как на рисунке 3.6.

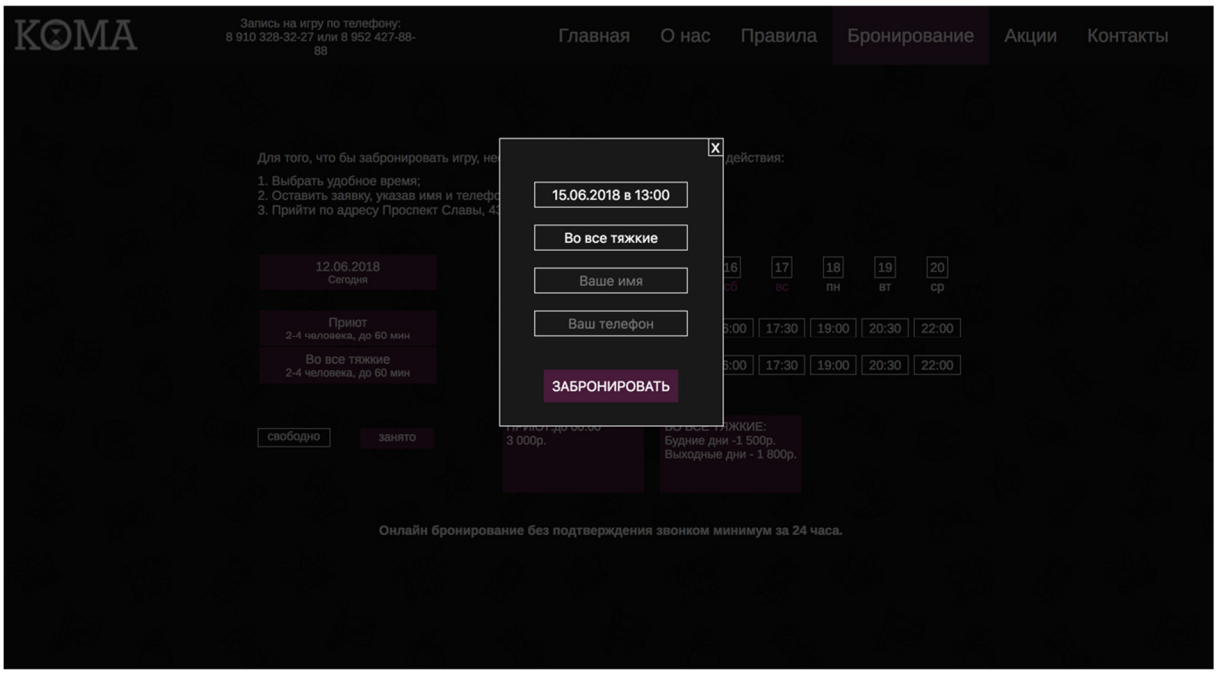

Рисунок 3.6 – Окно ввода данных клиента

Далее заказчик вводит в пустые подсвеченные поля свои данные

нажимает кнопку «ЗАБРОНИРОВАТЬ» как на рисунке 3.7.

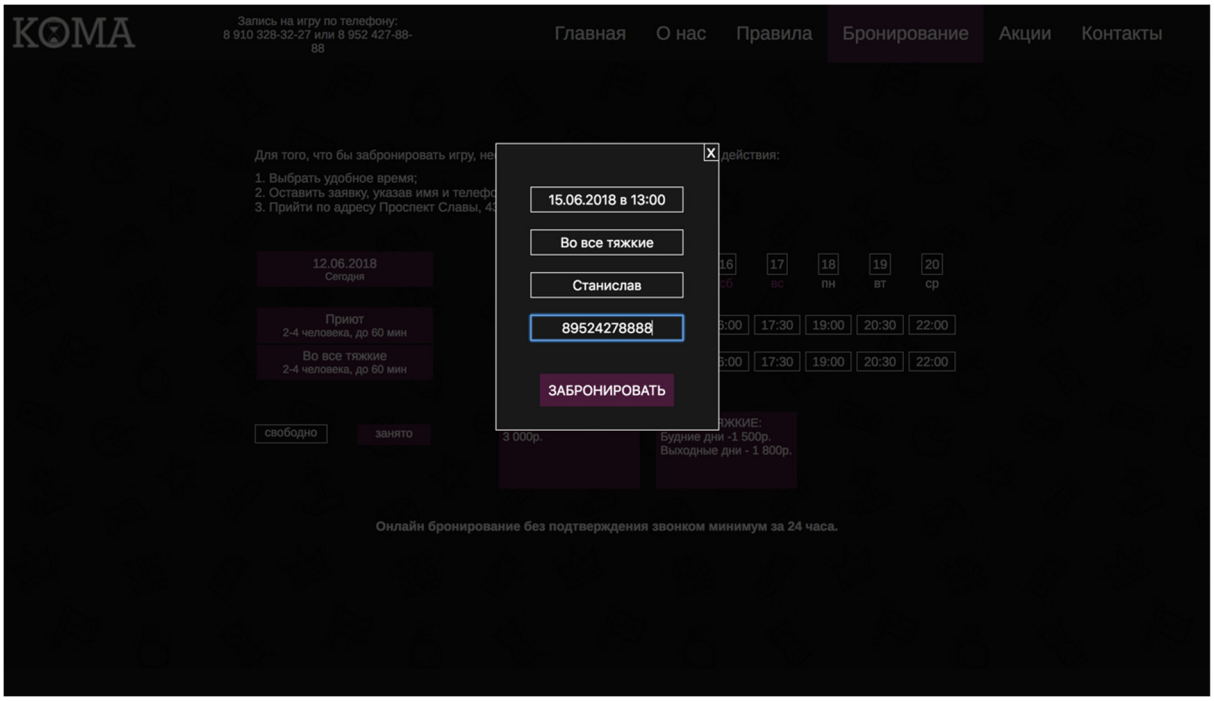

Рисунок 3.7 – Заполнение формы

После успешного оформления бронирования на привязанный e-mail ящик приходит уведомление о бронировании, для входа в который он вводит сначала логин как на рисунке 3.8.

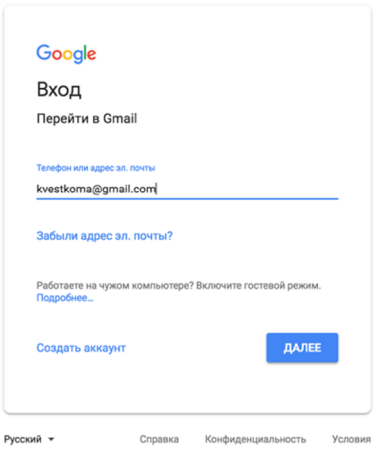

Рисунок 3.8 – Ввод логина почтового ящика

Затем администратор вводит пароль почтового ящика как на рисунке 3.9.

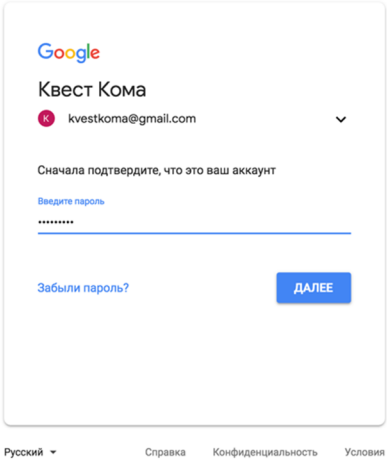

Рисунок 3.9 – Ввод пароля почтового ящика

Если всё данные верны, можно будет просмотреть историю входящих сообщений как на рисунке 3.10.

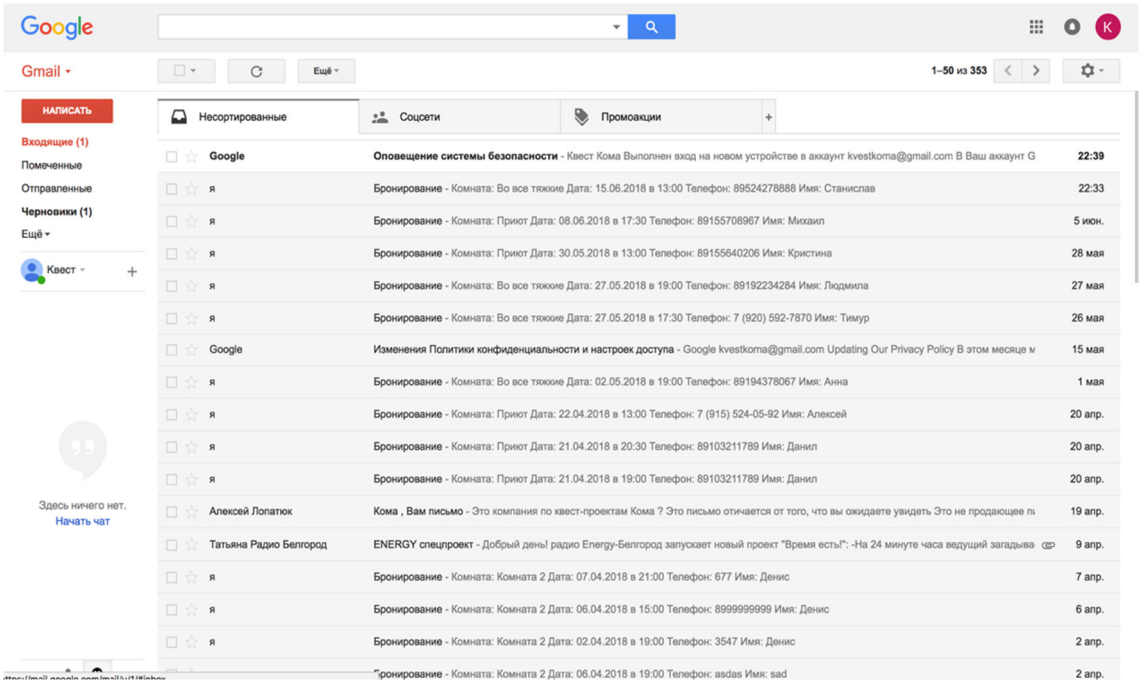

Рисунок 3.10 – История входящих сообщений

Далее администратору необходимо открыть последнее по дате сообщение как на рисунке 3.11.

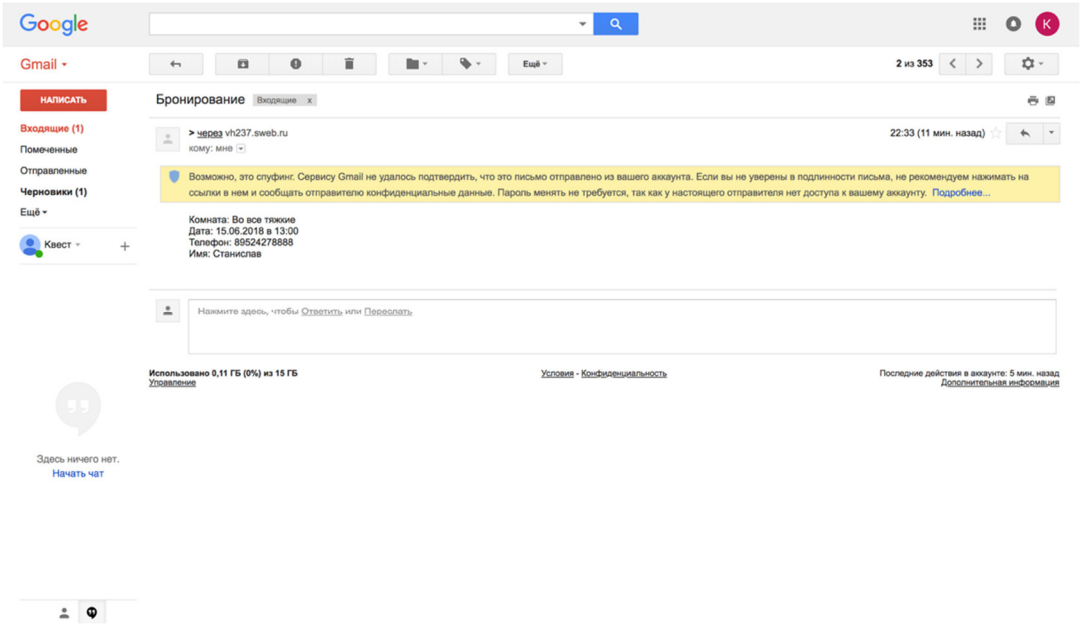

Рисунок 3.11 – Последнее сообщение с бронированием

Таким образом, был оформлен заказ и продемонстрированы основные функции системы автоматизированного онлайн бронирования.

3.5 Организационно-экономическая часть

Целесообразность разработки с экономической точки зрения

Автоматизированная система онлайн-бронирования – это программное средство, которое выполняет специализированные операционные задачи организации. Система онлайн-бронирования предназначается для повышения эффективности работы администраторов-операторов, с помощью организации автоматизированного процесса према заявки на платформе web-приложения.

Разработка системы онлайн-бронирования абсолютно оправдана и целесообразна с экономической и организационной точки зрения.

Во-первых, системой онлайн бронирования для работы используются

бесплатное программное обеспечение (MySQL, Apache, JavaScript, PHP).

Во-вторых, высокая популярность языка web-программирования и легкость освоения позволяют модернизировать код системы онлайн бронирования под новые задачи и условия работы организации без существенных финансовых и трудовых затрат. Для доработки кода web-приложения необходим базовый уровень знания языков php и html.

В-третьих, одно из главных преимуществ web-приложения перед win32-приложением в том, что не потребуется компиляция кода.

Внедрение системы онлайн бронирования в деятельность организации «Кома» приведёт к следующим результатам:

– повышение уровня обслуживания клиентов за счет оперативности и точности выполнения заявки;

– повышение уровня организации сотрудников за счет организованности работы;

– повышение качества продаж за счет анализа заказов и стабильного следования намеченному графику работы;

3.6 Выбор методики и расчет экономической эффективности разработки

В рамах выпускной квалификационной работы было разработана автоматизированная система онлайн-бронирования, которая является универсальным средством. Разработанную версию разработки можно распространять на рынке программного обеспечения и реализовывать в организациях и компаниях связанные с проведением квест игр. Но разработанный автоматизированной системе онлайн-бронирования достаточно универсален, который подойдет для любых различных компаний, где необходимо вести учет броинрования времени заявок клиентов, а также видения клиентской базы данных.

Исходные данные для проведения технико-экономического обоснования представлены в таблице 3.1.

Таблица 3.1 – Исходные данные

Обозначение	Наименование показателя	Единицы измерения	Значение показателя
$C_{ЭВМ}$	Стоимость электронно-вычислительной машины (ЭВМ)	руб.	30000
$Д_М$	Среднее количество дней в месяце	дни	30
$\rho_н$	Норматив рентабельности	–	0,2
$\omega_д$	Коэффициент, учитывающий дополнительную заработную плату разработчика программы	–	0,15
$\omega_с$	Коэффициент, учитывающий начисления органам социального страхования	–	0,356
q_I	Количество I-задач, решаемых потребителем	зад. год	50
$t'_{М.В.I}$	Время решения I-ой задачи базовой программой	маш. час	5
$n_п$	Количество организаций, которые приобретут данную программу	шт.	15
$Z_{ЭЛ}$	Тариф за 1 кВт/час	руб.	1,3
$\varepsilon_н$	Нормативный коэффициент эффективности капиталовложений	–	0,25
$T_С$	Срок службы разработанной программы	год	3
НДС	Налог на добавленную стоимость	%	18
$T_Р$	Количество рабочих дней в году	дн	365
$t_{СМ}$	Продолжительность смены	ч	8
α	Простои ЭВМ	%	20
P	Мощность, потребляемая электронно-вычислительной машиной (ЭВМ)	кВт	0,4
$S_Д$	Стоимость деталей, заменяемых при ремонте	руб.	1500
ОЗП	Основная заработная плата программиста	руб.	16000

Расчет затрат на разработку web-приложения

Суммарные затраты на разработку автоматизированной системы онлайн бронирования рассчитываются по следующей формуле:

$S_{РП} = S_{ЗП} + S_{НАК}$,

где:

$S_{ЗП}$ – затраты по заработной плате web-программиста;

$S_{НАК}$ – накладные расходы.

Затраты по заработной плате web-программиста рассчитываются по формуле:

$S_{ЗП} = ОЗП \cdot (1+\omega_с) \cdot (1+\omega_д) \cdot t_{рi}$,

где:

ОЗП – основная заработная плата web-программиста за месяц;

$t_{рi}$ – время, необходимое для разработки автоматизированной системы онлайн бронирования программистом i-го разряда (чел.-мес.);

$\omega_д$ – коэффициент, учитывающий дополнительную заработную плату web-программиста автоматизированной системы онлайн бронирования, в долях к сумме основной заработной платы;

$\omega_с$ – коэффициент, учитывающий начисления органам социального страхования на заработную плату разработчика автоматизированной системы онлайн бронирования, в долях к сумме основной заработной плате разработчика.

Автоматизированная система онлайн бронирования разрабатывалась 60 дней, если учесть, что в одном месяце 22 рабочих дня, то месяцев на разработку одним программистом было затрачено:

$t_{р1} = 60/22 = 2,73$ (чел. –мес.)

Таким образом, затраты по заработной плате программиста составят:

$S_{ЗП} = 16000 \cdot (1+0,356) \cdot (1+0,15) \cdot 2,73 = 68045,55$ (руб.)

Накладные затраты рассчитываются с учетом $\omega_н$ – коэффициента, определяющего уровень накладных расходов организации по формуле:

$S_{НАК} = ОЗП \cdot \omega_н \cdot t_{р12}$,

$S_{НАК} = 16000 \cdot 1{,}2 \cdot 2{,}73 = 52363{,}64$ (руб.)

Таким образом, суммарные затраты на разработку web-приложения составляют:

$S_{РП} = 68046{,}55 + 52363{,}64 = 120410{,}2$ (руб.)

Расчет цены разработанной web-приложения

Оптовая цена разработанной автоматизированной системы онлайн-бронирования определяется по следующей формуле:

$Z_П = S_{РП} + П$,

где:

$Z_П$ – оптовая цена (руб.);

$S_{РП}$ – суммарные затраты на разработку web-сайта (руб.);

$П$ – прибыль, рассчитанная по формуле:

$П = \rho_н \cdot S_{РП}$,

где:

$\rho_н$ – норматив рентабельности, учитывающий прибыль организации, разрабатывающей данную автоматизированную систему онлайн-бронирования в долях ко всем затратам данной организации на разработку автоматизированной системы онлайн бронирования.

Итак,

$Z_П = S_{РП} \cdot (1+ \rho_н)$,

$Z_П = 120410{,}2 \cdot (1+ 0{,}2) = 144492{,}2$ (руб.)

Розничная цена автоматизированной системы онлайн-бронирования рассчитывается с учетом налога на добавленную стоимость ($НДС$ = 18%) по формуле:

$$Z_{Пр} = Z_П + НДС = Z_П + \frac{Z_П * 18}{100} = Z_П * (1 + 0{,}18)$$

$Z_{Пр} = 144492{,}2 \cdot (1+0{,}18) = 170500{,}8$ (руб.)

Выручка от продаж при условии $n_п$ = 15 – количество организаций,

желающих прибрести автоматизированную систему онлайн-бронирования, составит:

$B = Z_{\text{Пр}} \cdot n_{\text{п}}$,

B = 170500,8 ·15 = 2557512 (руб.).

Расчет капитальных вложений

Капиталовложения, связанные с работой электронно-вычислительной машиной рассчитываются по формуле:

$K_{\text{ЭВМ}} = C_{\text{ЭВМ}}+S_{\text{Т}}+S_{\text{М}}+S_{\text{З}}+S_{\text{ПЛ}}$,

где:

$C_{\text{ЭВМ}}$ – стоимость ЭВМ (руб.);

$S_{\text{Т}}$ – стоимость транспортировки ЭВМ (руб.);

$S_{\text{М}}$ – стоимость монтажа ЭВМ (руб.);

$S_{\text{З}}$ – стоимость запасных частей (руб.);

$S_{\text{ПЛ}}$ – стоимость площади установки ЭВМ (руб.).

Расчет коэффициентов входящих в формулу расчета величины капиталовложений:

$S_{\text{Т}} = 0{,}008 \cdot C_{\text{ЭВМ}}$; $S_{\text{Т}} = 0{,}008 \cdot 30000 = 240$ (руб.);

$S_{\text{З}} = 0{,}1 \cdot C_{\text{ЭВМ}}$; $S_{\text{З}} = 0{,}1 \cdot 30000 = 3000$ (руб.);

$S_{\text{М}} = 0{,}05 \cdot (C_{\text{ЭВМ}} + S_{\text{З}})$; $S_{\text{М}} = 0{,}05 \cdot (30000+3000) = 1650$ (руб.).

Капиталовложения в электронно-вычислительной машине составляют:

$K_{\text{ЭВМ}}$ = 30000 + 240 + 3000 + 1650 = 34890 (руб.)

Расчет эксплуатационных расходов

Эксплуатационные расходы на электронно-вычислительную машину (ЭВМ) рассчитываются по формуле:

$$E = \left(T_{\text{М.В.}} * e_{\text{ч}}\right) + \frac{Z_{\text{П}}}{T_{\text{С}}}$$

где:

$T_{\text{М.В.}}$ – машинное время для решения задач с помощью разработанной автоматизированной системы онлайн бронирования, (маш.час/год);

$e_ч$ – эксплуатационные расходы, приходящиеся на 1 час работы электронно-вычислительной машины;

$Z_П$ – цена, по которой продается автоматизированная система онлайн бронирования (руб.);

$T_С$ – срок службы программы (г).

Полезный фонд времени работы электронно-вычислительной машины рассчитывается по формуле:

$$T_{ПОЛ} = T_{ОБЩ} * t_{СМ} * N_{СМ} * \left(1 - \frac{\alpha}{100}\right),$$

где:

$T_{ОБЩ}$ – общий фонд времени работы электронно-вычислительной машины (дни);

$T_{ОБЩ} = T_Р$;

$N_{СМ}$ – количество смен работы электронно-вычислительной машины;

$t_{СМ}$ –время одного рабочего дня (час);

α – простои ЭВМ (в % от общего фонда времени работы электронно-вычислительной машины).

Полезный универсальны фонд времени работы электронно-вычислительной машины получим:

Тпол = 365 * 8 * 1 * (1 – 20/100) = 2336 (маш.час/год)

Машинное время для решения задач с помощью разработанной автоматизированной системы онлайн бронирования рассчитывается по формуле:

$$T_{М.В.} = q_I * t_{М.В.I},$$

где:

q_I – количество I-задач, решаемых потребителем в год (шт.);

$t_{М.В.I}$ – время решения I-ой задачи, разработанной автоматизированной системы онлайн бронирования (маш.час).

$T_{М.В.} = 50 \cdot 1 = 50$ (маш.час / год).

Эксплуатационные расходы, приходящиеся на 1 час работы электронно-вычислительной машины, оцениваются по формуле:

$$e_ч = \frac{(A_O + S_{ЗП} + S_{ЭЛ} + R_{РМ})}{T_{ПОЛ}},$$

где:

$A_О$ – амортизационные отчисления (руб.);

$S_{ЗП}$ – затраты по заработной плате программиста в год (руб./год);

$S_{ЭЛ}$ – стоимость потребляемой энергии (руб.);

$R_{РМ}$ – затраты на ремонт электронно-вычислительной машины (руб.);

$T_{ПОЛ}$ – полезный годовой фонд работы ЭВМ, (маш.час/год).

Амортизационные отчисления рассчитываются с учетом нормы амортизации ($a_н$ =12,5 %);

$A_О = a_н \cdot K_{ЭВМ} = 0{,}125 \cdot 34890 = 4361{,}25$ (руб.).

Затраты по заработной плате программиста за год рассчитывается по формуле:

$S_{ЗП} = (1+\omega_с) \cdot (1+\omega_д) \cdot ОЗП \cdot 12,$

где:

$\omega_с$ – коэффициент, учитывающий начисления органам социального страхования на заработную плату разработчика программы, в долях к сумме основной заработной плате разработчика.

$\omega_д$ – коэффициент, учитывающий дополнительную заработную плату разработчика программы, в долях к сумме основной заработной платы;

ОЗП – основная заработная плата программиста за месяц (15000 руб.);

Рассчитываем годовые затраты по заработной плате и социальным отчислениям для инженера:

$S_{ЗП} = (1+ 0{,}356) \cdot (1+ 0{,}15) \cdot 16000 \cdot 12 = 299404{,}8$

(руб./ год)

Стоимость потребляемой энергии оценивается по формуле:

$S_{ЭЛ} = P \cdot T_{ПОЛ} \cdot Z_{ЭЛ}$,

где:

P – мощность, потребляемая ЭВМ (кВт);

$T_{ПОЛ}$ – полезный годовой фонд работы ЭВМ (маш. час/год);

$Z_{ЭЛ}$ – тариф за 1 кВт/час (руб. /кВт).

Итак, произведем расчет стоимости потребляемой энергии:

$S_{ЭЛ} = 0{,}4 \cdot 2336 \cdot 1{,}3 = 1214{,}72$ (руб.).

Затраты на ремонт ЭВМ вычисляются по формуле:

$R_{РМ} = N_{СР} \cdot S_{Д}$,

где:

$N_{СР}$ – среднее количество ремонтов в год;

$S_{Д}$ – стоимость деталей заменяемых при одном ремонте, в среднем.

$R_{РМ} = 2 \cdot 1500 = 3000$ (руб.).

Произведем вычисление эксплуатационных расходов, приходящихся на 1 час работы ЭВМ:

$e_{ч} = (4361{,}25 + 299404{,}8 + 1214{,}72 + 3000) / 2336 = 131{,}84$ (руб./час)

Далее вычислим эксплуатационные расходы на электронно-вычислительной машины:

E = (50 * 131,84) + 144492,4 / 3 = 54756,13 (руб.)

Расчет денежного годового экономического эффекта

Денежный годовой экономический эффект оценивается по следующей формуле:

$\Delta W_{ГЭ} = \Delta E_{М.Э.} + \varepsilon_{н} \cdot \Delta K_{Э}$,

где:

$\Delta E_{М.Э.}$ – экономия стоимости машинного времени (руб.);

$\varepsilon_{н}$ – нормативный коэффициент эффективности капитальных вложений;

$\Delta K_{Э}$ – экономия капитальных вложений (руб.).

Расчет экономии капитальных вложений производится по формуле:

$$\Delta K_{э} = \frac{(T_{м.в.1} - T_{м.в.2}) \cdot K_{эвм}}{T_{пол}},$$

где:

$T_{М.В.2}$ – машинное время для решения задач с помощью разработанной автоматизированной системы онлайн бронирования (маш. час/год);

$K_{ЭВМ}$ – капиталовложения в ЭВМ (руб.);

$T_{ПОЛ}$ – полезный годовой фонд работы ЭВМ (маш. час/год);

$T_{М.В.1}$ – машинное время для решения задач базовой программой рассчитывается с учетом $t'_{М.В.I}$ – время решения I-ой задачи базовой программой:

$T_{М.В.1} = q_I \cdot t'_{М.В.I}$,

$T_{М.В.1} = 50 \cdot 5 = 250$ (маш. час/год).

Произведем расчет экономии капитальных вложений по формуле:

$\Delta K_Э = (250 - 50) * 34890 / 2336 = 2987,16$ (руб.)

Расчет экономии стоимости машинного времени производится по формуле:

$\Delta E_{М.Э.} = e_ч \cdot (T_{М.В.1} - T_{М.В.2})$,

где:

$e_ч$ – эксплуатационные расходы, приходящиеся на 1 час работы ЭВМ;

$T_{М.В.1}$ – машинное время для решения задач базовой программой (маш. час/год);

$T_{М.В.2}$ – машинное время для решения задач с помощью разработанной программы (маш. час/год);

Произведем расчет экономии стоимости машинного времени по формуле:

$\Delta E_{М.Э.} = 131,84 \cdot (250 - 50) = 26368,22$ (руб.)

Теперь определим денежный годовой экономический эффект по формуле:

$\Delta W_{ГЭ} = 26368,22 + 0,25 \cdot 2987,16 = 27115,01$ (руб.)

Определение показателей эффективности инвестиций

Для определения показателей эффективности инвестиций необходимо рассчитать норму дисконта (*E*).

$E = a + b + c$,

где:

a – цена капитала;

b – коэффициент учитывающий риск;

c – уровень инфляции на валютном рынке.

$E = 0{,}14 + 0{,}04 + 0{,}02 = 0{,}20$.

Приведение осуществляется путем умножения затрат и результатов на коэффициент дисконтирования (*КД*), равный:

$$КД = \frac{1}{(1+E)^T},$$

где:

T – период дисконтирования (гг.)

Оценка проекта, сравнение вариантов и выбор оптимального производится с использованием следующих показателей:

– Чистая дисконтированная стоимость (текущая дисконтированная стоимость), т.е. доход;

– Внутренняя норма доходности (рентабельность);

– Индекс доходности;

– Срок окупаемости;

– Денежный поток.

Размер выручки от реализации определяется с учетом прогнозируемой средней потребности в разработанной программе (15 шт.) и розничной цены (144492,2 руб.).

Для того, чтобы рассчитать данные показатели, необходимо составить план денежных потоков как в таблице 3.2.

Таблица 3.2 – План денежных потоков

Показатель	Значения, руб.			
	0-й год	1-й год	2-й год	3-й год
Выручка от реализации	0	2557512,26	2557512,26	2557512,26
НДС (18 %)	0	460352,21	460352,21	460352,21
Выручка от реализации без НДС	0	2097160,05	2097160,05	2097160,05
Издержки на персонал	0	68046,55	68046,55	68046,55
Эксплуатационные расходы	0	54756,13	54756,13	54756,13
Амортизационные отчисления	0	4361,25	4361,25	4361,25
Прибыль от реализации	0	1969996,13	1969996,13	1969996,13
Налог на прибыль (24%)	0	472799,07	472799,07	472799,07
Чистая прибыль	0	1497197,06	1497197,06	1497197,06
Капитальные вложения	34890			
Прочие единовременные затраты	144492,2182			
Денежный поток	-179382,22	1497197,06	1497197,06	1497197,06

Чистая дисконтированная стоимость (*ЧДС*) определяется как сумма потоков реальных денег, приведенная за весь расчетный период к начальному году:

$$ЧДС = \sum_{t=0}^{T}\left(P_t - З_t\right)\cdot КД_t,$$

где:

P_t – результат в t-ом году;

$З_t$ – затраты в t-ом году;

T – период дисконтирования.

Вычисление чистой дисконтированной стоимости и текущей дисконтированной стоимости приведено в таблице 3.3.

Таблица 3.3 – Вычисление чистой дисконтированной стоимости и текущей дисконтированной стоимости

Год	Затраты (-) Результаты (+)	КД при Е = 0,20	ТДС (тыс. руб.)	ТДС нарастающим итогом
0	-179382,22	1	-179382,2182	-179382,22
1	1497197,06	0,8333	1247664,217	1068281,99
2	1497197,06	0,6944	1039720,181	2108002,18
3	1497197,06	0,5787	866433,4841	2974435,664
ЧДС = 2974435,66				

Внутренняя норма доходности (рентабельность) представляет собой ту ставку дисконта ($E_{ВН}$), при которой $ЧДС = 0$. Ее вычисление является итеративным процессом, который начинается с барьерной ставки (E), если при этом $ЧДС$ положительная, то в следующей итерации используют более высокую ставку, если отрицательная – то более низкую.

Точное значение $E_{ВН}$ вычисляется по формуле:

$$E_{ВН} = E_{ЧДС(+)} + \frac{ЧДС(+)}{ЧДС(+) - ЧДС(-)},$$

где:

$E_{ЧДС(+)}$ – значение ставки дисконта, при которой ЧДС принимало последнее положительное значение;

$ЧДС(+)$ – последнее положительное значение $ЧДС$;

$ЧДС(-)$ – последнее отрицательное значение $ЧДС$.

Чистая дисконтированная стоимость широко используется в экономике и финансах как инструмент сравнения потоков платежей, получаемых в разные сроки. Модель дисконтированной стоимости позволяет определить, какой объём финансовых вложений готов сделать инвестор для получения данного денежного потока.

Зависимость чистой дисконтированной стоимости от нормы дисконта представлена в таблице 3.4.

Таблица 3.4 – Зависимость чистой дисконтированной стоимости от нормы дисконта

Номер	Значение нормы дисконта (Е)	Значение ЧДС, тыс. руб.
1	0,2	2 974,44
2	0,4	2 199,55
3	0,6	1 706,74
4	0,8	1 371,21
5	1	1 130,67
6	2	541,49
7	3	311,89
8	4	191,92
9	5	118,67
10	6	69,42
11	7	34,09
12	8	7,51
13	8,2	2,97
14	8,4	-1,36

Данная зависимость отражена на рисунке 3.12.

Рисунок 3.12 – Зависимость ЧДС от нормы дисконта

Расчет внутренней нормы дисконта представлен в таблице 3.5.

Таблица 3.5 – Расчет внутренней нормы дисконта.

Год	Денежные потоки	E = 820 %		E = 820 %	
		КД	ТДС	КД	ТДС
0	-179382,22	1	-179382,2182	1	-179382,2182
1	1497197,06	0,108695652	162738,8109	0,106382979	159276,283
2	1497197,06	0,011814745	17689,00119	0,011317338	16944,28543
3	1497197,06	0,001284211	1922,71752	0,001203972	1802,583556
ЧДС = 2968,31				ЧДС = -1359,07	

Точное значение внутренней нормы доходности (рентабельность) составит:

Евн = 820% + (2968,31% / (2968,31% – (–1359,07%)) = 820,69%

Рассчитанное значение $E_{ВН}$, составляющее 820,69%, превышает фактическую норму дисконта $E_{ВН}$ = 20%, следовательно, инвестиции в данный проект оправданы.

Индекс доходности (*ИД*) рассчитывается по формуле:

$$ИД = \frac{1}{K} \cdot \sum_{t=0}^{T} (P_t - З_t) \cdot КД$$

,

где:

К – приведенная величина инвестиций, рассчитывающаяся по формуле:

$$K = \sum_{t=0}^{T} K_t \cdot КД$$

где:

K_t – величина инвестиций в *t*-ом году.

Индекс доходности (*ИД*) проекта составляет:

ИД = (1247664 + 1039720,2 + 866433,5) / -179382 = 17,58

Рассчитанное значение *ИД* = 17,58 больше единицы, следовательно, разработку системы можно считать эффективной и экономически обоснованной.

Средняя рентабельность разработки рассчитывается по формуле:

$$РП = \frac{ИД}{Т} \cdot 100\ \%,$$

где:

ИД – индекс доходности проекта;

Т – срок службы программы.

Средняя рентабельность разработки в нашем случае составит:

РП = 17,58 / 3 * 100% = 586,05%

Срок окупаемости инвестиционного проекта ($Т_{ОК}$) – это период времени, который потребуется для возмещения инвестиций. $Т_{ОК}$ определяют с учетом дисконтирования, путем суммирования ежегодных поступлений до определенного периода, в котором они превзойдут первоначальные расходы денежных средств.

Определим $Т_{ОК}$ графическим методом, график, изображенный на рисунке 3.13, строится по данным таблицы 3.5.

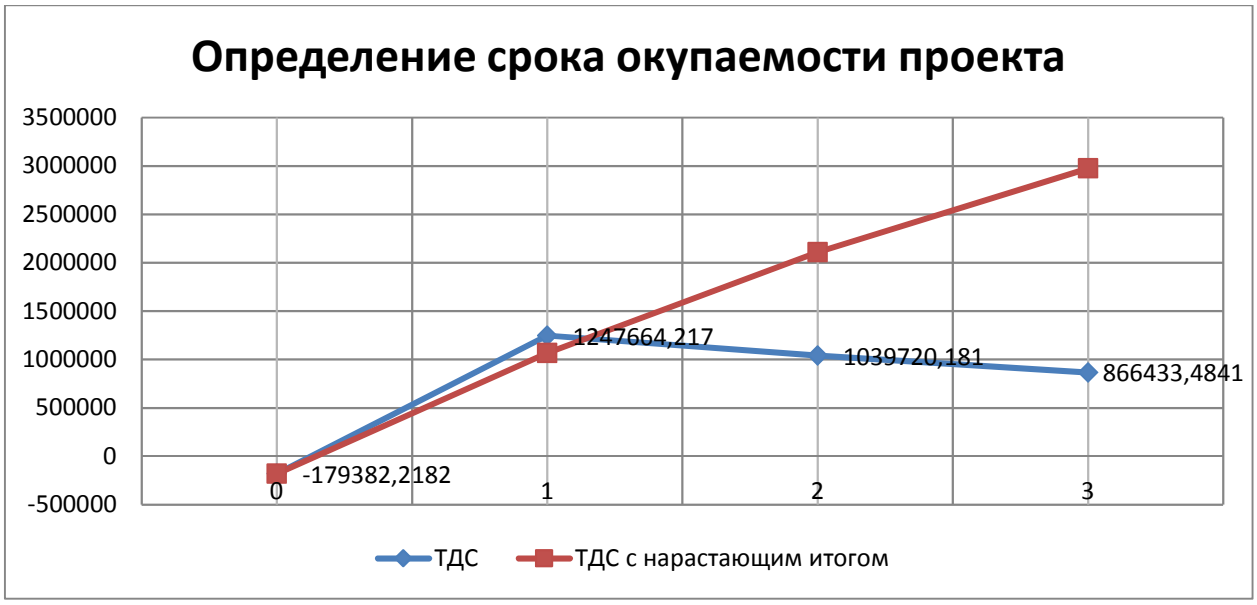

Рисунок 3.13 – Определение срока окупаемости проекта

Как видно из графика, значение $Т_{ОК}$ составляет меньше 1 года.

Выводы по третьему разделу.

Обобщенные технико-экономические показатели разработки программы сведены в таблицу 3.6.

Таблица 3.6 – Обобщенные технико-экономические показатели разработки программы

Показатель	Значение
Капитальные вложения (руб.)	34890
Эксплуатационные расходы (руб.)	54756,13
Оптовая цена (руб.)	144492,2182
Свободная отпускная цена (руб.)	170500,8175
Затраты на проектирование (руб.)	120410,1818
Чистая дисконтированная стоимость (при E = 20%) (руб.)	2974435,664
Внутренняя норма доходности (%)	820,6859377
Индекс доходности	17,58155248
Средняя рентабельность разработки (%)	586,0517492
Срок окупаемости (год)	1

По полученным результатам проведенных вычислений величина *ТДС* > 0, значение индекса доходности *ИД* > 1, а рассчитанная $E_{ВН}$ = 586,05% превышает фактическую норму дисконта $E_{ВН}$ = 20 %. Это позволяет сделать вывод о том, что вложение инвестиций в разработку данного проекта является экономически целесообразным.

Спроектирована информационная модель в текущей технологии работы. Дано описание бизнес-процессов рассматриваемой информационной модели.

Проведен реинжиниринг и показаны преимущества новой информационной модели в нотации IDEF3.

Далее в работе приведено описание первичных документов с нормативно-справочной информацией.

Спроектирована база данных с описанием таблиц, схем данных, необходимые запросы. Описана даталогическая, инфологическая и физическая модели баз данных.

Описано назначение системы онлайн бронирования, ее основные и служебные функции.

Представлена структура, задачи и назначения каждого программного модуля. Дано описание и методология архитектуры MVC. Описаны организация сбора, передачи, обработки и выдачи информации о бронировании.

ЗАКЛЮЧЕНИЕ

В результате проведения научно-технических исследований в компании «Кома» были изучены бизнес-процессы по продаже заявок на игры, а также онлайн процессы по оформлению заявки на игру в городе Белгороде. Определены задачи администратора-оператора, требования к первичным документам, проанализированы сильные и слабые стороны текущего технологического процесса.

В итоге проведения выпускной квалификационной работы была выполнена основная цель – автоматизирован процесс онлайн-бронирования организации «Кома».

Эта цель была достигнута за счет решения следующих задач:

– изучена финансово-предпринимательская деятельность организации «Кома»;

– обоснована необходимость технологических, технических, программных и информационных проектных решений;

– обоснован выбор программных средств по разработке и проектированию программного средства;

– проведена декомпозиция бизнес-процессов сайта организации и выявлены недостатки в текущей технологии работы;

– проведен реинжиниринг получения данных и предоставлена новая инфокоммуникационная модель;

– описано назначение и выполняемые задачи разрабатываемого программного обеспечения;

– проанализированы текущие разработки, описаны проектируемые технологии и разработка программного средства;

– спроектирована и разработана система онлайн-бронирования;

– внедрена и успешно запущена в работу система онлайн-бронирования.

Разработанная автоматизированное система онлайн-бронирования является программным продуктом готовым к внедрению в деятельность «Кома», выполнив при этом условия по техническому обеспечению. Для внедрения необходимо: приобрести стационарный компьютер в качестве web-инструмента, проинсталлировать дистрибутивы программ необходимых для работы web-сервера, приобрести электронный планшет или смартфонн в качестве рабочего места, распаковать рабочие файлы на web-сервер системы онлайн-бронирования администратора-оператора.

В дальнейшем систему онлайн-бронирования можно улучшить и модернезировать путём реализации следующих действий:

– Добавление функции автоматического e-mail и sms информирования клиентов об изменение информации по заявке или статуса заявки;

– Создание скриншота визуальной системы отображения и просмотра информации по заявкам и доступного времени бронирования;

– Разработка дополнительного модуля по работе с разработчиками сценариев.

Таким образом, автоматизированная система онлайн-бронирования – это инструмент рационализации и интенсификации управленческой деятельности организации или предприятия.

Профессиональные автоматизированные системы онлайн учета и ведения базы данных web-сайта – это главный инструмент для общения и взаимодействия человека с информационными системами, которые выполняют роль автономных рабочих мест, интеллектуальных терминалов больших ЭВМ, рабочих станций в локальных сетях. Автоматизированные системы онлайн учета и ведения базы данных web-сайта имеют открытую архитектуру и легко адаптируются под необходимые условия.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Абдикеев, Н.М. Реинжиниринг бизнес-процессов / Н.М. Абдикеев, Т.П. Данько, С.В. Ильдеменов, А.Д. Киселев. – Москва: Эксмо, 2013. – 562 с.

2. Бегг, К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / К. Бегг, Т. Коннолли. – М. : Вильямс, 2012. – 435 с.

3. Боуэн, Р. Apache. Настольная книга администратора / Р. Боуэн. – М. : ДиаСофтЮП, 2016. – 384 с.

4. Брябин, В. Программное обеспечение персональных ЭВМ / В. Брябин. – М. : Эксмо, 2014. – 262 с.

5. Бьюли, А. Изучаем SQL / А. Бьюли. – М. : Символ-Плюс, 2015. – 342 с.

6. Вейтман, В. Программирование для Web / В. Вейтман. – Москва: Диалектика, 2016 – 378 с.

7. Гагарина, Л. Разработка и эксплуатация автоматизированных информационных систем / Л. Гагарина, Д. Киселев. – М. : Инфра-М, 2015. – 374 с.

8. Дубейковский, В.И. Эффективное моделирование с CA ERwin Process Modeler и AllFusion Process Modeler / В.И. Дубейковский. – Москва: Диалог-МИФИ, 2014. – 385 с.

9. Дунаев, В. Базы данных. Язык SQL для студента / В. Дунаев. – СПб. : БХВ-Петербург, 2012. – 309 с.

10. Дюбуа, П. MySQL / П. Дюбуа. – М. : Вильямс, 2013. – 450 с.

11. Енин, А. Локальная СУБД своими руками. Учимся на примерах / А. Енин, Н. Енин. – М. : Салон-Пресс, 2015. – 219 с.

12. Калянов, Г. Case-технологии: Консалтинг в автоматизации бизнес-процессов / Г. Калянов. – М. : Телеком, 2014. – 270 с.

13. Калянов, Г. Моделирование, анализ, реорганизация и автоматизация бизнес-процессов / Г. Калянов. – М. : Финансы и статистика, 2015. – 249 с.

14. Колисниченко, Д.Н. PHP 5 и MySQL 6. Разработка Web-приложений / Д.Н. Колисниченко. – СПб: БХВ-Петербург, 2015. – 568 с.

15. Кроудер, Д. Создание веб-сайта для чайников / Д. Кроудер. – М. : Вильямс, 2015. – 346 с.

16. Куприенкова, И. Все о цветах и букетах / И. Куприенкова. – М. : Рипол Классик, 2015. – 152 с.

17. Маклаков, С.В. BPwin и ERwin. CASE – средства разработки информационных систем / С.В. Маклаков. – Москва: Диалог-МИФИ, 2015. – 216 с.

18. Маклаков, С.В. Моделирование бизнес-процессов с AllFusion Process Modeler / С.В. Маклаков. – Москва: Диалог-МИФИ, 2015. – 254 с.

19. Мержевич, В. HTML и CSS на примерах / В. Мержевич. – СПб. : БХВ-Петербург, 2016. – 488 с.

20. Мерсер, К. PHP 5 для начинающих / К. Мерсер. – Москва: Вильямс, 2016. – 746 с.

21. Мотев, А. Уроки MySQL / А. Мотев. – СПб. : БХВ-Петербург, 2015. – 201 с.

22. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript / Р. Никсон. – Спб: Питер, 2014. – 486 с.

23. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 / Р. Никсон. – СПб. : Питер, 2015. – 668 с.

24. Оболенски, Н. Практический реинжиниринг бизнеса / Н. Оболенски. – Москва: ЛОРИ, 2015. – 364 с.

25. Овчинников, В.В. Методология проектирования автоматизированных информационных систем. Основы системного подхода / В.В. Овчинников. – Москва: Компания Спутник, 2016. – 225 с.

26. Паттерсон, Д. Архитектура компьютера и проектирование компьютерных систем / Д. Паттерсон, Дж. Хеннесси. – СПб. : Питер, 2014. – 764 с.

27. Робсон, М., Уллах Ф. Практическое руководство по реинжинирингу бизнес процессов / М. Робсон, Ф. Уллах. – Москва: Аудит, ЮНИТИ, 2013. – 214 с.

28. Ручкин, В. Архитектура компьютерных сетей / В. Ручкин. – М . : Диалог-МИФИ, 2013. – 270 с.

29. Скиена, С. Алгоритмы. Руководство по разработке / С. Скиена. – СПб. : БХВ-Петербург, 2010. – 722 с.

30. Смирнов, А. Архитектура вычислительных систем / А. Смирнов. – М. : Эксмо, 2013. – 325 с.

31. Сойер, Д. JavaScript и jQuery. Исчерпывающее руководство / Д. Сойер. – М. : Эксмо, 2015. – 891 с.

32. Тельнов, Ю.Ф. Реинжиниринг бизнес-процессов : учебное пособие / Ю.Ф. Тельнов. – Москва: Московский международный институт эконометрики, информатики, финансов и права, 2014. – 98 с.

33. Федотова, Д.Э. CASE-технологии : практикум / Д.Э. Федотова, Ю.Д. Семенов, К.Н. Чижик. – Москва: Горячая Линия, 2016. – 161 с.

34. Филиппов, В. Многомерные СУБД при создании корпоративных информационных систем / В. Филиппов. – М. : Инфра-М, 2015. – 81 с.

35. Флэнаган, Д. JavaScript. Карманный справочник / Д. Флэнаган. – М. : Вильямс, 2016. – 321 с.

36. Харрис, Э. PHP/MySQL для начинающих / Э. Харрис. – Москва: Кудиц-образ, 2016. – 354 с.

37. Хокинс, С. Администрирование Web-сервера Apache и руководство по электронной коммерции / С. Хокинс. – Москва: Диалектика, 2011. – 326 с.

38. Хокинс, С. Администрирование Web-сервера Apache и руководство по электронной коммерции / С. Хокинс. – М. : Вильямс, 2015. – 326 с.

ПРИЛОЖЕНИЕ А

Даталогическая модель базы данных представлена в таблице А.1

Таблица А.1 – Даталогическая модель базы данных

Сущность	Идентификатор таблицы	Атрибут	Идентификатор поля	Тип поля	PK
Уровень доступа	Access	Номер	Access_id	Integer(11)	+
		Название	Name	Varchar(255)	
Клиенты	Clients	Номер	Clients_id	Integer(11)	+
		ФИО	Fio	Varchar(255)	
		E-mail	Email	Varchar (255)	
		Телефон	Tel	Varchar (255)	
		Размер скидки	Discount	Integer (11)	
Сотрудники	Employees	Номер	Employees_id	Integer(11)	+
		ФИО	Fio	Varchar(255)	
		E-mail	Email	Varchar(255)	
		Телефон	Tel	Varchar(255)	
		Пароль	Password	Varchar(255)	
		Фотография	Photo	Varchar(255)	
		Номер доступа	Access_id	Enum('1', '2', '3')	
Сценарии	Scenarious	Номер	Scenarious_id	Integer(11)	+
		Название	Name	Varchar(255)	
		Описание	Desc	Text	
		Изображение	Photo	Varchar(255)	
		Цена в закупке	Price1	Integer(11)	
		Цена в реализации	Price2	Integer(11)	
		Текущий остаток	Rest	Integer(11)	
		Необходимый остаток	Rest_need	Integer(11)	
Заявки	Orders	Номер	Orders_id	Integer(11)	+
		Дата ввода	Date_begin	Date	
		Дата закрытия	Date_close	Date	
		Номер клиента	Clients_id	Integer(11)	
		Адрес доставки	Delivery_address	Varchar(255)	

Продолжение таблицы А.1

Сущность	Идентификатор таблицы	Атрибут	Идентификатор поля	Тип поля	РК
		Сумма доставки	Delivery_sum	Integer(11)	
		Дата доставки	Delivery_date	Date	
		Размер скидки	Order_discount	Integer(11)	
		Сумма заказа	Order_sum	Integer(11)	
		Записка	Note	Text	
		Номер статуса заказа	Statuses_id	Integer(11)	
		Номер сотрудника	Employees_id	Integer(11)	
Заявки_Сценарии	Orders_scenarios	Номер	Orders_flowers_id	Integer(11)	+
		Номер заказа	Orders_id	Integer(11)	
		Номер сценария	Scenario_id	Integer(11)	
		Количество	Quantity	Integer(11)	
История заявки	Orders_history	Номер	Orders_history_id	Integer(11)	+
		Номер заявки	Orders_id	Integer(11)	
		Номер сотрудника	Employees_id	Integer(11)	
		Дата добавления	nowDateTime	Timestamp	
		Комментарий	Comment	Varchar(255)	
Статусы заявки	Statuses_orders	Номер	Statuses_id	Integer(11)	+
		Название	Name	Varchar(255)	
Задачи	Tasks	Номер	Tasks_id	Integer(11)	+
		Название	Name	Varchar(255)	
		Номер сотрудника	Employees_id	Integer(11)	
		Дата закрытия	Date_die	Date	
		Описание	Description	Text	
		Завершение задачи	Complete	enum('В процессе', 'Выполнено')	
		Отчет	Report	Text	
		Дата отчета	Report_date	Date	

Программные коды php-страниц

Index.php

```
<?php
session_start();

class SafeMySQL
{
	private $conn;
	private $stats;
	private $emode;
	private $exname;
	private $defaults = array(
		'host'	=> 'localhost',
		'user'	=> 'ipzaj_koma',
		'pass'	=> 'uHMd6Ge67tzr6Wdp',
		'db'	=> 'ipzaj_koma',
		'port'	=> NULL,
		'socket'	=> NULL,
		'pconnect'	=> FALSE,
		'charset'	=> 'utf8',
		'errmode'	=> 'error', //or exception
		'exception' => 'Exception', //Exception class name
	);
	const RESULT_ASSOC = MYSQLI_ASSOC;
	const RESULT_NUM   = MYSQLI_NUM;
	function __construct($opt = array())
	{
		$opt = array_merge($this->defaults,$opt);
		$this->emode  = $opt['errmode'];
		$this->exname = $opt['exception'];
		if ($opt['pconnect'])
		{
			$opt['host'] = "p:".$opt['host'];
		}
		@$this->conn = mysqli_connect($opt['host'], $opt['user'], $opt['pass'], $opt['db'], $opt['port'], $opt['socket']);
		if ( !$this->conn )
		{
			$this->error(mysqli_connect_errno()." ".mysqli_connect_error());
		}
		mysqli_set_charset($this->conn, $opt['charset']) or $this->error(mysqli_error($this->conn));
		unset($opt); // I am paranoid
	}
	/**
	 * Обычная функция выполнить запрос с заполнителями.Mysqli_query оболочка с поддержкой заполнителей
*
* Примеры:
	 * $db->query("DELETE FROM table WHERE id=?i", $id);
	 *
	 * @param string $query - an SQL query with placeholders
	 * @param mixed  $arg,... unlimited number of arguments to match placeholders in the query
	 * @return resource|FALSE whatever mysqli_query returns
	 */
	public function query()
	{
		return $this->rawQuery($this->prepareQuery(func_get_args()));
	}
	/**
	 * Обычные функции для извлечения одной строки.
	 *
	 * @param resource $result - myqli result
	 * @param int $mode - optional fetch mode, RESULT_ASSOC|RESULT_NUM, default RESULT_ASSOC
	 * @return array|FALSE whatever mysqli_fetch_array returns
	 */
	public function fetch($result,$mode=self::RESULT_ASSOC)
	{
		return mysqli_fetch_array($result, $mode);
	}
	/**
	 * Обычные функции получить число пострадавших строк.
	 *
```

```
     * @return int whatever mysqli_affected_rows returns
     */
    public function affectedRows()
    {
        return mysqli_affected_rows ($this->conn);
    }
    /**
     * Обычные функцию, чтобы получить последнюю вставки ID.
     *
     * @return int whatever mysqli_insert_id returns
     */
    public function insertId()
    {
        return mysqli_insert_id($this->conn);
    }
    /**
     * Обычные функцию, чтобы получить количество строк в наборе результатов.
     *
     * @param resource $result - myqli result
     * @return int whatever mysqli_num_rows returns
     */
    public function numRows($result)
    {
        return mysqli_num_rows($result);
    }
    /**
     * Обычные функции, чтобы освободить результирующего.
     */
    public function free($result)
    {
        mysqli_free_result($result);
    }
    /**
     * Помощник функцию, чтобы получить скалярное значение прямо из запросов и необязательных аргументов
*
* Примеры:
     * $name = $db->getOne("SELECT name FROM table WHERE id=1");
     * $name = $db->getOne("SELECT name FROM table WHERE id=?i", $id);
     *
     * @param string $query - an SQL query with placeholders
     * @param mixed  $arg,... unlimited number of arguments to match placeholders in the query
     * @return string|FALSE either first column of the first row of resultset or FALSE if none found
     */
    public function getOne()
    {
        $query = $this->prepareQuery(func_get_args());
        if ($res = $this->rawQuery($query))
        {
            $row = $this->fetch($res);
            if (is_array($row)) {
                return reset($row);
            }
            $this->free($res);
        }
        return FALSE;
    }
    /**
     * Helper function to get single row right out of query and optional arguments
     *
     * Examples:
     * $data = $db->getRow("SELECT * FROM table WHERE id=1");
     * $data = $db->getOne("SELECT * FROM table WHERE id=?i", $id);
     *
     * @param string $query - an SQL query with placeholders
     * @param mixed  $arg,... unlimited number of arguments to match placeholders in the query
     * @return array|FALSE either associative array contains first row of resultset or FALSE if none found
     */
    public function getRow()
    {
        $query = $this->prepareQuery(func_get_args());
        if ($res = $this->rawQuery($query)) {
            $ret = $this->fetch($res);
            $this->free($res);
            return $ret;
        }
        return FALSE;
    }
    /**
```

```
 * Helper function to get single column right out of query and optional arguments
 *
 * Examples:
 * $ids = $db->getCol("SELECT id FROM table WHERE cat=1");
 * $ids = $db->getCol("SELECT id FROM tags WHERE tagname = ?s", $tag);
 *
 * @param string $query - an SQL query with placeholders
 * @param mixed  $arg,... unlimited number of arguments to match placeholders in the query
 * @return array|FALSE either enumerated array of first fields of all rows of resultset or FALSE if none found
 */
public function getCol()
{
	$ret   = array();
	$query = $this->prepareQuery(func_get_args());
	if ( $res = $this->rawQuery($query) )
	{
		while($row = $this->fetch($res))
		{
			$ret[] = reset($row);
		}
		$this->free($res);
	}
	return $ret;
}
/**
 * Helper function to get all the rows of resultset right out of query and optional arguments
 *
 * Examples:
 * $data = $db->getAll("SELECT * FROM table");
 * $data = $db->getAll("SELECT * FROM table LIMIT ?i,?i", $start, $rows);
 *
 * @param string $query - an SQL query with placeholders
 * @param mixed  $arg,... unlimited number of arguments to match placeholders in the query
 * @return array enumerated 2d array contains the resultset. Empty if no rows found.
 */
public function getAll()
{
	$ret   = array();
	$query = $this->prepareQuery(func_get_args());
	if ( $res = $this->rawQuery($query) )
	{
		while($row = $this->fetch($res))
		{
			$ret[] = $row;
		}
		$this->free($res);
	}
	return $ret;
}
/**
 * Helper function to get all the rows of resultset into indexed array right out of query and optional arguments
 *
 * Examples:
 * $data = $db->getInd("id", "SELECT * FROM table");
 * $data = $db->getInd("id", "SELECT * FROM table LIMIT ?i,?i", $start, $rows);
 *
 * @param string $index - name of the field which value is used to index resulting array
 * @param string $query - an SQL query with placeholders
 * @param mixed  $arg,... unlimited number of arguments to match placeholders in the query
 * @return array - associative 2d array contains the resultset. Empty if no rows found.
 */
public function getInd()
{
	$args  = func_get_args();
	$index = array_shift($args);
	$query = $this->prepareQuery($args);
	$ret = array();
	if ( $res = $this->rawQuery($query) )
	{
		while($row = $this->fetch($res))
		{
			$ret[$row[$index]] = $row;
		}
		$this->free($res);
	}
	return $ret;
}
/**
```

```
        * Helper function to get a dictionary-style array right out of query and optional arguments
        *
        * Examples:
        * $data = $db->getIndCol("name", "SELECT name, id FROM cities");
        *
        * @param string $index - name of the field which value is used to index resulting array
        * @param string $query - an SQL query with placeholders
        * @param mixed  $arg,... unlimited number of arguments to match placeholders in the query
        * @return array - associative array contains key=value pairs out of resultset. Empty if no rows found.
        */
       public function getIndCol()
       {
               $args  = func_get_args();
               $index = array_shift($args);
               $query = $this->prepareQuery($args);
               $ret = array();
               if ( $res = $this->rawQuery($query) )
               {
                       while($row = $this->fetch($res))
                       {
                               $key = $row[$index];
                               unset($row[$index]);
                               $ret[$key] = reset($row);
                       }
                       $this->free($res);
               }
               return $ret;
       }
       /**
        * Функция для разбора заполнители либо в полном запроса или части запроса
* В отличие от отечественных подготовленных заявлений, позволяет любой части запроса, который будет обработан
*
* Полезно для отладки
* И чрезвычайно полезна для условного построения запросов
* Как добавление различных частей запроса, используя циклы, условия и т.д.
* Уже разобранные детали должны быть добавлены с помощью его стороны? Заполнителя
*
* Примеры:
        * $query = $db->parse("SELECT * FROM table WHERE foo=?s AND bar=?s", $foo, $bar);
        * echo $query;
        *
        * if ($foo) {
        *     $qpart = $db->parse(" AND foo=?s", $foo);
        * }
        * $data = $db->getAll("SELECT * FROM table WHERE bar=?s ?p", $bar, $qpart);
        *
        * @param string $query - whatever expression contains placeholders
        * @param mixed  $arg,... unlimited number of arguments to match placeholders in the expression
        * @return string - initial expression with placeholders substituted with data.
        */
       public function parse()
       {
               return $this->prepareQuery(func_get_args());
       }
       /**
        * function to implement whitelisting feature
        * sometimes we can't allow a non-validated user-supplied data to the query even through placeholder
        * especially if it comes down to SQL OPERATORS
        *
        * Example:
        *
        * $order = $db->whiteList($_GET['order'], array('name','price'));
        * $dir   = $db->whiteList($_GET['dir'],   array('ASC','DESC'));
        * if (!$order || !dir) {
        *     throw new http404(); //non-expected values should cause 404 or similar response
        * }
        * $sql  = "SELECT * FROM table ORDER BY ?p ?p LIMIT ?i,?i"
        * $data = $db->getArr($sql, $order, $dir, $start, $per_page);
        *
        * @param string $iinput   - field name to test
        * @param  array  $allowed - an array with allowed variants
        * @param  string $default - optional variable to set if no match found. Default to false.
        * @return string|FALSE    - either sanitized value or FALSE
        */
       public function whiteList($input,$allowed,$default=FALSE)
       {
               $found = array_search($input,$allowed);
               return ($found === FALSE) ? $default : $allowed[$found];
```

```
}
/**
 * function to filter out arrays, for the whitelisting purposes
 * useful to pass entire superglobal to the INSERT or UPDATE query
 * OUGHT to be used for this purpose,
 * as there could be fields to which user should have no access to.
 *
 * Example:
 * $allowed = array('title','url','body','rating','term','type');
 * $data    = $db->filterArray($_POST,$allowed);
 * $sql     = "INSERT INTO ?n SET ?u";
 * $db->query($sql,$table,$data);
 *
 * @param  array $input   - source array
 * @param  array $allowed - an array with allowed field names
 * @return array filtered out source array
 */
public function filterArray($input,$allowed)
{
	foreach(array_keys($input) as $key )
	{
		if ( !in_array($key,$allowed) )
		{
			unset($input[$key]);
		}
	}
	return $input;
}
/**
 * Function to get last executed query.
 *
 * @return string|NULL either last executed query or NULL if were none
 */
public function lastQuery()
{
	$last = end($this->stats);
	return $last['query'];
}
/**
 * Function to get all query statistics.
 *
 * @return array contains all executed queries with timings and errors
 */
public function getStats()
{
	return $this->stats;
}
/**
 * private function which actually runs a query against Mysql server.
 * also logs some stats like profiling info and error message
 *
 * @param string $query - a regular SQL query
 * @return mysqli result resource or FALSE on error
 */
private function rawQuery($query)
{
	$start = microtime(TRUE);
	$res   = mysqli_query($this->conn, $query);
	$timer = microtime(TRUE) - $start;
	$this->stats[] = array(
		'query' => $query,
		'start' => $start,
		'timer' => $timer,
	);
	if (!$res)
	{
		$error = mysqli_error($this->conn);

		end($this->stats);
		$key = key($this->stats);
		$this->stats[$key]['error'] = $error;
		$this->cutStats();

		$this->error("$error. Full query: [$query]");
	}
	$this->cutStats();
	return $res;
}
```

```
	private function prepareQuery($args)
	{
		$query = '';
		$raw   = array_shift($args);
		$array = preg_split('~(\?[nsiuap])~u',$raw,null,PREG_SPLIT_DELIM_CAPTURE);
		$anum  = count($args);
		$pnum  = floor(count($array) / 2);
		if ( $pnum != $anum )
		{
			$this->error("Количество аргументов ($anum) не соответствует числу placeholders ($pnum) в
[$raw]");
		}
		foreach ($array as $i => $part)
		{
			if ( ($i % 2) == 0 )
			{
				$query .= $part;
				continue;
			}
			$value = array_shift($args);
			switch ($part)
			{
				case '?n':
					$part = $this->escapeIdent($value);
					break;
				case '?s':
					$part = $this->escapeString($value);
					break;
				case '?i':
					$part = $this->escapeInt($value);
					break;
				case '?a':
					$part = $this->createIN($value);
					break;
				case '?u':
					$part = $this->createSET($value);
					break;
				case '?p':
					$part = $value;
					break;
			}
			$query .= $part;
		}
		return $query;
	}
	private function escapeInt($value)
	{
		if ($value === NULL)
		{
			return 'NULL';
		}
		if(!is_numeric($value))
		{
			$this->error("Integer (?i) placeholder ожидает числовое значение, ".gettype($value)." given");
			return FALSE;
		}
		if (is_float($value))
		{
			$value = number_format($value, 0, '.', ''); // may lose precision on big numbers
		}
		return $value;
	}
	private function escapeString($value)
	{
		if ($value === NULL)
		{
			return 'NULL';
		}
		return	"'".mysqli_real_escape_string($this->conn,$value)."'";
	}
	private function escapeIdent($value)
	{
		if ($value)
		{
			return "`".str_replace("`","``",$value)."`";
		} else {
			$this->error("Пустое значение для идентификатора (?n) placeholder");
		}
```

```
}
private function createIN($data)
{
	if (!is_array($data))
	{
		$this->error("Значение для IN (?a) placeholder должен быть массивом");
		return;
	}
	if (!$data)
	{
		return 'NULL';
	}
	$query = $comma = '';
	foreach ($data as $value)
	{
		$query .= $comma.$this->escapeString($value);
		$comma  = ",";
	}
	return $query;
}
private function createSET($data)
{
	if (!is_array($data))
	{
		$this->error("SET (?u) placeholder ожидается array, ".gettype($data)." given");
		return;
	}
	if (!$data)
	{
		$this->error("Пустой массив для SET (?u) placeholder");
		return;
	}
	$query = $comma = '';
	foreach ($data as $key => $value)
	{
		$query .= $comma.$this->escapeIdent($key).'='.$this->escapeString($value);
		$comma  = ",";
	}
	return $query;
}
private function error($err)
{
	$err  = __CLASS__.": ".$err;
	if ( $this->emode == 'error' )
	{
		$err .= ". Ошибка начата в ".$this->caller().", бросил";
		trigger_error($err,E_USER_ERROR);
	} else {
		throw new $this->exname($err);
	}
}
private function caller()
{
	$trace  = debug_backtrace();
	$caller = '';
	foreach ($trace as $t)
	{
		if ( isset($t['class']) && $t['class'] == __CLASS__ )
		{
			$caller = $t['file']." on line ".$t['line'];
		} else {
			break;
		}
	}
	return $caller;
}
/**
 * On a long run we can eat up too much memory with mere statsistics
 * Let's keep it at reasonable size, leaving only last 100 entries.
 */
private function cutStats()
{
	if ( count($this->stats) > 100 )
	{
		reset($this->stats);
		$first = key($this->stats);
		unset($this->stats[$first]);
	}
```

```
                }
        }

        ## Получиение списка комнат
        function getRooms($date,$db){
                $all=$db->getRow("SELECT * FROM bron_time WHERE date=?s",$date);
                if(!$all){$db->query("INSERT INTO
bron_time(id_room,date,seans_1,seans_2,seans_3,seans_4,seans_5,seans_6,seans_7,seans_8,seans_9)
VALUES('1','".$date."','0','0','0','0','0','0','0','0','0'),('2','".$date."','0','0','0','0','0','0','0','0','0'),('3','".$date."','0','0','0','0','0','0','0','0','0')");}
                $timtm=0;
                if(date("w",time()+$timtm)==0){$W='<span style="color:#AB4490;">вс</span>';}
                if(date("w",time()+$timtm)==1){$W='пн';}
                if(date("w",time()+$timtm)==2){$W='вт';}
                if(date("w",time()+$timtm)==3){$W='ср';}
                if(date("w",time()+$timtm)==4){$W='чт';}
                if(date("w",time()+$timtm)==5){$W='пт';}
                if(date("w",time()+$timtm)==6){$W='<span style="color:#AB4490;">сб</span>';}
                if(date("d.m.Y")==$date){$dop_cl='fiol';}else{$dop_cl='';}
                $result='<tr><td style="width: 35%;height:75px;"><div class="fiol" style="margin-top: -10px;">'.date("d.m.Y").'<br><span
style="font-size:12px;">Сегодня</span></td>
                <td><span onClick="days(\''.date("d.m.Y").'\');" class="days_b '.$dop_cl.'">'.date("d").'</span><div
class="top_w">'.$W.'</div></td>';
        $timtm=86400;
                $i=1;
                for($i;$i<9;$i++){
                if(date("d.m.Y",time()+$timtm)==$date){$dop_cl='fiol';}else{$dop_cl='';}
                $result.='<td><span onClick="days(\''.date("d.m.Y",time()+$timtm).'\');" class="days_b
'.$dop_cl.'">'.date("d",time()+$timtm).'</span>';
                if(date("w",time()+$timtm)==0){$W='<span style="color:#AB4490;">вс</span>';}
                if(date("w",time()+$timtm)==1){$W='пн';}
                if(date("w",time()+$timtm)==2){$W='вт';}
                if(date("w",time()+$timtm)==3){$W='ср';}
                if(date("w",time()+$timtm)==4){$W='чт';}
                if(date("w",time()+$timtm)==5){$W='пт';}
                if(date("w",time()+$timtm)==6){$W='<span style="color:#AB4490;">сб</span>';}
                $result.='<div class="top_w">'.$W.'</div></td>';
                $timtm=$timtm+86400;
                }
                $result.='</tr>';
                $rr=1;
                $rooms=$db->getAll("SELECT * FROM bron");
                foreach($rooms as $keys=>$vals){
                        if($rr==3){$rr++;}else{

        $room=$db->getRow("SELECT * FROM bron_time WHERE date=?s AND id_room=?i",$date,$vals['id']);
        $color4ik='rgba(81, 27, 66, 0.85);border:none;cursor:default';
        if($room['seans_1']==1){$seans_1='background:'.$color4ik.';';}
        if($room['seans_2']==1){$seans_2='background:'.$color4ik.';';}
        if($room['seans_3']==1){$seans_3='background:'.$color4ik.';';}
        if($room['seans_4']==1){$seans_4='background:'.$color4ik.';';}
        if($room['seans_5']==1){$seans_5='background:'.$color4ik.';';}
        if($room['seans_6']==1){$seans_6='background:'.$color4ik.';';}
        if($room['seans_7']==1){$seans_7='background:'.$color4ik.';';}
        if($room['seans_8']==1){$seans_8='background:'.$color4ik.';';}
        if($room['seans_9']==1){$seans_9='background:'.$color4ik.';';}
        if(date("d.m.Y")==$date){
                $yy=time()+1800;
        if($vals['seans_1']<=date("H:i",$yy)){$seans_1='background:'.$color4ik.';';}
        if($vals['seans_2']<=date("H:i",$yy)){$seans_2='background:'.$color4ik.';';}
        if($vals['seans_3']<=date("H:i",$yy)){$seans_3='background:'.$color4ik.';';}
        if($vals['seans_4']<=date("H:i",$yy)){$seans_4='background:'.$color4ik.';';}
        if($vals['seans_5']<=date("H:i",$yy)){$seans_5='background:'.$color4ik.';';}
        if($vals['seans_6']<=date("H:i",$yy)){$seans_6='background:'.$color4ik.';';}
        if($vals['seans_7']<=date("H:i",$yy)){$seans_7='background:'.$color4ik.';';}
        if($vals['seans_8']<=date("H:i",$yy)){$seans_8='background:'.$color4ik.';';}
        if($vals['seans_9']<=date("H:i",$yy)){$seans_9='background:'.$color4ik.';';}
        }
        if($rr==1){$da='приют';}
        if($rr==2){$da='во-все-тяжкие';}
        if($rr==3){$da='релакс-бар';}$rr++;
        $day_display_none = strftime("%a", strtotime($date));
        if($da=='приют' AND ($day_display_none=='Mon' or $day_display_none=='Tue')){

        $result.='<tr><td style="text-align:left;"><a href="/'.$da.'/" style="text-decoration:none;color:#fff;"><div class="fiol"
style="cursor:pointer;">'.$vals['name_room'].'<br><span style="font-size:13px;">2-4 человека, до 60 мин</span></div></a></td>
        <td onClick="bron_okno(\''.$date.'\',\'seans_1\',\''.$vals['id'].'\',\''.$vals['name_room'].'\',\''.$vals['seans_1'].'\');"><div class="bor_bron"
style="'.$seans_1.'" >'.$vals['seans_1'].'</div></td>
        <td onClick="bron_okno(\''.$date.'\',\'seans_2\',\''.$vals['id'].'\',\''.$vals['name_room'].'\',\''.$vals['seans_2'].'\');"><div class="bor_bron"
```

```
style="'.$seans_2.'">'.$vals['seans_2'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_3\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_3'].'\');"><div class="bor_bron"
style="'.$seans_3.'">'.$vals['seans_3'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_4\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_4'].'\');"><div class="bor_bron"
style="'.$seans_4.'">'.$vals['seans_4'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_5\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_5'].'\');"><div class="bor_bron"
style="'.$seans_5.'" >'.$vals['seans_5'].'</div></td>
            </tr>';
            }else{
            $result.='<tr><td style="text-align:left;"><a href="/'.$da.'/" style="text-decoration:none;color:#fff;"><div class="fiol"
style="cursor:pointer;">'.$vals['name_room'].'<br><span style="font-size:13px;">2-4 человека, до 60 мин</span></div></a></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_1\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_1'].'\');"><div class="bor_bron"
style="'.$seans_1.'" >'.$vals['seans_1'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_2\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_2'].'\');"><div class="bor_bron"
style="'.$seans_2.'">'.$vals['seans_2'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_3\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_3'].'\');"><div class="bor_bron"
style="'.$seans_3.'">'.$vals['seans_3'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_4\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_4'].'\');"><div class="bor_bron"
style="'.$seans_4.'">'.$vals['seans_4'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_5\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_5'].'\');"><div class="bor_bron"
style="'.$seans_5.'" >'.$vals['seans_5'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_6\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_6'].'\');"><div class="bor_bron"
style="'.$seans_6.'">'.$vals['seans_6'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_7\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_7'].'\');"><div class="bor_bron"
style="'.$seans_7.'">'.$vals['seans_7'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_8\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_8'].'\');"><div class="bor_bron"
style="'.$seans_8.'">'.$vals['seans_8'].'</div></td>
            <td onClick="bron_okno(\".$date.'\',\'seans_9\',\".$vals['id'].'\',\".$vals['name_room'].'\',\".$vals['seans_9'].'\');"><div class="bor_bron"
style="'.$seans_9.'">'.$vals['seans_9'].'</div></td>
            </tr>';
            }
            $seans_1=0;$seans_2=0;$seans_3=0;$seans_4=0;$seans_5=0;$seans_6=0;$seans_7=0;$seans_8=0;$seans_9=0;
            }
            }
            return $result;
            }

            ## комнаты для администрирования
            function getRooms_admin($date,$db){
                  $all=$db->getRow("SELECT * FROM bron_time WHERE date=?s",$date);
                  if(!$all){$db->query("INSERT INTO
bron_time(id_room,date,seans_1,seans_2,seans_3,seans_4,seans_5,seans_6,seans_7,seans_8,seans_9)
VALUES('1','".$date."','0','0','0','0','0','0','0','0','0'),('2','".$date."','0','0','0','0','0','0','0','0','0'),('3','".$date."','0','0','0','0','0','0','0','0','0')");}
                  $timtm=0;
                  if(date("w",time()+$timtm)==0){$W='<span style="color:#AB4490;">вс</span>';}
                  if(date("w",time()+$timtm)==1){$W='пн';}
                  if(date("w",time()+$timtm)==2){$W='вт';}
                  if(date("w",time()+$timtm)==3){$W='ср';}
                  if(date("w",time()+$timtm)==4){$W='чт';}
                  if(date("w",time()+$timtm)==5){$W='пт';}
                  if(date("w",time()+$timtm)==6){$W='<span style="color:#AB4490;">сб</span>';}
                  if(date("d.m.Y")==$date){$dop_cl='fiol';}else{$dop_cl='';}
                  $result='<tr><td style="width: 35%;height:75px;"><div class="fiol" style="margin-top: -10px;">'.date("d.m.Y").'<br><span
style="font-size:12px;">Сегодня</span></td>
                  <td><span onClick="days(\".date("d.m.Y").'\');" class="days_b '.$dop_cl.'">'.date("d").'</span><div
class="top_w">'.$W.'</div></td>';
            $timtm=86400;
                  $i=1;
                  for($i;$i<9;$i++){
                  if(date("d.m.Y",time()+$timtm)==$date){$dop_cl='fiol';}else{$dop_cl='';}
                  $result.='<td><span onClick="days(\".date("d.m.Y",time()+$timtm).'\');" class="days_b
'.$dop_cl.'">'.date("d",time()+$timtm).'</span>';
                  if(date("w",time()+$timtm)==0){$W='<span style="color:#AB4490;">вс</span>';}
                  if(date("w",time()+$timtm)==1){$W='пн';}
                  if(date("w",time()+$timtm)==2){$W='вт';}
                  if(date("w",time()+$timtm)==3){$W='ср';}
                  if(date("w",time()+$timtm)==4){$W='чт';}
                  if(date("w",time()+$timtm)==5){$W='пт';}
                  if(date("w",time()+$timtm)==6){$W='<span style="color:#AB4490;">сб</span>';}
                  $result.='<div class="top_w">'.$W.'</div></td>';
                  $timtm=$timtm+86400;
                  }
                  $result.='</tr>';

                  $rooms=$db->getAll("SELECT * FROM bron");
                  foreach($rooms as $keys=>$vals){
            $room=$db->getRow("SELECT * FROM bron_time WHERE date=?s AND id_room=?i",$date,$vals['id']);
            $color4ik='rgba(81, 27, 66, 0.85);border:none';
```

```
if($room['seans_1']==1){$seans_1='background:'.$color4ik.';';}
if($room['seans_2']==1){$seans_2='background:'.$color4ik.';';}
if($room['seans_3']==1){$seans_3='background:'.$color4ik.';';}
if($room['seans_4']==1){$seans_4='background:'.$color4ik.';';}
if($room['seans_5']==1){$seans_5='background:'.$color4ik.';';}
if($room['seans_6']==1){$seans_6='background:'.$color4ik.';';}
if($room['seans_7']==1){$seans_7='background:'.$color4ik.';';}
if($room['seans_8']==1){$seans_8='background:'.$color4ik.';';}
if($room['seans_9']==1){$seans_9='background:'.$color4ik.';';}
$result.='<tr><td style="text-align:left;"><div class="fiol">'.$vals['name_room'].'<br><span style="font-size:13px;">2-4 человека, до
60 мин</span></div></td>
<td onClick="bron(\".$date.'\',\'seans_1\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_1.'" >'.$vals['seans_1'].'</div></td>
<td onClick="bron(\".$date.'\',\'seans_2\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_2.'">'.$vals['seans_2'].'</div></td>
<td onClick="bron(\".$date.'\',\'seans_3\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_3.'">'.$vals['seans_3'].'</div></td>
<td onClick="bron(\".$date.'\',\'seans_4\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_4.'">'.$vals['seans_4'].'</div></td>
<td onClick="bron(\".$date.'\',\'seans_5\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_5.'" >'.$vals['seans_5'].'</div></td>
<td onClick="bron(\".$date.'\',\'seans_6\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_6.'">'.$vals['seans_6'].'</div></td>
<td onClick="bron(\".$date.'\',\'seans_7\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_7.'">'.$vals['seans_7'].'</div></td>
<td onClick="bron(\".$date.'\',\'seans_8\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_8.'">'.$vals['seans_8'].'</div></td>
<td onClick="bron(\".$date.'\',\'seans_9\',\".$vals['id'].'\');"><div class="bor_bron" style="'.$seans_9.'">'.$vals['seans_9'].'</div></td>
</tr>';
$seans_1=0;$seans_2=0;$seans_3=0;$seans_4=0;$seans_5=0;$seans_6=0;$seans_7=0;$seans_8=0;$seans_9=0;
}
return $result;
}


##Получение отзывов и работа с ними
function getOtziv(){
    $db=func_get_arg(1);
        $numar=func_num_args();
        $param=func_get_arg(0);
        if($numar>2){
        $ar1=func_get_arg(2);
            $bade_menu = $db->getAll("SELECT * FROM $param ORDER by date DESC Limit ?i",$ar1);
            $param=func_get_arg(2);
        }else{
        $bade_menu = $db->getAll("SELECT * FROM $param ORDER by date DESC");
        }
        if(empty($bade_menu)){
            $file_menu='Запрос к базе данных не принес результата!';
        }else{
            $file_menu='';
                foreach($bade_menu as $key=>$val){
                $file_menu.=file_get_contents ('./old_otziv.php');
                    foreach($val as $key_i=>$val_i){
                    $file_menu=str_replace('{'.$key_i.'}',$val_i,$file_menu);
                }
            }
        }
    return $file_menu;
    }
```

ПРИЛОЖЕНИЕ В

SQL-запрос на создание таблиц, первичных и внешних ключей, индексов, добавление данных

```
--
-- —труктура таблицы `bron_time`
--

CREATE TABLE `bron_time` (
  `id` int(255) NOT NULL,
  `id_room` int(11) NOT NULL,
  `date` varchar(255) NOT NULL,
  `seans_1` int(11) NOT NULL DEFAULT '0',
  `seans_2` int(11) NOT NULL DEFAULT '0',
  `seans_3` int(11) NOT NULL DEFAULT '0',
  `seans_4` int(11) NOT NULL DEFAULT '0',
  `seans_5` int(11) NOT NULL DEFAULT '0',
  `seans_6` int(11) NOT NULL DEFAULT '0',
  `seans_7` int(11) NOT NULL DEFAULT '0',
  `seans_8` int(11) NOT NULL DEFAULT '0',
  `seans_9` int(11) NOT NULL DEFAULT '0'
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

--
-- ƒамп данных таблицы `bron_time`
--

INSERT INTO `bron_time` (`id`, `id_room`, `date`, `seans_1`, `seans_2`, `seans_3`, `seans_4`, `seans_5`, `seans_6`, `seans_7`, `seans_8`,
`seans_9`) VALUES
(1, 1, '25.04.2016', 1, 1, 0, 0, 0, 0, 0, 0, 0),
(2, 2, '25.04.2016', 0, 0, 1, 0, 0, 0, 0, 0, 0),
(3, 3, '25.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(13, 1, '26.04.2016', 1, 1, 1, 1, 1, 1, 1, 1, 1),
(14, 2, '26.04.2016', 1, 1, 1, 1, 1, 1, 1, 1, 1),
(15, 3, '26.04.2016', 1, 1, 1, 1, 1, 1, 1, 1, 1),
(16, 1, '27.04.2016', 0, 0, 0, 0, 0, 0, 0, 1, 0),
(17, 2, '27.04.2016', 0, 0, 0, 0, 0, 0, 0, 1, 0),
(18, 3, '27.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(19, 1, '28.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(20, 2, '28.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(21, 3, '28.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(22, 1, '29.04.2016', 0, 0, 0, 0, 0, 1, 0, 1, 0),
(23, 2, '29.04.2016', 0, 0, 0, 0, 0, 1, 0, 1, 0),
(24, 3, '29.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(61, 1, '12.05.2016', 0, 0, 0, 0, 1, 0, 1, 1, 0),
(25, 1, '30.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(26, 2, '30.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(27, 3, '30.04.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(28, 1, '01.05.2016', 0, 0, 0, 0, 0, 0, 1, 1, 0),
(29, 2, '01.05.2016', 0, 0, 0, 0, 0, 0, 1, 1, 0),
(30, 3, '01.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(31, 1, '02.05.2016', 0, 1, 0, 0, 0, 0, 1, 1, 1),
(32, 2, '02.05.2016', 0, 1, 0, 0, 0, 0, 1, 1, 1),
(33, 3, '02.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(34, 1, '03.05.2016', 0, 0, 0, 0, 0, 0, 1, 1, 0),
(35, 2, '03.05.2016', 0, 0, 0, 0, 0, 0, 1, 1, 0),
(36, 3, '03.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(37, 1, '04.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(38, 2, '04.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(39, 3, '04.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(60, 3, '11.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(59, 2, '11.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(58, 1, '11.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(57, 3, '10.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(56, 2, '10.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(55, 1, '10.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(54, 3, '09.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
(53, 2, '09.05.2016', 0, 0, 0, 1, 1, 0, 0, 0, 0),
(43, 1, '06.05.2016', 0, 0, 0, 0, 0, 0, 0, 0, 0),
```

```
(44, 2, '06.05.2016', 0, 0, 0, 0, 0, 0, 0, 0),
(45, 3, '06.05.2016', 0, 0, 0, 0, 0, 0, 0, 0),
(52, 1, '09.05.2016', 0, 0, 0, 1, 1, 0, 0, 0),
(47, 2, '07.05.2016', 0, 0, 0, 0, 0, 0, 0, 0);

--
-- »ндексы сохранЄнных таблиц
--

--
-- »ндексы таблицы `bron_time`
--
ALTER TABLE `bron_time`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT для сохранЄнных таблиц
--

--
-- AUTO_INCREMENT для таблицы `bron_time`
--
ALTER TABLE `bron_time`
  MODIFY `id` int(255) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2332;
--
CREATE TABLE `bron` (
  `id` int(11) NOT NULL,
  `name_room` text NOT NULL,
  `seans_1` varchar(255) NOT NULL,
  `seans_2` varchar(255) NOT NULL,
  `seans_3` varchar(255) NOT NULL,
  `seans_4` varchar(255) NOT NULL,
  `seans_5` varchar(255) NOT NULL,
  `seans_6` varchar(255) NOT NULL,
  `seans_7` varchar(255) NOT NULL,
  `seans_8` varchar(255) NOT NULL,
  `seans_9` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- ƒамп данных таблицы `bron`
--

INSERT INTO `bron` (`id`, `name_room`, `seans_1`, `seans_2`, `seans_3`, `seans_4`, `seans_5`, `seans_6`, `seans_7`, `seans_8`, `seans_9`)
VALUES
(1, 'sриют', '10:00', '11:30', '13:00', '14:30', '16:00', '17:30', '19:00', '20:30', '22:00'),
(2, '¬о все т€жкие', '10:00', '11:30', '13:00', '14:30', '16:00', '17:30', '19:00', '20:30', '22:00'),
(3, 'RELAX-бар', '10:30', '12:00', '13:30', '15:00', '16:30', '18:00', '19:30', '21:00', '22:30');

--
-- »ндексы сохранЄнных таблиц
--

--
-- »ндексы таблицы `bron`
--
ALTER TABLE `bron`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT для сохранЄнных таблиц
--

--
-- AUTO_INCREMENT для таблицы `bron`
--
ALTER TABLE `bron`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;
```

Обозначение	Наименование	Дополнительные сведения
	Текстовые документы	
1. 11070026.09.03.02.033.ПЗВКР	Пояснительная записка	91 с.
	Графические документы	
2. 11070026.09.03.02.033.ДМВКР	Демонстрационные материалы (презентация)	14 сл.
	Демонстрационные материалы (пл. ф. А4)	14 *5 экз.
	Другие документы	
3.	Документы на компакт-диске	1 CD

					11070026.09.03.02.033.ПЗВКР			
Изм.	Лист.	Номер докум.	Подп.	Дата				
Разработал		Сидоров А.А.			Разработка и автоматизация системы онлайн учета и ведения базы данных web-сайта Ведомость ВКР	Лит.	Лист	Листов
Проверил		Щербинина Н.В				У	92	93
Н.контр.		Нестерова Е.В.				НИУ «БелГУ» гр. 07001353		
Утвердил		Иващук О.А.						

Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

«___» _____ _____ г.

_____ _____

(подпись) *(Ф.И.О.)*