МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет	*Кафедра*
вычислительной техники	«МОиПЭВМ»

Направление подготовки: 09.03.02 «Информационные системы и технологии»

Профиль подготовки: _____

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

на тему

Информационная система по продаже билетов в театр

Студент _____ (подпись, дата) Наврузов Зафар Бахрамович (ФИО полностью)

Руководитель _____ (подпись, дата) Кольчугина Е.А. (фамилия, инициалы)

Нормоконтролёр _____ (подпись, дата) Такташкин Д.В. (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____№_____)

Заведующий кафедрой _____ (подпись) Макарычев П.П. (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____№____)

Секретарь ГЭК _____ (подпись) Попова Н.А. (фамилия, инициалы)

Пенза, 2021

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет	*Кафедра*
вычислительной техники	«МОиПЭВМ»

«Утверждаю»
Заведующий кафедрой
_____П.П. Макарычев
«__»_____2021 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

1. Студент Наврузов Зафар Бахрамович гр. 17ВИ1 факультета ВТ
2. Тема работы Информационная система по продаже билетов в театр

Тема утверждена приказом ПГУ №_____от "_____"_____2021 г.

3. Руководитель ВКР профессор кафедры «МОиПЭВМ», д.т.н. Кольчугина Е.А.
4. Задание на работу (назначение разработки, исходные данные и т.п.)

1) Назначение: информационная система по продаже билетов в театр предназначена для автоматизации процесса бронирования и продажи билетов в театр, учет актеров, спектаклей, сеансов, билетов и пользователей системы.

2) Среда выполнения: операционная система Windows, начиная с версии 7.

Средства разработки: языки программирования Java Script, PHP, SQL, язык разметки HTML и язык стилей CSS; инструментальные средства «Visual Studio Code», «MS Office», «phpMyAdmin».

Технологии разработки: RUP, Ext JS, JSON, React.

3) Основные функции распределенного клиент-серверного приложения:

а) Ведение учета актеров, спектаклей, сеансов, билетов и пользователей системы.

б) Модификация записей.

в) Просмотр афиш, сеансов, актеров.

д) Бронирование и покупка билета «онлайн».

е) Регистрация, аутентификация и авторизация в системе.

ж) Поиск записей.

5. Перечень подлежащих разработке вопросов

1) Анализ предметной области

2) Анализ требований

3) Проектирование

4) Реализация

5) Тестирование

6) План разработки и оценка бюджета

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований	20	13.04.21-18.04.21	
2	Проектирование	14	20.04.21-25.04.21	
3	Реализация	5	11.05.21-27.05.21	
4	Контроль качества	5	27.05.21-30.05.21	
5	План разработки бюджета	8	30.05.21-31.05.21	
6	Оформление пояснительной записки	10	01.06.21-3.06.21	

Дата выдачи задания "___"_____ 2021 г.

Руководитель выпускной квалификационной работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить

Декан факультета _____
(подпись, дата)

Реферат

Пояснительная записка содержит 112 листов, 41 рисунок, 4 приложения, 23 использованных источника и 5 таблиц.

JAVASCRIPT OBJECT NOTATION, REACT, EXT JS, JAVASCRIPT, PHP, БАЗА ДАННЫХ, ТЕАТР, ПОКУПКА БИЛЕТОВ, СУБД, БРОНИРОВАНИЕ БИЛЕТОВ, АКТЕРЫ, СПЕКТАКЛИ, СЕАНСЫ, IDEF1X, RUTIONAL UNIFIED PROCESS, UML, MYSQL, VISUAL STUDIO CODE

Объект разработки – клиент-серверное приложение для продажи билетов в театр.

Целью выпускной квалификационной работы является разработка клиент-серверного приложения с удаленной базой данных, позволяющее автоматизировать работу сотрудников театра, а также автоматизировать процесс бронирования и продажи билетов.

Технология разработки – Rutional Unified Process, JavaScrit Object notation, Sencha Ext JavaScript, React.

Результатом работы является клиент-серверное приложение для продажи билетов в театр с разработанной базой данных.

Для разработки, проектирования и кодирование программного обеспечения использовались языки программирования JavaScript, PHP, язык разметки HTML, язык стилей CSS, язык запросов к базе данных SQL, текстовый редактор «Visual Studio Code», СУБД MySQL, инструменты для проектирования и создания UML-диаграмм Microsoft Office 2013 и ОС Windows Seven.

Содержание

Введение 5
1 Анализ предметной области и требований 7
1.1 Анализ предметной области и постановка задачи 7
1.1.1 Анализ программ аналогов 8
1.1.1.1 Анализ «Радарио» 8
1.1.1.2 Анализ «Единое поле» 9
1.1.1.3 Анализ «TicketsCloud» 10
1.1.1.4 Анализ «INTICKETS» 11
1.1.2 Постановка задачи 14
1.1.2.1 Назначение 14
1.1.2.2 Предполагаемая аудитория 14
1.1.2.3 Общий взгляд на продукт 14
1.1.2.4 Предположения и зависимости 15
1.1.3 Выбор технологий разработки 15
1.1.3.1 Методология RUP 15
1.1.3.2 Технология JSON 16
1.1.3.3 Framework «React» 16
1.1.3.4 Framework «Ext JS» 17
1.1.4 Выбор инструментальных средств 18
1.2 Анализ требований 20
1.2.1 Функциональные требования 20
1.2.2 Нефункциональные требования 25
1.2.2.1 Интерфейсы пользователей 25
1.2.2.2 Требования к производительности 25
1.2.2.3 Требования к безопасности 25
2 Проектирование 26
2.1 Базовая архитектура системы 26
2.2 Логическая модель базы данных 27
2.3 Пользовательский интерфейс клиентского приложения 28

2.4 Планирование разработки и оценка бюджета 33
2.4.1 Планирование проекта 33
3 Реализация 41
3.1 Физическая модель базы данных 41
3.2 Серверная часть приложения 41
3.2.1 Реализация представлений 42
3.3 Модель реализации клиентского приложения 42
4 Контроль качества программного обеспечения 46
4.1 Функциональное тестирование 46
Заключение 51
Список использованных источников 52
Приложение А. Глоссарий 55
Приложение Б. Листинг клиентского приложения 57
Приложение В. Результаты тестирования 102
Приложение Г. Листинг скрипта DDL 110

Введение

Масштабное развитие IT в России является одним из наиболее важных направлений государственной политики. Не так давно принята и реализуется Государственная программа РФ «Информационное общество» с периодом на 2011–2024 годы [1].

Целю Государственной программы является повышение конкурентоспособность экономики страны путём повышения значимости государственного управления, снижения издержек, а также улучшение качества продукции и обеспечение роста качества жизни населения.

Развитие информационных технологий коснулось почти всех сфер, которые так или иначе взаимодействуют с web-технологиями, облачными хранилищами и удаленным доступом. Особенно стоит отметить область услуг продажи и покупки через интернет. Данная сфера постоянно обновляется новыми web-сайтами, предложениями и конкуренция между фирмами, которые прибегают к таким новым технологиям, сильно растет. В сети интернет можно встретить множество сервисов по покупке и продаже билетов в кино, театры, парки и т.д. Информационные системы, которые используются в данной сфере помогают фирмам оперативно размещать афиши, использовать инструменты маркетинга данных сервисов, продавать билеты, тем самым разгружать сотрудников и исключать скопления очередей на кассах, что очень важно в условиях пандемии. Клиентам данных фирм, в свою очередь, удобнее просматривать афишу, расписание сеансов, покупать и бронировать билеты удаленно через данные сервисы.

В настоящее время на территории Российской Федерации существует большое количество уже готовых решений по данной теме за немалую цену. Но также находятся фирмы, которые прибегают к варианту «сайт под ключ», иными словами – нанимают профессионального программиста для более гибкой разработки сервиса. Таким фирмам, естественно, не выгодно делится полученным решением, как минимум из-за конкуренции на рынке [2].

Готовые программные решения сервисов по бронированию и продажи билетов в основном располагаются у поставщика. Это означает что сайт фирмы будет привязан к сервису поставщика и будет напрямую зависеть от его пропускной способности, безопасности и стабильности. К таким сервисам относятся, например, «Радарио», «Единое поле», «TicketsCloud», «INTICKETS».

Поэтому актуальной является задача разработки собственной, независимой от сервисов поставщиков информационной системы, которая поможет автоматизировать процесс бронирования и продажи билетов.

На основе вышеизложенного можно сделать вывод, что тема выпускной квалификационной работы актуальна.

Целью данной работы является разработка клиент-серверного приложения с удаленной базой данных, где будет автоматизирован процесс бронирования и продажи билетов в театр, учет актеров, спектаклей, сеансов, билетов и пользователей системы.

1 Анализ предметной области и требований

1.1 Анализ предметной области и постановка задачи

Благодаря интернет-технологиям, в настоящее время можно не выходя из дома выбирать сеанс спектакля и заранее заказывать билеты на наиболее удобные места. Система получила название «Online бронирование».

Процесс бронирования и покупки билетов через интернет сводится к следующим шагам:

а) пользователь системы регистрируется на сайте театра, где хочет приобрести онлайн билет, либо забронировать конкретное место для дальнейшей покупки (после регистрации пользователю открывается функционал бронирования и покупки билетов);

б) пользователь просматривает и выбирает по своему вкусу афиши и спектакли, которые должны состояться в ближайшее время;

в) после выбора спектакля пользователь переходит к выбору ряда и места в зале;

г) пользователь бронирует определенное место и, при желании, покупает билет «online».

Предприятия индустрии развлечений заинтересованы в увеличении каналов распространения билетов, в прозрачном учете и контроле за продажами. На рынке представлено множество программных решений для этих целей, но большинство из них предполагает участие оператора в процессе реализации билетов. А это значит, что предприятию придется устанавливать обычную билетную кассу, нанимать персонал, решать другие организационные вопросы. Замена традиционных точек продаж «online» кассирами позволит автоматизировать значительную часть этих операций, минимизировать ошибки сотрудников, снизить операционные расходы [3].

1.1.1 Анализ программ аналогов

Информационные системы по бронированию и продаже билетов онлайн, такие как «Радарио», «Единое поле», «TicketsCloud», «INTICKETS» представляют собой клиент-серверное приложение.

1.1.1.1 Анализ «Радарио»

«Радарио» – это профессиональный сервис для онлайн-продажи билетов и доступов к трансляциям [4].

Сервис «Радарио» представляет собой мощную надстройку для билетного решения, где есть все необходимые инструменты для продвижения событий и работы со зрителями

На рисунке 1 представлен пример бронирования билета через данную систему.

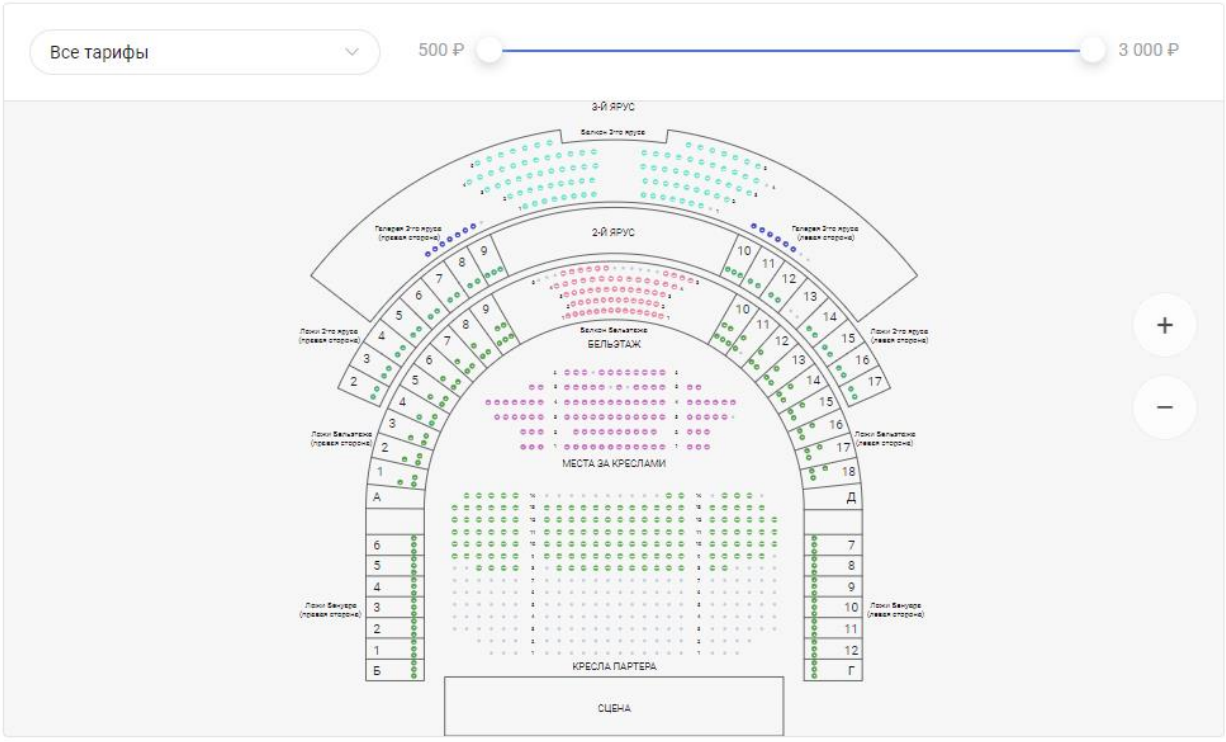

Рисунок 1 – Окно «Бронирование билета» в системе «Радарио»

У данного программного обеспечения присутствует богатый функционал и большое количество настроек.

Продукт «Радарио» имеет много плюсов, например, множество функций для работы с отдельными бизнес-процессами, легко читаемый интерфейс пользователя, набор отдельных модулей программы для обеспечения анализа. Минусами данной программы являются недостаточно подробная сноска о спектакле (например, нельзя добавить список актеров), платный доступ к программе, комиссия за каждую покупку билета, что влечет за собой возникновение больших убытков компании на покупку данного сервиса, а также на постоянное отчисление комиссий.

1.1.1.2 Анализ «Единое поле»

«Единое поле» – сервис по продаже билетов. Сервис предоставляет одностраничный сайт, разработанный специально для продажи билетов на мероприятие [5].

Окно для онлайн бронирования в системе «Единое поле» представлено на рисунке 2.

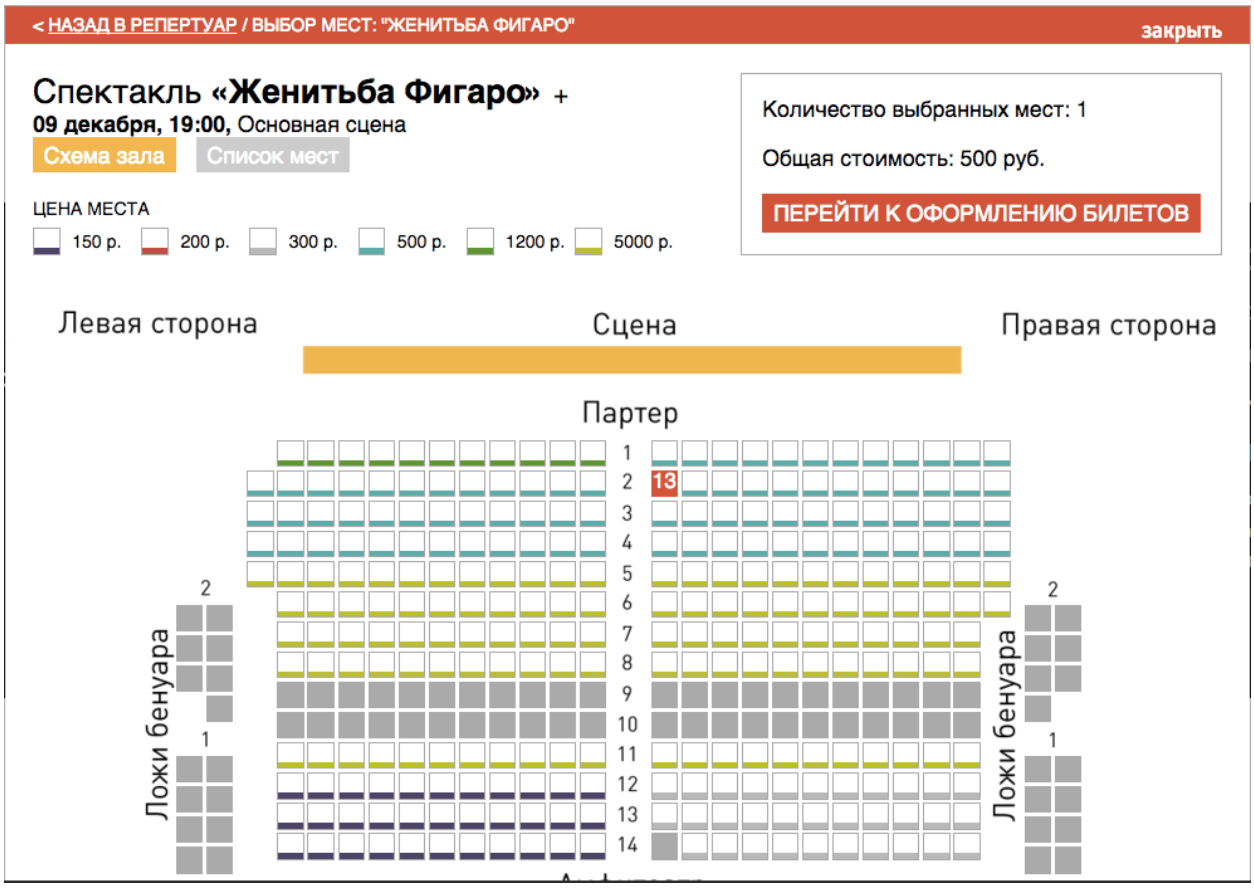

Рисунок 2 – Окно для онлайн бронирования в системе «Единое поле»

В целом «Единое поле» – это быстрая и простая система для бронирования билетов. В данном программном обеспечении имеются следующие недостатки:

– сложная интеграция с сайтом фирмы;

– высокая стоимость программного обеспечения;

– комиссия за каждый проданный билет.

1.1.1.3 Анализ «TicketsCloud»

«TicketsCloud» – глобальная платформа по продаже билетов.

Окно для редактирования информации о концерте представлено на рисунке 3.

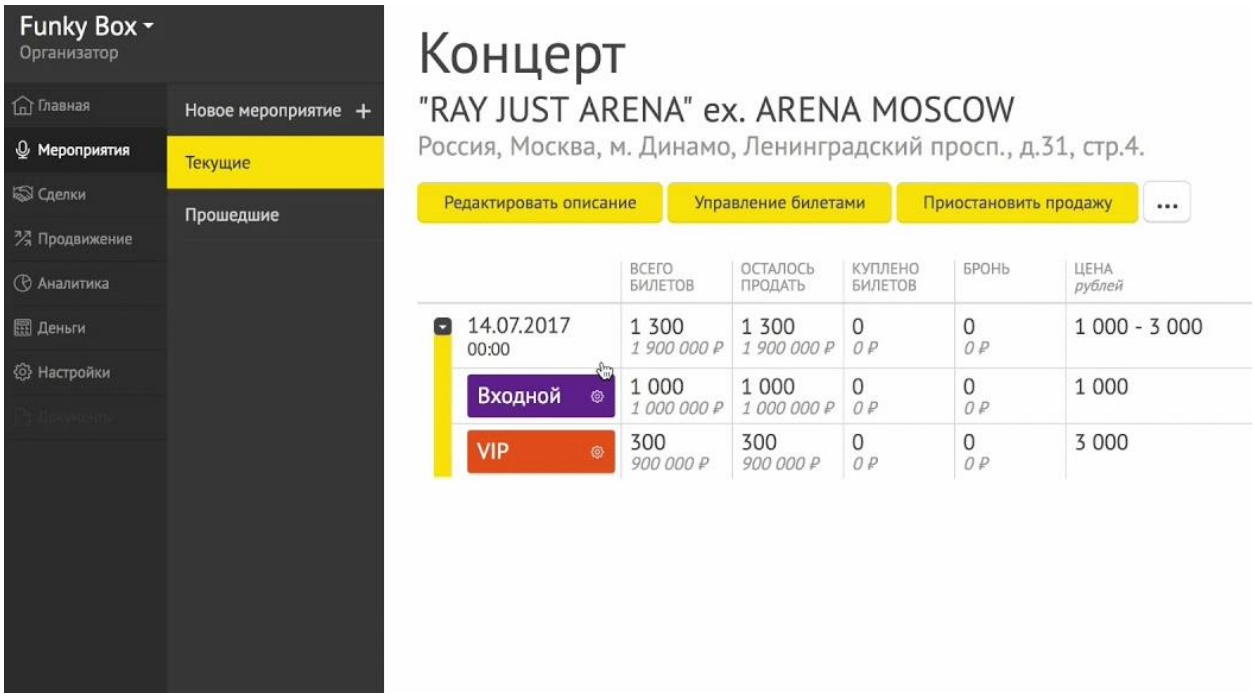

Рисунок 3 – Окно для редактирования информации о концерте в системе «TicketsCloud»

Достоинства данного программного обеспечения следующие: сервис легко кастомизируется под мероприятие любой сложности, имеется система промокодов, аналитика по каналам продаж, возможность реализовывать нестандартные размещения внутри зала [6]. Данный программный продукт имеет следующие недостатки:

– зависимость сервиса от сайта-поставщика;

– комиссия за каждый проданный билет.

1.1.1.4 Анализ «INTICKETS»

«INTICKETS» – современный облачный сервис для контроля и управления продажей билетов в online и offline среде на культурно-массовые мероприятия. «INTICKETS» представляет собой профессиональный продукт для автоматизации всех бизнес-процессов с набором современных инструментов для управления полным циклом продаж в учреждениях культуры, у билетных агентов и организаторов зрелищных мероприятий, деятельность которых связана с продажей билетов [7].

Система позволяет билетному столу театра или зрелищной площадки автоматизировать весь процесс от планирования до контроля поступления выручки, включая собственную и агентскую продажу электронных билетов, билетов на бланках строгой отчетности в кассах.

Окно для бронирования билете в системе «INTICKETS» представлено на рисунке 4.

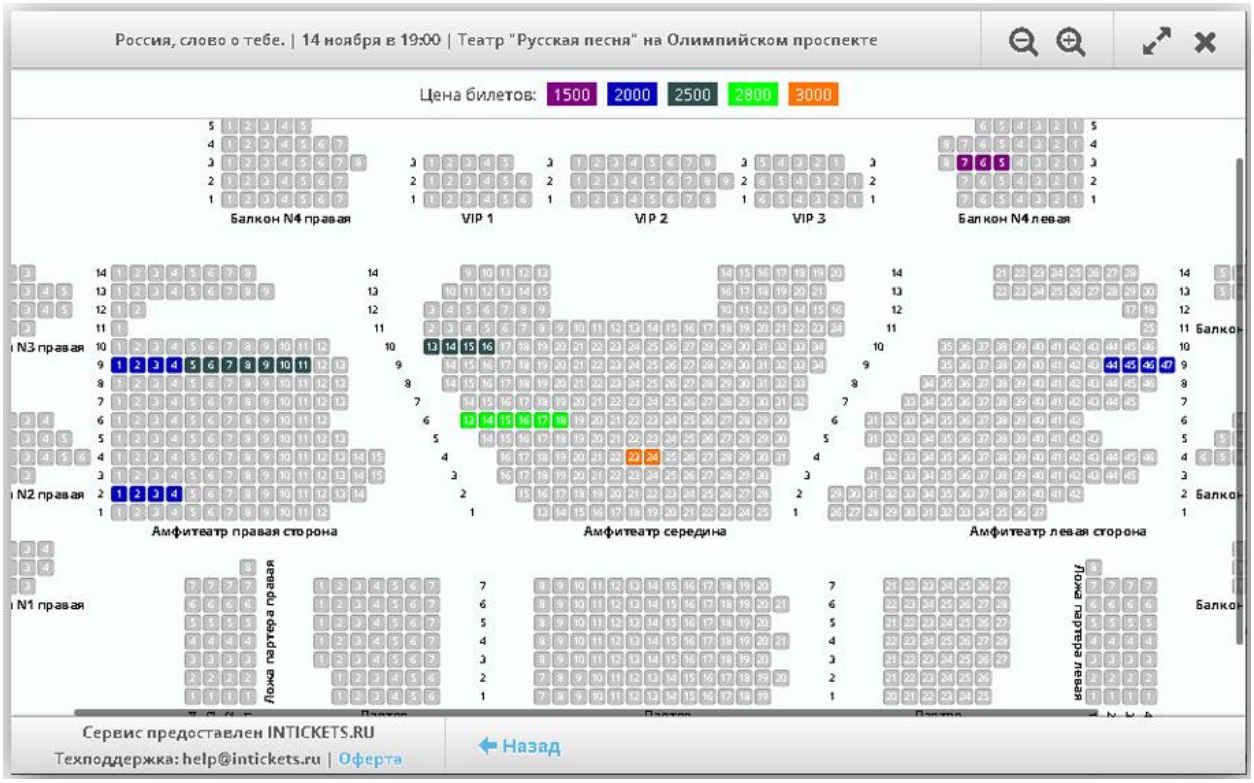

Рисунок 4 – Окно для бронирования билета в системе «INTICKETS»

Достоинства данной информационной системы:

– позволяет работать в offline режиме;

– интеграция с 1С;

– формирование отчетов для анализа.

У данного программного продукта выявлен только один недостаток – высокая стоимость.

Для сравнительного анализа программ по продаже и бронированию билетов были выделены такие критерии сравнения, как:

– удобство интерфейса;

– стоимость продукта;

– отчисление процента комиссии за каждый проданный билет;

– достаточность информации о спектакле;

– зависимость программного продукта от других платформ.

Результаты сравнительного анализа программ-аналогов по бронированию и продаже билетов представлен в таблице 1.

Таблица 1 – Сравнительный анализ программ-аналогов

Критерий	Объект анализа			
	«Радарио»	«Единое поле»	«TicketsCloud»	«INTICKETS»
Удобство интерфейса	Интуитивно понятный интерфейс	Интуитивно понятный интерфейс	Интуитивно понятный интерфейс	Сложный для восприятия интерфейс
Стоимость продукта	30 000 рублей в месяц	От 15 000 рублей в месяц	От 10 000 рублей в месяц	По согласованию с компанией
Отчисление процента комиссии за каждый проданный билет	От 2,5% до 8% за каждый проданный билет	От 1,5% до 3% за каждый проданный билет	5% за каждый проданный билет	От 2,5 до 10% за каждый проданный билет

Продолжение таблицы 1

Критерий	Объект анализа			
	«Радарио»	«Единое поле»	«TicketsCloud»	«INTICKETS»
Достаточность информации о спектакле	Информации о спектакле недостаточно (нельзя добавить список актеров)	Полная информация о спектакле	Полная информация о спектакле	Полная информация о спектакле
Зависимость продукта от других платформ	Модуль программы зависит от платформы поставщика	Модуль программы зависит от платформы поставщика	Модуль программы зависит от платформы поставщика	«Online» режим зависит от платформы поставщика, «Offline» режим не зависит.

В результате проведенного анализа можно прийти к выводу, что каждая из рассмотренных программ имеет свои недостатки и преимущества.

После проведения анализа программ-аналогов было решено реализовать свою автоматизированную информационную систему по бронированию и покупке билетов в театр, где исключены недостатки, описанные выше: высокая стоимость покупки программного продукта, отчисление комиссии за каждый проданный билет, зависимость от сайта-поставщика, недостаточность информации о спектакле. Стоит обозначить, что проанализированные программные продукты в основном предназначены для размещения в качестве модуля на собственном сайте компании, а разрабатываемое программное обеспечение – напротив, нацелено на одну полноценную систему, где функции продажи и бронирования билетов будут взаимосвязаны с учетом спектаклей и сеансов.

Разрабатываемое клиент-серверное приложение с удаленной базой данных для автоматизации процессов продажи и бронирования билетов бу-

дет актуально для фирм, которым необходим функционал, который повышает качество и скорость работы.

1.1.2 Постановка задачи

1.1.2.1 Назначение

Разрабатываемая ИС предназначена для автоматизации процесса бронирования и продажи билетов в театр, учета актеров, спектаклей, сеансов, билетов и пользователей системы. Система ориентирована на конечного пользователя, не обладающего высокой квалификацией в области вычислительной техники. Поэтому клиент-серверное приложение автоматизированной информационной системы должно обладать простым, удобным и интуитивно понятным интерфейсом. Также должна быть разработана база данных, которая будет обеспечивать надежное и безопасное хранение и доступ к информации.

1.1.2.2 Предполагаемая аудитория

Автоматизированная ИС предназначена для сотрудников театра, которые контролируют учет актеров, спектаклей, сеансов, билетов, и для клиентов театра, которые могут просмотреть афиши и сеансы для дальнейшего бронирования или покупки билета на определенный сеанс выбранного спектакля.

1.1.2.3 Общий взгляд на продукт

Программный продукт состоит из трех частей. Первая часть – это база данных, которая находится на сервере, для обеспечения хранения информации об актерах, пользователях системы, спектаклях, сеансах и билетах. Вторая часть – программное обеспечение, где автоматизированы большинство функций системы, связанные с учетом. Третья часть – программное обеспечение, которое находится на стороне клиента, а именно будет работать под управлением браузера на языке JavaScript. Третья часть программного обес-

печения представляет собой оболочку, построенную на базе фреймворков «React» и «Sencha Ext JS».

1.1.2.4 Предположения и зависимости

Стабильность работы программного продукта зависит от наличия подключения к глобальной сети интернет. Так же стоит отметить, что немаловажным является стабильность сервера, где будет развернута система управления базами данных и некоторые файлы программы для осуществления запросов к CGI-программам.

1.1.3 Выбор технологий разработки

1.1.3.1 Методология RUP

Рациональный унифицированный процесс разработки программного обеспечения (RUP – Rational Unified Process) является частным случаем унифицированного процесса (UP – Unified Process). В основу рационального унифицированного процесса положена итеративная разработка программного обеспечения. В рамках RUP разработка выполняется в виде нескольких краткосрочных итераций продолжительностью от двух до шести недель. Итерация по существу является мини-проектом фиксированной длительности, в результате которой расширяется и дополняется функциональность разрабатываемой системы. Поэтому унифицированный процесс разработки иногда называют итеративной и инкрементальной разработкой.

В результате каждой итерации получается работающая, но не полнофункциональная система, которая еще не является коммерческой и не подлежит распространению. Продолжительность создания коммерческой версии программной системы составляет 10–15 итераций.

Но результат каждой итерации нельзя рассматривать и в виде прототипа системы. Правильнее сказать, что в результате каждой итерации создается окончательная версия некоторой части всех системы [8].

1.1.3.2 Технология JSON

JSON (JavaScript Object Notation) – текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми. Наиболее частое распространенное использование JSON – пересылка данных от сервера к браузеру. Обычно данные JSON доставляются с помощью AJAX, который позволяет обмениваться данными браузеру и серверу без необходимости перезагружать страницу. Также можно использовать JSON для отправки данных от браузера на сервер, передавая строку JSON в качестве параметра запросов GET или POST. Но данный метод имеет меньшее распространение, так как передача данных через запросы AJAX может быть упрощена. Например, ID продукта может быть включен в адрес URL как часть запроса GET. Хоть JSON и прост для понимания и использования, он является очень полезным и гибким инструментом для передачи данных между приложениями и компьютерами, особенно при использовании AJAX [9].

1.1.3.3 Framework «React»

В качестве одно из фрейморков для разработки клиентского программного обеспечения был выбран React. React – это библотека JS для разработки web-приложений и пользовательских интиерфейсов. «React» был создан компанией Facebook, а первый релиз библиотеки увидел свет в марте 2013 года. Текущей версий на данный момент является версия React v17.0.

Первоначально «React» предназначался для создания веб-сайтов, однако позже появилась платформа «React Native», которая уже предназначалась для мобильных устройств. «React» представляет собой идеальный инструмент для создания масштабируемых веб-приложений, особенно в тех ситуациях, когда приложение представляет SPA (одностраничное приложение).

React-разработка заключается в описании того, что нужно вывести на страницу (а не в составлении инструкций для браузера, посвящённых тому,

как это делать). Это означает значительное сокращение объёмов шаблонного кода.

Отличительной чертой библиотеки является концентрация на компонентах – мы можем создать отдельные компоненты и затем их легко переносить из проекта в проект. Еще одна особенность «React» – использование JSX. JSX представляет комбинацию кода JavaScript и XML и предоставляет простой и интуитивно понятный способ для определения кода визуального интерфейса.

«React» развивается как открытый проект, и все сайты библиотеки доступны на официальном сайте. Кроме того, данной библиотеке посвящано много статей и книг, а также есть собранная документация, которая описывает функции и классы данной библиотеки.

«React» относительно прост в освоении, имеет понятный и лаконичный синтаксис [10].

1.1.3.4 Framework «Ext JS»

Вторым фреймворком для проектирования и разработки приложения выбран Ext JS. Фреймверк Ext JS поддерживается всеми современными браузерами, начиная от Internet Explorer 6 и кончая последней версией Google Chrome [11]. Он предоставляет разработчику целый набор графических компонентов, начиная от простых кнопок и расширенных элементов обычных HTML-форм, до сложнейших компонентов вроде таблиц, слоев и деревьев. В библиотеке также есть достаточно много невидимых пользователю компонентов, которые и обеспечивают работу того, что видно на экране. Это и получение данных с сервера в фоновом режиме (в формате JSON или XML), обновление частей страницы, локальные хранилища данных, поддержка cookie и многое другое.

Самым мощным и известным компонентом является таблица или Grid, функционал которого в ExtJS описан крайне удачно. Здесь присутствует и группировка, и сортировка, и поддержка выделения, и редактируемые табли-

цы (при этом можно задавать отдельные редакторы для разных колонок и типов данных в них), и разбиение на страницы с последующей навигацией, и многое другое.

Каждый компонент в ExtJS позволяет себя конфигурировать путем настройки разнообразных опций, генерирует множество событий в ответ почти на любое изменение своего состояния, а также гибко настраивается через задание необходимых свойств оформления в CSS-стилях. В архитектуре библиотеки есть и такое понятие как тема оформления – это обычный CSS-файл, изменяя который можно мгновенно изменить внешний вид всех элементов интерфейса. Уже создано множество различных пользовательских тем, в разной цветовой гамме и стиле, хотя, на мой взгляд, именно стандартная тема наиболее приятная и привычная для большинства пользователей [12].

1.1.4 Выбор инструментальных средств

Для обеспечения хранения информации об актерах, пользователях системы, спектаклях, сеансах и билетах была задействована реляционная база данных. В рамках данной выпускной квалификационной работы предпочтение было уделено СУБД MySQL, где используется язык SQL. MySQL – это свободная реляционная система управления базами данных (СУБД) с открытым кодом. MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Денвер, XAMPP, VertrigoServ. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы. Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых ти-

пов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц [13].

Для разработки программного обеспечения была выбран редактор исходного кода «Visual Studio Code». «Visual Studio Code» позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией. «Visual Studio» также позволяет заменять кодовую страницу при сохранении документа, символы перевода строки и язык программирования текущего документа. На март 2019 года посредством встроенного в продукт пользовательского интерфейса можно загрузить и установить несколько тысяч расширений только в категории «programming languages» (языки программирования) [14].

Для кодирования программного обеспечения используются JavaScript и PHP. JavaScript прекрасно подходит для кодирования оболочки программного обеспечения (интерфейса пользователя). PHP был выбран как общий «стандарт» для промежуточной обработки данных между оболочкой системы и СУБД [15].

Для создания шаблонов печати был выбран язык XSLT. XSLT – язык преобразования XML-документов. Спецификация XSLT входит в состав XSL и является рекомендацией W3C. При применении таблицы стилей XSLT, состоящей из набора шаблонов, к XML-документу образуется конечное дерево, которое может быть сериализовано в виде XML-документа, XHTML-документа, HTML-документа или простого текстового файла [16].

В качестве языка моделирования был выбран UML. UML объединил лучшие современные технические приемы моделирования и разработки про-

граммного обеспечения. UML-диаграммы легко воспринимаются и при этом без труда генерируются компьютерами [17].

Для проектирования UML-диаграмм использовался продукт Microsoft Office Visio 2013.

Логическое и физическое проектирование базы данных клиент-серверного приложения выполнялось при помощи ERWin Data Modeler.

Для планирования и распределения ресурсов по задачам был выбран продукт Microsoft Project 2013.

1.2 Анализ требований

Требование можно определить, как «подробное описание того, что должно быть реализовано».

1.2.1 Функциональные требования

Диаграммы вариантов использования (use case diagram) используются для более простого взаимодействия с будущими пользователями разрабатываемой системы. Use-case диаграммы показывают что система должна делать.

Были спроектированы две диаграммы вариантов использования, разбитые на функционал администратора и на функционал клиента.

Диаграммы вариантов использования с функционалом администратора представлена на рисунке 5.

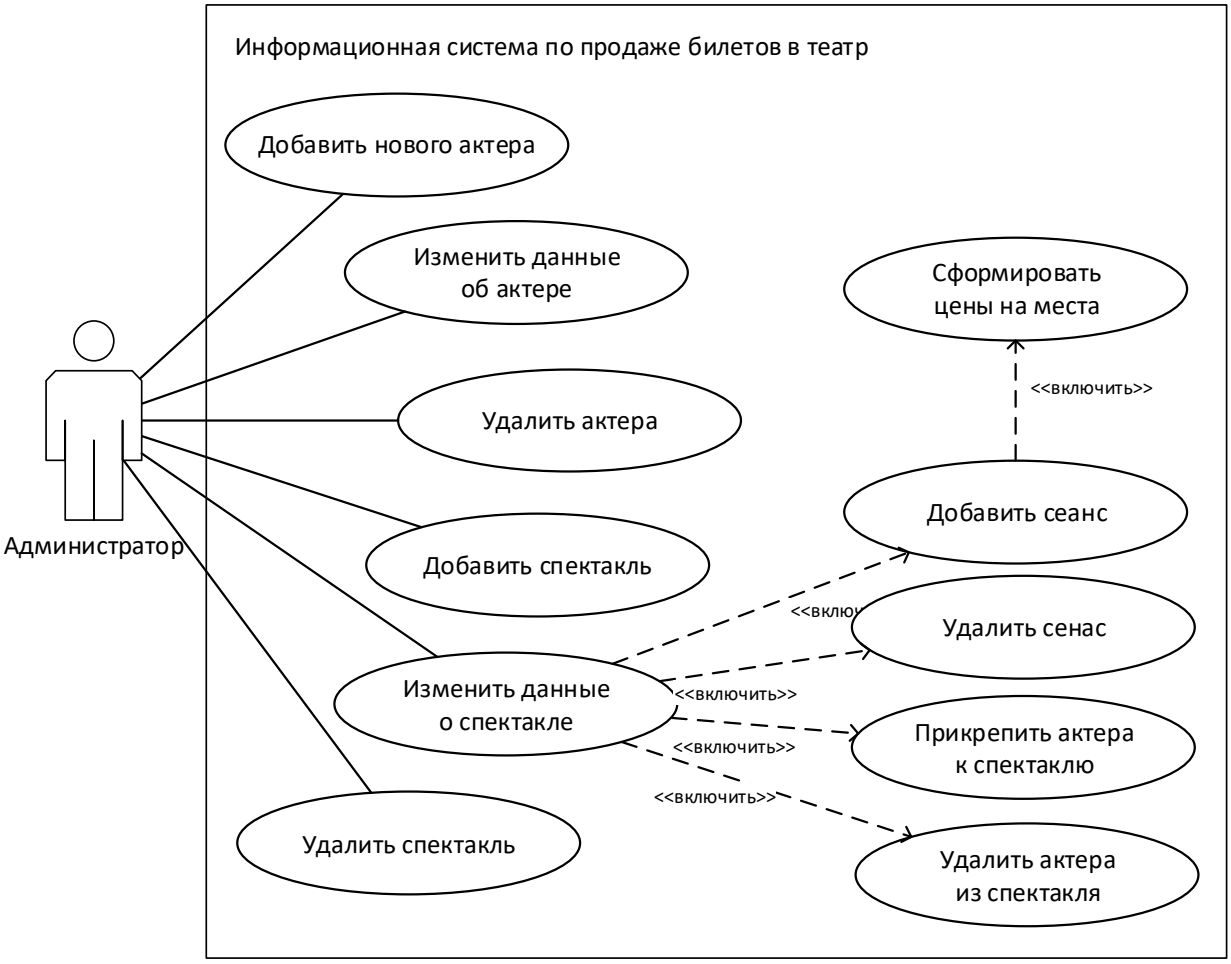

Рисунок 5 – Диаграмма вариантов использования
(функционал администратора)

Вариант использования «Изменить данные о спектакле» включает в себя варианты: «Добавить сеанс», «Удалить сеанс», «Прикрепить актера к спектаклю», «Удалить актера из спектакля».

Вариант использования «Добавить сеанс» включает в себя вариант: «Сформировать цены на места».

Диаграммы вариантов использования с функционалом клиента системы представлена на рисунке 6.

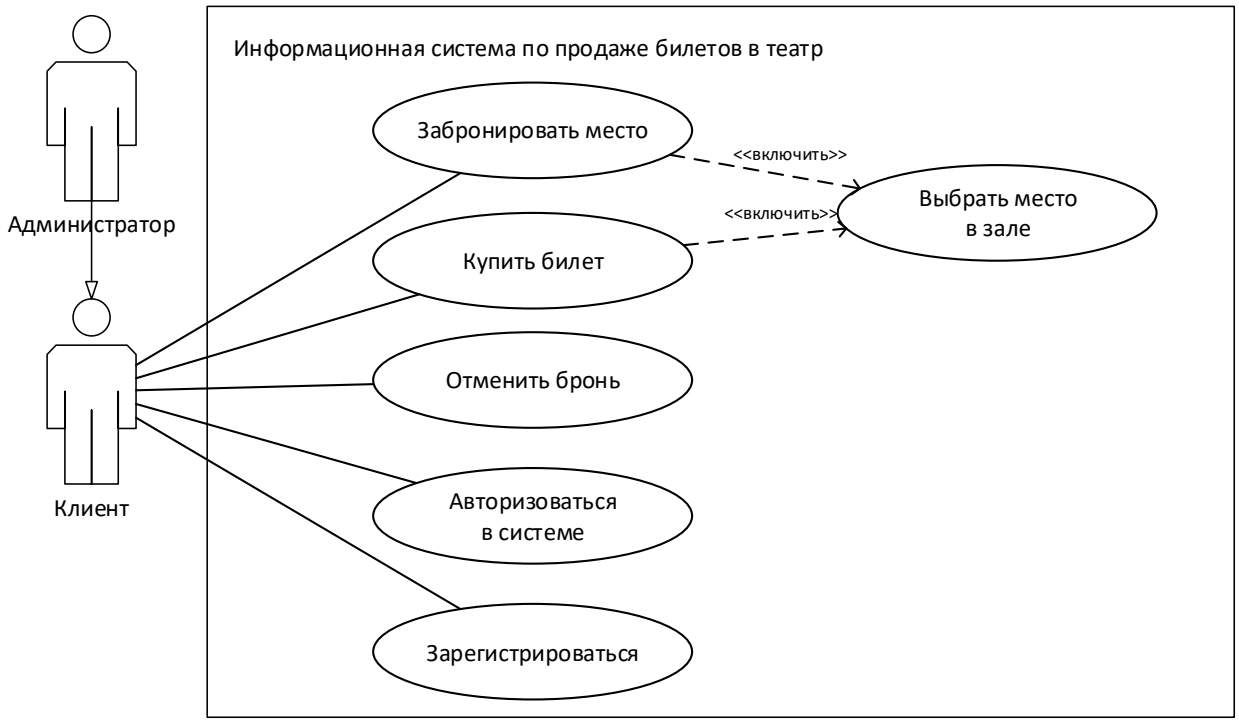

Рисунок 6 – Диаграмма вариантов использования
(функционал клиента)

Варианты использования «Забронировать место» и «Купить билет» включают в себя вариант: «Выбрать место в зале».

В информационной системе по бронированию билетов в театр присутствуют 3 типа пользователей: «Администратор», «Зарегистрированный пользователь» и «Гость». Так как у пользователя «Гость» функционал, кроме просмотра страниц и сайта отсутствует, было принято решение не выделять под данный тип отдельную роль. В соответствии с описанными типами пользователей на диаграмме вариантов использования было выделено только две роли: «Администратор» и «Клиент» (зарегистрированный пользователь).

Диаграммы деятельности представляют собой графическое представление рабочих процессов поэтапных действий и действий с поддержкой выбора, итерации и параллелизма [18].

Диаграмма деятельности для прецедента «Купить билет» расположена на рисунке 7.

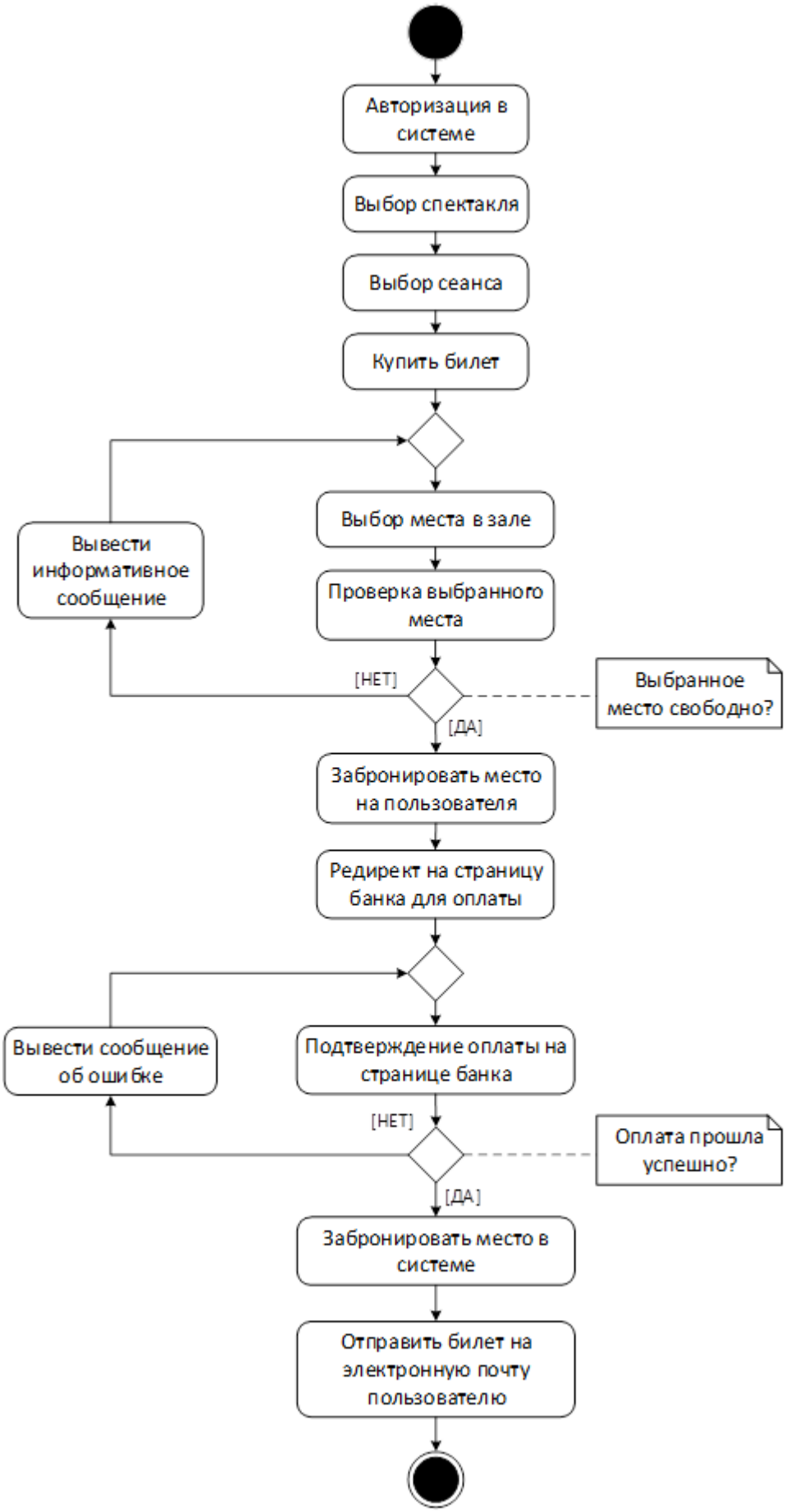

Рисунок 7 – Диаграмма деятельности (купить билет)

Диаграмма последовательности моделирует взаимодействие объектов на основе временной последовательности. Она показывает, как одни объекты взаимодействуют с другими в конкретном прецеденте [19].

В рамках данной работы была спроектирована диаграмма последовательности для прецедента «Купить билет» (рисунок 8).

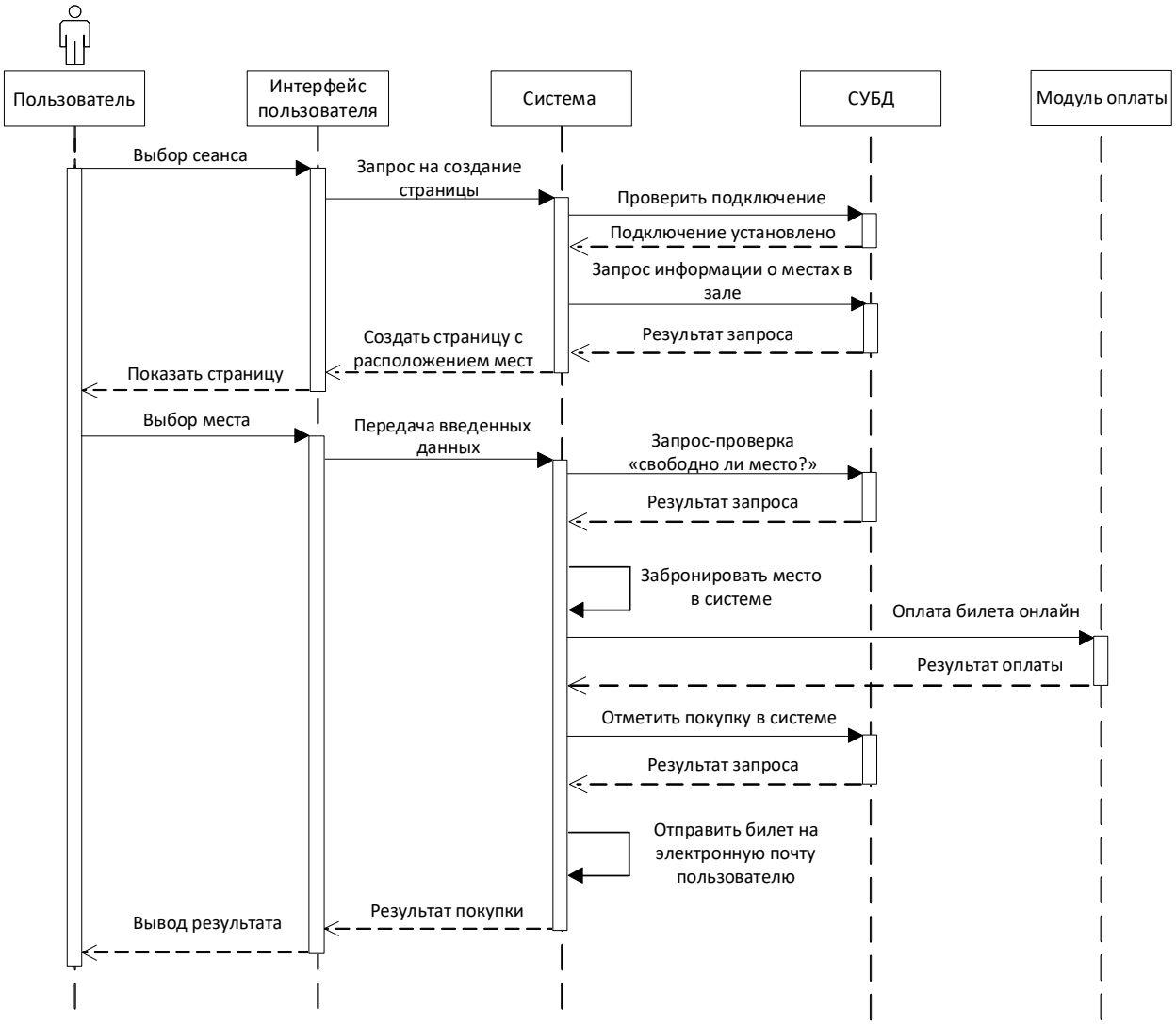

Рисунок 8 – Диаграмма последовательности для прецедента «Купить билет»

На диаграмме последовательности расположены пять компонентов: пользователь, интерфейс пользователя, система, СУБД, модуль оплаты. Второй компонент – «Интерфейс пользователя» представлен в виде оболочки фреймворка «React» и «Sencha ExtJS». Модуль оплаты представляет собой систему оплаты онлайн при помощи банковской карты. Данный модуль может быть организован, например, через «Яндекс кассу».

1.2.2 Нефункциональные требования

Нефункциональные требования показывают какими важными характеристиками и свойствами должна обладать разрабатываемая система и программное обеспечение.

1.2.2.1 Интерфейсы пользователей

Программное обеспечение для покупки и бронирования билетов в театр должно обладать читаемым и простым графическим интерфейсом пользователя.

1.2.2.2 Требования к производительности

Программное обеспечение должно обеспечивать стабильную работу при наличии установленного стабильного соединения с сервером по средствам сети интернет.

1.2.2.3 Требования к безопасности

Доступ к клиентской части приложения должен осуществляться при помощи аутентификации. А доступ аутентифицированного пользователя к определенным функциям системы должен осуществляться при помощи авторизации.

2 Проектирование

2.1 Базовая архитектура системы

Архитектура информационной системы представлена в виде модели «клиент-сервер-сервер».

Схема архитектуры АИС представлена на рисунке 9.

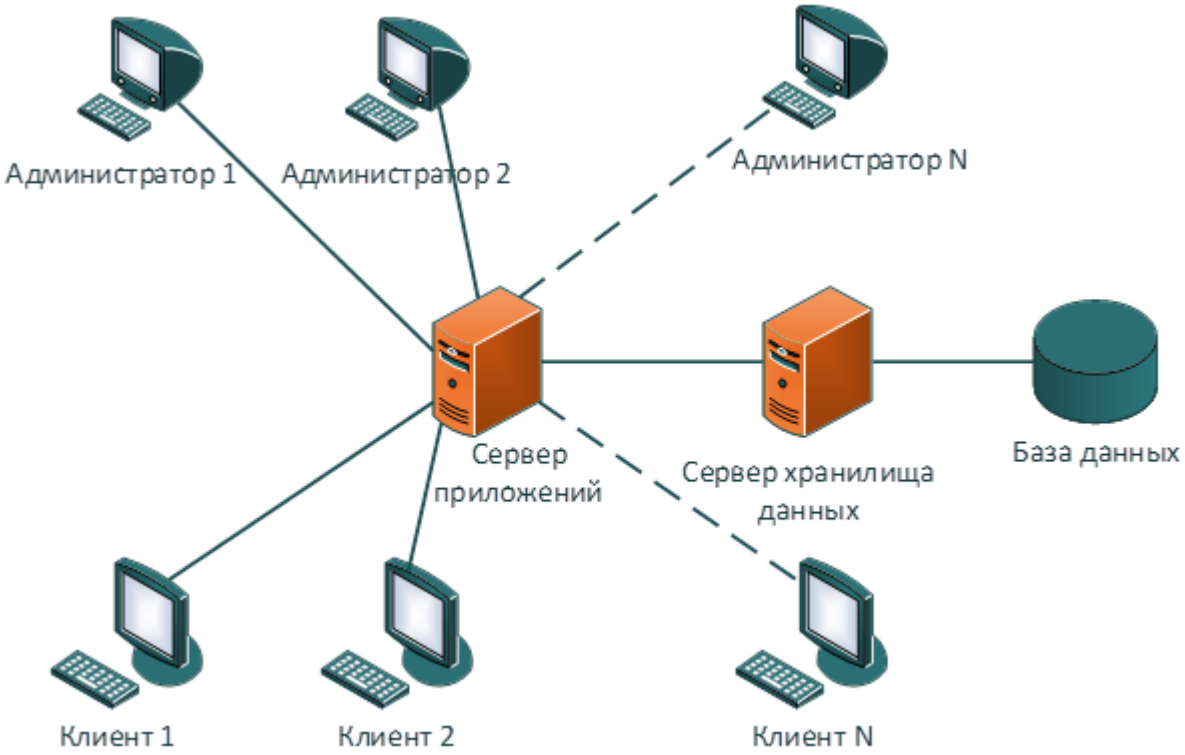

Рисунок 9 – Схема архитектуры АИС

В представленной схеме архитектуры на рисунке 9 расположены 3 главных компонента системы. Первое звено описывается как машина клиента (персональный компьютер), где выполняется часть приложения, написанная на JavaScript, средствами установленного модуля Adobe Flash через web-браузер. Второе звено представляет собой web-приложение (написанное на языке программирования PHP), располагающееся на сервере приложений, к которому удаленно подключаются пользователи АИС (администраторы и клиенты). Третьим звеном является север сервер базы данных, который располагается на дополнительной машине и хранит базу данных.

2.2 Логическая модель базы данных

Логическая модель строится на основе концептуальной модели с учетом выбранной модели данных, но не учитывая особенности целевой СУБД. Устранение избыточности модели производится, как правило, за счет декомпозиции отношений таким образом, чтобы в каждом отношении хранились только первичные факты.

Нормализация предназначена для приведения структуры базы данных к виду, обеспечивающему минимальную избыточность. Конечной целью нормализации является уменьшение потенциальной противоречивости хранимой в БД информации [20].

Построим логическую модель базы данных в нотации IDEF1X при помощи case-средства ERwin Data Modeler. Логическая модель представлена на рисунке 10.

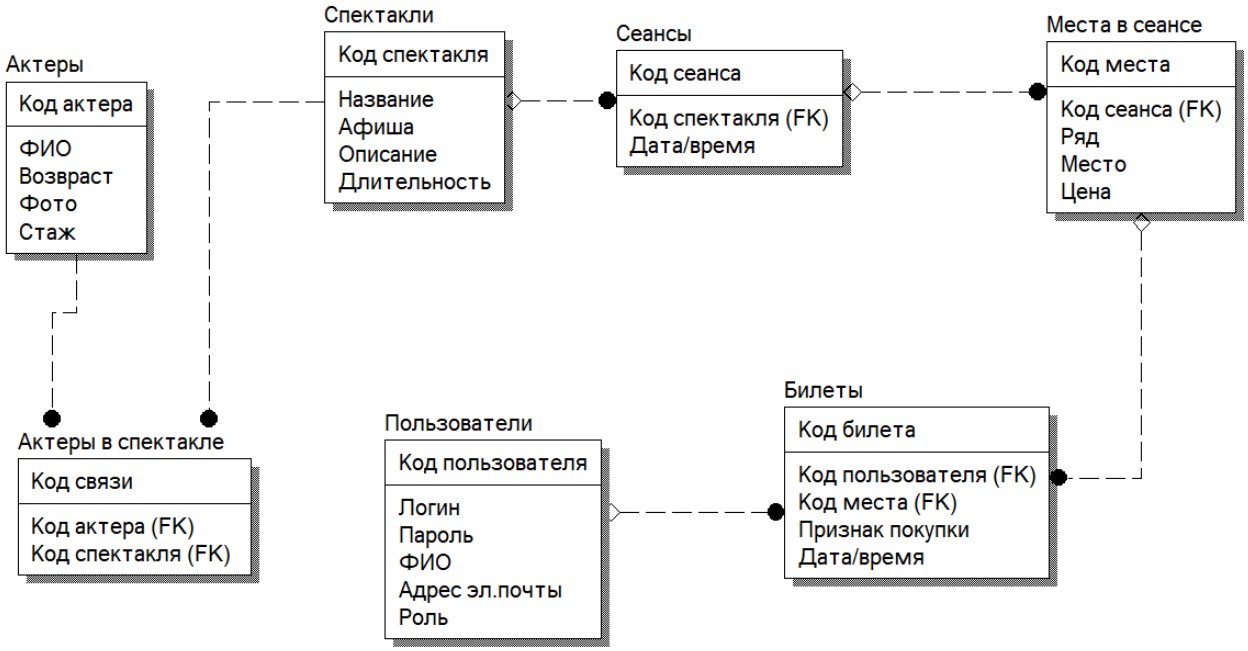

Рисунок 10 – Логическая модель базы данных

Разработанная логическая модель базы данных не содержит транзитивных зависимостей. Таким образом можно сделать вывод, что модель соответствует третьей нормальной форме.

2.3 Пользовательский интерфейс клиентского приложения

На основе диаграммы прецедентов и анализа функциональных и нефункциональных требований был разработан простой и интуитивно понятный графический интерфейс пользователя.

Графический интерфейс пользователя был построен с помощью фреймворков «Ext JS» и «React». На рисунке 11 представлен графический пользовательский интерфейс главной страницы web-приложения с функционалом администратора.

Рисунок 11 – Графический интерфейс главной страницы web-приложения (функции для администратора)

Рассмотрим основные элементы интерфейса главной страницы разработанного приложения, представленные на рисунке 11:

1 – кнопка для перехода на главную страницу;

2 – кнопка для перехода на страницу работы с данными об актерах театра;

3 – кнопка для перехода на страницу работы с данными о спектаклях;

4 – кнопка для перехода в личный кабинет с информацией о бронях;

5 – кнопка для выхода из системы;

6 – область описания театра;

7 – таблица с ближайшими сеансами;

8 – кнопка для перехода на страницу работы с сеансами (для администратора).

Данная страница представляет собой описание театра (историю), информацию о ближайших сеансах и меню для работы с сайтом.

Графический интерфейс страницы с информацией об актерах театра представлен (пользователь без авторизации) на рисунке 12.

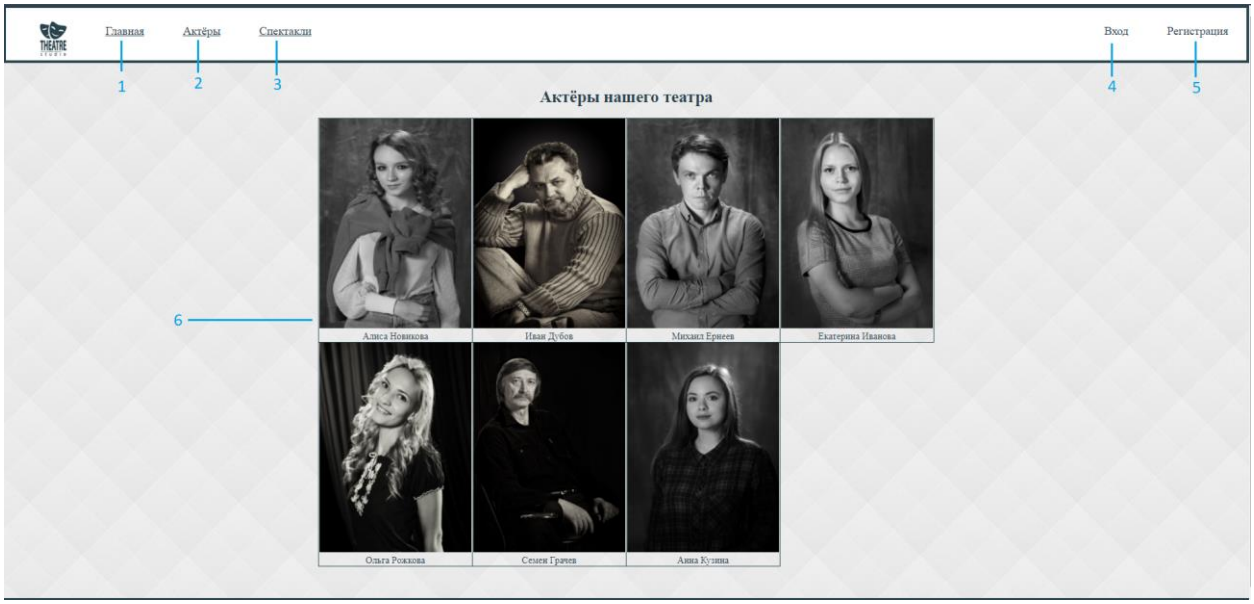

Рисунок 12 – Графический интерфейс страницы с информацией об актерах театра

Рассмотрим основные элементы интерфейса страницы с информацией об актерах театра разработанного приложения, представленные на рисунке 12:

1 – кнопка для перехода на главную страницу;

2 – кнопка для перехода на страницу работы с данными об актерах театра;

3 – кнопка для перехода на страницу работы с данными о спектаклях;

4 – кнопка для перехода на страницу входа в личный кабинет;

5 – кнопка для перехода на страницу регистрации пользователя;

6 – область со перечнем актеров театра (для перехода к странице «Информация об актере» следует нажать на фотографию, либо имя актера).

Графический интерфейс страницы работы с данными о спектаклях представлен на рисунке 13.

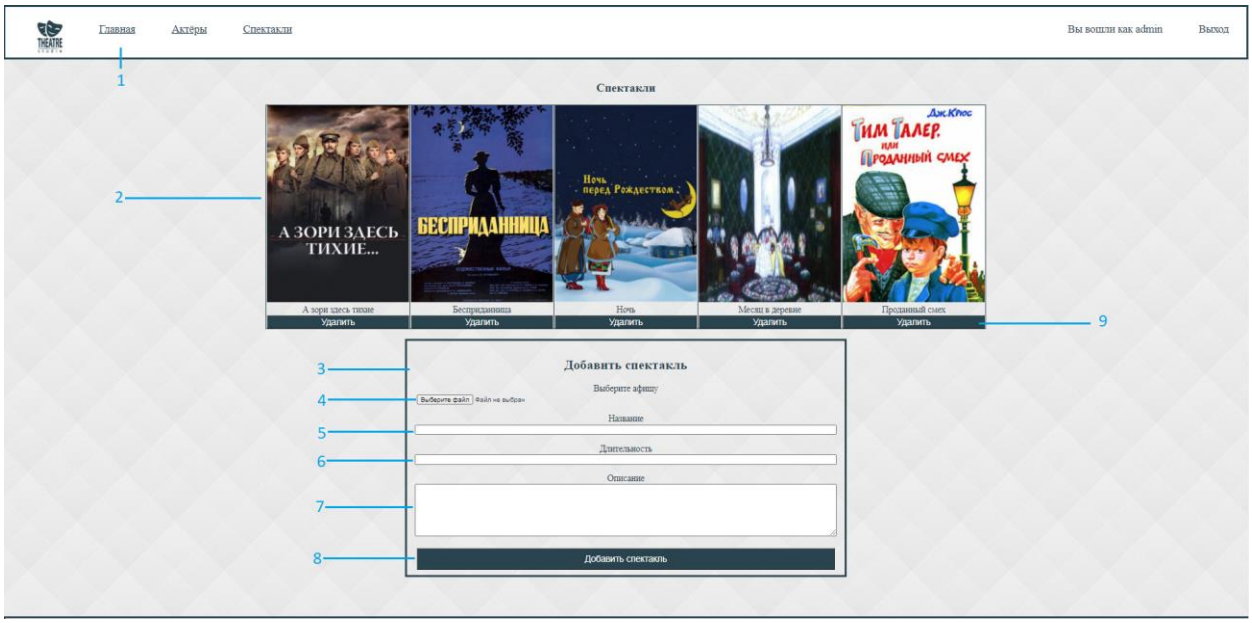

Рисунок 13 – Графический интерфейс окна корректировки строки инвентаризационной описи

Рассмотрим основные элементы интерфейса страницы окна корректировки строки инвентаризационной описи разработанного приложения, представленные на рисунке 13:

1 – область меню;

2 – область с перечнем спектаклей (для перехода к странице «Информация о спектакле» следует нажать на афишу, либо на название спектакля);

3 – окно для добавления нового спектакля;

4 – кнопка для выбора файла-афиши;

5 – текстовое поле для ввода название спектакля;

6 – числовое поле для ввода длительности спектакля;

7 – текстовое поле для ввода информации о спектакле;

8 – кнопка для подтверждения добавления спектакля;

9 – кнопка для удаления выбранного спектакля.

Графический интерфейс страницы бронирования места представлен на рисунке 14.

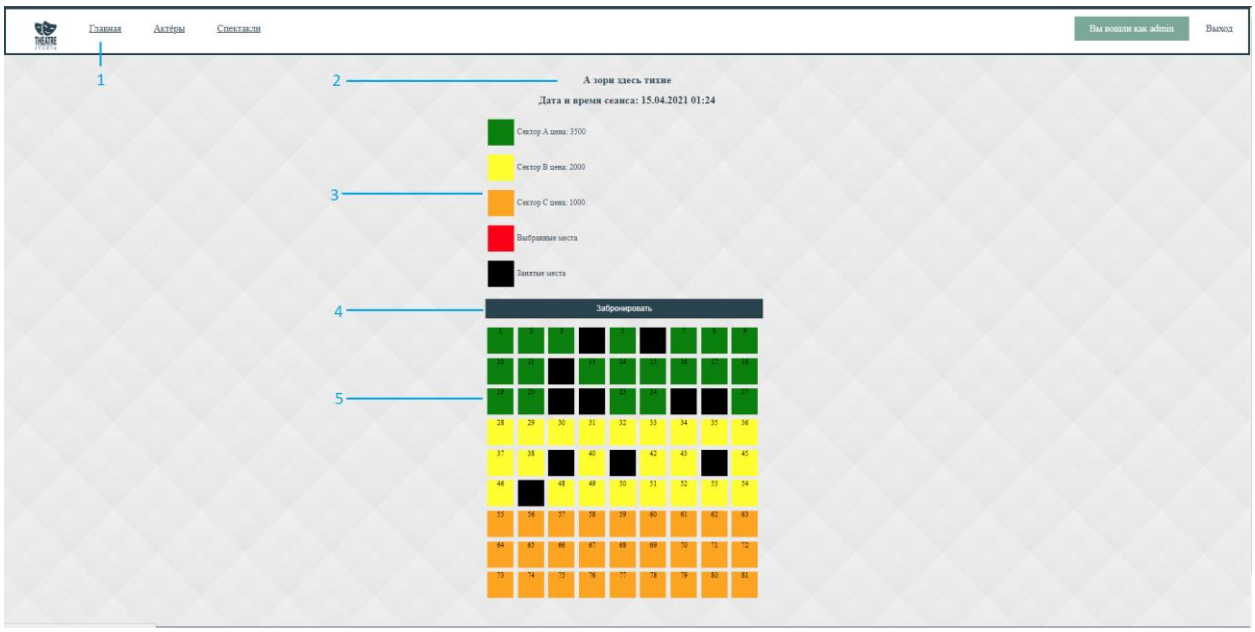

Рисунок 14 – Графический интерфейс страницы бронирования места

Рассмотрим основные элементы интерфейса страницы бронирования места разработанного приложения, представленные на рисунке 14:

1 – область меню;

2 – информация о спектакле;

3 – область легенды секторов и их стоимости;

4 – кнопка для подтверждения бронирования места;

5 – область для выбора места в зале с информацией об уже занятых местах.

Графический интерфейс страницы личного кабинета (забронированные места) представлен на рисунке 15.

Рисунок 15 – Графический интерфейс страницы личного кабинета

Рассмотрим основные элементы интерфейса страницы личного кабинета разработанного приложения, представленные на рисунке 15:

1 – область меню;

2 – таблица с информацией о забронированных местах;

3 – кнопка для отмены брони.

Графический интерфейс страницы регистрации нового пользователя представлен на рисунке 16.

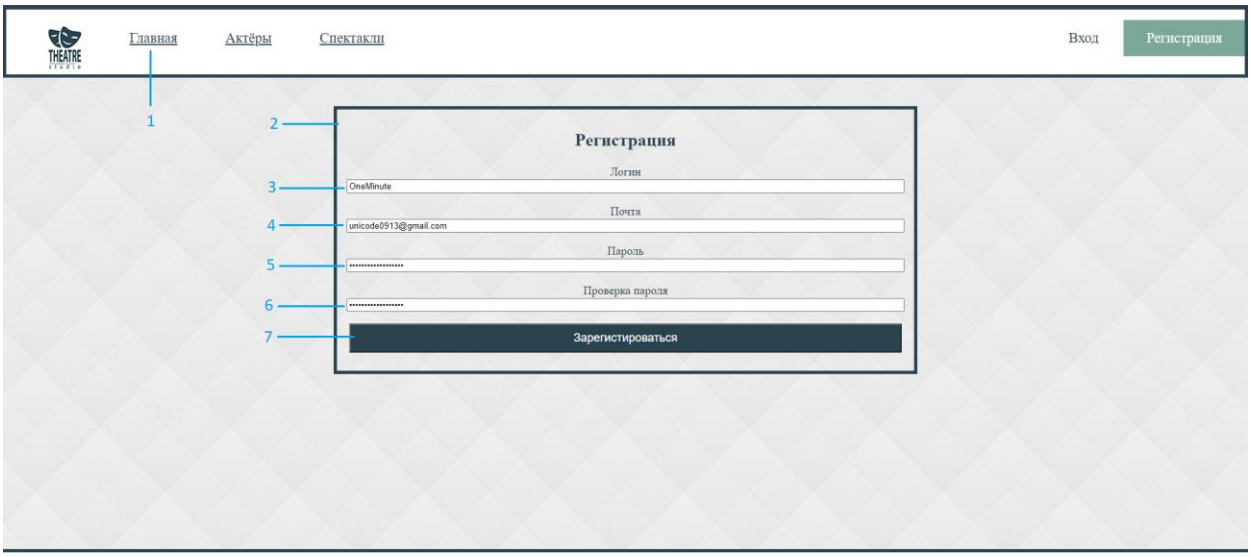

Рисунок 16 – Графический интерфейс страницы регистрации нового пользователя

Рассмотрим основные элементы интерфейса страницы регистрации нового пользователя разработанного приложения, представленные на рисунке 16:

1 – область меню;

2 – таблица с информацией о регистрируемом пользователе;

3 – текстовое поле для ввода имя регистрируемого пользователя;

4 – текстовое поле для ввода адреса электронной почты;

5 – текстовое поле для ввода пароля пользователя;

6 – текстовое поле для ввода подтверждения пароля;

7 – кнопка для подтверждения регистрации.

Оставшиеся окна приложения имеют схожий графический интерфейс.

Спроектированный графический интерфейс пользоватя является простым, удобным и интуитивно понятным.

2.4 Планирование разработки и оценка бюджета

Управление проектами – совокупность процессов по планированию, координации и контролю работ для реализации целей проектов с учетом ограничений на ресурсы, бюджет и требований качества [21].

2.4.1 Планирование проекта

В ходе планирования проекта было выделено 12 задач. Список задач проекта, их виды, длительность и предшественники приведены в таблице 2.

Таблица 2 – Задачи проекта

№	Название	Вид Задачи	Предшественники	Длительность
1	Начало проекта	Веха	-	-
2	Анализ предметной области	Фаза	-	-
3	Просмотр рынка информационных систем	Задача	-	4
4	Сбор информации о существующих информационных системах	Задача	3	6
5	Анализ требований	Фаза	-	-
6	Анализ функциональных требований	Задача	4	12
7	Анализ используемых технологий	Задача	6;7	8
8	Проектирование	Фаза	-	-
9	Проектирование базовой архитектуры	Задача	6;7	4
10	Проектирование логической модели базы данных	Задача	6	2,5
11	Проектирование интерфейса пользователя	Задача	6;7;9	14
12	Реализация	Фаза	-	-
13	Реализация физической модели базы данных	Задача	10	4
14	Реализация серверной части приложения	Задача	13	8
15	Реализация клиентской части приложения	Задача	9;13;11	22
16	Тестирование	Фаза	-	-
17	Функциональное тестирование	Задача	16;17	10
18	Завершение проекта	Фаза	-	-
19	Оформление документации	Задача	17	10
20	Конец проекта	Веха	-	-

В MS Project 2013 создадим проект с начальной датой 01.12.20. Занесем выше перечисленные задачи в созданный проект.

Выделим необходимые ресурсы для выполнения каждой из задач данного проекта. Ресурсы и затраты представлены в таблице 3.

Таблица 3 – Ресурсы проекта

Название ресурса	Тип	Затраты			Затраты на исп.
		Таблица норм	Станд. Ставка	Ставка сверхур.	
Программист 1	Т	А	22000 р./мес	300 р./ч	-
Программист 2	Т	А	18000 р./мес	280 р./ч	-
Аналитик	Т	А	15000 р./мес	250 р./ч	-
Тестировщик	Т	А	15000 р./мес	250 р./ч	-
Компьютер	М	А	15000 р.		-
Принтер	М	А	3000 р.		-
Бумага	М	А	200 р.		-
Интернет	З				-

Распределение ресурсов по задачам представлено в таблице 4.

Таблица 4 – Распределение ресурсов по задачам

Название	Ресурс	Единицы (затраты)	Таблица норм затрат
Начало проекта	Компьютер	4	А
Просмотр рынка информационных систем	Аналитик, Интернет	100	А 250
Сбор информации о существующих информационных системах	Аналитик, Интернет	100	А 300
Анализ функциональных требований	Аналитик, Программист 1	100 100	А А
Анализ используемых технологий	Программист 1, Аналитик	100 30	А А
Проектирование базовой архитектуры	Программист 1	100	А
Проектирование логической модели базы данных	Программист 2	100	А
Проектирование интерфейса пользователя	Программист 1	100	А
Реализация физической модели базы данных	Программист 2	100	А
Реализация серверной части приложения	Программист 2	100	А
Реализация клиентской части приложения	Программист 1, Программист 2	100 100	А А
Функциональное тестирование	Тестировщик, Программист 2	50 100	А А
Оформление документации	Программист 1, Тестировщик, Принтер, Бумага	100 100 1 1	А А А А

Запишем выделенные ресурсы в ранее подготовленный проект. Результаты данного этапа представлены на рисунке 17.

	Название ресурса	Тип	Единицы измерения материалов	Краткое название	Макс. единиц	Стандартная ставка	Ставка сверхурочных	Начисление	Базовый календарь	Добавить новый столбец
1	Программист1	Трудовой		Прог1	100%	22 000,00р./мес	300,00р./ч	Пропорциональное	Стандартный	
2	Программист2	Трудовой		Прог2	100%	18 000,00р./мес	280,00р./ч	Пропорциональное	Стандартный	
3	Аналитик	Трудовой		Аналит	100%	15 000,00р./мес	250,00р./ч	Пропорциональное	Стандартный	
4	Тестировщик	Трудовой		Тестир	100%	15 000,00р./мес	250,00р./ч	Пропорциональное	Стандартный	
5	Компьютер	Материальный	Штука	Комп		15 000,00р.		Пропорциональное		
6	Принтер	Материальный	Штука	Принт		3 000,00р.		Пропорциональное		
7	Бумага	Материальный	Пачка	Бум		200,00р.		Пропорциональное		
8	Интернет	Затраты		Инет				Пропорциональное		

Рисунок 17 – Ресурсы проекта

Прикрепим соответствующие ресурсы к выделенным задачами проекта. После, «Microsoft Office Project» может представить графическую иллюстрацию календарного плана в виде диаграммы «Ганта». Диаграмма «Ганта» представлена на рисунке 18.

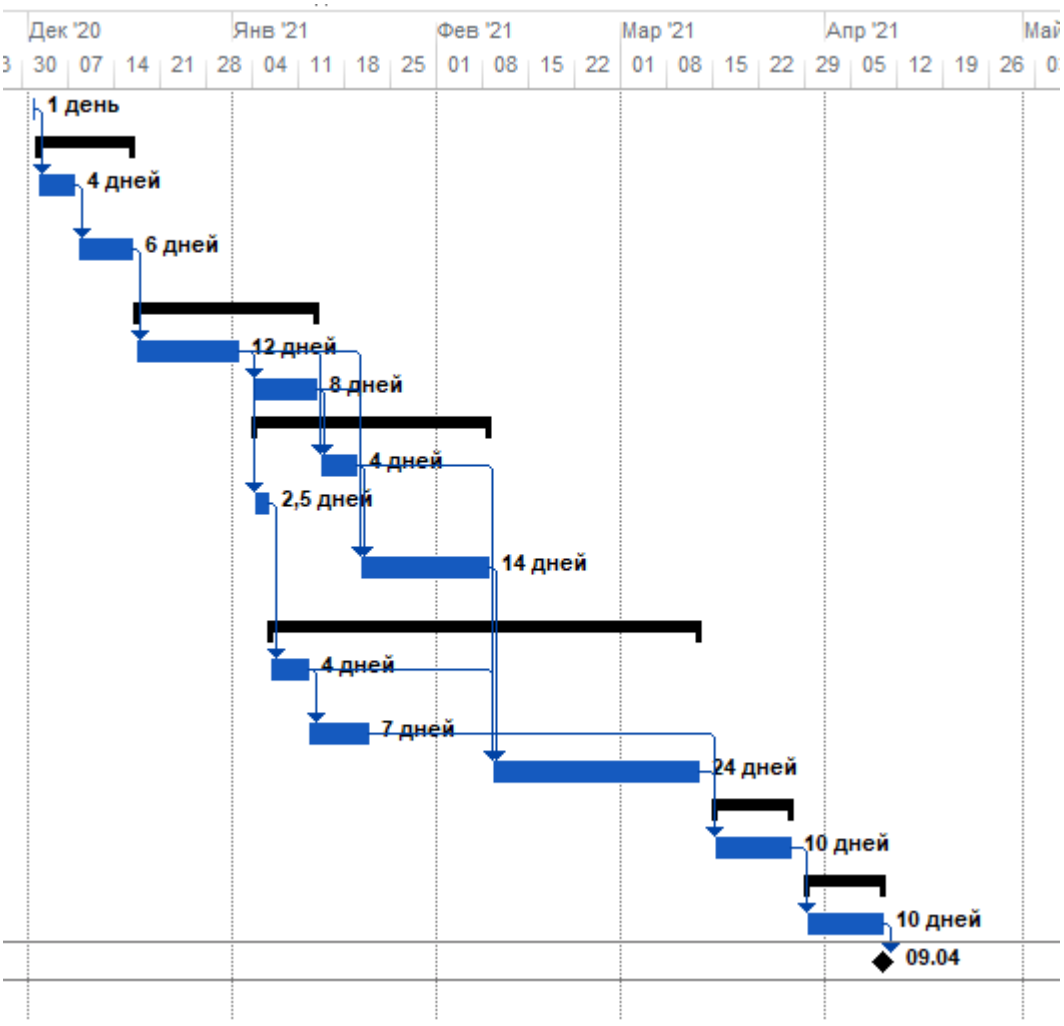

Рисунок 18 – Диаграмма Ганта

Сетевой график рабочего проекта представлен на рисунке 16. Красным цветом на рисунке 19 показан критический путь.

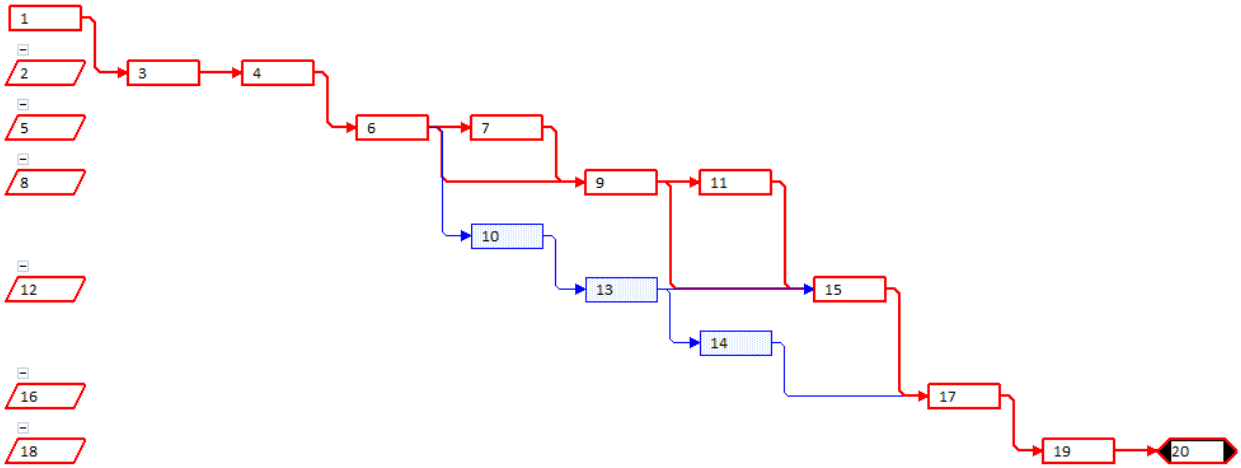

Рисунок 19 – Сетевой график

Графики загрузки трудовых ресурсов программиста 1, программиста 2, аналитика и тестировщика представлены на рисунках 20–23.

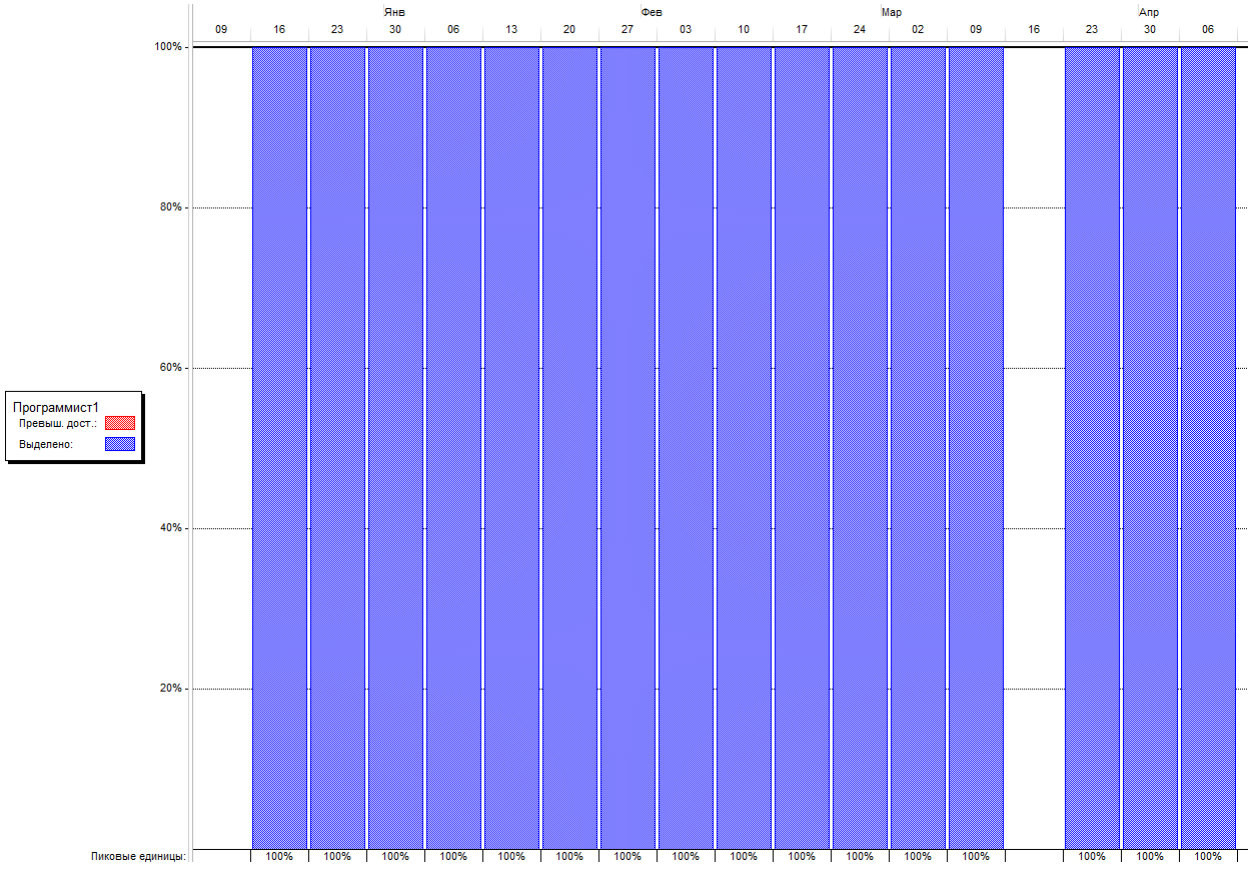

Рисунок 20 – График загрузки программиста 1

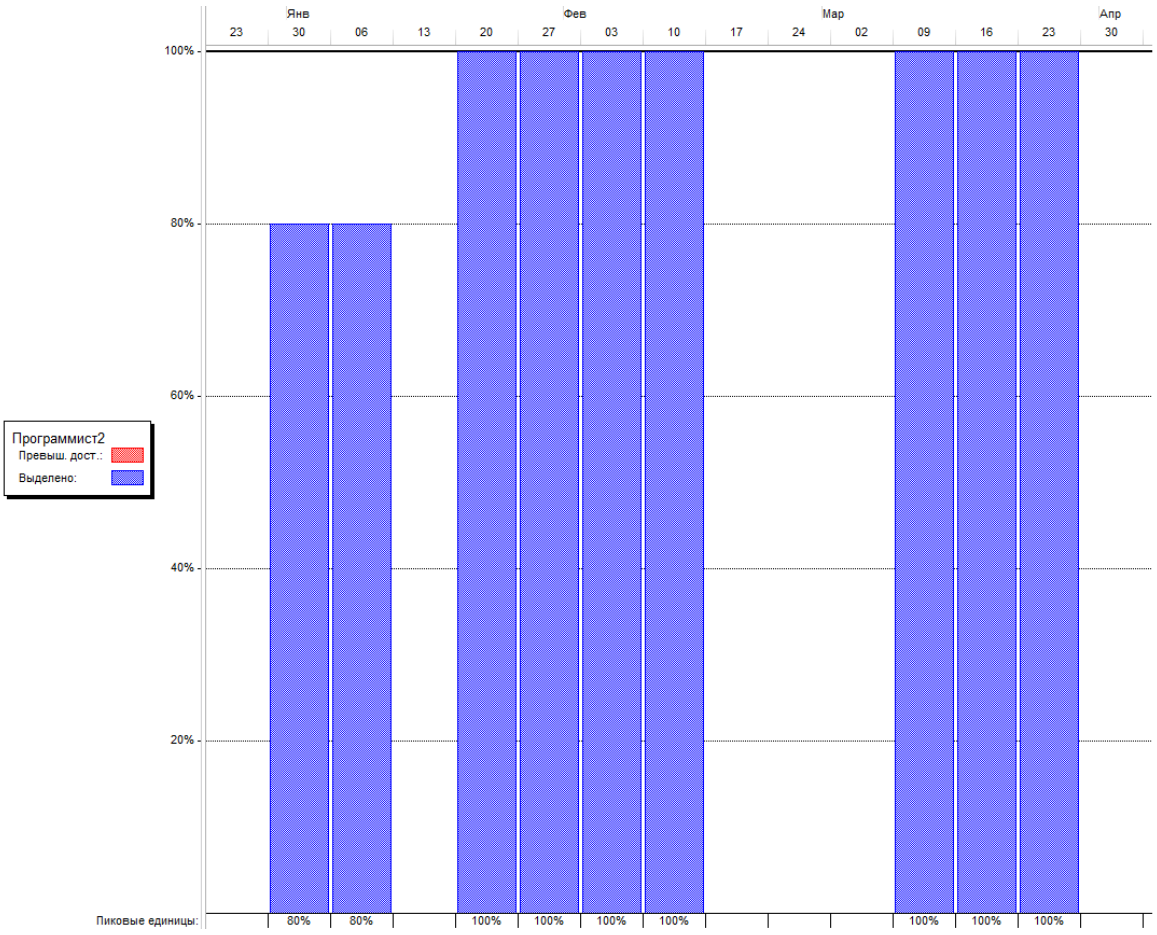

Рисунок 21 – График загрузки программиста 2

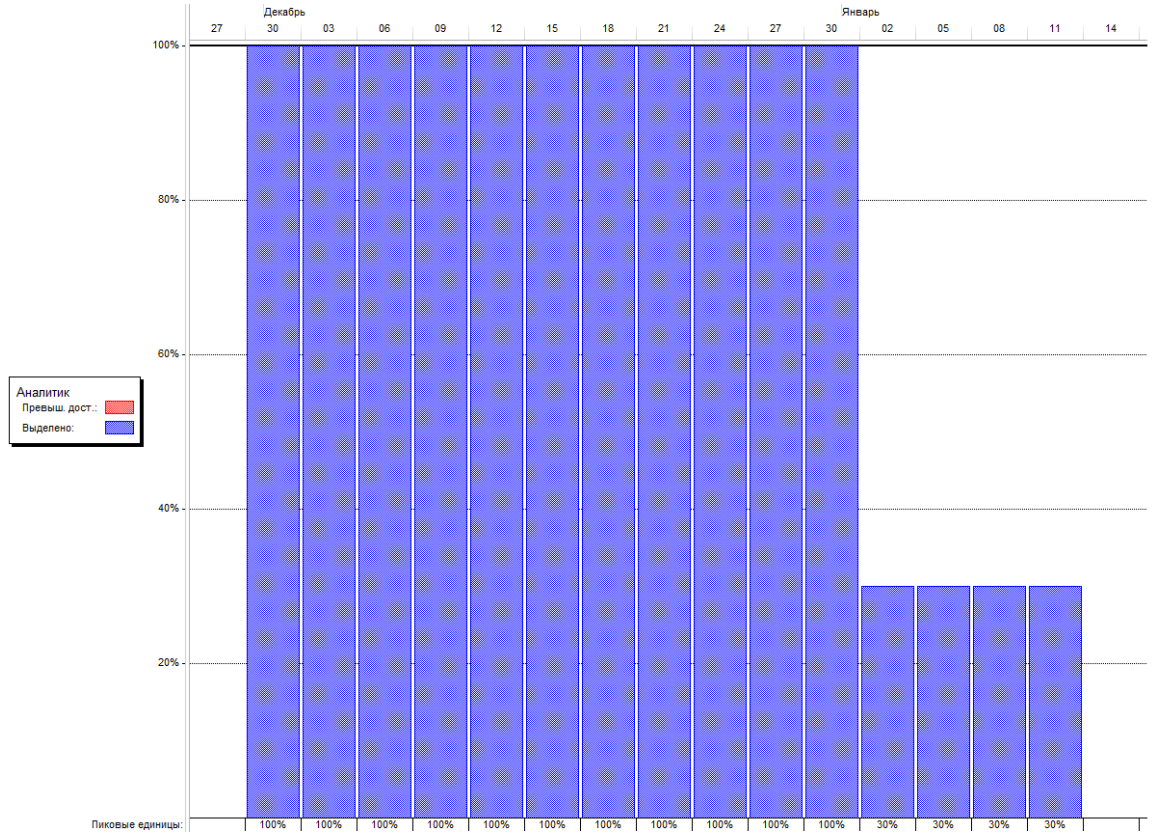

Рисунок 22 – График загрузки аналитика

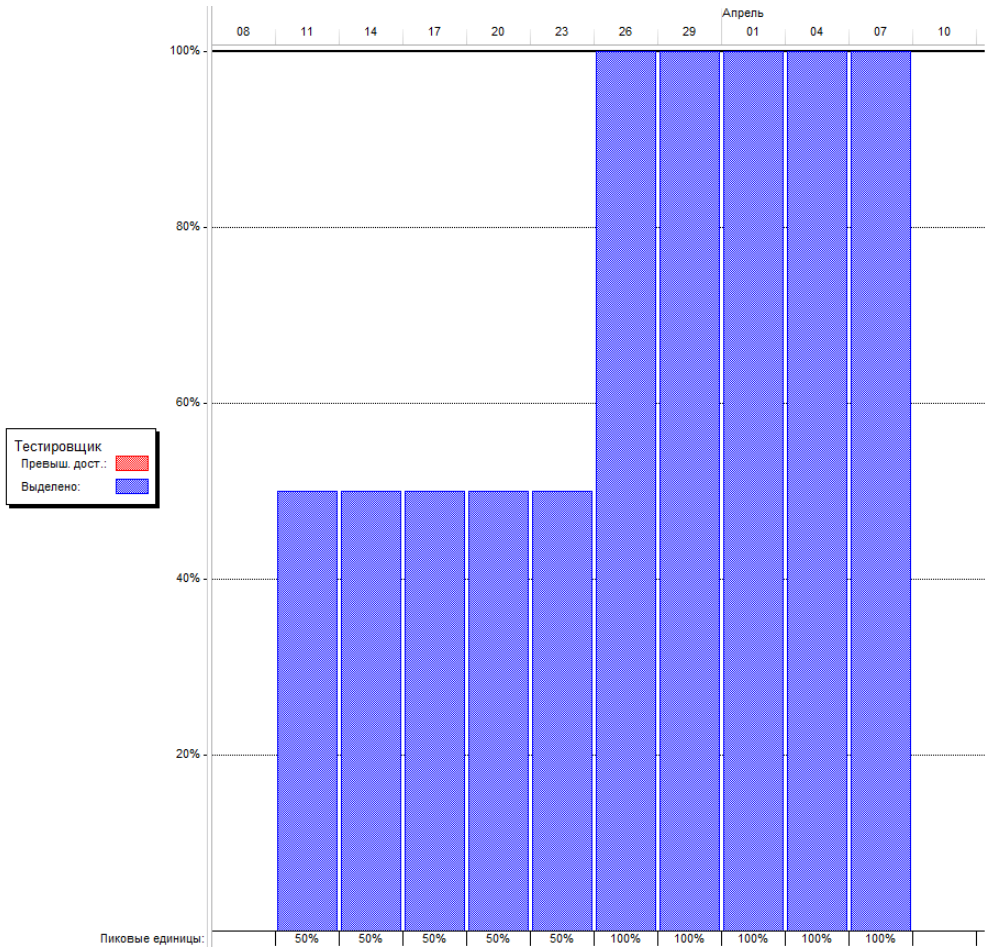

Рисунок 23 – График загрузки тестировщика

На предоставленных выше графиках загрузки трудовых ресурсов показано, что «Программист 1» загружен полностью, «Программист 2» большую часть времени загружен на 100%, аналитик, аналогично, большую часть времени имеет загрузку 100%, тестировщик в загружен на 75%.

Следует отметить, что главную часть работ по анализу, разработке и кодированию программы автоматизированной информационной системы выполняет «Программист 1». «Программист 2» выступает в качестве помощника «Программиста 1».

«Программист 1» занимается кодированием и отрисовкой оболочки web-приложения. «Программист 2» проектирует и кодирует программу, которая относится к серверной части приложения, а именно обработку информации (CGI-программы на языке программирования PHP).

Сводка по проекту представлена на рисунке 24. На сводке указаны длительность в днях, общие затраты, а также график затрат по фазам проекта.

Рисунок 24 – Сводка по проекту

В сводке по проекту показано, что проект был разбит на пять фаз:

- анализ предметной области;
- анализ требований;
- проектирование;
- реализация;
- тестирование.

Также в сводке по проекту обозначены 2 вехи: «Начало проекта» и «Завершение проекта».

Продолжительность проекта составила 94 дня, а его бюджет составил 153 443 рубля.

3 Реализация

3.1 Физическая модель базы данных

Физическая модель строится на основе логической модели данных и учитывает особенности целевой СУБД. В качестве СУБД был выбран продукт MySQL.

При помощи case-средства ERWin Data Modeler была разработана физическая модель, представленная на рисунке 25.

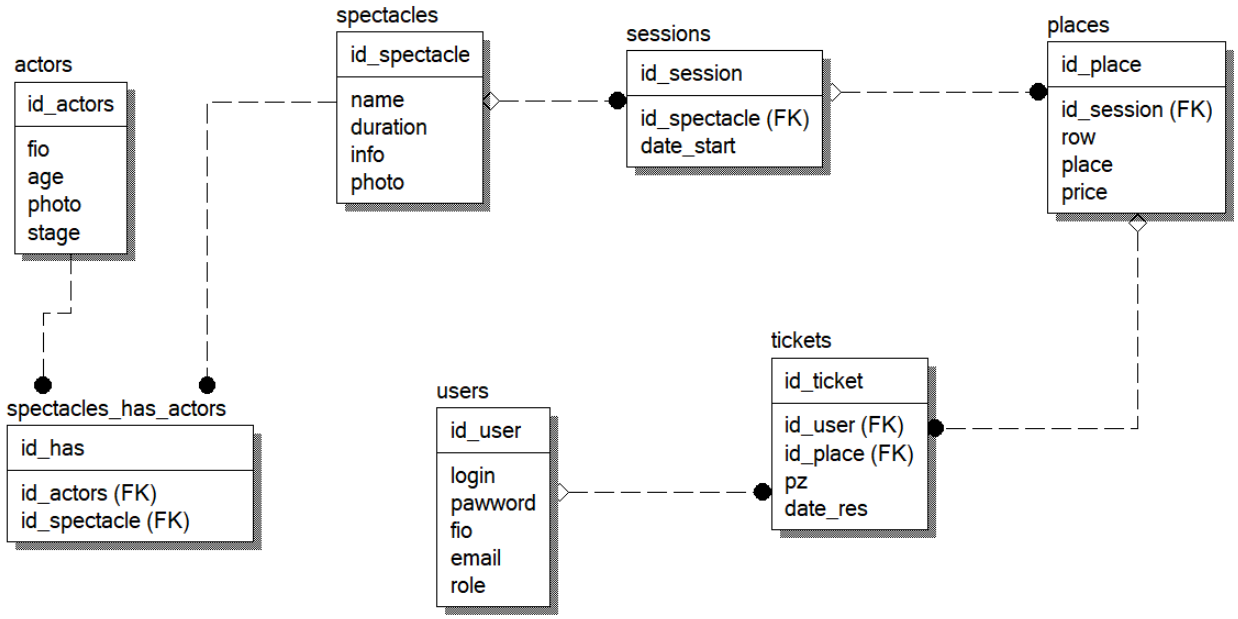

Рисунок 25 – Физическая модель базы данных

Сгенерированный скрипт DDL в ERWin Data Modeler (приложение Г) был выполнен в СУБД MySQL.

3.2 Серверная часть приложения

В ходе разработки клиент-серверного приложения для бронирования и покупки билетов в театр было принято решение описать логику приложения для работы с базой данных в web-приложении серверной части. Серверная часть приложения представляет собой вторую часть приложения на языке программирования PHP и базу данных, которая включает в себя таблицы и представление.

3.2.1 Реализация представлений

Исходный код представления для отображения списка забронированных мест в личном кабинете на языке SQL представлен в листинге 1.

Листинг 1 – Представление для отображения забронированных мест

```
SELECT login,name,to_char(date_start,'dd.mm.yyyy') date_start,row,place,
     price,CASE pz WHEN 1 THEN 'Бронь' WHEN 2 THEN 'Покупка'
     ELSE 'Ошибка статуса' END pz
FROM main.users a
JOIN main.tickets b ON a.id_uesr=b.id_user
JOIN main.places c ON b.id_place=c.id_place
JOIN main.session d ON c.id_session=d.id_session
JOIN main.spectacles e ON d.id_spectacle=e.id_spectacle
WHERE pz in (1,2)
```

Исходный код представления для отображения списка спектаклей и играющих в них актеров на языке SQL представлен в листинге 2.

Листинг 2 – Представление для отображения списка спектаклей и играющих в них актеров

```
SELECT name,duration,info,a.photo,fio,age,b.photo,stage
FROM spectacles a
LEFT OUTER JOIN spectacles_has_actors b ON a.id_spectacle=b.id_spectacle
LEFT OUTER JOIN actors c ON b.id_actor=c.id_actor
```

Результат обращения к представлениям соответствует ожидаемым результатам, вследствие чего можно сделать вывод, что представления работают корректно.

3.3 Модель реализации клиентского приложения

Модель реализации описывает, как реализуются в виде компонентов элементы модели проектирования.

Для отображения реализации используется диаграмма компонентов. Диаграмма компонентов представлена на рисунке 26.

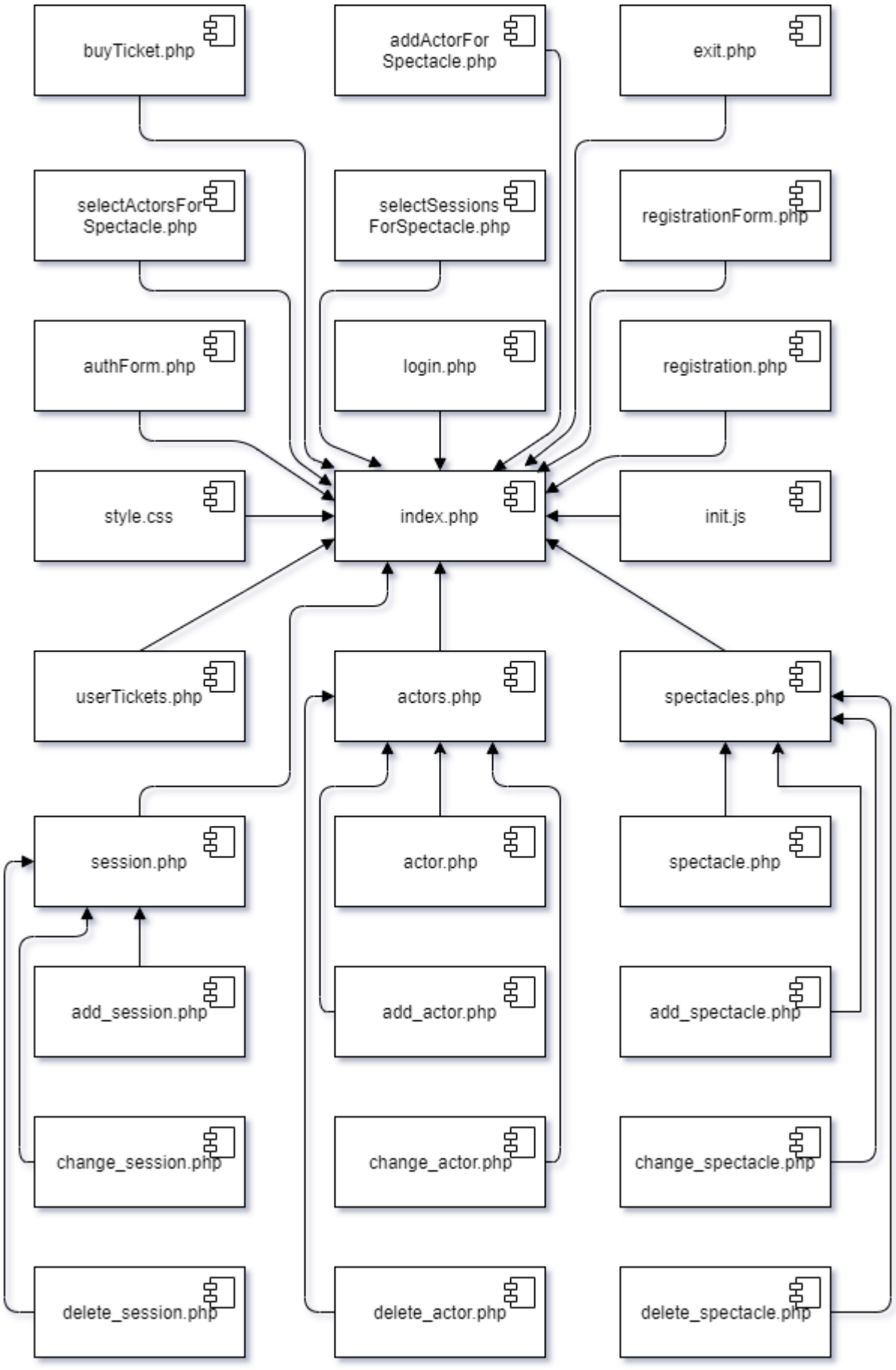

Рисунок 26 – Диаграмма компонентов

Следует сказать, что на диаграмме компонентов основным является файл «index.php». В данном файле находится точка входа в программу и запуск web-приложения начинается именно с него. Точка входа в программу на стороне JavaScript располагается в файле «init.js». Таблица стилей находится в файле «style.css».

Исходный код клиентского приложения на языках программирования JavaScript и PHP приведен в приложении Б.

Одним из способов моделирования статического вида системы с точки зрения развертывания является диаграмма развертывания.

Диаграмма развертывания предназначена для визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения. При этом представляются только компоненты-экземпляры программы, являющиеся исполнимыми файлами или динамическими библиотеками. Те компоненты, которые не используются на этапе исполнения, на диаграмме развертывания не показываются.

Диаграмма развертывания содержит графические изображения процессоров, устройств, процессов и связей между ними. В отличие от диаграмм логического представления, диаграмма развертывания является единой для системы в целом, поскольку должна всецело отражать особенности ее реализации. Эта диаграмма, по сути, завершает процесс ООАП для конкретной программной системы и ее разработка, как правило, является последним этапом спецификации модели.

Диаграмма развертывания представлена на рисунке 27.

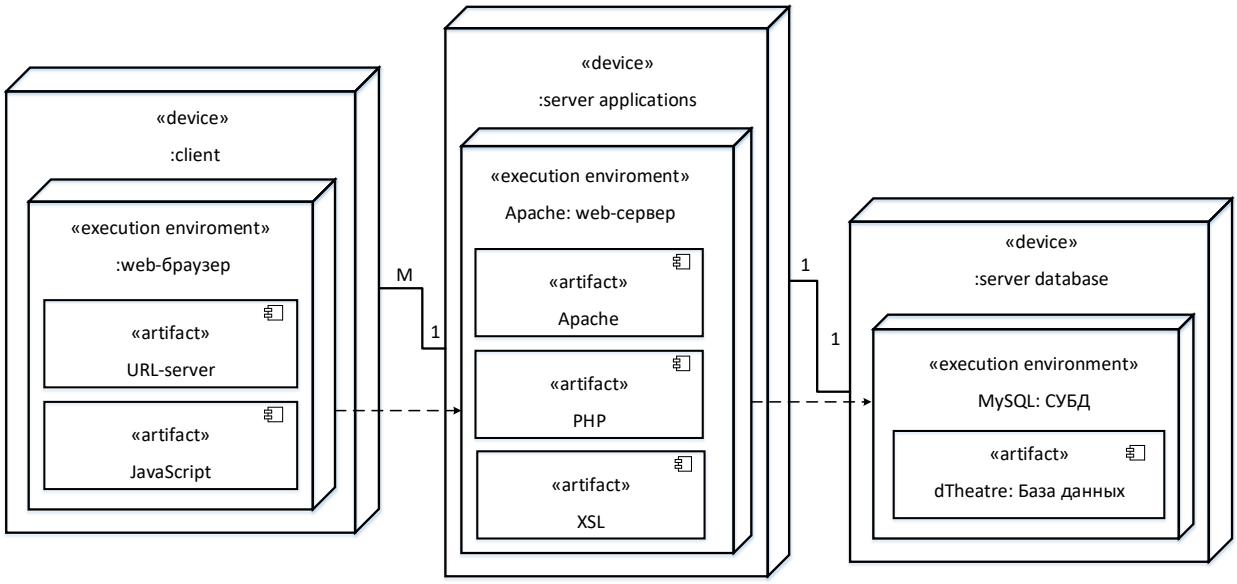

Рисунок 27 – Диаграмма развертывания

На рисунке 27 обозначено, что на машине клиента должен присутствовать веб-браузер с модулем Adobe Flash для запуска JavaScript.

После ввода URL-адреса сервера, web-браузер клиента обратится к веб-серверу, который расположен на сервере приложений, с целью запуска клиентского веб-приложения. После определенных действий пользователя клиентское веб-приложение, написанное на языке программирования JavaScript, обратится серверному веб-приложению, написанному на языке программирования PHP, для обмена информацией. Далее серверное веб-приложение обратится к системе управления базой данных, которая располагается на сервере хранилища данных, с целью выполнения некоторых запросов к данным и объектам БД.

4 Контроль качества программного обеспечения

В процессе разработки программного продукта важно соблюдать ряд требований, чтобы обеспечить качество программы. Обеспечение качества включает в себя список характеристик и свойств программы, которые должны соблюдаться во время разработки программы и выполняться при ее работе. Обеспечение качества производится на всех этапах развития программного обеспечения: от начала разработки, до выпуска программы в пользование.

Контроль качества - означает проверку выполнения требований к программному обеспечению и общую оценку разработки. Основная задача контроля качества – поиск ошибок в программе и их дальнейшее исправление.

Обеспечение контроля и качества программного обеспечения определяется в соответствии с различными требованиями в зависимости от особенностей проекта [22].

Если программа соответствует потребностям пользователей и ожиданиям разработчиков, то можно сказать, что программа успешно прошла оценку качества и готова к выпуску.

В данной выпускной квалификационной работе была выбрана методология функционального тестирования.

4.1 Функциональное тестирование

Функциональное тестирование относится к категории тестирования «черного ящика». Каждая функция программы тестируется ввода её исходных данных и анализа выходных. При этом внутренняя структура программы учитывается очень редко [23].

Функциональное тестирование предполагает составление плана тестирования, который строится на основе вариантов использования и включает в себя описание тестов и их результаты.

План тестирования приведен в таблице 5.

Таблица 5 – План тестирования

Вариант использования	Тест	Результат
Авторизоваться в системе	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим верный логин и пароль.	Тест выполнен. Вход в систему разрешен.
Авторизоваться в системе	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим неверный логин или пароль.	Тест выполнен. Вход в систему запрещен (рисунок В.1).
Зарегистрироваться	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Регистрация», заводим логин, пароль, электронную почту. Затем нажимаем кнопку «Зарегистрироваться».	Тест выполнен. Новый пользователь успешно добавлен в систему (рисунок В.2).
Зарегистрироваться	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Регистрация», заводим логин, который уже присутствует в системе, пароль, электронную почту. Затем нажимаем кнопку «Зарегистрироваться».	Тест выполнен. Системы вывела ошибку (рисунок В.3).
Забронировать место	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим верный логин и пароль. Далее выбираем конкретный сеанс. Выбираем незанятое место в зале. Затем нажимает кнопку «Забронировать».	Тест выполнен. Место успешно забронировано (рисунок В.4).
Купить билет	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим верный логин и пароль. Далее выбираем конкретный сеанс. Выбираем незанятое место в зале. Затем нажимает кнопку «Забронировать». Заходим на электронную почту и открываем письмо с предложение об оплате забронированного билета. Далее переходим на страницу оплаты банка. Подтверждаем операцию оплаты.	Тест выполнен. Билет успешно куплен и выслан на электронную почту.

Продолжение таблицы 5

Вариант использования	Тест	Результат
Отменить бронь	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим верный логин и пароль. Нажимаем кнопку «Вы вошли как, …» (переход в личный кабинет). Далее выбираем конкретный сеанс с которого будет снята бронь. Затем нажимаем кнопку «Снять бронь».	Тест выполнен. Система сняла бронь с забронированного места (рисунок В.5).
Добавить нового актера	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Актеры». На открывшейся странице появилось окно «Добавить актера». Вводим требуемую информацию об актере: фото, имя, фамилию, возраст, стаж, биографию. Затем нажимаем кнопку «Добавить актера».	Тест выполнен. Новый актер успешно добавлен в систему (рисунок В.6, рисунок В.7).
Изменить данные об актере	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Актеры». Система отобразит страницу с перечнем актеров. Выбираем из перечня нужного нам актера и изменяем у него возраст на 35. Затем нажимаем кнопку «Изменить данные актера».	Тест выполнен. Новая информация об актере успешно сохранена.
Удалить актера	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Актеры». Система отобразит страницу с перечнем актеров. Выбираем из перечня нужного нам актера и нажимаем кнопку «Удалить».	Тест выполнен. Выбранный актер успешно удален из системы.

Продолжение таблицы 5

Вариант использования	Тест	Результат
Добавить спектакль	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Спектакли». На открывшейся странице появилось окно «Добавить спектакль». Вводим требуемую информацию о новом спектакле: афишу, название, длительность, описание. Затем нажимаем кнопку «Добавить спектакль».	Тест выполнен. Новый спектакль успешно добавлен в систему (рисунок В.8, В.9, В.10).
Изменить данные о спектакле	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Спектакли». Система отобразит страницу с перечнем спектаклей. Выбираем из перечня нужный нам спектакль и изменяем у него длительность на 1 час. Затем нажимаем кнопку «Изменить запись».	Тест выполнен. Новая информация о спектакле успешно сохранена.
Удалить спектакль	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Спектакли». Система отобразит страницу с перечнем спектаклей. Выбираем из перечня нужный нам спектакль и нажимаем кнопку «Удалить».	Тест выполнен. Выбранный спектакль успешно удален из системы.
Добавить сеанс	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Спектакли». Система отобразит страницу с перечнем спектаклей. Выбираем из перечня нужный нам спектакль. Затем система откроет страницу с информацией о выбранном спектакле и окно для работы с сеансами. Заводим информацию о новом сеансе и нажимаем кнопку «Добавить запись».	Тест выполнен. Новый сеанс успешно добавлен в систему (рисунок В.11).

Окончание таблицы 5

Вариант использования	Тест	Результат
Удалить сеанс	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Спектакли». Система отобразит страницу с перечнем спектаклей. Выбираем из перечня нужный нам спектакль. Затем система откроет страницу с информацией о выбранном спектакле и окно для работы с сеансами. Выбираем из списка сеанс, который требуется удалить. Далее нажимаем кнопку «Удалить запись».	Тест выполнен. Выбранный сеанс успешно удален из системы (рисунок В.12).
Прикрепить актера к спектаклю	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Спектакли». Система отобразит страницу с перечнем спектаклей. Выбираем из перечня нужный нам спектакль. Затем система откроет страницу с информацией о выбранном спектакле. В области «Актеры» в специальном поле выбираем нужного актера и нажимаем кнопку «Добавить актера».	Тест выполнен. Выбранный актер успешно прикреплен к спектаклю (рисунок В.13).
Удалить актера из спектакля	Заходим в web-приложение путем ввода URL-адреса в браузере. На появившейся странице нажимаем кнопку «Войти», заводим логин и пароль администратора системы. Затем нажимаем в меню кнопку «Спектакли». Система отобразит страницу с перечнем спектаклей. Выбираем из перечня нужный нам спектакль. Затем система откроет страницу с информацией о выбранном спектакле. В области «Актеры» выбираем нужного актера и нажимаем кнопку «Удалить актера».	Тест выполнен. Выбранный актер успешно удален из спектакля (рисунок В.14).

На основе анализа результатов тестирования можно сделать вывод, что разработанная информационная система по продаже билетов в театр работает корректно.

Заключение

При выполнении выпускной квалификационной работы была разработана АИС по продаже билетов в театр, состоящая из приложения, располагающегося на станции клиента, приложения, располагающегося на серверной машине, и серверной базы данных.

Было проведено логическое и физическое проектирование базы данных, проектирование графического интерфейса пользователя.

Клиентская часть приложения реализована при помощи объектно-ориентированого языка программирвоания JavaScript. Серверная часть приложения реализована при помощи языка программирования PHP в среде разработки «Visual Studio Code». В системе управления базой данных при помощи языка SQL была реализована база данных, которая включает в себя объекты: таблицы и представления. База данных была реализована в СУБД «MySQL 5.7.31».

В рамках данной выпускной квалификационной работы было проведено тестирование. На основе результатов тестирования можно сделать вывод, что разработанное программное обеспечение по продаже билетов в театр работает исправно.

В ходе выполнения работы также было выполнено планирование проекта. Длительность выполнения проекта составила 94 дня, а бюджет равен 153 443 рубля. Оценка сроков и бюджета говорит о том, что данная разработка экономически выгодна.

Таким образом, в результате выполнения выпускной квалификационной работы было разработано программное обеспечение, отвечающее всем требованиям технического задания.

Список использованных источников

1. Распоряжение Правительства Российской Федерации от 20 октября 2010 г. N 1815-р г. Москва «О государственной программе Российской Федерации Информационное общество (2011-2020 годы)» (ред. от 29.03.2019 N 356-24).
2. Технологии для кинотеатра. Как устроена билетная система: статья // ООО «ОНЛАЙН СИНЕМАПЛЕКС1С»: сайт. 2014. [Электронный ресурс]. URL: https://cinemaplex.ru/2016/09/21/tehnologii-dlya-kinoteatra-kak-ustroena-biletnaya -sistema.html (дата обращения: 01.05.2021).
3. Почему покупать билеты в кинотеатр через интернет выгодно: статья // Яндекс.Дзен: сайт. 2015. [Электронный ресурс]. URL: https://zen.yandex.ru/media/kin0manka/pochemu-pokupat-bilety-v-kinoteatr-cherez-internet-vygodno-5e7b8e436952617566befd73 (дата обращения: 01.05.2021).
4. «Радарио»: описание программного продукта // Radario: сайт. 2012. [Электронный ресурс]. URL: https://business.radario.ru/ (дата обращения: 01.05.2021).
5. «Единое поле»: описание программного продукта // Единое поле: сайт. 2014. [Электронный ресурс]. URL: https://edinoepole.ru/ (дата обращения: 01.05.2021).
6. «TicketsCloud»: описание программного продукта // Tickets Cloud: сайт. 2013. [Электронный ресурс]. URL: https://ticketscloud.com/ (дата обращения: 01.05.2021).
7. «INTICKETS»: описание программного продукта // INTICKETS.RU: сайт. 2009. [Электронный ресурс]. URL: https://intickets.ru/ (дата обращения: 01.05.2021).
8. Крачтен Ф. Введение в Rational Unified Process / Ф. Крачтен. – М.: Вильямс, 2002. – 240 с.
9. Флэнаган Д. JavaScript. Подробное руководство, 6-е издание. / Д. Флэнаган. – Пер. с англ. – СПб: Символ-Плюс, 2012. – 1080 с.

10. Бэнкс А. React и Redux: функциональная веб-разработка / А. Бэнкс, Е. Порселло. – СПб.: Питер, 2018. – 336 с.

11. Представляем Sencha Ext JS: статья // Хабрахабр: сайт. 2006. [Электронный ресурс]. URL: https://habrahabr.ru/post/225169/ (дата обращения: 01.05.2021).

12. Общий взгляд на AJAX фреймворк ExtJS: статья // Интерфейс: сайт. 2010. [Электронный ресурс]. URL: http://www.interface.ru/home.asp?artid=22109 (дата обращения: 01.05.2021).

13. Гольцман В. MySQL 5.0. Библиотека программиста / В. Гольцман. – СПб.: Питер, 2010. – 253 с.

14. Visual Studio Code: описание программного продукта // Microsoft: сайт. 2007. [Электронный ресурс]. URL: https://visualstudio.microsoft.com/ru/ (дата обращения: 01.05.2021).

15. Мишель Е. Дэвис. Изучаем PHP и MySQL / Е. Мишель Дэвис. – Пер. с англ. – СПб.: Символ-Плюс, 2008. – 448 с.

16. Валиков А. Технология XSLT / А. Валиков. – СПб.: БХВ-Петербург, 2002. – 79 с.

17. Арлоу Д. UML 2 и Унифицированный процесс / Д. Арлоу, А. Нейштадт. – СПб.: Символ-Плюс, 2007. – 624 с.

18. Пайлон Д. UML 2 для программистов / Д. Пайлон, Н.Питмен. – СПб.: Питер, 2012. – 240 с.

19. Коналлен Дж. Разработка Web-приложений с использованием UML: пер. с англ. / Дж. Коналлен. – М.: Издательский дом «Вильямс», 2001. – 288 с.

20. Конноли Т. Базы данных: Проектирование, реализация и сопровождение. Теория и практика. Второе издание. / Т. Конноли, К. Бегг, А. Страчан. – М.: Вильямс, 2000. – 1139 с.

21. Новиков Д.А. Управление проектами: Организационные механизмы / Д.А. Новиков. – М.: ПМСОФТ, 2007. – 140 с.

22. Котляров В.П. Основы тестирования программного обеспечения / В.П. Котляров, Т.В. Коликова. – М.: Бином. Лаборатория знаний, 2006. – 285 с.

23. Канер С. Тестирование программного обеспечения / С. Канер, Д. Фолк, Е.К. Нгуен. – К.: ДиаСофт, 2001. – 544 с.

Глоссарий

Приложение А

(справочное)

XML (Extensible Markup Language) – это расширяемый язык разметки, разработанный специально для размещения информации в World Wide Web.

AJAX (Asynchronous JavaScript and XML) – подход к построению интерактивных пользовательских интерфейсов веб-приложений, заключающийся в «фоновом» обмене данными браузера с веб-сервером.

JSON (JavaScript Object Notation) – текстовый формат обмена данными, основанный на JavaScript.

Фреймворк (Framework) – программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

URL-адрес – это специальный и индивидуальный адрес различных интернет ресурсов.

CSS (Cascading Style Sheets) – это каскадные таблицы стилей, используются для описания внешнего вида документа, написанного языком разметки.

DDL (Data Definition Language) – язык описания данных, это семейство компьютерных языков, используемых в компьютерных программах для описания структуры баз данных.

Приложение Б

КОД ПРОГРАММЫ

(рекомендуемое)

Листинг Б.1 – Исходный код файла index.php

```
<?php
session_start();
if (isset($_COOKIE['token'])) {
	$token = $_COOKIE['token'];
	ConnectDB();
	$res = $mysqli->query("SELECT * FROM users WHERE token = '$token'");
	if ($res->num_rows == 1) {
		$user = $res->fetch_object();
		LogIn($user->id);
	}
}


function ConnectDB()
{
	global $mysqli;
	$mysqli = new mysqli('localhost', 'root', '', 'theatre'); $mysqli-
>query('SET CHARACTER SET utf8');
}

function LogIn($userid)
{
	global $mysqli;
	$_SESSION['userid'] = $userid;
	$token = sprintf('%08x%08x%08x%08x', rand(), rand(), rand(), rand());
	setcookie('token', $token, time() + 3600 * 24 * 30);
	$res = $mysqli->query("UPDATE users SET token = '$token' WHERE id =
'$userid'");
}
?>

<!DOCTYPE html>
<html>

<head>
	<meta charset="utf-8">
	<title> Театр </title>
	<link href="style.css" rel="stylesheet" type="text/css">
</head>

<body>
	<div class="header">
		<a class="logo" href="index.php"><img src="logoT.jpg" alt=""></a>
		<div class="top-menu">
			<ul>
				<li><a href="index.php">Главная</a></li>
				<li><a href="pages/actors.php">Актёры</a></li>
				<li><a href="pages/spectacles.php">Спектакли</a></li>
				<li><a href="pages/session.php">Сеансы</a></li>
				<div class="auth">
					<?php
					if (isset($_SESSION['userid'])) {
						$userid = $_SESSION['userid'];
```

```
                                    $mysqli = new mysqli('localhost', 'root',
'', 'theatre'); $mysqli->query('SET CHARACTER SET utf8');

                                    $res = $mysqli->query("SELECT login From
users WHERE id='$userid'");
                                    $user = $res->fetch_object();
                                    $login = $user->login;

                                    echo                                    "<li><a
href='pages/userTickets.php'>Вы вошли как $login </a></li>
                              <li><a href='exit.php'>Выход</a></li>";
                                    } else {
                                    echo                                    "<li><a
href='authForm.php'>Вход</a></li>
                              <li><a
href='registrationForm.php'>Регистрация</a></li>";
                                    }
                                    ?>
                        </ul>
                  </div>
            </div>
            </div>
            <div class="content">
                  <?php
                  echo "<h2>О театре</h2>";
                  echo            "<div            class='about'><table><tr><td><img
src='Files/theatreHouse.png' alt=''><br></td><td><P>Первый театральный сезон
открылся 27 ноября 1993 премьерой спектакля «Приключения Буратино в Стране
Дураков» на сцене ДК «Современник». Основа труппы театра — участники Народно-
го театра юного зрителя при ДК «Современник» (руководитель Лидия Михайловна
Ершова с 1971 г.). Собственное здание (бывший киноклуб «Дружба») театр полу-
чил в декабре 1995 г. и вселился в него, не ожидая
реконструкции.</p></td></tr></table>

                  <p>В ноябре 2003 года завершились ремонт и реконструкция театра, в
результате которой к основному зданию был сооружён пристрой, позволивший ор-
ганизовать закулисное пространство и получить подсобные помещения. Зрительный
зал театра рассчитан на 81 место.</p>

                  <p> Ежегодно театр выпускает 5-6 премьер, каждая из которых стано-
вится заметным событием в культурной жизни города. За 27 творческих сезонов
было поставлено более 170 спектаклей и представлений для детей и взрос-
лых.</p></div>";
                  echo "<h2>Все сеансы</h2>";
                  $mysqli = new mysqli('localhost', 'root', '', 'theatre'); $mysqli-
>query('SET CHARACTER SET utf8');
                  $res     =     $mysqli->query("SELECT     sessions.id     as     id,
DATE_FORMAT(sessions.date, '%d.%m.%Y %H:%i') as date, spectacles.name as name
FROM sessions INNER JOIN spectacles ON spectacles.id=sessions.spectacle_id
ORDER BY date");
                  echo "<div id= allSessions>";
                  echo "<table>";

                  echo "<tr><th>Спектакль</th><th>Дата сеанса</th></tr>";
                  $indexRow = 0;
                  while ($session =  $res->fetch_object()) {
```

```
                $id = $session->id;
                $name = $session->name;
                $date = $session->date;
                echo                "<tr><td><a                id='actor'
href='pages/session.php?session_id=$id'>$name</a></td><td><a        id='actor'
href='pages/session.php?session_id=$id'>$date</a></td></tr>";
            }
            echo "</tr></table></div>";
            ?>
            <div class='mainPage'>
            </div>
      </div>
      <div class="footer">
               г. Пенза телефон 65-55-21<br>
            Горячая линия
      </div>
</body>

</html>
```

Листинг Б.2 – Исходный код файла init.js

```
var actorsForSpectacle = [];
Ext.onReady(function(){

    store = new Ext.data.JsonStore({
            url:'../selectActorsForSpectacle.php',
            totalProperty: 'totalCount',
            root: 'items',
            fields:[
            {name:'id'},
                  {name:'name'},
                  {name:'surname'},
                  {name:'age'},
                  {name:'experience'}
            ],
            remoteSort:true
      });

    store.load();

    var cm = new Ext.grid.ColumnModel({
            columns:[
                  {header:"<center>id</center>",  align:"center",  width:144,
dataIndex:'id'},
                  {header:"<center>Фамилия</center>",align:"center",
width:217, dataIndex:'surname'},
            {header:"<center>Имя</center>",align:"center", width:217, dataIn-
dex:'name'},
            {header:"<center>Возраст</center>",align:"center", width:144, da-
taIndex:'age'},
            {header:"<center>Стаж</center>",align:"center",  width:144,  da-
taIndex:'experience'},
            ]
      });

    var selectModel = new Ext.grid.CheckboxSelectionModel({
            width:25,
            header:'<div style="background:#F0F1F3; height:90%;></div>',
            listeners: {
```

```
                rowselect: function(sm, row, rec) {
                // строка grid выбрана
                      id = rec.data.id;
                },
                rowdeselect: function(sm, row, rec) {
                // не выбрана ни одна строка
                      id = '';
                }
            }
      });

    grid = new Ext.grid.EditorGridPanel({
        title: 'Выберите актеров, которые будут играть в спектакле',
            store : store,
            cm          : cm,
        height    : 500,
        sm: selectModel,
            enableColLock: false,
            loadMask:{msg:'Идет загрузка данных...'},
            maskDisabled: false,
            loadMask: true,
            collapsible: false,
            frame:true
      });

    grid.render('tabs1');
});
```

Листинг Б.3 – Исходный код файла style.css

```
html {
    height: 100%;
}

body {
    margin: 0;
    color: black;
    /* Растягиваем body по высоте html */
    min-height: 100%;
    display: grid;
    grid-template-rows: auto 1fr auto;
}

.formSearch {
    color: #28444f;
    float: right;
    font-size: 18px;
}

.formSearch button {
    border: 1px solid #28444f;
    background-color: #27263D;
    color: #28444f;
}

.formSearch button:hover {
    border: 1px solid red;
    color: #7aa899;
}

.selection {
    background-color: #7aa899;
```

```
}

.top-menu {
    height: 100px;
    background-color: #ffffff;
    border: 5px solid #28444f;
    text-align: left;
}

body {
    background-image: url(cell.png);
}

.top-menu ul li {
    display: inline-block;
    font-size: 22px;
}

h2 {
    text-align: center;
    margin-left: auto;
    margin-right: auto;
    color: #28444f;
}

h1 {
    text-align: center;
    margin-left: auto;
    margin-right: auto;
    color: #28444f;
}

h3 {
    text-align: left;
    margin-left: auto;
    margin-right: auto;
    color: #28444f;
}

.top-menu a {
    display: block;
    /* ссылка растягивается на весь пункт li */
    padding: 15px 35px;
    /* задаем внутренние отступы */
    color: #28444f;
    /* цвет текста */
    /*text-decoration: none; /* убрать нижнее подчёркивание у ссылок */
}

.top-menu a:hover {
    /*background-color: #7aa899;*/
    color: #7aa899;
}

.logo {
    display: inline-block;
    padding: 25px;
    margin-left: 30px;
    float: left;
}

.auth {
    display: inline-block;
    float: right;
```

```
}

.auth ul li {
    display: inline-block;
    /*text-align: center;*/
    font-size: 22px;
}

.auth a {
    display: block;
    /* ссылка растягивается на весь пункт li */
    padding: 15px 35px;
    /* задаем внутренние отступы */
    color: #28444f;
    /* цвет текста */
    text-decoration: none;
    /* убрать нижнее подчёркивание у ссылок */
}

.auth a:hover {
    background-color: #7aa899;
    color: white;
}

.content {
    /*background-color: white;*/
    /*min-height: 800px;*/
    flex-grow: 1;
}

.footer {
    width: 100%;
    height: 100px;
    background-color: #ffffff;
    border: 5px solid #28444f;
    text-align: left;
    font-size: 14px;
    color: #28444f;
}

#tableActor table {
    margin-left: auto;
    margin-right: auto;
}

#tableActor td {
    width: 400px;
    vertical-align: top;
    font-size: 20px;
    color: #28444f;
}

#tableActor a {
    color: #28444f;
    /* цвет текста */
    text-decoration: none;
    /* убрать нижнее подчёркивание у ссылок */
}

#tableActor a:hover {
    color: #7aa899;
}

#allSessions table {
```

```
    border: 2px solid #28444f;
    margin-left: auto;
    margin-right: auto;
    border-collapse: collapse;
    margin: center;
    text-align: left;
    color: #28444f;
}

#allSessions td {
    text-align: center;
    border: 2px solid #28444f;
}

.about {
    width: 60%;
    margin-left: auto;
    margin-right: auto;
    border-collapse: collapse;
    margin: center;
    text-align: left;
    color: #28444f;
    font-size: 20pt;
}

#allSessions th {
    text-align: center;
    border: 2px solid #28444f;
}

#ajaxDiv table {
    margin-left: auto;
    margin-right: auto;
    border-collapse: collapse;
    margin: center;
    text-align: center;
}

#ajaxDiv th {
    background-color: #27263D;
    height: 50px;
    color: #28444f;
}

#ajaxDiv td,
th {
    border: 2px solid #28444f;
    font-size: 18px;
    margin: center;
    width: 200px;
    color: #28444f;
}

#ajaxDiv {
    margin: center;
    text-align: center;
}

#ajaxDiv button {
    background-color: #28444f;
    color: white;
    width: 290px;
    font-size: 18px;
}
```

```
#ajaxDiv button:hover {
    background-color: #28444f;
    color: red;
    width: 290px;
    font-size: 18px;
}

.formInput {
    text-align: center;
    border: 5px solid #28444f;
    /*background-color: white;*/
    color: #28444f;
    font-size: 18px;
    width: 870px;
    margin: 20px;
    margin-left: auto;
    margin-right: auto;
    padding: 10px;
}

.formInput input {
    width: 98%;
}

.formInput th {
    color: red;
}

.formInput textarea {
    width: 98%;
    height: 100px;
}

.formInput button {
    background-color: #28444f;
    color: white;
    width: 98%;
    padding: 10px;
    font-size: 18px;
}

.formInput button:hover {
    color: #7aa899;
    width: 98%;
    padding: 10px;
}

#actor {
    color: #28444f;
    /* цвет текста */
    text-decoration: none;
    /* убрать нижнее подчёркивание у ссылок */
}

#actor:hover {
    color: #7aa899;
    /* цвет текста */
}

#buttonA {
    text-decoration: none;
    display: inline-block;
    vertical-align: middle;
```

```
    width: 198px;
    height: 198px;
    font-size: 18px;
    background-color: #27263D;
    color: #28444f;
}

#buttonA:hover {
    border: 1px solid red;
    background-color: #28444f;
    color: #7aa899;
    width: 198px;
    height: 198px;
    font-size: 18px;
}

.mainPage {
    text-align: center;
    /*background-color: white;*/
    color: #28444f;
    font-size: 18px;
    margin: 20px;
    margin-left: auto;
    margin-right: auto;
    padding: 10px;
}

#imgBorder {
    border: 5px solid #28444f;
}

.cinoteatr {
    width: 650px;
    margin-left: auto;
    margin-right: auto;
}

.cinoteatr button {
    background-color: #28444f;
    color: white;
    width: 98%;
    padding: 10px;
    font-size: 18px;
}

.cinoteatr button:hover {
    background-color: #28444f;
    color: #7aa899;
    width: 98%;
    font-size: 18px;
}

.mesta {
    float: left;
    width: 60px;
    height: 60px;
    background: silver;
    margin: 5px;
    text-align: center;
    cursor: pointer;
}

.mestaExample {
    float: left;
```

```
    width: 60px;
    height: 60px;
    background: silver;
    margin: 5px;
    text-align: center;
}

#actorsFromSpectacle button {
    background-color: #28444f;
    color: white;
}

#actorsFromSpectacle button:hover {
    background-color: #28444f;
    color: red;
}

#count_ {
    color: #28444f;
    font-size: 18pt;
}

#status {
    color: green;
    font-size: 18pt;
}

.legend table {
    border-collapse: collapse;
    margin: center;
    text-align: left;
}

.legend td {
    border: none;
    font-size: 18px;
    margin: center;
    height: 80px;
    color: #28444f;
}
```

Листинг Б.4 – Исходный код файла userTickets.php

```
<?php
global $admin;
$admin = false;
global $login;
$login = null;

session_start();
if (isset($_SESSION['userid'])) {
    $userid = $_SESSION['userid'];
    $mysqli = new mysqli('localhost', 'root', '', 'theatre'); $mysqli-
>query('SET CHARACTER SET utf8');

    $res = $mysqli->query("SELECT login, role From users WHERE
id='$userid'");
    $user = $res->fetch_object();
    $login = $user->login;
    $role = $user->role;
```

```
    if($role == 'admin'){
        $admin = true;
    }
}
?>

<!DOCTYPE html>
<html>

<head>
      <meta charset="utf-8">
      <title> Театр </title>
      <link href="../style.css" rel="stylesheet" type="text/css">
</head>

<body>
      <div class="header">
            <a class="logo" href="../index.php"><img src="logoT.jpg"
alt=""></a>
            <div class="top-menu">
                  <ul>
                        <li><a href="index.php">Главная</a></li>
                        <li><a href="actors.php">Актёры</a></li>
                        <li><a href="spectacles.php">Спектакли</a></li>
                        <div class="auth">
                              <?php
                              if (isset($_SESSION['userid'])) {
                                    $userid = $_SESSION['userid'];
                                    $mysqli = new mysqli('localhost', 'root',
'', 'theatre'); $mysqli->query('SET CHARACTER SET utf8');

                                    $res = $mysqli->query("SELECT login From
users WHERE id='$userid'");
                                    $user = $res->fetch_object();
                                    $login = $user->login;

                                    echo "<li><a href='userTickets.php'>Вы во-
шли как $login </a></li>
                              <li><a href='../exit.php'>Выход</a></li>";
                              } else {
                                    echo "<li><a
href='authForm.php'>Вход</a></li>
                        <li><a
href='../registrationForm.php'>Регистрация</a></li>";
                              }
                              ?>
                  </ul>
            </div>
      </div>
      </div>
      <div class="content">
            <?php
        if($admin){
            echo "<h2>Забронированные билеты</h2>";
        }
        else{
```

```
            echo "<h2>Ваши билеты</h2>";
        }
        if(!$admin){
            $res = $mysqli->query("SELECT places.id as idPlace,
DATE_FORMAT(tickets.date_res, '%d.%m.%Y %H:%i') as date, tickets.pz as pz,
places.row as r, places.place as place, places.price as price, places.number
as number, spectacles.name as name FROM tickets INNER JOIN places ON tick-
ets.place_id=places.id INNER JOIN sessions ON places.session_id=sessions.id
INNER JOIN spectacles ON sessions.spectacle_id=spectacles.id WHERE tick-
ets.user_id=$userid");
        }
        else {
            $res = $mysqli->query("SELECT users.login as login, users.email
as email, places.id as idPlace, DATE_FORMAT(tickets.date_res, '%d.%m.%Y
%H:%i') as date, tickets.pz as pz, places.row as r, places.place as place,
places.price as price, places.number as number, spectacles.name as name FROM
tickets INNER JOIN places ON tickets.place_id=places.id INNER JOIN sessions
ON places.session_id=sessions.id INNER JOIN spectacles ON ses-
sions.spectacle_id=spectacles.id INNER JOIN users ON tickets.user_id=users.id
WHERE tickets.pz='Бронь'");
        }
            echo "<div id= allSessions>";
            echo "<table>";
        if(!$admin){
            echo "<tr><th>Спектакль</th><th>Дата сеан-
са</th><th>Ряд</th><th>Место</th><th>Цена</th><th>Статус</th></tr>";
        }
        else{
            echo "<tr><th>Логин пользователя</th><th>Спектакль</th><th>Дата
сеанса</th><th>Ряд</th><th>Место</th><th>Цена</th><th>Статус</th></tr>";
        }
            while ($ticket =  $res->fetch_object()) {
                  $date = $ticket->date;
            $pz = $ticket->pz;

            $row = $ticket->r;
            $place = $ticket->place;
            $price = $ticket->price;
            $id = $ticket->idPlace;

            if($admin){
                $login = $ticket->login;
            }

                  $name = $ticket->name;
            if($pz === 'Бронь'){
                if($admin){
                   echo
"<tr><td>$login</td><td>$name</td><td>$date</td><td>$row</td><td>$place</td><
td>$price</td><td>$pz</td><td><button onclick=\"deleteReserv($id)\">Снять
бронь</button></td></tr>";

                }
                else
                {
```

```
                    echo
"<tr><td>$name</td><td>$date</td><td>$row</td><td>$place</td><td>$price</td><
td>$pz</td><td><button onclick=\"deleteReserv($id)\">Снять
бронь</button></td><td><button on-
click=\"buyTicket($id)\">Оплатить</button></td></tr>";
                }
            }
            else{
                echo
"<tr><td>$name</td><td>$date</td><td>$row</td><td>$place</td><td>$price</td><
td>$pz</td></tr>";
            }
            }
            echo "</table></div>";
            ?>
            <div class='mainPage'>
            </div>
      </div>
      <div class="footer">
               г. Пенза телефон 65-55-21<br>
            Горячая линия
      </div>
</body>

<script>
      function deleteReserv(place_id) {
            var httpRequest = new XMLHttpRequest();
            httpRequest.onreadystatechange = function() {
                  alertResponse(httpRequest);
            };

            httpRequest.open('POST', '../deleteReservPlace.php', true);
            httpRequest.setRequestHeader("Content-type", "application/x-www-
form-urlencoded");
            let value = "place_id=" + place_id;
            httpRequest.send(value);
      }

    function buyTicket(place_id) {
            var httpRequest = new XMLHttpRequest();
            httpRequest.onreadystatechange = function() {
                  alertResponse(httpRequest);
            };

            httpRequest.open('POST', '../buyTicket.php', true);
            httpRequest.setRequestHeader("Content-type", "application/x-www-
form-urlencoded");
            let value = "place_id=" + place_id;
            httpRequest.send(value);
      }

      function alertResponse(httpRequest) {
            if (httpRequest.readyState == 4) {
                  if (httpRequest.status == 200) {
                        var responseDiv = docu-
ment.getElementById('allSessions');
```

```
                        responseDiv.innerHTML = httpRequest.responseText;
                  } else {
                        alert('Возникли проблемы с получением ответа от серве-
pa.');
                  }
            }
      }
</script>
</html>
```

Листинг Б.5 – Исходный код файла actors.php

```
<?PHP SESSION_START();
GLOBAL $ADMIN;
$ADMIN = FALSE;
GLOBAL $LOGIN;
$LOGIN = NULL;

?>

<!DOCTYPE HTML>
<HTML>

<HEAD>
      <META CHARSET="UTF-8">
      <TITLE> TEATP </TITLE>
      <LINK HREF="../STYLE.CSS" REL="STYLESHEET" TYPE="TEXT/CSS">
</HEAD>

<BODY>
      <DIV CLASS="HEADER">
            <A CLASS="LOGO" HREF="../INDEX.PHP"><IMG SRC="../LOGOT.JPG"
ALT=""></A>
            <DIV CLASS="TOP-MENU">
                  <UL>
                        <LI><A HREF="../INDEX.PHP">ГЛАВНАЯ</A></LI>
                        <LI><A HREF="ACTORS.PHP">АКТЁРЫ</A></LI>
                        <LI><A HREF="SPECTACLES.PHP">СПЕКТАКЛИ</A></LI>
                        <DIV CLASS="AUTH">
                              <?PHP
                              IF (ISSET($_SESSION['USERID'])) {
                                    $USERID = $_SESSION['USERID'];
                                    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT',
'', 'THEATRE'); $MYSQLI->QUERY('SET CHARACTER SET UTF8');

                                    $RES = $MYSQLI->QUERY("SELECT LOGIN, ROLE
FROM USERS WHERE ID='$USERID'");
                                    $USER = $RES->FETCH_OBJECT();
                                    $LOGIN = $USER->LOGIN;
                                    $ROLE = $USER->ROLE;
                                    IF ($ROLE == 'ADMIN') {
                                          $ADMIN = TRUE;
                                    }
```

```
                                    ECHO "<LI><A HREF='USERTICKETS.PHP'>ВЫ
ВОШЛИ КАК $LOGIN </A></LI>
                              <LI><A HREF='../EXIT.PHP'>ВЫХОД</A></LI>";
                              } ELSE {
                                    ECHO "<LI><A
HREF='../AUTHFORM.PHP'>ВХОД</A></LI>
                              <LI><A
HREF='../REGISTRATIONFORM.PHP'>РЕГИСТРАЦИЯ</A></LI>";
                              }
                              ?>
                  </UL>
            </DIV>
      </DIV>
      </DIV>
      <DIV CLASS="CONTENT">
            <H1>АКТЁРЫ НАШЕГО ТЕАТРА</H1>
            <?PHP
            $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');
            IF ($ADMIN) {
                  ECHO "
            <DIV CLASS= FORMINPUT>
               <H2> ДОБАВИТЬ АКТЕРА </H2>
               <FORM ID='UPLOAD-CONTAINER' ACTION='../ADDACTOR.PHP' METHOD
='POST' ENCTYPE='MULTIPART/FORM-DATA'>";
                  IF (ISSET($_GET['STATUS'])) {
                        $STATUS = $_GET['STATUS'];
                        SWITCH ($STATUS) {
                              CASE 1:
                                    ECHO '<DIV STYLE="COLOR: GREEN;"> АКТЕР
ДОБАВЛЕН!</DIV><HR>';
                                    BREAK;
                              CASE 2:
                                    ECHO '<DIV STYLE="COLOR: RED;"> НЕ УДАЛОСЬ
ЗАГРУЗИТЬ ФОТО</DIV><HR>';
                                    BREAK;
                              CASE 3:
                                    ECHO '<DIV STYLE="COLOR: RED;"> ВЫ НЕ
ВВЕЛИ ИМЯ АКТЕРУ</DIV><HR>';
                                    BREAK;
                              CASE 4:
                                    ECHO '<DIV STYLE="COLOR: RED;"> ВЫ НЕ
ВВЕЛИ ФАМИЛИЮ АКТЕРУ</DIV><HR>';
                                    BREAK;
                              CASE 5:
                                    ECHO '<DIV STYLE="COLOR: RED;"> ВЫ НЕ
ВОЗРАСТ АКТЕРУ</DIV><HR>';
                                    BREAK;
                        }
                  }
                  ECHO "<P>ВЫБЕРИТЕ ФОТО<INPUT TYPE='FILE' CLASS='CHOOSEFILE'
NAME='USERFILE' ACCEPT='IMAGE/*' /></P>
               <P>ИМЯ <INPUT TYPE='TEXT' NAME='NAME'></P>
               <P>ФАМИЛИЯ <INPUT TYPE='TEXT' NAME='SURNAME'></P>
               <P>ВОЗРАСТ<INPUT TYPE='NUMBER' NAME='AGE'></P>
               <P>СТАЖ<INPUT TYPE='NUMBER' NAME='EXPERIENCE'></P>
```

```
                <P>БИОГРАФИЯ<TEXTAREA TYPE='TEXT'
NAME='BIOGRAPHY'></TEXTAREA></P>
                <BUTTON> ДОБАВИТЬ АКТЕРА </BUTTON>
                </FORM>
          </DIV>";
             }

             $MYSQLI->QUERY('SET CHARACTER SET UTF8');
             $RES = $MYSQLI->QUERY('SELECT * FROM ACTORS');
             ECHO "<DIV ID=AJAXDIV>";
             ECHO "<TABLE>";

             ECHO "<TR>"; // NEW
             $INDEXROW = 0;
             WHILE ($ACTOR =  $RES->FETCH_OBJECT()) {
                   $ID = $ACTOR->ID;
                   $PHOTO = $ACTOR->PHOTO;
                   IF ($INDEXROW > 3) {
                         ECHO "<TR>";
                         $INDEXROW = 0;
                   }
                   IF ($PHOTO == "") {
                         $PHOTO = '../FILES/NOPHOTO.JPG';
                   }
                   IF ($ADMIN) {
                         ECHO "<TD><A ID='ACTOR'
HREF='ACTOR.PHP?ACTOR_ID=$ID'><IMG SRC='$PHOTO' ALT = 'ФОТО АКТЕРА'
WIDTH='290' HEIGHT='400'></IMG> $ACTOR->NAME $ACTOR->SURNAME</A><BUTTON ON-
CLICK=\"DELETEACTOR($ID)\">УДАЛИТЬ</BUTTON><BR></TD>";
                   } ELSE {
                         ECHO "<TD><A ID='ACTOR'
HREF='ACTOR.PHP?ACTOR_ID=$ID'><IMG SRC='$PHOTO' ALT = 'ФОТО АКТЕРА'
WIDTH='290' HEIGHT='400'></IMG> $ACTOR->NAME $ACTOR->SURNAME</A></TD>";
                   }
                   $PHOTO_STR = JSON_ENCODE($PHOTO);
                   $INDEXROW++;
             }
             ECHO "</TR></TABLE></DIV>";
             ?>
      </DIV>
      <DIV CLASS='MAINPAGE'>
      </DIV>
      <DIV CLASS="FOOTER">
               Г. ПЕНЗА ТЕЛЕФОН 65-55-21<BR>
             ГОРЯЧАЯ ЛИНИЯ
      </DIV>
</BODY>

<SCRIPT>
      FUNCTION DELETEACTOR(ACTOR_ID) {
             VAR HTTPREQUEST = NEW XMLHTTPREQUEST();
             HTTPREQUEST.ONREADYSTATECHANGE = FUNCTION() {
                   ALERTRESPONSE(HTTPREQUEST);
             };

             HTTPREQUEST.OPEN('POST', '../DELETEACTOR.PHP', TRUE);
```

```
            HTTPREQUEST.SETREQUESTHEADER("CONTENT-TYPE", "APPLICATION/X-WWW-
FORM-URLENCODED");
            LET VALUE = "ACTOR_ID=" + ACTOR_ID;
            HTTPREQUEST.SEND(VALUE);
      }

      FUNCTION ALERTRESPONSE(HTTPREQUEST) {
            IF (HTTPREQUEST.READYSTATE == 4) {
                  IF (HTTPREQUEST.STATUS == 200) {
                        VAR RESPONSEDIV = DOCUMENT.GETELEMENTBYID('AJAXDIV');
                        RESPONSEDIV.INNERHTML = HTTPREQUEST.RESPONSETEXT;
                  } ELSE {
                        ALERT('ВОЗНИКЛИ ПРОБЛЕМЫ С ПОЛУЧЕНИЕМ ОТВЕТА ОТ СЕРВЕ-
РА.');
                  }
            }
      }
</SCRIPT>

</HTML>
```

Листинг Б.6 – Исходный код файла spectacle.php

```
<?PHP SESSION_START();
GLOBAL $ADMIN;
$ADMIN = FALSE;
GLOBAL $LOGIN;
$LOGIN = NULL;

GLOBAL $SPECTACLE_ID;
$SPECTACLE_ID = NULL;
?>

<!DOCTYPE HTML>
<HTML>

<HEAD>
      <META CHARSET="UTF-8">
      <TITLE> ТЕАТР </TITLE>
      <LINK REL="STYLESHEET" TYPE="TEXT/CSS" HREF="/EXT3/RESOURCES/CSS/EXT-
ALL.CSS"/>
      <LINK HREF="../STYLE.CSS" REL="STYLESHEET" TYPE="TEXT/CSS">
      <SCRIPT TYPE="TEXT/JAVASCRIPT" SRC="/EXT3/ADAPTER/EXT/EXT-
BASE.JS"></SCRIPT>
      <SCRIPT TYPE="TEXT/JAVASCRIPT" SRC="/EXT3/EXT-ALL.JS"></SCRIPT>
      <SCRIPT TYPE="TEXT/JAVASCRIPT"
SRC="/EXT3/EXAMPLES/UX/SLIDERTIP.JS"></SCRIPT>
      <SCRIPT TYPE="TEXT/JAVASCRIPT" SRC="../JS/SESSIONSFORM.JS"></SCRIPT>

</HEAD>

<BODY>
      <DIV CLASS="HEADER">
            <A CLASS="LOGO" HREF="../INDEX.PHP"><IMG SRC="../LOGOT.JPG"
ALT=""></A>
```

```
            <DIV CLASS="TOP-MENU">
                  <UL>
                        <LI><A HREF="../INDEX.PHP">ГЛАВНАЯ</A></LI>
                        <LI><A HREF="ACTORS.PHP">АКТЁРЫ</A></LI>
                        <LI><A HREF="SPECTACLES.PHP">СПЕКТАКЛИ</A></LI>
                        <DIV CLASS="AUTH">
                              <?PHP
                              IF (ISSET($_SESSION['USERID'])) {
                                    $USERID = $_SESSION['USERID'];
                                    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT',
'', 'THEATRE'); $MYSQLI->QUERY('SET CHARACTER SET UTF8');

                                    $RES = $MYSQLI->QUERY("SELECT * FROM USERS
WHERE ID='$USERID'");
                                    $USER = $RES->FETCH_OBJECT();
                                    $LOGIN = $USER->LOGIN;
                                    $ROLE = $USER->ROLE;
                                    IF ($ROLE == 'ADMIN') {
                                          $ADMIN = TRUE;
                                    }
                                    ECHO "<LI><A HREF='USERTICKETS.PHP'>ВЫ
ВОШЛИ КАК $LOGIN </A></LI>
                              <LI><A HREF='../EXIT.PHP'>ВЫХОД</A></LI>";
                              } ELSE {
                                    ECHO "<LI><A
HREF='../AUTHFORM.PHP'>ВХОД</A></LI>
                              <LI><A
HREF='../REGISTRATIONFORM.PHP'>РЕГИСТРАЦИЯ</A></LI>";
                              }
                              ?>
                  </UL>
            </DIV>
      </DIV>
      </DIV>
      <DIV CLASS="CONTENT">

            <?PHP
            IF (ISSET($_GET['SPECTACLE_ID'])) {
                  $SPECTACLE_ID = $_GET['SPECTACLE_ID'];
                  $_SESSION['SPECTACLE_ID'] = $SPECTACLE_ID;
            }
            $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');
            $RES = $MYSQLI->QUERY("SELECT * FROM SPECTACLES WHERE
ID=$SPECTACLE_ID");
            ECHO "<DIV ID= TABLEACTOR>";
            $SPECTACLE =  $RES->FETCH_OBJECT();
            $NAME = $SPECTACLE->NAME;
            $POSTER = $SPECTACLE->POSTER;
            $DESCRIPTION = $SPECTACLE->DESCRIPTION;
            $DURATION = $SPECTACLE->DURATION;

            IF ($ADMIN) {
                  ECHO "<DIV CLASS= FORMINPUT>
            <FORM ACTION='../CHANGESPECTACLE.PHP' METHOD ='POST'
ENCTYPE='MULTIPART/FORM-DATA'>
```

```
            <H2> ИЗМЕНИТЬ СПЕКТАКЛЬ </H2>
            <P>ЗАМЕНИТЬ АФИШУ<INPUT TYPE='FILE' CLASS='CHOOSEFILE'
NAME='USERFILE' ACCEPT='IMAGE/*' VALUE=''/></P>
            <INPUT TYPE='HIDDEN' NAME ='ID' VALUE=$SPECTACLE_ID>
            <INPUT TYPE='HIDDEN' NAME ='OLDPHOTO' VALUE=$POSTER>
            <P>НАЗВАНИЕ <INPUT TYPE='TEXT' NAME ='NAME' VALUE='$NAME'></P>
            <P>ДЛИТЕЛЬНОСТЬ <INPUT TYPE='NUMBER' NAME ='DURATION' VAL-
UE=$DURATION></P>
            <P>ОПИСАНИЕ <TEXTAREA TYPE='TEXT'
NAME='DESCRIPTION'>$DESCRIPTION</TEXTAREA></P>
            <BUTTON> ИЗМЕНИТЬ ЗАПИСЬ </BUTTON></DIV>
            </FORM>";
            }
            IF ($POSTER == "") {
                  $POSTER = '../FILES/NOPHOTO.JPG';
            }
            ECHO "<H2>$NAME</H2><BR>";
            ECHO "<TABLE><TR>";
            ECHO "<TD><IMG SRC=$POSTER ALT = 'ФОТО АКТЕРА' WIDTH='348'
HEIGHT='480'></IMG></TD>";
            ECHO "<TD>$DESCRIPTION<BR>";
            ECHO "<BR>ДЛИТЕЛЬНОСТЬ: $DURATION ЧАСОВ<BR>";

            ECHO "<BR><H3>АКТЕРЫ:</H3>";
            $RES = $MYSQLI->QUERY("SELECT ACTORS.NAME AS NAME, ACTORS.SURNAME
AS SURNAME, ACTORS.ID AS ID FROM SPECTALES_HAS_ACTORS
            INNER JOIN ACTORS ON SPECTALES_HAS_ACTORS.ACTOR_ID=ACTORS.ID WHERE
SPECTALES_HAS_ACTORS.SPECTACLE_ID=$SPECTACLE_ID");
            ECHO "<DIV ID='ACTORSFROMSPECTACLE'><UL>";
            WHILE ($ACTOR = $RES->FETCH_OBJECT()) {
                  IF ($ADMIN) {
                        ECHO "<LI><A HREF='ACTOR.PHP?ACTOR_ID=$ACTOR-
>ID'>$ACTOR->NAME $ACTOR->SURNAME</A> <BUTTON ON-
CLICK=\"DELETEACTORFORSPECTACLE($SPECTACLE_ID, $ACTOR->ID)\">УДАЛИТЬ
АКТЕРА</BUTTON></LI>";
                  } ELSE {
                        ECHO "<LI><A HREF='ACTOR.PHP?ACTOR_ID=$ACTOR-
>ID'>$ACTOR->NAME $ACTOR->SURNAME</A></LI>"; //<A HREF='#'> | УДАЛИТЬ</A>
                  }
            }
            ECHO "</UL><BR>";
            IF ($ADMIN) {
                  ECHO "<H3>ДОБАВИТЬ АКТЕРА: </H3><SELECT ID='ACTORS'>";
                  $RES = $MYSQLI->QUERY("SELECT ACTORS.NAME AS NAME, AC-
TORS.SURNAME AS SURNAME, ACTORS.ID AS ID FROM ACTORS");
                  WHILE ($ACTOR = $RES->FETCH_OBJECT()) {
                        ECHO "<OPTION>$ACTOR->ID | $ACTOR->NAME $ACTOR-
>SURNAME</OPTION>";
                  }
                  ECHO "</SELECT> <BUTTON ON-
CLICK=\"ADDACTORFORSPECTACLE($SPECTACLE_ID)\">ДОБАВИТЬ АКТЕРА</BUTTON>";
            }
            ECHO '</DIV><BR>';
            IF($ADMIN) {
                  ECHO "<DIV ID='SESSIONS'></DIV>";
            }
```

```
            ELSE {
                  ECHO "<H3>СЕАНСЫ:</H3>";
                  ECHO "<DIV ID='SEANSFORSPECTACLE'><UL>";
                  $RES = $MYSQLI->QUERY("SELECT ID, DATE_FORMAT(DATE,
'%D.%M.%Y %H:%I') AS DATE FROM SESSIONS WHERE SPECTACLE_ID=$SPECTACLE_ID");
                  WHILE ($SESSION = $RES->FETCH_OBJECT()) {
                        ECHO "<LI><A HREF='SESSION.PHP?SESSION_ID=$SESSION-
>ID'>$SESSION->DATE</A></LI>";
                  }
                  ECHO "</UL></DIV>";
            }
            ECHO "</TD></TR></TABLE>";
            ?>
      </DIV>
      <DIV CLASS='MAINPAGE'>
      </DIV>
      <DIV CLASS="FOOTER">
               Г. ПЕНЗА ТЕЛЕФОН 65-55-21<BR>
            ГОРЯЧАЯ ЛИНИЯ
      </DIV>
</BODY>


<SCRIPT>

      FUNCTION ADDACTORFORSPECTACLE(IDSPECTACLE) {
            VAR IDACTOR = DOCUMENT.GETELEMENTBYID('ACTORS').VALUE;
            IDACTOR = IDACTOR.SPLIT('|')[0];
            VAR HTTPREQUEST = NEW XMLHTTPREQUEST();
            HTTPREQUEST.ONREADYSTATECHANGE = FUNCTION() {
                  ALERTRESPONSE(HTTPREQUEST);
            };
            HTTPREQUEST.OPEN('POST', '../ADDACTORFORSPECTACLE.PHP', TRUE);
            HTTPREQUEST.SETREQUESTHEADER("CONTENT-TYPE", "APPLICATION/X-WWW-
FORM-URLENCODED");
            LET VALUE = "ACTOR_ID=" + IDACTOR + "&SPECTACLE_ID=" + IDSPECTA-
CLE;
            HTTPREQUEST.SEND(VALUE);
      }

      FUNCTION DELETEACTORFORSPECTACLE(IDSPECTACLE, IDACTOR) {
            VAR HTTPREQUEST = NEW XMLHTTPREQUEST();
            HTTPREQUEST.ONREADYSTATECHANGE = FUNCTION() {
                  ALERTRESPONSE(HTTPREQUEST);
            };
            HTTPREQUEST.OPEN('POST', '../DELETEACTORFORSPECTACLE.PHP', TRUE);
            HTTPREQUEST.SETREQUESTHEADER("CONTENT-TYPE", "APPLICATION/X-WWW-
FORM-URLENCODED");
            LET VALUE = "ACTOR_ID=" + IDACTOR + "&SPECTACLE_ID=" + IDSPECTA-
CLE;
            HTTPREQUEST.SEND(VALUE);
      }

      FUNCTION ALERTRESPONSE(HTTPREQUEST) {
            IF (HTTPREQUEST.READYSTATE == 4) {
                  IF (HTTPREQUEST.STATUS == 200) {
```

```
                        VAR RESPONSEDIV = DOCU-
MENT.GETELEMENTBYID('ACTORSFROMSPECTACLE');
                        RESPONSEDIV.INNERHTML = HTTPREQUEST.RESPONSETEXT;
                  } ELSE {
                        ALERT('ВОЗНИКЛИ ПРОБЛЕМЫ С ПОЛУЧЕНИЕМ ОТВЕТА ОТ СЕРВЕ-
РА.');
                  }
            }
      }
</SCRIPT>

</HTML>
```

Листинг Б.7 – Исходный код файла session.php

```
<?PHP SESSION_START();
GLOBAL $ADMIN;
$ADMIN = FALSE;
GLOBAL $LOGIN;
$LOGIN = NULL;
GLOBAL $RESERVATIONPLACES;
$RESERVATIONPLACES = ARRAY();
GLOBAL $PRICE1;
$PRICE1 = 0;
GLOBAL $PRICE2;
$PRICE2 = 0;
GLOBAL $PRICE3;
$PRICE3 = 0;
?>

<!DOCTYPE HTML>
<HTML>

<HEAD>
    <META CHARSET="UTF-8">
    <TITLE> TEATP </TITLE>
    <LINK HREF="../STYLE.CSS" REL="STYLESHEET" TYPE="TEXT/CSS">
</HEAD>

<BODY>
    <DIV CLASS="HEADER">
        <A CLASS="LOGO" HREF="../INDEX.PHP"><IMG SRC="../LOGOT.JPG"
ALT=""></A>
        <DIV CLASS="TOP-MENU">
            <UL>
                <LI><A HREF="../INDEX.PHP">ГЛАВНАЯ</A></LI>
```

```
                <LI><A HREF="ACTORS.PHP">АКТЁРЫ</A></LI>
                <LI><A HREF="SPECTACLES.PHP">СПЕКТАКЛИ</A></LI>
                <DIV CLASS="AUTH">
                    <?PHP
                    IF (ISSET($_SESSION['USERID'])) {
                        $USERID = $_SESSION['USERID'];
                        $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEA-
TRE'); $MYSQLI->QUERY('SET CHARACTER SET UTF8');

                        $RES = $MYSQLI->QUERY("SELECT LOGIN FROM USERS WHERE
ID='$USERID'");
                        $USER = $RES->FETCH_OBJECT();
                        $LOGIN = $USER->LOGIN;

                                    ECHO "<LI><A HREF='USERTICKETS.PHP'>ВЫ ВО-
ШЛИ КАК $LOGIN </A></LI>
                                <LI><A HREF='../EXIT.PHP'>ВЫХОД</A></LI>";
                    } ELSE {
                        ECHO "<LI><A HREF='../AUTHFORM.PHP'>ВХОД</A></LI>
                            <LI><A
HREF='../REGISTRATIONFORM.PHP'>РЕГИСТРАЦИЯ</A></LI>";
                    }
                    ?>
            </UL>
        </DIV>
    </DIV>
    </DIV>
    <DIV CLASS="CONTENT">
        <?PHP
        IF (ISSET($_GET['SESSION_ID'])) {
            $SESSION_ID = $_GET['SESSION_ID'];
        }
        $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');

        $RES = $MYSQLI->QUERY("SELECT SPECTACLES.NAME AS SPECTACLE_NAME,
DATE_FORMAT(SESSIONS.DATE, '%D.%M.%Y %H:%I') AS DATE FROM SESSIONS
        INNER JOIN SPECTACLES ON SPECTACLES.ID=SESSIONS.SPECTACLE_ID WHERE
SESSIONS.ID = $SESSION_ID");

        $SESSION =  $RES->FETCH_OBJECT();

        ECHO "<DIV CLASS='CINOTEATR'>";
```

```
        ECHO "<H2>$SESSION->SPECTACLE_NAME</H2>";
        ECHO "<H2>ДАТА И ВРЕМЯ СЕАНСА: $SESSION->DATE</H2>";

        $RES = $MYSQLI->QUERY("SELECT PRICE FROM PLACES WHERE NUMBER=1 AND
SESSION_ID=$SESSION_ID");
        $PRICE1 = $RES->FETCH_OBJECT();
        $PRICE1 = $PRICE1->PRICE;

        $RES = $MYSQLI->QUERY("SELECT PRICE FROM PLACES WHERE NUMBER=28 AND
SESSION_ID=$SESSION_ID");
        $PRICE2 = $RES->FETCH_OBJECT();
        $PRICE2 = $PRICE2->PRICE;

        $RES = $MYSQLI->QUERY("SELECT PRICE FROM PLACES WHERE NUMBER=81 AND
SESSION_ID=$SESSION_ID");
        $PRICE3 = $RES->FETCH_OBJECT();
        $PRICE3 = $PRICE3->PRICE;

        ECHO "<DIV CLASS='LEGEND'><TABLE>
        <TR><TD><DIV CLASS='MESTAEXAMPLE'></DIV></TD>
        <TD>СЕКТОР A ЦЕНА: $PRICE1</TD><TR>
        <TR><TD><DIV CLASS='MESTAEXAMPLE'></DIV></TD>
        <TD>СЕКТОР B ЦЕНА: $PRICE2</TD><TR>
        <TR><TD><DIV CLASS='MESTAEXAMPLE'></DIV></TD>
        <TD>СЕКТОР C ЦЕНА: $PRICE3</TD><TR>
        <TR><TD><DIV CLASS='MESTAEXAMPLE'></DIV></TD>
        <TD>ВЫБРАННЫЕ МЕСТА</TD><TR>
        <TR><TD><DIV CLASS='MESTAEXAMPLE'></DIV></TD>
        <TD>ЗАНЯТЫЕ МЕСТА</TD><TR>
        </TABLE></DIV>";
        IF ($LOGIN != NULL) {
            $USERID = $_SESSION['USERID'];
            ECHO "<P><BUTTON ON-
CLICK=\"GETCHOOSENPLACES('../RESERVATIONTICKET.PHP',$SESSION_ID,$USERID)\">ЗА
БРОНИРОВАТЬ</BUTTON></P>";
        } ELSE {
            ECHO "<H2>ТОЛЬКО ЗАРЕГИСТРИРОВАННЫЕ ПОЛЬЗОВАТЕЛИ МОГУТ БРОНИРО-
ВАТЬ БИЛЕТЫ</H2>";
        }

        $RES = $MYSQLI->QUERY("SELECT PLACES.NUMBER AS NUMBER FROM PLACES IN-
NER JOIN TICKETS ON TICKETS.PLACE_ID=PLACES.ID WHERE PLAC-
ES.SESSION_ID=$SESSION_ID");
```

```
        WHILE ($PLACES = $RES->FETCH_OBJECT()) {
            ARRAY_PUSH($RESERVATIONPLACES, (INT)$PLACES->NUMBER);
        }
        ECHO "<DIV ID='COUNT_'></DIV>";
        ECHO "<DIV ID='STATUS'></DIV>";
        $ROW = 1;
        FOR ($I = 1; $I < 82; $I++) {
            IF ($I % 9 == 0) {
                ECHO "<DIV CLASS='MESTA'>$I</DIV>";
                $ROW++;
            } ELSE {
                ECHO "<DIV CLASS='MESTA'>$I</DIV>";
            }
        }
        ECHO '</DIV>'
        ?>
        <DIV ID="AJAXDIV"></DIV>
    </DIV>
    <DIV CLASS='MAINPAGE'>
    </DIV>
    <DIV CLASS="FOOTER">
           Г. ПЕНЗА ТЕЛЕФОН 65-55-21<BR>
        ГОРЯЧАЯ ЛИНИЯ
    </DIV>
</BODY>


<SCRIPT>
    VAR EXMAPLEPLACES = DOCUMENT.GETELEMENTSBYCLASSNAME('MESTAEXAMPLE');
    FOR (I = 0; I < EXMAPLEPLACES.LENGTH; I++) {
        SWITCH (I) {
            CASE 0:
                EXMAPLEPLACES[I].STYLE.BACKGROUND = 'GREEN';
                BREAK;
            CASE 1:
                EXMAPLEPLACES[I].STYLE.BACKGROUND = 'YELLOW';
                BREAK;
            CASE 2:
                EXMAPLEPLACES[I].STYLE.BACKGROUND = 'ORANGE';
                BREAK;
            CASE 3:
                EXMAPLEPLACES[I].STYLE.BACKGROUND = 'RED';
                BREAK;
```

```
            CASE 4:
                EXMAPLEPLACES[I].STYLE.BACKGROUND = 'BLACK';
                BREAK;
        }
    }
    VAR L = DOCUMENT.GETELEMENTSBYCLASSNAME('MESTA');

    FOR (LET I = 0; I < L.LENGTH; I++) {
        IF (L[I].INNERHTML < 28) {
            L[I].STYLE.BACKGROUND = 'GREEN';
        }
        IF (L[I].INNERHTML > 27 && L[I].INNERHTML < 55) {
            L[I].STYLE.BACKGROUND = 'YELLOW';
        }
        IF (L[I].INNERHTML > 54) {
            L[I].STYLE.BACKGROUND = 'ORANGE';
        }
    }
    VAR RESERVPLACES = JSON.PARSE('<?PHP ECHO
JSON_ENCODE($RESERVATIONPLACES); ?>');
    CONSOLE.LOG(RESERVPLACES);
    FOR (LET I = 0; I < L.LENGTH; I++) {
        FOR (LET J = 0; J < RESERVPLACES.LENGTH; J++) {
            IF (L[I].INNERHTML == RESERVPLACES[J]) {
                L[I].STYLE.BACKGROUND = 'BLACK';
            }
        }
    }

    FUNCTION COUNT_MEST() {
        LET Z_MEST = 0;
        FOR (LET I = 0; I < L.LENGTH; I++) {
            IF (L[I].STYLE.BACKGROUND == 'RED') {
                Z_MEST++;
            }
        }
        RETURN Z_MEST;
    }

    FUNCTION GETSUM(){
        VAR SUM = 0;
        FOR (LET I = 0; I < L.LENGTH; I++) {
            IF (L[I].STYLE.BACKGROUND == 'RED') {
```

```
                SUM += GETPRICEFORPLACE(NUMBER(L[I].INNERHTML));
            }
        }
        RETURN SUM;
    }
    FUNCTION GETPLACES() {
        VAR PLACES = [];
        FOR (LET I = 0; I < L.LENGTH; I++) {
            IF (L[I].STYLE.BACKGROUND == 'RED') {
                PLACES.PUSH(L[I].INNERHTML);
            }
        }
        RETURN PLACES;
    }
    FOR (LET I = 0; I < L.LENGTH; I++) {
        L[I].ONCLICK = FUNCTION() {
            LET BACKGROUND = THIS.STYLE.BACKGROUND;
            IF (<?PHP ECHO JSON_ENCODE($LOGIN); ?>) {
                IF (BACKGROUND != 'BLACK') {
                    IF (BACKGROUND == 'RED') {
                        IF (L[I].INNERHTML < 28) {
                            THIS.STYLE.BACKGROUND = 'GREEN';
                        }
                        IF (L[I].INNERHTML > 27 && L[I].INNERHTML < 55) {
                            THIS.STYLE.BACKGROUND = 'YELLOW';
                        }
                        IF (L[I].INNERHTML > 54) {
                            THIS.STYLE.BACKGROUND = 'ORANGE';
                        }

                    } ELSE {
                        THIS.STYLE.BACKGROUND = 'RED';
                    }
                }
            } ELSE {
                DOCUMENT.GETELEMENTBYID('STATUS').INNERHTML = "<A
HREF='../AUTHFORM.PHP'>ДЛЯ БРОНИРОВАНИЯ МЕСТ НЕОБХОДИМО
АВТОРИЗОВАТЬСЯ!</HREF>";
            }
            IF(COUNT_MEST() > 0){
                DOCUMENT.GETELEMENTBYID('COUNT_').INNERHTML = 'ВЫБРАНО МЕСТ :
' + COUNT_MEST() + ' НА СУММУ: ' + GETSUM() + ' РУБЛЕЙ';
            } ELSE{
```

```
                DOCUMENT.GETELEMENTBYID('COUNT_').INNERHTML = '';
            }
        }
    }

    FUNCTION GETPRICEFORPLACE(PLACENUMBER) {
        VAR PRICE1 = NUMBER(JSON.PARSE('<?PHP ECHO JSON_ENCODE($PRICE1);
?>'));
        VAR PRICE2 = NUMBER(JSON.PARSE('<?PHP ECHO JSON_ENCODE($PRICE2);
?>'));
        VAR PRICE3 = NUMBER(JSON.PARSE('<?PHP ECHO JSON_ENCODE($PRICE3);
?>'));
        IF(PLACENUMBER < 28){
            RETURN PRICE1;
        }
        IF(PLACENUMBER > 28 && PLACENUMBER < 55){
            RETURN PRICE2;
        }
        RETURN PRICE3;
    }

    FUNCTION GETCHOOSENPLACES(URL, SESSION, USERID) {
        VAR HTTPREQUEST = NEW XMLHTTPREQUEST();
        HTTPREQUEST.ONREADYSTATECHANGE = FUNCTION() {
            ALERTRESPONSE(HTTPREQUEST);
        };
        VAR PLACES = GETPLACES();
        VAR NOW = NEW DATE();
        VAR MONTH = NOW.GETMONTH() + 1;
        IF (MONTH < 10) {
            MONTH = '0' + MONTH;
        }
        VAR DAY = NOW.GETDATE();
        IF (DAY < 10) {
            DAY = '0' + DAY;
        }
        VAR DATENOW = NOW.GETFULLYEAR() + '-' + MONTH + '-' + DAY + ' ' +
NOW.GETHOURS() + ':' + NOW.GETMINUTES() + ':' + NOW.GETSECONDS();
        CONSOLE.LOG(PLACES);
        HTTPREQUEST.OPEN('POST', URL, TRUE);
        HTTPREQUEST.SETREQUESTHEADER("CONTENT-TYPE", "APPLICATION/X-WWW-FORM-
URLENCODED"); //ТИП ПЕРЕДАВАЕМЫХ ДАННЫХ ЗАКОДИРОВАН.
```

```
        LET VALUE = "PLACES=" + PLACES + "&SESSION=" + SESSION + "&USERID=" +
USERID + "&DATE=" + DATENOW;
        HTTPREQUEST.SEND(VALUE);
    }

    FUNCTION ALERTRESPONSE(HTTPREQUEST) {
        IF (HTTPREQUEST.READYSTATE == 4) {
            IF (HTTPREQUEST.STATUS == 200) {
                VAR RESPONSEDIV = DOCUMENT.GETELEMENTBYID('STATUS');
                VAR RESERVPLACES = JSON.PARSE(HTTPREQUEST.RESPONSETEXT);
                IF (COUNT_MEST() > 1) {
                    RESPONSEDIV.INNERHTML = "МЕСТА УСПЕШНО ЗАБРОНИРОВАНЫ";
                } ELSE {
                    RESPONSEDIV.INNERHTML = "МЕСТО УСПЕШНО ЗАБРОНИРОВАНО";
                }
                CONSOLE.LOG(RESERVPLACES);
                FOR (LET I = 0; I < L.LENGTH; I++) {
                    FOR (LET J = 0; J < RESERVPLACES.LENGTH; J++) {
                        IF (L[I].INNERHTML == RESERVPLACES[J]) {
                            L[I].STYLE.BACKGROUND = 'BLACK';
                        }
                    }
                }
            } ELSE {
                ALERT('ВОЗНИКЛИ ПРОБЛЕМЫ С ПОЛУЧЕНИЕМ ОТВЕТА ОТ СЕРВЕРА.');
            }
        }
    }
</SCRIPT>
</HTML>
```

Листинг Б.8 – Исходный код файла authForm.php

```
<!DOCTYPE HTML>
<HTML>
<HEAD>
      <META CHARSET = "UTF-8">
      <TITLE> ТЕАТР </TITLE>
      <LINK HREF="STYLE.CSS" REL="STYLESHEET" TYPE="TEXT/CSS">
</HEAD>
<BODY>
      <DIV CLASS="HEADER">
            <A CLASS="LOGO" HREF="INDEX.PHP"><IMG SRC="LOGOT.JPG" ALT=""></A>
```

```
            <DIV CLASS="TOP-MENU">
            <UL>
                  <LI><A HREF="INDEX.PHP">ГЛАВНАЯ</A></LI>
                  <LI><A HREF="PAGES/ACTORS.PHP">АКТЁРЫ</A></LI>
                  <LI><A HREF="PAGES/SPECTACLES.PHP">СПЕКТАКЛИ</A></LI>
            <DIV CLASS="AUTH">
                  <?PHP
                  IF(ISSET($_SESSION['USERID']))
                  {
                        $USERID = $_SESSION['USERID'];
                        $MYSQLI                        =                        NEW
MYSQLI('LOCALHOST','ROOT','','SHOE_STORE');

                        $RES  =  $MYSQLI->QUERY("SELECT  LOGIN  FROM  USERS  WHERE
ID='$USERID'");
                        $USER = $RES->FETCH_OBJECT();
                        $LOGIN = $USER->LOGIN;

                        ECHO "<LI><A HREF='PAGES/USERTICKETS.PHP'>ВЫ ВОШЛИ КАК
$LOGIN </A></LI>
                        <LI><A HREF='EXIT.PHP'>ВЫХОД</A></LI>";
                  }
                  ELSE
                  {
                        ECHO"<LI><A HREF='AUTHFORM.PHP'>ВХОД</A></LI>
                        <LI><A
HREF='REGISTRATIONFORM.PHP'>РЕГИСТРАЦИЯ</A></LI>";
                  }
                  ?>
            </UL>
            </DIV>
            </DIV>
      </DIV>
      <DIV CLASS="CONTENT">
        <H2>ВХОД</H2>
        <DIV CLASS='FORMINPUT'>
        <?PHP
        IF(ISSET($_GET['ERROR']))
        {
            $ERROR = $_GET['ERROR'];
            ECHO '<DIV STYLE="COLOR: RED;">'.$ERROR.'</DIV><HR>';
        }
        ?>
```

```
        <FORM ACTION='LOGIN.PHP' METHOD ='POST'>
          <P>ЛОГИН <INPUT TYPE='TEXT' NAME ='LOGIN'></P>
          <P>ПАРОЛЬ <INPUT TYPE='PASSWORD' NAME ='PASSWORD'></P>
          <P><BUTTON>ВХОД</BUTTON></P>
        </FORM>
        </DIV>
    </DIV>
      <DIV CLASS="FOOTER">
         Г. ПЕНЗА ТЕЛЕФОН 65-55-21<BR>
                  ГОРЯЧАЯ ЛИНИЯ
      </DIV>
</BODY>
</HTML>
```

Листинг Б.9 – Исходный код файла buyTicket.php

```
<?PHP
GLOBAL $ADMIN;
$ADMIN = FALSE;
GLOBAL $LOGIN;
$LOGIN = NULL;

SESSION_START();
IF (ISSET($_SESSION['USERID'])) {
    $USERID = $_SESSION['USERID'];
    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');

    $RES = $MYSQLI->QUERY("SELECT LOGIN, ROLE FROM USERS WHERE
ID='$USERID'");
    $USER = $RES->FETCH_OBJECT();
    $LOGIN = $USER->LOGIN;
    $ROLE = $USER->ROLE;
    IF($ROLE == 'ADMIN'){
        $ADMIN = TRUE;
    }
}

IF(ISSET($_POST['PLACE_ID'])) {
    $ID = $_POST['PLACE_ID'];
    $PZ = 'КУПЛЕНО';
      $STMT = $MYSQLI->PREPARE("UPDATE TICKETS SET PZ=? WHERE PLACE_ID=?");
      $STMT->BIND_PARAM('SI',$PZ,$ID);
```

```
      $STMT->EXECUTE();
}
ECHO '<DIV CLASS=FORMINPUT> <H3>БИЛЕТ КУПЛЕН<H3></DIV><HR>';
$RES = $MYSQLI->QUERY("SELECT PLACES.ID AS IDPLACE,
DATE_FORMAT(TICKETS.DATE_RES, '%D.%M.%Y %H:%I') AS DATE, TICKETS.PZ AS PZ,
PLACES.ROW AS R, PLACES.PLACE AS PLACE, PLACES.PRICE AS PRICE, PLACES.NUMBER
AS NUMBER, SPECTACLES.NAME AS NAME FROM TICKETS INNER JOIN PLACES ON TICK-
ETS.PLACE_ID=PLACES.ID INNER JOIN SESSIONS ON PLACES.SESSION_ID=SESSIONS.ID
INNER JOIN SPECTACLES ON SESSIONS.SPECTACLE_ID=SPECTACLES.ID WHERE TICK-
ETS.USER_ID=$USERID");
ECHO "<TABLE>";
IF(!$ADMIN){
    ECHO "<TR><TH>СПЕКТАКЛЬ</TH><TH>ДАТА
СЕАНСА</TH><TH>РЯД</TH><TH>МЕСТО</TH><TH>ЦЕНА</TH><TH>СТАТУС</TH></TR>";
}
ELSE{
    ECHO "<TR><TH>ЛОГИН ПОЛЬЗОВАТЕЛЯ</TH><TH>СПЕКТАКЛЬ</TH><TH>ДАТА
СЕАНСА</TH><TH>РЯД</TH><TH>МЕСТО</TH><TH>ЦЕНА</TH><TH>СТАТУС</TH></TR>";
}
WHILE ($TICKET =  $RES->FETCH_OBJECT()) {
    $DATE = $TICKET->DATE;
    $PZ = $TICKET->PZ;

    $ROW = $TICKET->R;
    $PLACE = $TICKET->PLACE;
    $PRICE = $TICKET->PRICE;
    $ID = $TICKET->IDPLACE;

    IF($ADMIN){
        $LOGIN = $TICKET->LOGIN;
    }

    $NAME = $TICKET->NAME;
    IF($PZ === 'БРОНЬ'){
        IF($ADMIN){
            ECHO
"<TR><TD>$LOGIN</TD><TD>$NAME</TD><TD>$DATE</TD><TD>$ROW</TD><TD>$PLACE</TD><
TD>$PRICE</TD><TD>$PZ</TD><TD><BUTTON ONCLICK=\"DELETERESERV($ID)\">СНЯТЬ
БРОНЬ</BUTTON></TD></TR>";

        }
        ELSE
        {
```

```
            ECHO
"<TR><TD>$NAME</TD><TD>$DATE</TD><TD>$ROW</TD><TD>$PLACE</TD><TD>$PRICE</TD><
TD>$PZ</TD><TD><BUTTON ONCLICK=\"DELETERESERV($ID)\">СНЯТЬ
БРОНЬ</BUTTON></TD><TD><BUTTON ON-
CLICK=\"BUYTICKET($ID)\">ОПЛАТИТЬ</BUTTON></TD></TR>";
        }
    }
    ELSE{
        ECHO
"<TR><TD>$NAME</TD><TD>$DATE</TD><TD>$ROW</TD><TD>$PLACE</TD><TD>$PRICE</TD><
TD>$PZ</TD></TR>";
    }
}
ECHO "</TABLE>";
```

Листинг Б.10 – Исходный код файла deleteActor.php

```
<?PHP
GLOBAL $ADMIN;
$ADMIN = FALSE;
GLOBAL $LOGIN;
$LOGIN = NULL;

SESSION_START();
IF (ISSET($_SESSION['USERID'])) {
    $USERID = $_SESSION['USERID'];
    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');

    $RES = $MYSQLI->QUERY("SELECT LOGIN, ROLE FROM USERS WHERE
ID='$USERID'");
    $USER = $RES->FETCH_OBJECT();
    $LOGIN = $USER->LOGIN;
    $ROLE = $USER->ROLE;
    IF ($ROLE == 'ADMIN') {
        $ADMIN = TRUE;
    }
}

IF (ISSET($_POST['ACTOR_ID'])) {
    $ACTOR_ID = $_POST['ACTOR_ID'];
    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');
```

```
    $RES = $MYSQLI->QUERY("DELETE FROM ACTORS WHERE ID=$ACTOR_ID");
    IF(!$RES){
        DIE;
    }
}

$MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI->QUERY('SET
CHARACTER SET UTF8');
$RES = $MYSQLI->QUERY('SELECT * FROM ACTORS');
ECHO "<TABLE>";

ECHO "<TR>"; // NEW
$INDEXROW = 0;
WHILE ($ACTOR =  $RES->FETCH_OBJECT()) {
    $ID = $ACTOR->ID;
    $PHOTO = $ACTOR->PHOTO;
    IF ($INDEXROW > 3) {
        ECHO "<TR>";
        $INDEXROW = 0;
    }
    IF ($PHOTO == "") {
        $PHOTO = '../FILES/NOPHOTO.JPG';
    }
    IF ($ADMIN) {
        ECHO "<TD><A ID='ACTOR' HREF='ACTOR.PHP?ACTOR_ID=$ID'><IMG
SRC='$PHOTO' ALT = 'ФОТО АКТЕРА' WIDTH='290' HEIGHT='400'></IMG> $ACTOR->NAME
$ACTOR->SURNAME</A><BUTTON ON-
CLICK=\"DELETEACTOR($ID)\">УДАЛИТЬ</BUTTON><BR></TD>";
    } ELSE {
        ECHO "<TD><A ID='ACTOR' HREF='ACTOR.PHP?ACTOR_ID=$ID'><IMG
SRC='$PHOTO' ALT = 'ФОТО АКТЕРА' WIDTH='290' HEIGHT='400'></IMG> $ACTOR->NAME
$ACTOR->SURNAME</A></TD>";
    }
    $PHOTO_STR = JSON_ENCODE($PHOTO);
    $INDEXROW++;
}
ECHO "</TR></TABLE>";
```

Листинг Б.11 – Исходный код файла deleteReservPlace.php

```
<?PHP
GLOBAL $ADMIN;
$ADMIN = FALSE;
```

```
GLOBAL $LOGIN;
$LOGIN = NULL;

SESSION_START();
IF (ISSET($_SESSION['USERID'])) {
    $USERID = $_SESSION['USERID'];
    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');

    $RES = $MYSQLI->QUERY("SELECT LOGIN, ROLE FROM USERS WHERE
ID='$USERID'");
    $USER = $RES->FETCH_OBJECT();
    $LOGIN = $USER->LOGIN;
    $ROLE = $USER->ROLE;
    IF($ROLE == 'ADMIN'){
        $ADMIN = TRUE;
    }
}

IF(ISSET($_POST['PLACE_ID'])) {
    $ID = $_POST['PLACE_ID'];
    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');
    $STMT = $MYSQLI->PREPARE("DELETE FROM TICKETS WHERE PLACE_ID=?");
    $STMT->BIND_PARAM('I', $ID);
    $STMT->EXECUTE();
}
ECHO '<DIV CLASS=FORMINPUT> <H3>БРОНЬ СНЯТА<H3></DIV><HR>';
$RES = $MYSQLI->QUERY("SELECT USERS.LOGIN AS LOGIN, USERS.EMAIL AS EMAIL,
PLACES.ID AS IDPLACE, DATE_FORMAT(TICKETS.DATE_RES, '%D.%M.%Y %H:%I') AS
DATE, TICKETS.PZ AS PZ, PLACES.ROW AS R, PLACES.PLACE AS PLACE, PLACES.PRICE
AS PRICE, PLACES.NUMBER AS NUMBER, SPECTACLES.NAME AS NAME FROM TICKETS INNER
JOIN PLACES ON TICKETS.PLACE_ID=PLACES.ID INNER JOIN SESSIONS ON PLAC-
ES.SESSION_ID=SESSIONS.ID INNER JOIN SPECTACLES ON SES-
SIONS.SPECTACLE_ID=SPECTACLES.ID INNER JOIN USERS ON TICKETS.USER_ID=USERS.ID
WHERE TICKETS.PZ='БРОНЬ'");
ECHO "<TABLE>";
IF(!$ADMIN){
    ECHO "<TR><TH>СПЕКТАКЛЬ</TH><TH>ДАТА
СЕАНСА</TH><TH>РЯД</TH><TH>МЕСТО</TH><TH>ЦЕНА</TH><TH>СТАТУС</TH></TR>";
}
ELSE{
```

```
    ECHO      "<TR><TH>ЛОГИН      ПОЛЬЗОВАТЕЛЯ</TH><TH>СПЕКТАКЛЬ</TH><TH>ДАТА
СЕАНСА</TH><TH>РЯД</TH><TH>МЕСТО</TH><TH>ЦЕНА</TH><TH>СТАТУС</TH></TR>";
}
WHILE ($TICKET =  $RES->FETCH_OBJECT()) {
    $DATE = $TICKET->DATE;
    $PZ = $TICKET->PZ;

    $ROW = $TICKET->R;
    $PLACE = $TICKET->PLACE;
    $PRICE = $TICKET->PRICE;
    $ID = $TICKET->IDPLACE;

    IF($ADMIN){
        $LOGIN = $TICKET->LOGIN;
    }

    $NAME = $TICKET->NAME;
    IF($PZ === 'БРОНЬ'){
        IF($ADMIN){
            ECHO
"<TR><TD>$LOGIN</TD><TD>$NAME</TD><TD>$DATE</TD><TD>$ROW</TD><TD>$PLACE</TD><
TD>$PRICE</TD><TD>$PZ</TD><TD><BUTTON      ONCLICK=\"DELETERESERV($ID)\">СНЯТЬ
БРОНЬ</BUTTON></TD></TR>";

        }
        ELSE
        {
            ECHO
"<TR><TD>$NAME</TD><TD>$DATE</TD><TD>$ROW</TD><TD>$PLACE</TD><TD>$PRICE</TD><
TD>$PZ</TD><TD><BUTTON                      ONCLICK=\"DELETERESERV($ID)\">СНЯТЬ
БРОНЬ</BUTTON></TD><TD><BUTTON>ОПЛАТИТЬ</BUTTON></TD></TR>";
        }
    }
    ELSE{
        ECHO
"<TR><TD>$NAME</TD><TD>$DATE</TD><TD>$ROW</TD><TD>$PLACE</TD><TD>$PRICE</TD><
TD>$PZ</TD></TR>";
    }
}
ECHO "</TABLE>";
```

Листинг Б.12 – Исходный код файла deleteSpectacle.php

```
<?PHP
GLOBAL $ADMIN;
$ADMIN = FALSE;
GLOBAL $LOGIN;
$LOGIN = NULL;

SESSION_START();
IF (ISSET($_SESSION['USERID'])) {
    $USERID = $_SESSION['USERID'];
    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');

    $RES = $MYSQLI->QUERY("SELECT LOGIN, ROLE FROM USERS WHERE
ID='$USERID'");
    $USER = $RES->FETCH_OBJECT();
    $LOGIN = $USER->LOGIN;
    $ROLE = $USER->ROLE;
    IF($ROLE == 'ADMIN'){
        $ADMIN = TRUE;
    }
}

IF(ISSET($_POST['SPECTACLE_ID'])) {
    $ID = $_POST['SPECTACLE_ID'];
    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI-
>QUERY('SET CHARACTER SET UTF8');
    $STMT = $MYSQLI->PREPARE("DELETE FROM SPECTACLES WHERE ID=?");
    $STMT->BIND_PARAM('I', $ID);
    $STMT->EXECUTE();
}

$MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI->QUERY('SET
CHARACTER SET UTF8');
$RES = $MYSQLI->QUERY('SELECT * FROM SPECTACLES');
ECHO "<TABLE>";
ECHO "<TR>"; // NEW
$INDEXROW = 0;
WHILE ($SPECTACLE = $RES->FETCH_OBJECT()) {
    $ID = $SPECTACLE->ID;
    $PHOTO = $SPECTACLE->POSTER;
```

```
    $NAME = $SPECTACLE->NAME;
    IF ($INDEXROW > 4) {
        ECHO "<TR>";
        $INDEXROW = 0;
    }
    IF ($PHOTO == "") {
        $PHOTO = '../FILES/NOPHOTO.JPG';
    }
    IF ($ADMIN) {
        ECHO  "<TD><A  ID='ACTOR'  HREF='SPECTACLE.PHP?SPECTACLE_ID=$ID'><IMG
SRC='$PHOTO'        ALT        =        'ФОТО        АФИШИ'        WIDTH='290'
HEIGHT='400'></IMG>$NAME</A><BR><BUTTON                                    ON-
CLICK=\"DELETESPECTACLE('../DELETESPECTACLE.PHP',$ID)\">УДАЛИТЬ</BUTTON></TD>
";
    } ELSE {
        ECHO  "<TD><A  ID='ACTOR'  HREF='SPECTACLE.PHP?SPECTACLE_ID=$ID'><IMG
SRC='$PHOTO'        ALT        =        'ФОТО        АФИШИ'        WIDTH='290'
HEIGHT='400'></IMG>$NAME</A></TD>";
    }
    $PHOTO_STR = JSON_ENCODE($PHOTO);
    $INDEXROW++;
}
ECHO "</TR></TABLE>";
```

Листинг Б.13 – Исходный код файла login.php

```
<?PHP
SESSION_START();
$ERRORS = ARRAY();
IF(ISSET($_SESSION['USERID']))
{
      //АУТЕНТИФИКАЦИЯ ИЗ СЕССИИ
      $USERID=$_SESSION['USERID'];
      HEADER("LOCATION:INDEX.PHP");
}
ELSE IF(ISSET($_COOKIE['TOKEN']))
{
      //АУТЕНТИФИКАЦИЯ ИЗ COOKIE
      $TOKEN = $_COOKIE['TOKEN'];
      CONNECTDB();
      $RES = $MYSQLI->QUERY("SELECT * FROM USERS WHERE TOKEN = '$TOKEN'");
      IF($RES->NUM_ROWS==1)
      {
```

```
            $USER = $RES->FETCH_OBJECT();
            LOGIN($USER->ID);
            HEADER("LOCATION:INDEX.PHP");
      }
}
ELSE IF(ISSET($_POST['LOGIN']) && ISSET($_POST['PASSWORD']))
{
            //АУТЕНТИФИКАЦИЯ ИЗ ФОРМЫ
            $LOGIN = $_POST['LOGIN'];
            $PWD = $_POST['PASSWORD'];
            CONNECTDB();
            $RES = $MYSQLI->QUERY("SELECT * FROM USERS WHERE LOGIN =
'$LOGIN'");
            IF($RES->NUM_ROWS == 1)
            {
                  $USER = $RES->FETCH_OBJECT();
                  $RANDOM = $USER->RND;
                  $PWDHASH = SHA1($PWD.$RANDOM);
                  IF($PWDHASH == $USER->PASSWORD)
                  {
                        LOGIN($USER->ID);
                        HEADER("LOCATION:INDEX.PHP");
                  }
                  ELSE
                  {
                        $ERRORS[] = "НЕПРАВИЛЬНЫЙ ПАРОЛЬ.";
                  }
            }
            ELSE
            {
                  $ERRORS[] = "ПОЛЬЗОВАТЕЛЯ С ТАКИМ ЛОГИНОМ НЕ СУЩЕСТВУЕТ.";
            }
            IF(!EMPTY($ERRORS))
            {
                  $ERROR = ARRAY_SHIFT($ERRORS);
                  HEADER("LOCATION:AUTHFORM.PHP?ERROR=$ERROR");
            }
}
?>

<?PHP
FUNCTION CONNECTDB()
{
```

```
      GLOBAL $MYSQLI;
      $MYSQLI = NEW MYSQLI('LOCALHOST','ROOT','','THEATRE');
}

FUNCTION LOGIN($USERID)
{
      GLOBAL $MYSQLI;
      $_SESSION['USERID'] = $USERID;
      $TOKEN = SPRINTF('%08X%08X%08X%08X', RAND(), RAND(), RAND(), RAND());
      SETCOOKIE('TOKEN',$TOKEN,TIME()+3600*24*30);
      $RES = $MYSQLI->QUERY("UPDATE USERS SET TOKEN = '$TOKEN' WHERE ID =
'$USERID'");
}
?> }
    $PHOTO_STR = JSON_ENCODE($PHOTO);
    $INDEXROW++;
}
ECHO "</TR></TABLE>";
```

Листинг Б.14 – Исходный код файла registration.php

```
<?PHP
IF(ISSET($_POST['LOGIN']) && ISSET($_POST['PASSWORD'])&& IS-
SET($_POST['PASSWORD2']))
{
      $LOGIN = $_POST['LOGIN'];
      $PWD = $_POST['PASSWORD'];
    $PWD2 = $_POST['PASSWORD2'];
    $EMAIL = $_POST['EMAIL'];

    IF($LOGIN=='')
    {
        $ERROR = 'ВЫ НЕ ВВЕЛИ ЛОГИН.';
            HEADER("LOCATION:REGISTRATIONFORM.PHP?ERROR=$ERROR");
    }
    ELSE
    {
        IF($PWD==$PWD2)
        {
            IF($PWD!='')
            {
                $MYSQLI = NEW MYSQLI('LOCALHOST','ROOT','','THEATRE');
```

```
                $ANSW = $MYSQLI->QUERY("SELECT * FROM USERS WHERE LOGIN =
'$LOGIN'");
                IF($ANSW->NUM_ROWS >= 1)
                {
                    $ERROR = 'ЭТОТ ЛОГИН УЖЕ ЗАНЯТ.';
                    HEADER("LOCATION:REGISTRATIONFORM.PHP?ERROR=$ERROR");
                }
                ELSE
                {
                    IF(!STRISTR($EMAIL, '@')){
                        $ERROR = 'НЕКОРРЕКТНАЯ ПОЧТА';
                        HEADER("LOCATION:REGISTRATIONFORM.PHP?ERROR=$ERROR");
                    }
                    ELSE{
                        $MYSQLI = NEW
MYSQLI('LOCALHOST','ROOT','','THEATRE');
                        $ANSW = $MYSQLI->QUERY("SELECT * FROM USERS WHERE
EMAIL = '$EMAIL'");
                        IF($ANSW->NUM_ROWS >= 1)
                        {
                            $ERROR = 'ЭТА ПОЧТА УЖЕ ИСПОЛЬЗУЕТСЯ.';
                            HEAD-
ER("LOCATION:REGISTRATIONFORM.PHP?ERROR=$ERROR");
                        }

                        $RND =SPRINTF('%08X%08X%08X%08X', RAND(), RAND(),
RAND(), RAND());
                        $PWDHASH = SHA1($PWD.$RND);
                        $TOKEN = 'A';
                        $SQL = "INSERT INTO USERS (LOGIN, PASSWORD, RND, TO-
KEN, EMAIL) VALUES ('$LOGIN','$PWDHASH','$RND','$TOKEN','$EMAIL')";
                        ECHO $SQL;
                        $RES = $MYSQLI->QUERY($SQL);
                        IF($RES){
                            //HEADER("LOCATION:REGISTRATIONFORM.PHP?ACCESS");
                        }
                        ELSE{
                            $ERROR = 'НЕ УДАЛОСЬ ЗАРЕГИСТРИРОВАТЬ ПОЛЬЗОВАТЕ-
ЛЯ';

//HEADER("LOCATION:REGISTRATIONFORM.PHP?ERROR=$ERROR");
                        }
                    }
```

```
                }
            }
            ELSE
            {
                $ERROR = 'ВЫ НЕ ВВЕЛИ ПАРОЛЬ.';
                    HEADER("LOCATION:REGISTRATIONFORM.PHP?ERROR=$ERROR");
            }
        }
        ELSE
        {
            $ERROR = 'ПАРОЛИ НЕ СОВПАДАЮТ';
            HEADER("LOCATION:REGISTRATIONFORM.PHP?ERROR=$ERROR");
        }
    }
}
?>
```

Листинг Б.15 – Исходный код файла registrationForm.php

```
<!DOCTYPE HTML>
<HTML>

<HEAD>
      <META CHARSET="UTF-8">
      <TITLE>TEATP </TITLE>
      <LINK HREF="STYLE.CSS" REL="STYLESHEET" TYPE="TEXT/CSS">
</HEAD>

<BODY>
      <DIV CLASS="HEADER">
            <A CLASS="LOGO" HREF="INDEX.PHP"><IMG SRC="LOGOT.JPG" ALT=""></A>
            <DIV CLASS="TOP-MENU">
                  <UL>
                        <LI><A HREF="INDEX.PHP">ГЛАВНАЯ</A></LI>
                        <LI><A HREF="PAGES/ACTORS.PHP">АКТЁРЫ</A></LI>
                        <LI><A HREF="PAGES/SPECTACLES.PHP">СПЕКТАКЛИ</A></LI>
                        <DIV CLASS="AUTH">
                              <?PHP
                              IF (ISSET($_SESSION['USERID'])) {
                                    $USERID = $_SESSION['USERID'];
                                    $MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT',
'', 'SHOE_STORE');
```

```
                                    $RES = $MYSQLI->QUERY("SELECT LOGIN FROM
USERS WHERE ID='$USERID'");
                                    $USER = $RES->FETCH_OBJECT();
                                    $LOGIN = $USER->LOGIN;

                                    ECHO "<LI><A
HREF='PAGES/USERTICKETS.PHP'>ВЫ ВОШЛИ КАК $LOGIN </A></LI>
                        <LI><A HREF='EXIT.PHP'>ВЫХОД</A></LI>";
                              } ELSE {
                                    ECHO "<LI><A
HREF='AUTHFORM.PHP'>ВХОД</A></LI>
                        <LI><A
HREF='REGISTRATIONFORM.PHP'>РЕГИСТРАЦИЯ</A></LI>";
                              }
                              ?>
                  </UL>
            </DIV>
      </DIV>
      <DIV CLASS="CONTENT">
            <DIV CLASS='FORMINPUT'>
                  <H2>РЕГИСТРАЦИЯ</H2>
                  <?PHP
                  IF (ISSET($_GET['ERROR'])) {
                        $ERROR = $_GET['ERROR'];
                        ECHO '<DIV STYLE="COLOR: RED;">' . $ERROR .
'</DIV><HR>';
                  }
                  IF (ISSET($_GET['ACCESS'])) {
                        ECHO '<DIV STYLE="COLOR: GREEN;">РЕГИСТРАЦИЯ ПРОШЛА
УСПЕШНО!</DIV><HR>';
                  }
                  ECHO "<FORM ACTION='REGISTRATION.PHP' METHOD ='POST'>
            <P>ЛОГИН <INPUT TYPE='TEXT' NAME ='LOGIN' VALUE=''></P>
                  <P>ПОЧТА <INPUT TYPE='TEXT' NAME ='EMAIL' VALUE=''></P>
            <P>ПАРОЛЬ <INPUT TYPE='PASSWORD' NAME ='PASSWORD' VALUE=''></P>
            <P>ПРОВЕРКА ПАРОЛЯ <INPUT TYPE='PASSWORD' NAME ='PASSWORD2' VAL-
UE=''></P>
            <P><BUTTON>ЗАРЕГИСТИРОВАТЬСЯ</BUTTON></P>
            </FORM>";
                  ?>
            </DIV>
      </DIV>
      <DIV CLASS="FOOTER">
```

```
            Г. ПЕНЗА ТЕЛЕФОН 65-55-21<BR>
          ГОРЯЧАЯ ЛИНИЯ
      </DIV>
</BODY>
</HTML>?>
```

Листинг Б.16 – Исходный код файла selectSessionsForSpectacle.php

```
<?PHP
SESSION_START();
REQUIRE_ONCE('JSON/JSON.PHP');
//REQUIRE('PAGES/SPECTACLE.PHP');

$ITEMS = ARRAY();

$JSON = NEW SERVICES_JSON();

$SPECTACLE_ID = $_SESSION['SPECTACLE_ID'];

$MYSQLI = NEW MYSQLI('LOCALHOST', 'ROOT', '', 'THEATRE'); $MYSQLI->QUERY('SET
CHARACTER SET UTF8');
$RESSESSIONS = $MYSQLI->QUERY("SELECT ID, DATE_FORMAT(DATE, '%D.%M.%Y %H:%I')
AS DATE FROM SESSIONS WHERE SPECTACLE_ID=$SPECTACLE_ID");

$NROWS = $RESSESSIONS->NUM_ROWS;
$ITEMS = ARRAY();

$PRICESECTOR1 = 0;
$PRICESECTOR2 = 0;
$PRICESECTOR3 = 0;

FOR($I=0; $I<$NROWS; $I++)
{
    $SESSION = $RESSESSIONS->FETCH_OBJECT();
    $IDSESSION = $SESSION->ID;

    $SQL = "SELECT PRICE FROM PLACES WHERE SESSION_ID=$IDSESSION AND PLAC-
ES.NUMBER=1";

    $RES = @$MYSQLI->QUERY("SELECT PRICE FROM PLACES WHERE SES-
SION_ID=$IDSESSION AND PLACES.NUMBER=1");
    IF($RES){
        $ANSW = $RES->FETCH_OBJECT();
```

```
        $PRICESECTOR1 = $ANSW->PRICE;
    }



    $RES = @$MYSQLI->QUERY("SELECT PRICE FROM PLACES WHERE SES-
SION_ID=$IDSESSION AND PLACES.NUMBER=28");
    IF($RES){
        $ANSW = $RES->FETCH_OBJECT();
        $PRICESECTOR2 = $ANSW->PRICE;
    }

    $RES = @$MYSQLI->QUERY("SELECT PRICE FROM PLACES WHERE SES-
SION_ID=$IDSESSION AND PLACES.NUMBER=81");
    IF($RES){
        $ANSW = $RES->FETCH_OBJECT();
        $PRICESECTOR3 = $ANSW->PRICE;
    }

    $ITEMS[$I] = ARRAY(
        'ID' => $SESSION->ID,
        'DATE' => $SESSION->DATE,
        'PRICESECTOR1' => $PRICESECTOR1,
        'PRICESECTOR2' => $PRICESECTOR2,
        'PRICESECTOR3' => $PRICESECTOR3

    );
}
/*WHILE($ACTOR = $RES->FETCH_OBJECT())
{
    $ITEMS[$I] = ARRAY(
        'SURNAME' => $ACTOR->SURNAME,
        'NAME' => $ACTOR->NAME,
        'AGE' => $ACTOR->AGE,
        'EXPERIENCE' => $ACTOR->EXPERIENCE
    );
    $I++;
}*/

$ARR = ARRAY('ITEMS'=> $ITEMS, 'TOTALCOUNT'=> $NROWS);
$OUTPUT = $JSON->ENCODE($ARR);
ECHO($OUTPUT);
?>
```

Приложение В

ТЕСТИРОВАНИЯ

(рекомендуемое)

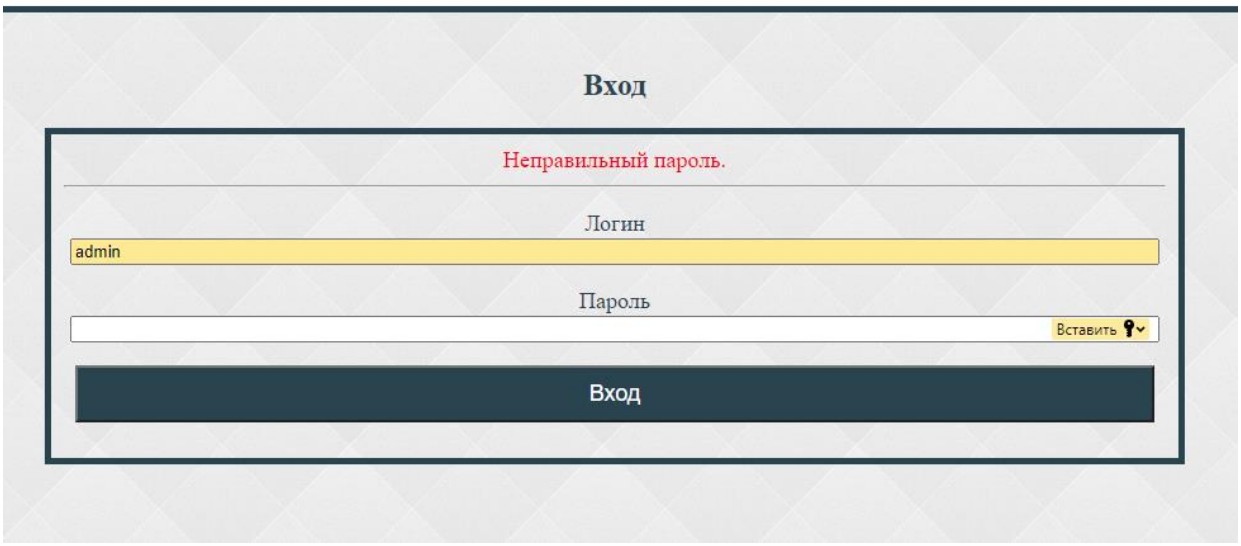

Рисунок В.1 – Результат выполнения теста №2

Рисунок В.2 – Результат выполнения теста №3

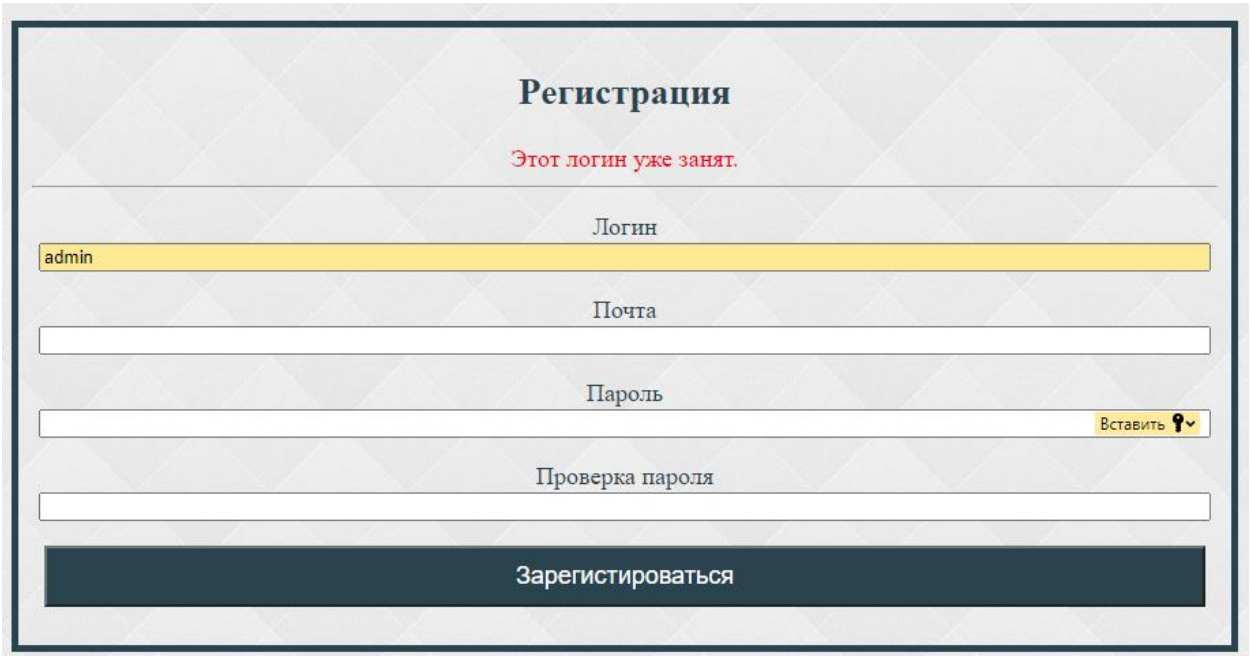

Рисунок В.3 – Результат выполнения теста №4

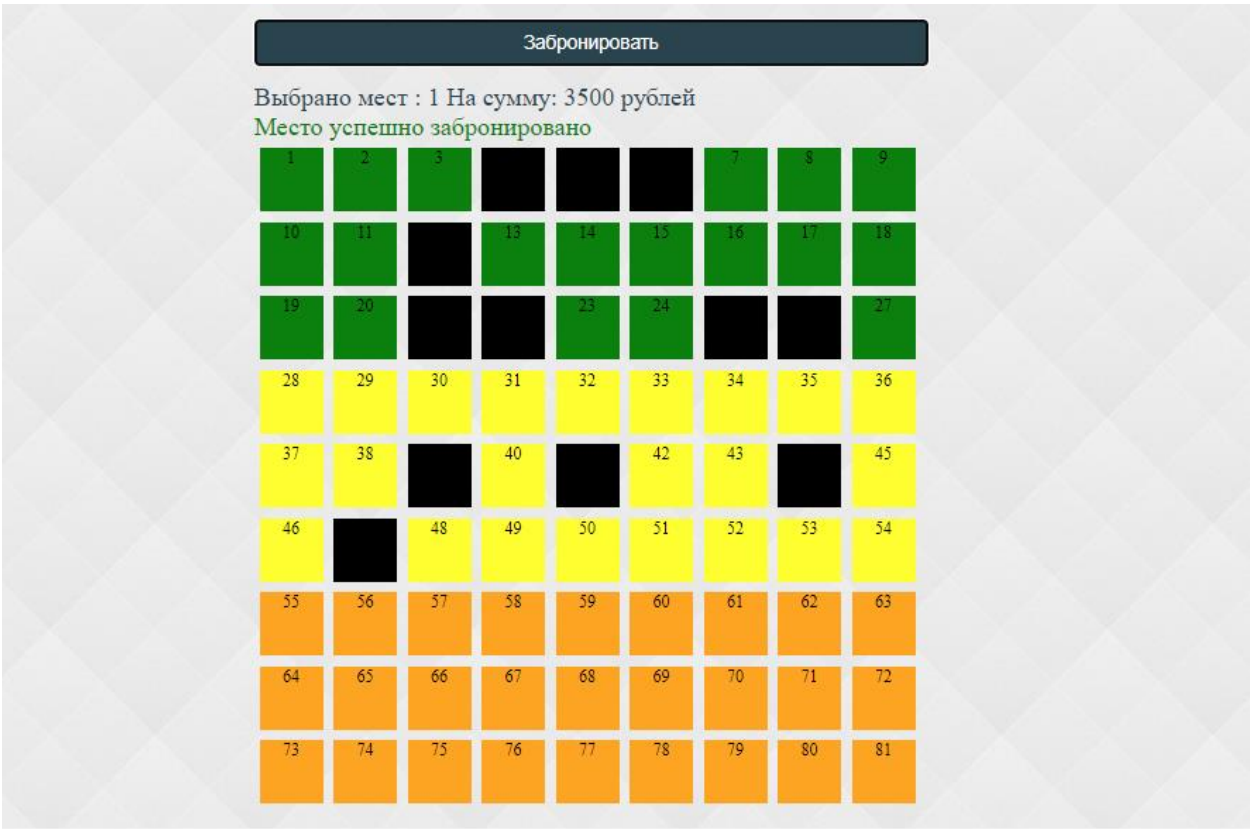

Рисунок В.4 – Результат выполнения теста №5

Рисунок В.5 – Результат выполнения теста №7

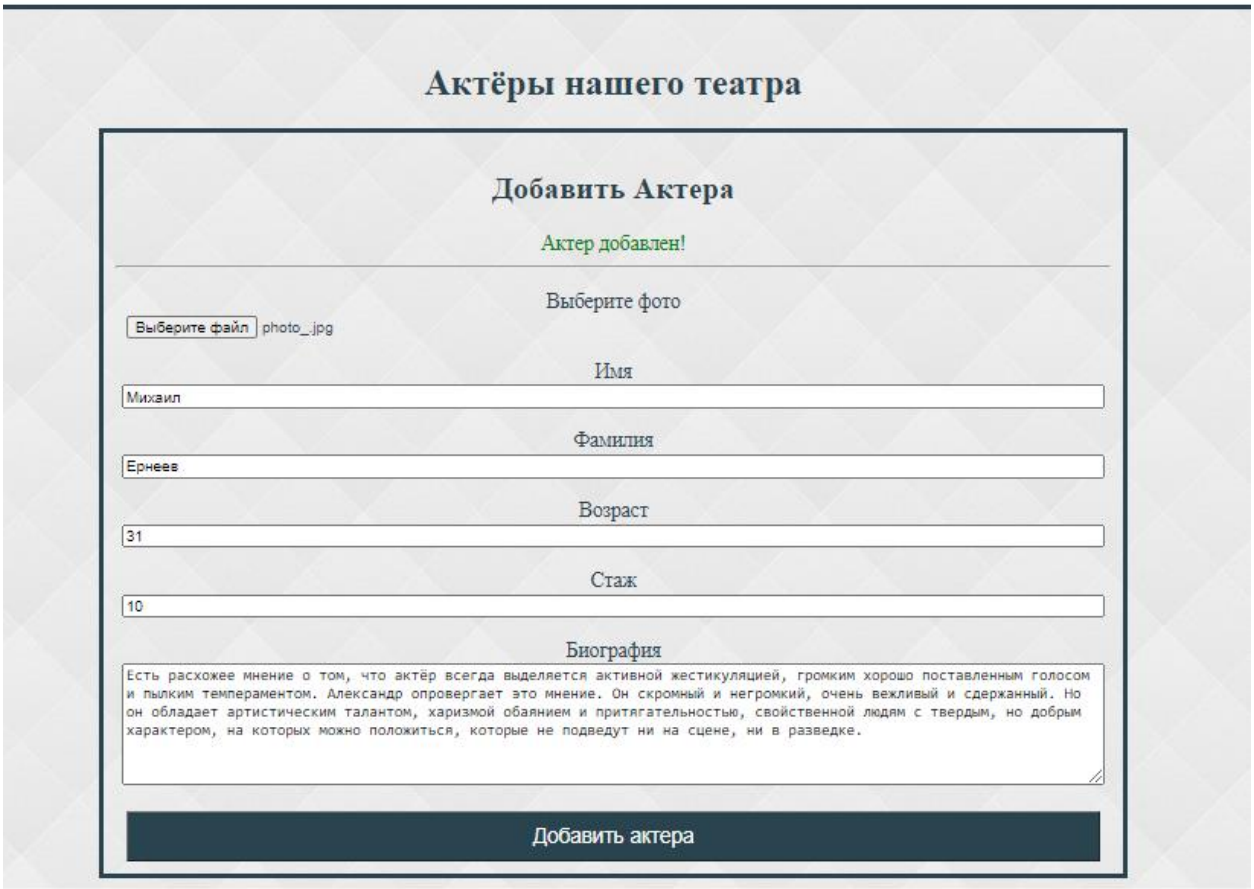

Рисунок В.6 – Результат выполнения теста №8

Рисунок В.7 – Результат выполнения теста №9

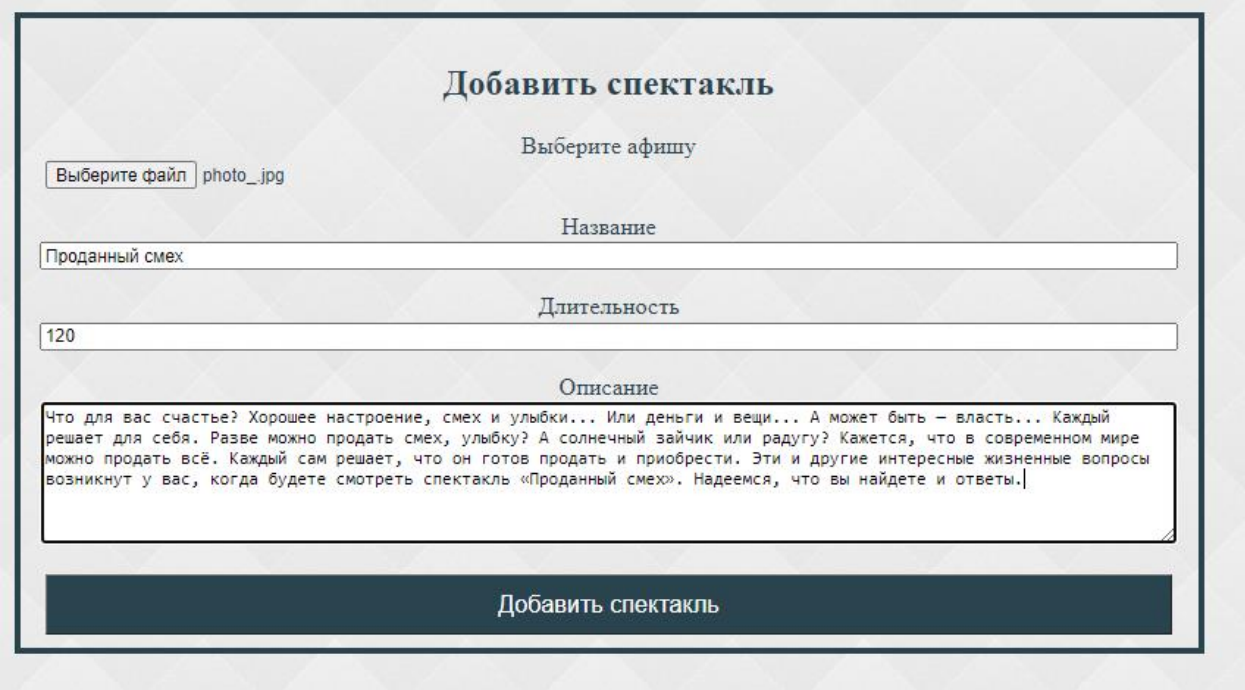

Рисунок В.8 – Результат выполнения теста №12

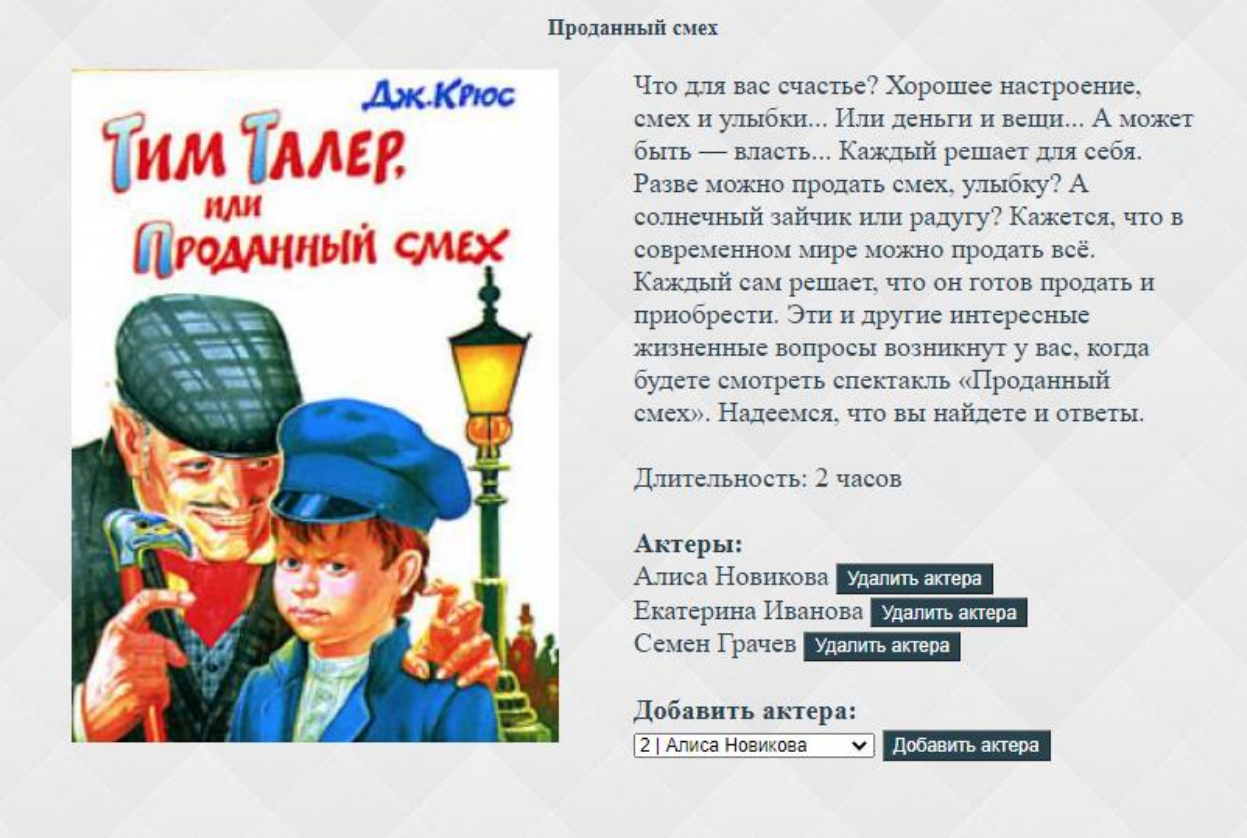

Рисунок В.9 – Результат выполнения теста №12

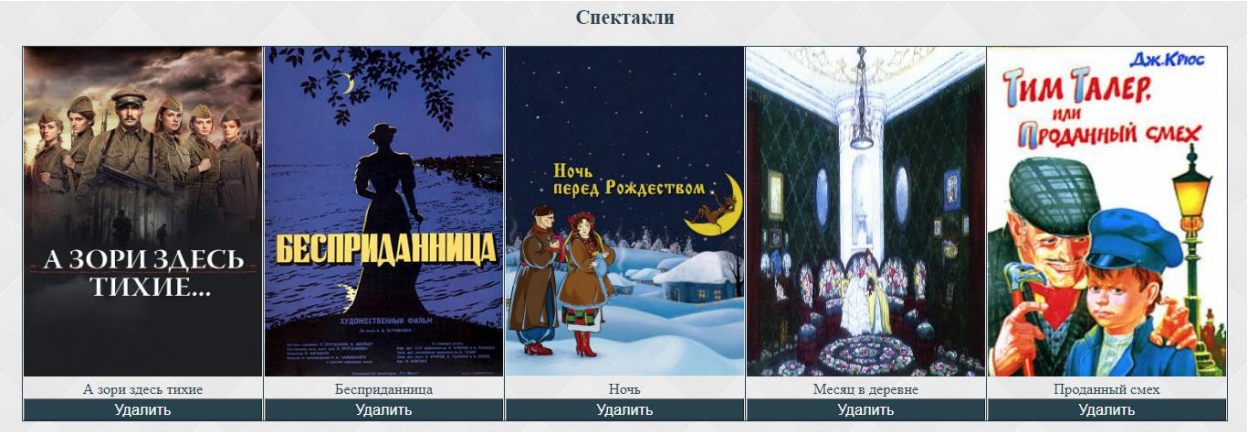

Рисунок В.10 – Результат выполнения теста №12

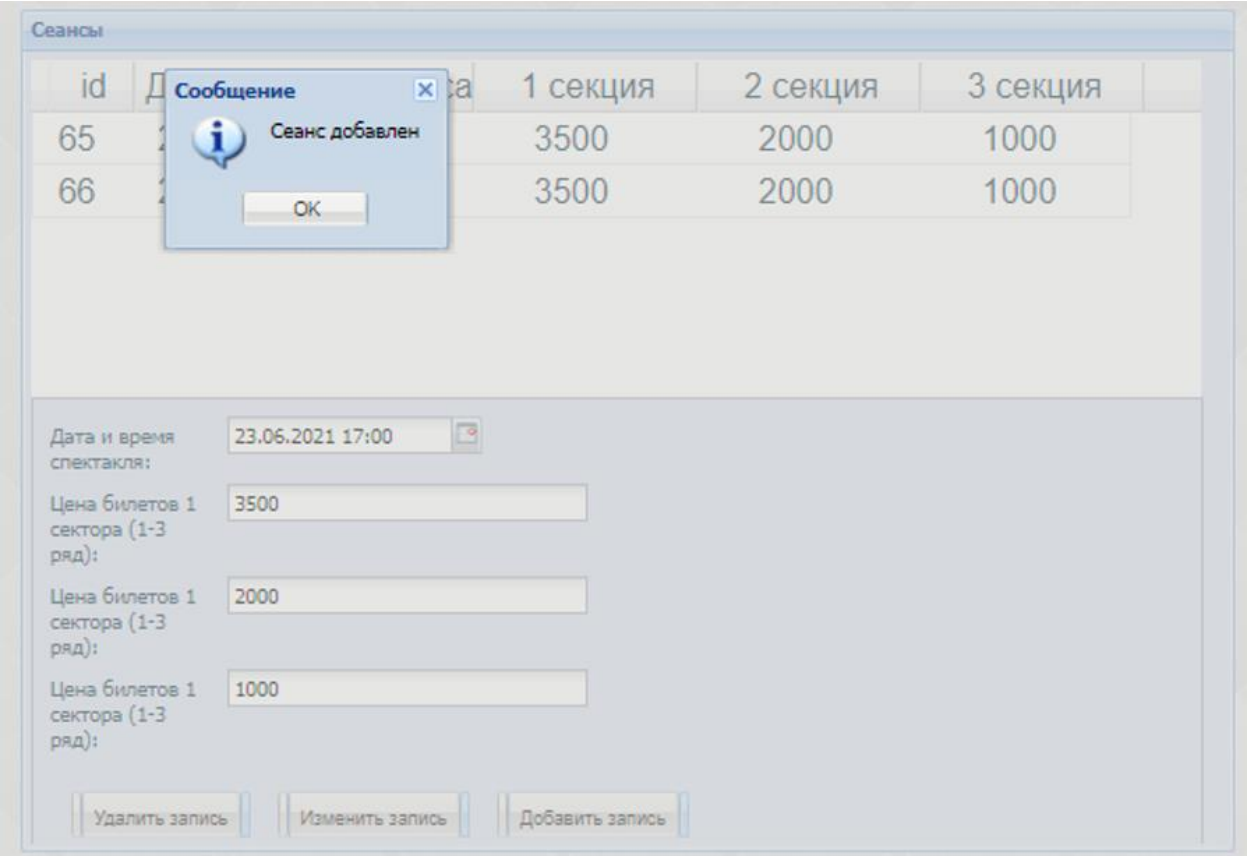

Рисунок В.11 – Результат выполнения теста №15

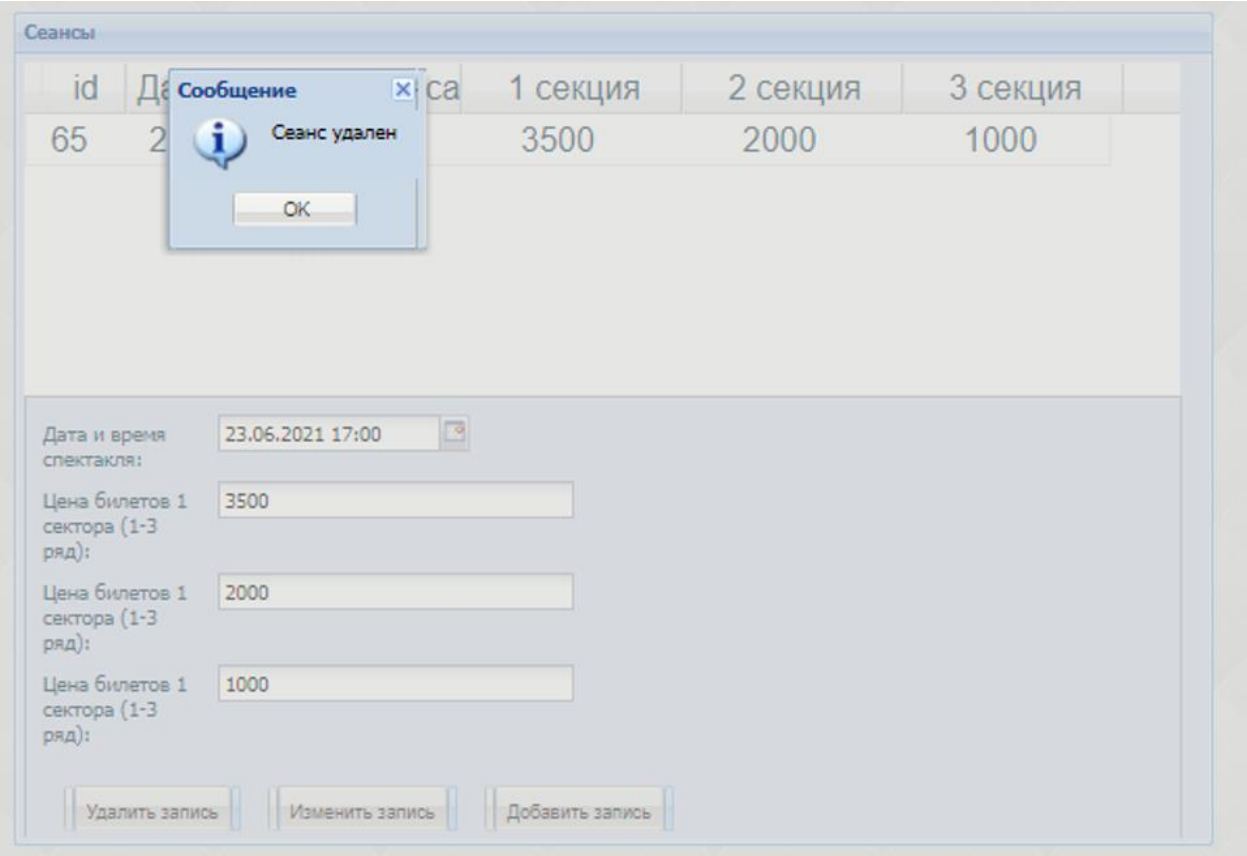

Рисунок В.12 – Результат выполнения теста №16

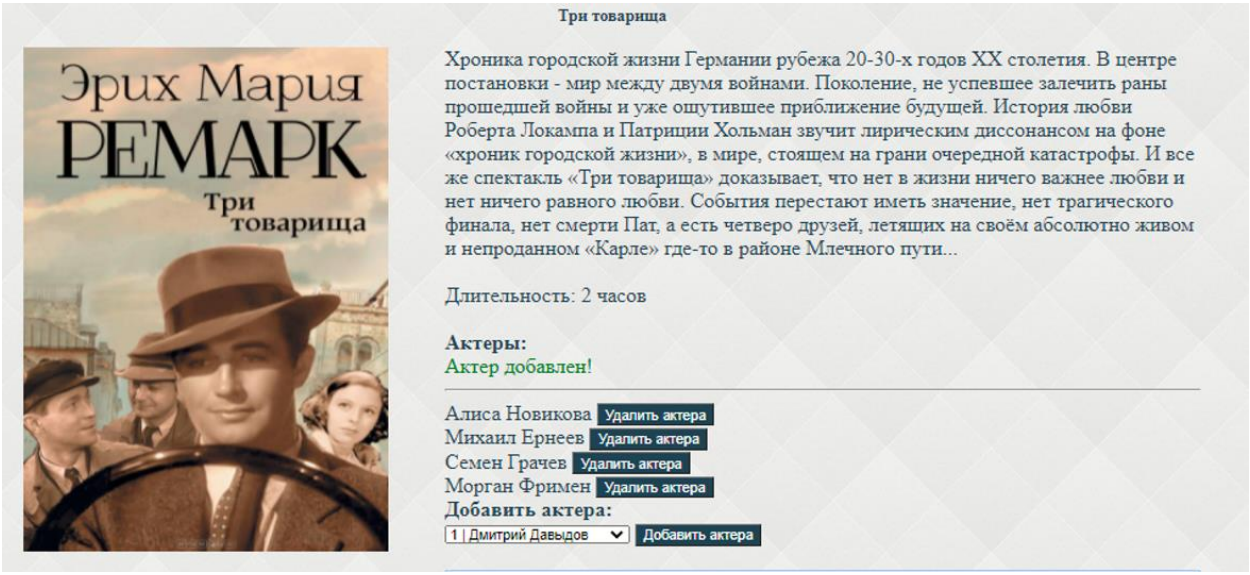

Рисунок В.13 – Результат выполнения теста №17

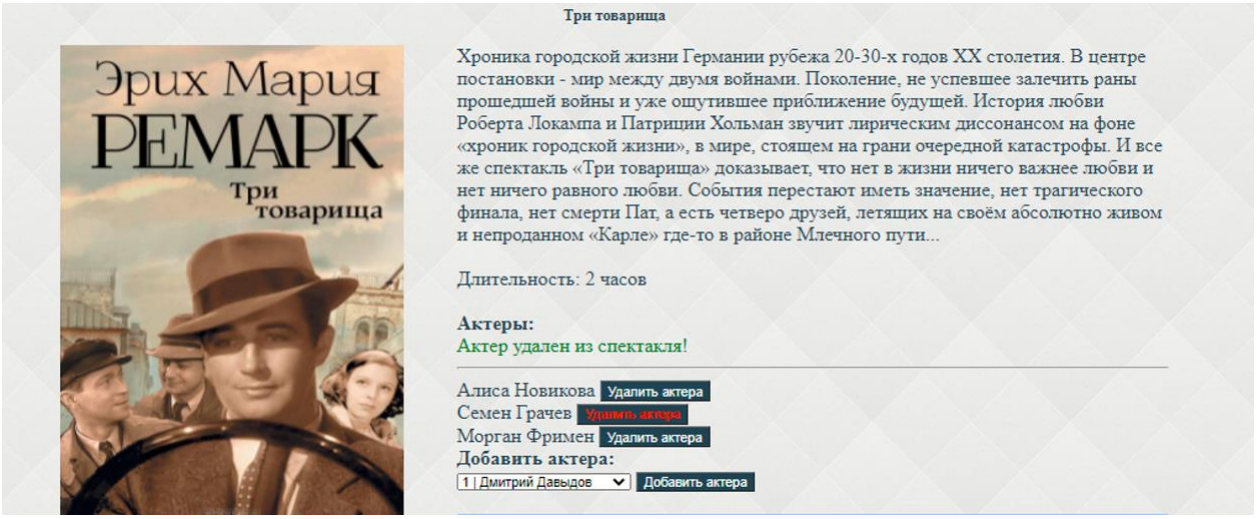

Рисунок В.14 – Результат выполнения теста №18

Приложение Г

СКРИПТ DDL

(рекомендуемое)

Листинг Г.1 – Скрипт создания таблиц базы данных на языке SQL

```
CREATE TABLE actors
(
      id_actors  CHAR(18) NULL,
      fio  CHAR(18) NULL,
      age  CHAR(18) NULL,
      photo  CHAR(18) NULL,
      stage  CHAR(18) NULL
);

ALTER TABLE actors
      ADD  PRIMARY KEY (id_actors);

CREATE TABLE places
(
      id_place  CHAR(18) NULL,
      id_session  CHAR(18) NULL,
      row  CHAR(18) NULL,
      place  CHAR(18) NULL,
      price  CHAR(18) NULL
);

ALTER TABLE places
      ADD  PRIMARY KEY (id_place);

CREATE TABLE sessions
(
      id_session  CHAR(18) NULL,
      id_spectacle  CHAR(18) NULL,
      date_start  CHAR(18) NULL
);

ALTER TABLE sessions
      ADD  PRIMARY KEY (id_session);

CREATE TABLE spectacles
(
      id_spectacle  CHAR(18) NULL,
      name  CHAR(18) NULL,
      duration  CHAR(18) NULL,
      info  CHAR(18) NULL,
      photo  CHAR(18) NULL
);

ALTER TABLE spectacles
      ADD  PRIMARY KEY (id_spectacle);

CREATE TABLE spectacles_has_actors
(
      id_actors  CHAR(18) NOT NULL,
      id_spectacle  CHAR(18) NOT NULL,
```

```
      id_has  CHAR(18) NULL
);

ALTER TABLE spectacles_has_actors
      ADD  PRIMARY KEY (id_has);

CREATE TABLE tickets
(
      id_ticket  CHAR(18) NULL,
      id_user  CHAR(18) NULL,
      pz  CHAR(18) NULL,
      date_res  CHAR(18) NULL,
      id_place  CHAR(18) NULL
);

ALTER TABLE tickets
      ADD  PRIMARY KEY (id_ticket);

CREATE TABLE users
(
      id_user  CHAR(18) NULL,
      login  CHAR(18) NULL,
      pawword  CHAR(18) NULL,
      fio  CHAR(18) NULL,
      role  CHAR(18) NULL,
      email  CHAR(18) NULL
);

ALTER TABLE users
      ADD  PRIMARY KEY (id_user);

ALTER TABLE places
      ADD FOREIGN KEY R_8 (id_session) REFERENCES sessions(id_session);

ALTER TABLE sessions
      ADD FOREIGN KEY R_7 (id_spectacle) REFERENCES spectacles(id_spectacle);

ALTER TABLE spectacles_has_actors
      ADD FOREIGN KEY R_1 (id_actors) REFERENCES actors(id_actors);

ALTER TABLE spectacles_has_actors
      ADD FOREIGN KEY R_2 (id_spectacle) REFERENCES spectacles(id_spectacle);

ALTER TABLE tickets
      ADD FOREIGN KEY R_3 (id_user) REFERENCES users(id_user);

ALTER TABLE tickets
      ADD FOREIGN KEY R_10 (id_place) REFERENCES places(id_place);
```