Частное учреждение образования

«Колледж бизнеса и права»

«Создания программного средства для записи на участие в квестах»

Пояснительная записка

к курсовому проекту по дисциплине

«Основы алгоритмизации и программирование»

КП Т.295015.401

Руководитель проекта (Дж.А. Миронова)

Учащийся (И.О.Погорельский)

2024

[Введение 5](#_Toc170269309)

[1 Описание задачи 6](#_Toc170269310)

[1.1 Анализ предметной области 6](#_Toc170269311)

[1.2 Постановка задачи 6](#_Toc170269312)

[2 Проектирование системы 8](#_Toc170269313)

[2.1 Требования к приложению 8](#_Toc170269314)

[2.2 Проектирование модели 8](#_Toc170269315)

[2.3 Концептуальный прототип 9](#_Toc170269316)

[3 Описание реализации программного средства 3](#_Toc170269317)

[3.1 Инструменты разработки и применяемые технологии 3](#_Toc170269318)

[3.2 Функции: логическая и физическая организация 3](#_Toc170269319)

[3.3 Входные и выходные данные 6](#_Toc170269320)

[3.4 Функциональное тестирование 7](#_Toc170269321)

[4 Применение 11](#_Toc170269324)

[4.1 Назначение программного средства 11](#_Toc170269325)

[4.2 Условия применения 11](#_Toc170269326)

[Заключение 12](#_Toc170269327)

[Список информационных источников 14](#_Toc170269328)

[Приложение А Текст Программных Модулей 15](#_Toc170269329)

[Приложение Б Формы входных и выходных документов 39](#_Toc168924701)

Изм.

Лист

№ докум.

Подпись

Датаа

Лист

3

КП Т.295015.401

Разраб.

Погорельский И.О

Провер.

Миронова Дж.А

Реценз.

Н. Контр.

Утверд.

*СОЗДАНИЯ ПРОГРММНОГО СРЕДСТВА ДЛЯ ЗАПИСИ НА УЧАСТИЕ В КВЕСТАХ*

Лит.

Листов

КБиП

у

# Введение

В современном мире трудно представить жизнь без различных форм развлечений и досуга, и одним из самых захватывающих и популярных видов активного отдыха стали квесты. Эти игры, в которых участники решают головоломки и выполняют задания в реальных пространствах, привлекают внимание людей всех возрастов. Квесты предоставляют возможность испытать свои командные навыки и весело провести время.

Сначала может показать что работа с квестами сложная и запутанная, особенно когда дело касается организации и записи на участие, но предоставленное приложение поможет разобраться со всеми нюансами и сделает процесс регистрации и участия максимально простым и удобным.

Целью разработки курсового проектирования на тему «Создание программного средства для записи на участие в квестах» является реализация программного средства, которое позволит пользователям регистрироваться или авторизоваться, просматривать доступные квесты , выбирать удобное время и дату, а также редактировать свои записи и получать необходимую информацию.

Квесты остаются актуальным видом развлечения, привлекающим внимание как новичков, так и опытных игроков. Программное средство для записи на участие в квестах становится важным инструментом для организаторов, обеспечивая удобство для клиентов эффектное управление расписанием квестов, повышение коммуникации с участниками. Эти функции помогают оптимизировать процесс организации квестов , улучшить качество обслуживания и удовлетворить потребности участников , что является ключевым для успешного развития бизнеса в сфере развлечений.

В первом разделе «Описание задачи» рассмотрены основные задачи и функции программного средства для записи на участие в квестах.

Второй раздел «Проектирование системы» представлено разработка механик и функции необходимых для корректной работы приложения.

Третий раздел «Описание реализации программного средства» описывает используемые средства и компоненты для создания программы.

Четвёртый раздел «Применение» рассматривает процесс использования приложения со стороны пользователей.

В Заключении будет проведен анализ результатов и сделаны выводы о дальнейших перспективах развитых проектов.

# Описание задачи

## Анализ предметной области

Целью разработки курсового проектирования на тему «Создание программного средства для записи на участие в квестах» является реализация программного средства, где пользователь имеет возможность входить в свой, или создавать новый аккаунт, просматривать доступные квесты, записываться на участие, добавлять квесты в избранное, узнавать краткое описание квестов.

Квест – это увлекательная игра, в которой ваша команда в течение определенного времени пытается общими усилиями достичь поставленной цели. Обычно квесты проходят в помещении со специальными декорациями, звуком и другими атрибутами, которые помогают создать идеальную атмосферу в зависимости от выбранного квеста.

Компьютерная обработка значительно упрощает взаимодействие с записью на квесты, в отличие от реального визита или записи на месте проведения квеста. Также компьютерная обработка повышает спрос клиентов и их заинтересованность, обеспечивает быстрый доступ к доступным квестам и их стоимости.

Потенциальным пользователем программного средства являются люди разного возраста, так как квесты популярны как среди молодежи, так и среди более возрастных групп. Программное средство разработано с целью охвата наибольшей клиентской базы и максимального упрощения использования.

## Постановка задачи

Задачей проекта является разработка программного средства для просмотра и записи на участие в квестах. Данная программа обеспечивает просмотр и ознакомление с квестами, их бронирование, просмотр цен и оформление заказа на участие.

Программное средство будет включать в себя функционал для множественного поиска по названию квеста, что позволяет пользователям ориентироваться в небольшом, но достаточном объёме данных. Также будет возможность просмотра всех забронированных квестов.

На рисунке 1.1 представлен аналог разрабатываемого программного средства «EXTRA REALLITY»

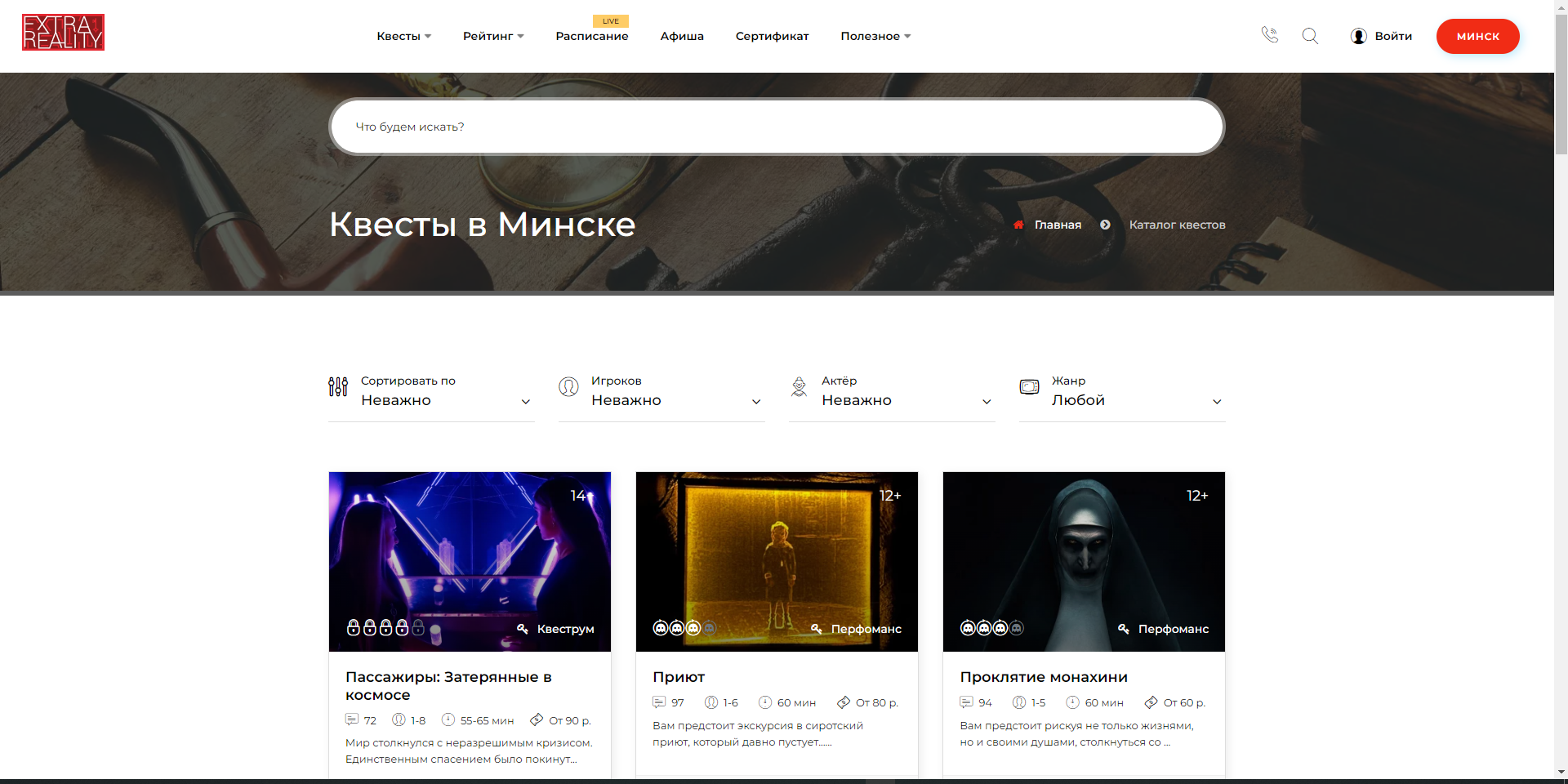


Рисунок 1.1 – Внешний вид

Достоинства аналога:

1.Интерфейс: Простой и интуитивно понятный интерфейс облегчает навигацию и использования приложения.

2.Доступность: Возможность записывать на квесты в любое время и из любого места , используя различные устройства (ПК, планшет, смартфоны).

3.Фильтрация: Разнообразные параметры фильтрации ( по жанру, стоимости, уровню сложности, количеству актёров) помогают быстро находить интересующие квесты.

4.История прохождения квеста. Хранения истории пройденных квестов

Недостатки аналога:

1) Зависимость от интернета: Доступ к приложению: Необходимость постоянного подключения к интернету может ограничивать использование приложения в условиях слабого или отсутствующего интернета

2)Бронирование: Нельзя посмотреть все забронированные квесты для определённого пользователя.

3) Технические проблемы: Ошибки и баги: Возможные ошибки и сбои в работе приложения могут вызвать неудобства и снизить доверие пользователей.

4) Актуальность информации: Обновление данных: Необходимость постоянного обновления информации о доступных квестах и их расписании может потребовать значительных усилий от организаторов.

Разрабатываемое программное средство будет включать в себя различные особенности , позволяющие более комфортнее взаимодействовать с приложением.

# Проектирование системы

## Требования к приложению

Для обеспечения безопасности в приложении на тему «Создания программного средства для записи на участие в квестах» используется система аутентификации, включающая логин и пароль. Это позволяет гарантировать, что доступ к функционалу приложения получают только авторизованные пользователи. При регистрации пользователь должен создать уникальный логин и надежный пароль. При входе пользователь вводит свой логин и пароль для доступа к аккаунту.

Для хранения данных используется файловая система. Все данные пользователей (логины, пароли, информация о квестах и записях) сохраняются в файлах.

Интерфейс программного средства представляет собой модальное окно с несколькими немодальными окнами. Главный экран содержит поля для ввода логина и пароля , с кнопками для регистрации и входа. После авторизации появляется форма со списком доступных квестов, сортировкой квестов по жанру, сложности, стоимости и количеству актёров, с краткой и важной информацией по ним , а также кнопкой для записи. Экран пользователя - информация о пользователе, список посещенных квестов , квесты на которые он записан, возможность отмены записи.

Все элементы управления логически сгруппированы , чтобы пользователь мог легко находить нужные функции. Также все элементы достаточно большие и не выделяются на общем плане , для более удобного взаимодействия. Используется легко читаемый шрифт Trebuchet MS , размер шрифта от 12 и до 23 , в местах где необходимо используется полужирное начертание. Сама форма выполнена в приятном цвете , основная часть в тёмно-сером фоне , а кнопки , поля в бирюзовом цвете с белым шрифтом.

Минимальные аппаратные требования заключается в использовании оперативной системы Windows 10,7,8 .Из основных важных требований к программному средству является доступ к интернету , клавиатура и мышь для взаимодействия с интерфейсом приложения.

## Проектирование модели

Для отображения функциональной структуры объектов программного средства используются 2 диаграммы: Блок-схема алгоритмов и диаграмма классов.

Диаграмма работы алгоритмов — это графическое представление последовательности операций, выполняемых алгоритмом. Она помогает визуализировать логику алгоритма и его структуру, делая процесс понятным и легко читаемым. Данная диаграмма полезна при проектировании, отладке и понимании алгоритмов. Они помогают программистам и разработчикам лучше организовать свои мысли, выделить ключевые шаги и легко обнаружить ошибки или улучшить производительность алгоритма. Блок-схема алгоритма редактирования записей на квесты представлена в графической части на листе 1[1].

Диаграмма классов — это структурная диаграмма в языке UML, которая отображает структуру системы, описывая классы, их атрибуты, методы и отношения между ними. Диаграмма классов помогает визуализировать архитектуру программного обеспечения и отношения между объектами. Данная диаграмма помогает разработчикам лучше понять структуру системы, улучшить проектирование программного обеспечения и облегчить коммуникацию между участниками проекта. Они являются важным инструментом для проектирования и документирования программных систем. Диаграмма классов представлена в графической части на листе 2[2].

## Концептуальный прототип

Концептуальный прототип представляет собой раннюю версию приложения, демонстрирующую основные идеи и функциональные возможности. Он служит для визуализации концепции и получения обратной связи от пользователей до начала разработки полной версии. Прототип может включать в себя наброски экранов, основные сценарии взаимодействия и примерный дизайн, но не содержит всей функциональности и конечной детализации.

Такой прототип позволяет разработчикам и заинтересованным сторонам оценить удобство интерфейса, выявить возможные проблемы и внести необходимые коррективы на ранних стадиях разработки.   
 Начальное окно имеет две формы. Первая форма имеет два текстовых поля и кнопку «Войти» которая отвечает за переход на форму каталога и проверку данных. На рисунке 2.1 представлена форма входа

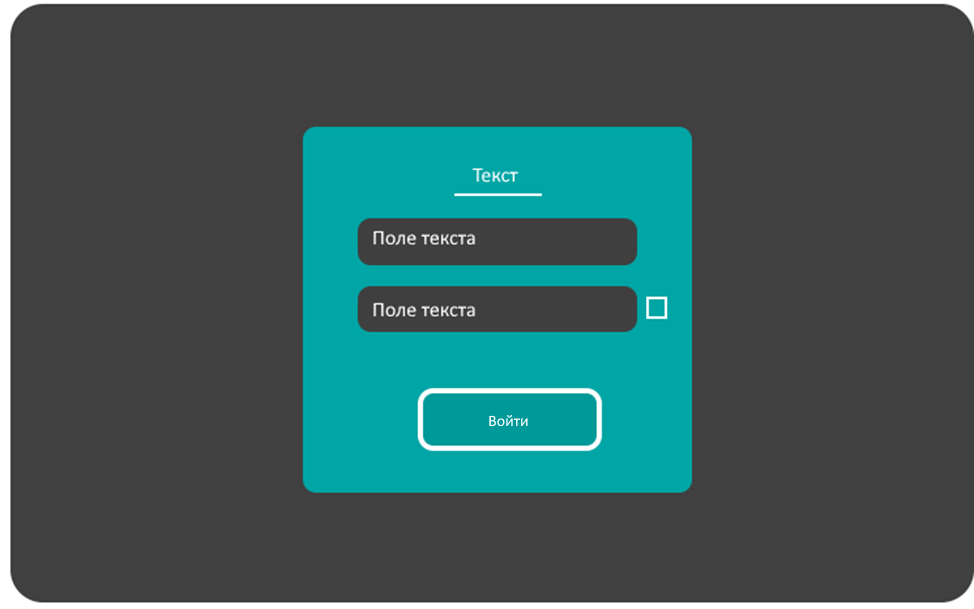
****

Рисунок 2.1 – Форма входа

Вторая форма имеет три текстовых поля и кнопку «Зарегистрироваться» которая отвечает за регистрацию и проверку данных. На рисунке 2.2 представлена форма регистрации

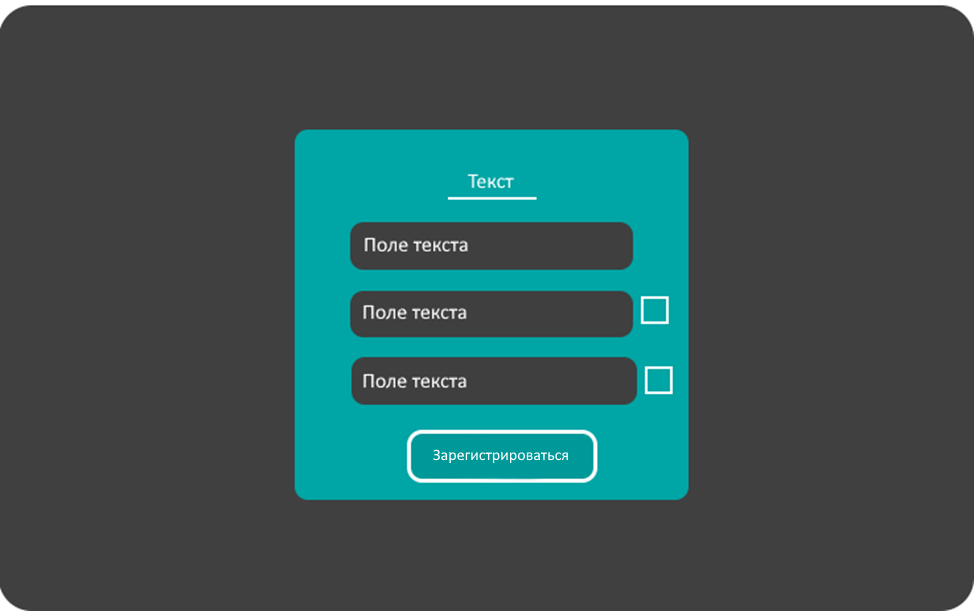
****

Рисунок 2.2 – Форма регистрации

Форма каталога квестов имеет много одно текстовое поле , которое выполняет роль поиска. Кнопка в виде «Круга» Которая открывает профиль пользователя, и кнопка «Забронироваться» которая позволяет забронировать время. На рисунке 2.3 представлена форма каталога квестов.

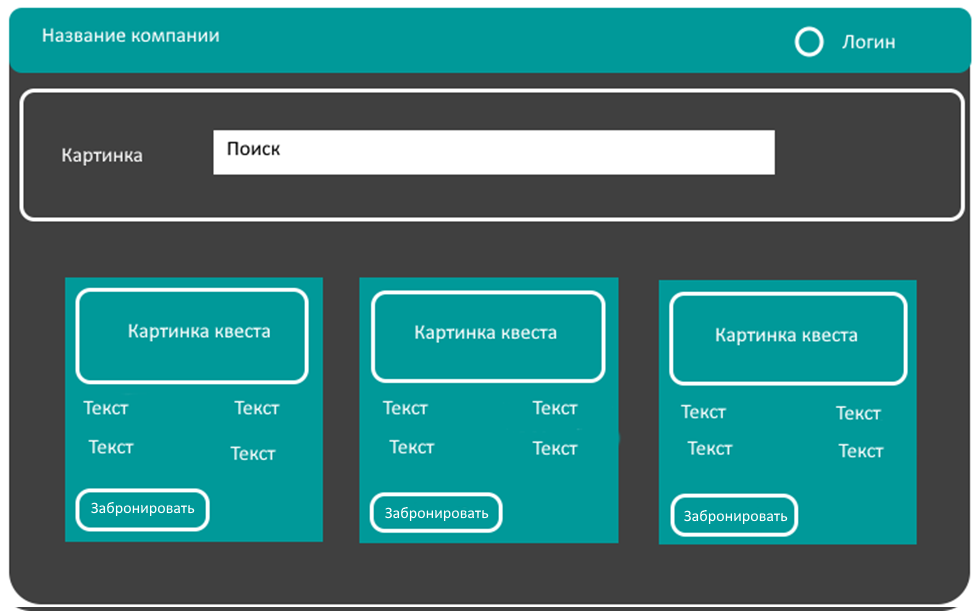


Рисунок 2.3 – Форма каталога квестов

Форма профиль, имеет две кнопки и два текстовых поля, которые позволяют выводить необходимую информацию, кнопка «Все забронированные квесты», покажет все забронированные квесты для данного пользователя, и кнопка «Отчёт за выбранный промежуток времени» выведет все забронированные квеста , в диапазоне тех дат которые буду введены в «Дата». На рисунке 2.4 представлена форма профиля

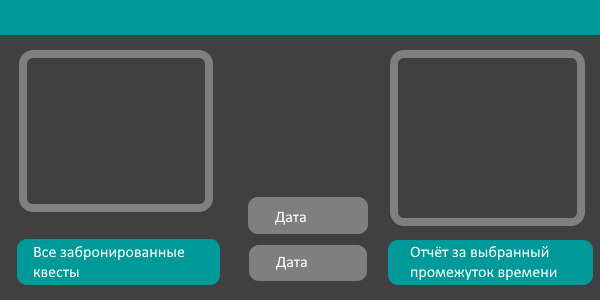


Рисунок 2.4 – Форма профиля

Форма бронирования времени имеет 8 кнопки, при их активации квест бронируется, при нажатии на кнопку «Отмена» бронь со времени снимается. На рисунке 2.5 представлена форма брони.

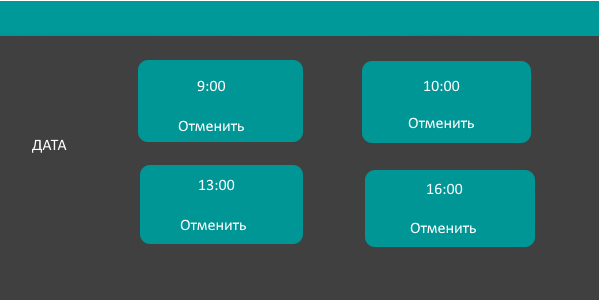


Рисунок 2.5 – Форма брони

# Описание реализации программного средства

## Инструменты разработки и применяемые технологии

Программирование на языке C++ [5]обладает значительной мощностью и гибкостью, что делает его идеальным для разработки высокопроизводительных приложений, включая системы регистрации для участия в квестах. CLI (Common Language Infrastructure) предоставляет среду выполнения для управляемых приложений, позволяя использовать C++ в экосистеме .NET[4].

Среда разработки Visual Studio [3]предоставляет удобные инструменты для создания приложений на C++, включая поддержку Windows Forms для создания графических интерфейсов. Windows Forms позволяют разрабатывать интерактивные приложения с использованием стандартных элементов управления.

Примеры приложений, которые можно разработать с использованием C++, CLI, Windows Forms и Visual Studio, включают системы регистрации для квестов, клиент-серверные приложения, игры, системы управления базами данных и другие программы, требующие высокой производительности и качественного графического интерфейса.

В разрабатываемом программном обеспечении (ПО) эта технология используется для создания клиентской части, которая взаимодействует с интернетом для получения данных о квестах и регистрации участников. Windows Forms обеспечивают удобный способ создания графического интерфейса для пользователя, а C++ позволяет достичь высокой производительности и эффективного управления ресурсами.

Для улучшения пользовательского интерфейса применяются расширения «Bunifu\_UI\_v1.53» и «Guna\_UI2». Эти расширения предлагают расширенные возможности настройки кнопок и других элементов управления, делая программное обеспечение более удобным и привлекательным для пользователей.

## Функции: логическая и физическая организация

Реализация регистрации данных пользователя происходит с помощью функции guna2Button14\_Click и представлена кодом ниже:  
private: System::Void guna2Button4\_Click(System::Object^ sender, System::EventArgs^ e) {

String^ username = guna2TextBox3->Text;

String^ password = guna2TextBox4->Text;

String^ confirmPassword = guna2TextBox5->Text;

if (!String::IsNullOrEmpty(username) && !String::IsNullOrEmpty(password) && !String::IsNullOrEmpty(confirmPassword)) {

guna2Button4->Enabled = true;

if (password == confirmPassword)

{

try

{

array<String^>^ files = Directory::GetFiles(".", "\*.txt");

for each (String ^ file in files)

{

StreamReader^ reader = gcnew StreamReader(file);

String^ fileUsername = reader->ReadLine();

reader->Close();

if (fileUsername == username)

{

MessageBox::Show("Такой аккаунт уже зарегистрирован.");

return;

}

}

String^ fileName = username + ".txt";

StreamWriter^ writer = gcnew StreamWriter(fileName);

writer->WriteLine(username);

writer->WriteLine(password);

writer->Close();

MessageBox::Show("Вы успешно зарегистрировались!");

bunifuPages1->SetPage(1);

}

catch (Exception^ ex)

{

MessageBox::Show("Произошла ошибка при сохранении учетных данных: " + ex->Message);

}

}

else

{

guna2TextBox3->Text = "";

guna2TextBox4->Text = "";

guna2TextBox5->Text = "";

MessageBox::Show("Пароли не совпадают. Пожалуйста, повторите попытку.");

}

}

else

{

MessageBox::Show("Пожалуйста, заполните все поля!");

}

}

Реализация системы входа в профиль происходит с помощью функции guna2Button3\_Click и представлена кодом ниже:  
private: System::Void guna2Button3\_Click(System::Object^ sender, System::EventArgs^ e) {

String^ enteredUsername = guna2TextBox1->Text;

String^ enteredPassword = guna2TextBox2->Text;

try {

String^ fileName = enteredUsername + ".txt";

StreamReader^ reader = gcnew StreamReader(fileName);

String^ savedUsername = reader->ReadLine();

String^ savedPassword = reader->ReadLine();

reader->Close();

if (enteredPassword == savedPassword) {

katalogformh^ form2 = gcnew katalogformh();

form2->Show();

form2->Login = enteredUsername;

}

else {

guna2TextBox1->Text = "";

guna2TextBox2->Text = "";

MessageBox::Show("Неверные учетные данные. Пожалуйста, повторите попытку.");

}

}

catch (Exception^ ex) {

MessageBox::Show("Данного аккаунта не существует ");

return;

}

}

Реализация сохранения забронированного квеста происходит с помощью функции SaveBron() и представлена кодом ниже:

private: Void SaveBron() {

String^ configFilePath = "bron.xml";

ClassConfig^ bron = gcnew ClassConfig();

MessageWarning->Show("Сохранение конфигурации...");

if (bron->btn1 != bron1 || bron->UserBtn1 != User1) {

bron->UserBtn1 = bron1 ? User1 : "";

bron->btn1 = bron1;

}

if (bron->btn2 != bron2 || bron->UserBtn2 != User2) {

bron->UserBtn2 = bron2 ? User2 : "";

bron->btn2 = bron2;

}

if (bron->btn3 != bron3 || bron->UserBtn3 != User3) {

bron->UserBtn3 = bron3 ? User3 :

bron->btn3 = bron3;

}

if (bron->btn4 != bron4 || bron->UserBtn4 != User4) {

bron->UserBtn4 = bron4 ? User4 : "";

bron->btn4 = bron4;

}

bron->SaveConfig(); //

}

Загрузка бронирования в xml файл происходит с помощью функции LoadBron() и представлена кодом нижу:  
private: Void LoadBron() {

if (File::Exists("bron.xml")) {

ClassConfig^ bron = bron->LoadConfig();

bron1 = bron->btn1;

bron2 = bron->btn2;

bron3 = bron->btn3;

bron4 = bron->btn4;

User1 = bron->UserBtn1;

User2 = bron->UserBtn2;

User3 = bron->UserBtn3;

User4 = bron->UserBtn4;

guna2Button1->Enabled = (!bron1 && User1 == "") || User1 == Login;

guna2Button2->Enabled = (!bron2 && User2 == "") || User2 == Login;

guna2Button3->Enabled = (!bron3 && User3 == "") || User3 == Login;

guna2Button5->Enabled = (!bron4 && User4 == "") || User4 == Login;

if (bron1) guna2Button1->Enabled = false;

if (bron2) guna2Button2->Enabled = false;

if (bron3) guna2Button3->Enabled = false;

if (bron4) guna2Button5->Enabled = false;

}

}

Текст остальных программных модулей расположен в приложении А.

## Входные и выходные данные

Входные данные:

* Логин пользователя: Строка, представляющая логин пользователя.
* Пароль пользователя: Строка, представляющая пароль для входа в аккаунт.
* Выбранное время пользователя: Кнопка, необходимая для бронирования времени.
* Забронированные квесты пользователя: Кнопка , необходимая для отображения забронированных квестов.
* Картинка профиля пользователя:

Выходные данные:

* Файл «login.txt»: В файл сохраняется логин и пароль, введенные пользователем и в последующем хранятся в нем.
* Файл «bron.xml»: В файл сохраняется логин пользователя и время бронирования.
* Директория «Users»: Включает в себе файлы вида «логин-пользователя.txt», в которые сохраняются забронированные квесты пользователя.
* Директория «Users»: Записывает, в себя фалы вида «логин.txt» с отчетом по записям на квесты за выбранный промежуток время.

## Функциональное тестирование

Функциональное тестирование включает проверку того, что приложение работает в соответствии с заданными требованиями. Оценка проводится путем сравнения ожидаемых результатов с фактическими (согласно функциональной спецификации), при этом функции тестируются на различных значениях. В процессе проектирования программного обеспечения важно убедиться, что все функции выполняются корректно.

Результаты функционального тестирования представлен в таблице 3.1

Таблица 3.1 – Результат функционального тестирования.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №№ | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 11 | Осуществление авторизации пользователя | 1. Заполнить поля логина и пароля. 2. Нажать на кнопку: «Войти». | Открытие главной формы. | Открытие главной формы. Результат представлен на рисунке Б.1 |
| 22 | Регистрация пользователя. | 1. Заполнить поля логина и пароля. 2. Нажать на кнопку: «Регистрация». | Возвращения на окно входа в аккаунт. | Возвращения на окно входа в аккаунт. Результат представлен на рисунке Б.3 |
| 43 | Поиск квестов | 1. Войти 2. Нажать на поле поиска.  Ввести название квеста. | Появление все возможные названия. | Появление все возможные квесты. Результат представлен на рисунке Б.4 |
| 54 | Бронирования времени на выбранный квест | 1. Войти. 2. Нажать на кнопку «Забронировать» под понравившимся квестом 3. Выбрать время. | Запись квеста в файл и блокирования времени для других пользователей. | Запись квеста в файл и блокирования времени для других пользователей. Результат представлен на рисунке Б.5 |
| 65 | Просмотр всех своих записей на квесты в профиле | 1. Войти. 2. Нажать на иконку профиля. 3. Нажать на кнопку «Забронированные квесты». | Просмотр всех забронированных квестов | Просмотр всех забронированных квестов. Результат представлен на рисунке Б.11 |
| 76 | Просмотр истории бронирования в профиле | 1. Войти. 2. Нажать на иконку профиля 3. Нажать на кнопку «Отчёт по записям» 4. Выбрать диапазон нужных дат 5. Нажать на кнопку: «Отобразить записи на данном промежутке». | Вывод информации о бронировании | Вывод информации о бронировании. Результат представлен на рисунке Б.13 |
| 87 | Просмотр всей информации о интересующем квесте | 1. Войти. 2. Выбрать необходимый квест. 3. Нажать на квест : | Отображение формы с квестом и всей информацией о нём | Отображение формы с квестом и всей информацией о нём . Результат представлен на рисунке Б.14 |
| 98 | Удаление бронирования на квест | 1. Войти. 2. Нажать на кнопку «Забронировать» под понравившимся квестом 3. Под забронированным временем нажать на кнопку «Отменит» | Отмена бронирования | Отмена бронирования Результат представлен на рисунке Б.17 |
| 19 | Редактирования бронирования на квест | 1.Войти  2. Нажать на кнопку «Редактировать мою запись»  3.В открывшимся окне выбрать ново время. | Редактирования бронирования | Редактирование бронирования Результат представлен на рисунке Б.19 |
| 110 | Ознакомление со справочной системой. | 1. Зайти в приложение. 2. Нажать на кнопку: | Открытие и ознакомление со справочной системой. | Открытие и ознакомление со справочной системой. Результат представлен на рисунке Б.21 |

## Описание справочной системы

Средством разработки для создания справочной системы является Dr.Explain, который позволяет легко создавать справочные системы для программного обеспечения, веб-приложений и других продуктов. С его помощью можно быстро и эффективно документировать функциональность и интерфейс приложений, что делает его очень полезным инструментом для разработчиков и технических писателей.

Справочная система имеет начальную страницу, оглавление раздел 1 и раздел 2.

В 1 разделе описывается меню программного средства, где описывается способ открытия и его назначение.

Во 2 разделе описывается главное окно, где отображается главная информация и реализованы основные функции программного средства.

В 3 разделе описывается окно профиля отображается вся необходимая информация для пользотвателя.

Справочная система вызывается с помощью специальной кнопки , которая расположена справа в нижнем углу. На рисунке 3.1 представлена кнопка справочной системы.



Рисунок 3.1 – Кнопка вызова справочной системы.

# Применение

## Назначение программного средства

Программное средcтво «QUEST WORDL» предназначено для отображения краткой информации о интересующих квестах, ознакомлении с ценой , количеством участников и жанром квеста. Пользователи могу сразу же выбрать и забронировать подходящее время . Также просмотреть информацию о всех своих забронированных квестах , а не искать по квестам.

Ограничения, накладываемые на представленное программное средство, включают следующее: необходимость наличия операционной системы Windows.

## Условия применения

Для корректной работы программы требуется операционная система Windows и установочный файл с расширением .exe. Программа также использует несколько вспомогательных файлов формата .txt, содержащих необходимые данные и конфигурации.

Процесс установки программного средства включает следующие шаги:

1. Подготовка операционной системы Windows: Убедитесь, что на вашем компьютере установлена операционная система Windows, поскольку программа «QUEST WORLD» предназначена для работы в данной среде.
2. Размещение вспомогательных .txt файлов: Убедитесь, что все необходимые файлы .txt расположены в той же директории, что и .exe файл программы. Эти файлы могут содержать конфигурационные данные, историю записей, списки пользователей и другие необходимые данные..
3. Запуск программы: После завершения установки и размещения всех требуемых файлов .txt, запустите программу "QUEST WORLD" с помощью ярлыка на рабочем столе или напрямую из папки установки.

Для работы программы необходимы следующие условия.

1. Операционная система:Windows 10 или новее.
2. Процессор: с тактовой частотой 1 ГГц или выше.
3. Оперативная память:не менее 2 ГБ
4. **Место на жестком диске:** не менее 500 МБ для установки программы и дополнительных файлов.
5. Необходимая периферия : Клавиатура и мышь

# Заключение

Целью курсового проектирования было создание программного средства , для предоставлению пользователю всех удобств и информации для бронирования квеста. Предоставление краткой информации, регистрации и личного кабинета с текущими квестами

Для достижения этой цели было реализовано приложение с использованием среды разработки Visual Studio, языка программирования C++ и FrameWork CLI .NET. Программное средство предназначено для обеспечения удобств для пользователей.

Были разработаны модуль авторизации , для безопасного входа в систему. Управление квестами, функция просмотра квестов ,записи на квесты , отмены записи и краткое ознакомление. Был разработан модуль отчётов , который сверяет даты и показывает квесты в данном промежутке времени.

Программное средство «QUEST WORLD» предназначено для упрощения процесса записи на участие в квестах. Оно позволяет пользователям просматривать доступные квесты, записываться на них, отменять свои записи, а также генерировать отчеты. Программное средство обеспечивает удобный и интуитивно понятный интерфейс, что делает работу с ним максимально эффективной и приятной.

В процессе разработки возникли трудности с интеграцией системы бронирования и управления данными, однако они были успешно преодолены с помощью самодельной системы управления записями и бронирования. Это позволило обеспечить корректное отображение информации о доступных квестах и управлении записями пользователей.

Одним из главных достоинств программного средства является простота его использования. Удобный и интуитивно понятный интерфейс позволяет пользователям быстро освоить основные функции без необходимости изучения сложных инструкций. Пользователь может легко и быстро записаться на квест, отменить запись.

Недостатком является отсутствие возможности сохранения детальной истории участий, что может быть важным для пользователей, которые хотят анализировать свои предыдущие квесты или отслеживать прогресс. Без этой функции пользователи могут столкнуться с ограничениями при попытке оценить свои достижения или планировать будущие участия на основе прошлых результатов.

Улучшение данного программного средства возможно за счет перехода на другой язык программирования (например Java), а также смены среды разработки на более мощные и гибкие платформы. Это позволит расширить функциональность приложения, улучшить производительность и предоставить пользователям дополнительные возможности для управления своими записями и участиями в квестах.  
**JAVA** язык программирования с достаточно богатыми библиотеками и фреймворками, которые могут значительно упростить добавление новых функций. Например, использование библиотек для работы с базами данных, создания веб-интерфейсов, машинного обучения и других возможностей, которые могут быть полезны для расширения функционала.Или же использовать язык Python по этой же причине.Платформа Unity, известная своей мощностью и гибкостью, позволяет создавать сложные графические интерфейсы и интегрировать их с другими системами. Это может быть особенно полезно для создания визуально привлекательных и интерактивных элементов интерфейса.

# Список информационных источников

1. Блок-схема [Электронный ресурс]. – Wikipedia, 2024. – Режим доступа: <https://ru.wikipedia.org/wiki/Блок-схема>. – Дата доступа: 14.05.2024.
2. Диаграмма классов [Электронный ресурс]. – Wikipedia, 2024. – Режим доступа: <https://ru.wikipedia.org/wiki/Диаграмма_классов>. – Дата доступа: 16.04.2024.
3. Visual Studio 2019 [Электронный ресурс]. – Microsoft, 2020. – Режим доступа: <https://visualstudio.microsoft.com/ru/vs>. – Дата доступа: 22.04.2020.
4. Обзор NET CLI [электронный ресурс]. – Microsoft, 2024. – Режим доступа: <https://learn.microsoft.com/en-us/dotnet/core/tools/>. – Дата доступа: 21.06.2024.
5. Объектно-ориентированное программирование в С++/ [Роберт Лафоре](https://oz.by/people/more901438.html). – 4-е изд. – СПб. : Питер, 2019. – 928 с.
6. Помощь и обучение по Windows [Электронный ресурс]. – Microsoft,2024. – Режим доступа: <https://support.microsoft.com/ru-ru/windows#WindowsVersion=Windows_10&ID0EBBD=Windows_10>. – Дата доступа: 22.05.2024.
7. Шаляпин Ю.В. Методические указания по выполнению дипломного проекта для учащихся по специальности 2-40 01 01 «Программное обеспечение технологий» / Ю.В. Шаляпин. – Минск : КБП, 2023. – 30 c.

# Приложение А

**Текст программных модулей**

**MyFrom.cpp**

#include "MyForm.h"

#include <Windows.h>

#include"broni2.h"

#include "Profil.h"

using namespace System;

using namespace System::Windows::Forms;

using namespace Project9;

[STAThreadAttribute]

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int) {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

Application::Run(gcnew MyForm);

**}**

**MyForm.h**

#pragma once

#include "katalogformh.h"

namespace Project9 {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: Bunifu::UI::WinForms::BunifuButton::BunifuButton^ bunifuButton1;

private: Bunifu::UI::WinForms::BunifuDropdown^ bunifuDropdown1;

private: Guna::UI2::WinForms::Guna2PictureBox^ guna2PictureBox1;

private: Bunifu::UI::WinForms::BunifuLabel^ bunifuLabel1;

private: System::Windows::Forms::Panel^ panel1;

private: Guna::UI2::WinForms::Guna2DragControl^ guna2DragControl1;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button1;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button2;

private: Bunifu::UI::WinForms::BunifuPages^ bunifuPages1;

private: System::Windows::Forms::TabPage^ tabPage1;

private: System::Windows::Forms::TabPage^ tabPage2;

private: System::Windows::Forms::TabPage^ tabPage3;

private: Guna::UI2::WinForms::Guna2TextBox^ guna2TextBox1;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button3;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button4;

private: Guna::UI2::WinForms::Guna2TextBox^ guna2TextBox5;

private: Guna::UI2::WinForms::Guna2TextBox^ guna2TextBox4;

private: Guna::UI2::WinForms::Guna2TextBox^ guna2TextBox3;

private: Guna::UI2::WinForms::Guna2TextBox^ guna2TextBox2;

private: Guna::UI2::WinForms::Guna2Elipse^ guna2Elipse1;

private: Guna::UI2::WinForms::Guna2AnimateWindow^ guna2AnimateWindow1;

private: Guna::UI2::WinForms::Guna2AnimateWindow^ guna2AnimateWindow2;

private: System::Windows::Forms::PictureBox^ pictureBox1;

private: Guna::UI2::WinForms::Guna2DragControl^ guna2DragControl2;

private:

private: System::ComponentModel::IContainer^ components;

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->components = (gcnew System::ComponentModel::Container());

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(MyForm::typeid));

Utilities::BunifuPages::BunifuAnimatorNS::Animation^ animation1 = (gcnew Utilities::BunifuPages::BunifuAnimatorNS::Animation());

this->guna2PictureBox1 = (gcnew Guna::UI2::WinForms::Guna2PictureBox());

this->bunifuLabel1 = (gcnew Bunifu::UI::WinForms::BunifuLabel());

this->panel1 = (gcnew System::Windows::Forms::Panel());

this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());

this->bunifuPages1 = (gcnew Bunifu::UI::WinForms::BunifuPages());

this->tabPage1 = (gcnew System::Windows::Forms::TabPage());

this->tabPage2 = (gcnew System::Windows::Forms::TabPage());

this->guna2Button3 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2TextBox1 = (gcnew Guna::UI2::WinForms::Guna2TextBox());

this->guna2TextBox2 = (gcnew Guna::UI2::WinForms::Guna2TextBox());

this->tabPage3 = (gcnew System::Windows::Forms::TabPage());

this->guna2Button4 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2TextBox5 = (gcnew Guna::UI2::WinForms::Guna2TextBox());

this->guna2TextBox3 = (gcnew Guna::UI2::WinForms::Guna2TextBox());

this->guna2TextBox4 = (gcnew Guna::UI2::WinForms::Guna2TextBox());

this->guna2Button2 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2Button1 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2DragControl1 = (gcnew Guna::UI2::WinForms::Guna2DragControl(this->components));

this->guna2Elipse1 = (gcnew Guna::UI2::WinForms::Guna2Elipse(this->components));

this->guna2AnimateWindow1 = (gcnew Guna::UI2::WinForms::Guna2AnimateWindow(this->components));

this->guna2AnimateWindow2 = (gcnew Guna::UI2::WinForms::Guna2AnimateWindow(this->components));

this->guna2DragControl2 = (gcnew Guna::UI2::WinForms::Guna2DragControl(this->components));

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->guna2PictureBox1))->BeginInit();

this->panel1->SuspendLayout();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->BeginInit();

this->bunifuPages1->SuspendLayout();

this->tabPage2->SuspendLayout();

this->tabPage3->SuspendLayout();

this->SuspendLayout();

//

// guna2PictureBox1

//

this->guna2PictureBox1->Image = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"guna2PictureBox1.Image")));

this->guna2PictureBox1->ImageRotate = 0;

this->guna2PictureBox1->Location = System::Drawing::Point(29, 92);

this->guna2PictureBox1->Name = L"guna2PictureBox1";

this->guna2PictureBox1->Size = System::Drawing::Size(320, 397);

this->guna2PictureBox1->TabIndex = 3;

this->guna2PictureBox1->TabStop = false;

//

// bunifuLabel1

//

this->bunifuLabel1->AllowParentOverrides = false;

this->bunifuLabel1->AutoEllipsis = false;

this->bunifuLabel1->Cursor = System::Windows::Forms::Cursors::Default;

this->bunifuLabel1->CursorType = System::Windows::Forms::Cursors::Default;

this->bunifuLabel1->Font = (gcnew System::Drawing::Font(L"Algerian", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->bunifuLabel1->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(50)), static\_cast<System::Int32>(static\_cast<System::Byte>(108)),

static\_cast<System::Int32>(static\_cast<System::Byte>(109)));

this->bunifuLabel1->Location = System::Drawing::Point(82, 453);

this->bunifuLabel1->Name = L"bunifuLabel1";

this->bunifuLabel1->RightToLeft = System::Windows::Forms::RightToLeft::No;

this->bunifuLabel1->Size = System::Drawing::Size(213, 36);

this->bunifuLabel1->TabIndex = 4;

this->bunifuLabel1->Text = L"QUEST WORLD";

this->bunifuLabel1->TextAlignment = System::Drawing::ContentAlignment::TopLeft;

this->bunifuLabel1->TextFormat = Bunifu::UI::WinForms::BunifuLabel::TextFormattingOptions::Default;

this->bunifuLabel1->Click += gcnew System::EventHandler(this, &MyForm::bunifuLabel1\_Click);

//

// panel1

//

this->panel1->BackColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(50)), static\_cast<System::Int32>(static\_cast<System::Byte>(109)),

static\_cast<System::Int32>(static\_cast<System::Byte>(108)));

this->panel1->Controls->Add(this->pictureBox1);

this->panel1->Controls->Add(this->bunifuPages1);

this->panel1->Controls->Add(this->guna2Button2);

this->panel1->Controls->Add(this->guna2Button1);

this->panel1->Location = System::Drawing::Point(381, 0);

this->panel1->Name = L"panel1";

this->panel1->Size = System::Drawing::Size(715, 641);

this->panel1->TabIndex = 5;

this->panel1->Paint += gcnew System::Windows::Forms::PaintEventHandler(this, &MyForm::panel1\_Paint);

//

// pictureBox1

//

this->pictureBox1->BackgroundImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox1.BackgroundImage")));

this->pictureBox1->Location = System::Drawing::Point(656, 12);

this->pictureBox1->Name = L"pictureBox1";

this->pictureBox1->Size = System::Drawing::Size(30, 30);

this->pictureBox1->TabIndex = 11;

this->pictureBox1->TabStop = false;

this->pictureBox1->Click += gcnew System::EventHandler(this, &MyForm::pictureBox1\_Click);

//

// bunifuPages1

//

this->bunifuPages1->Alignment = System::Windows::Forms::TabAlignment::Bottom;

this->bunifuPages1->AllowTransitions = true;

this->bunifuPages1->Controls->Add(this->tabPage1);

this->bunifuPages1->Controls->Add(this->tabPage2);

this->bunifuPages1->Controls->Add(this->tabPage3);

this->bunifuPages1->Location = System::Drawing::Point(119, 177);

this->bunifuPages1->Multiline = true;

this->bunifuPages1->Name = L"bunifuPages1";

this->bunifuPages1->Page = this->tabPage3;

this->bunifuPages1->PageIndex = 2;

this->bunifuPages1->PageName = L"tabPage3";

this->bunifuPages1->PageTitle = L"tabPage3";

this->bunifuPages1->SelectedIndex = 0;

this->bunifuPages1->Size = System::Drawing::Size(460, 422);

this->bunifuPages1->TabIndex = 8;

animation1->AnimateOnlyDifferences = true;

animation1->BlindCoeff = (cli::safe\_cast<System::Drawing::PointF>(resources->GetObject(L"animation1.BlindCoeff")));

animation1->LeafCoeff = 0;

animation1->MaxTime = 1;

animation1->MinTime = 0;

animation1->MosaicCoeff = (cli::safe\_cast<System::Drawing::PointF>(resources->GetObject(L"animation1.MosaicCoeff")));

animation1->MosaicShift = (cli::safe\_cast<System::Drawing::PointF>(resources->GetObject(L"animation1.MosaicShift")));

animation1->MosaicSize = 0;

animation1->Padding = System::Windows::Forms::Padding(0, 0, 0, 0);

animation1->RotateCoeff = 0;

animation1->RotateLimit = 0;

animation1->ScaleCoeff = (cli::safe\_cast<System::Drawing::PointF>(resources->GetObject(L"animation1.ScaleCoeff")));

animation1->SlideCoeff = (cli::safe\_cast<System::Drawing::PointF>(resources->GetObject(L"animation1.SlideCoeff")));

animation1->TimeCoeff = 0;

animation1->TransparencyCoeff = 0;

this->bunifuPages1->Transition = animation1;

this->bunifuPages1->TransitionType = Utilities::BunifuPages::BunifuAnimatorNS::AnimationType::HorizBlind;

//

// tabPage1

//

this->tabPage1->BackColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(50)), static\_cast<System::Int32>(static\_cast<System::Byte>(109)),

static\_cast<System::Int32>(static\_cast<System::Byte>(108)));

this->tabPage1->Location = System::Drawing::Point(4, 4);

this->tabPage1->Name = L"tabPage1";

this->tabPage1->Padding = System::Windows::Forms::Padding(3);

this->tabPage1->Size = System::Drawing::Size(452, 396);

this->tabPage1->TabIndex = 0;

this->tabPage1->Text = L"tabPage1";

//

// tabPage2

//

this->tabPage2->BackColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(50)), static\_cast<System::Int32>(static\_cast<System::Byte>(109)),

static\_cast<System::Int32>(static\_cast<System::Byte>(108)));

this->tabPage2->Controls->Add(this->guna2Button3);

this->tabPage2->Controls->Add(this->guna2TextBox1);

this->tabPage2->Controls->Add(this->guna2TextBox2);

this->tabPage2->Location = System::Drawing::Point(4, 4);

this->tabPage2->Name = L"tabPage2";

this->tabPage2->Padding = System::Windows::Forms::Padding(3);

this->tabPage2->Size = System::Drawing::Size(452, 396);

this->tabPage2->TabIndex = 1;

this->tabPage2->Text = L"tabPage2";

//

// guna2Button3

//

this->guna2Button3->Animated = true;

this->guna2Button3->BackColor = System::Drawing::Color::Transparent;

this->guna2Button3->BorderRadius = 20;

this->guna2Button3->BorderStyle = System::Drawing::Drawing2D::DashStyle::Dot;

this->guna2Button3->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button3->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button3->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button3->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button3->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2Button3->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold));

this->guna2Button3->ForeColor = System::Drawing::Color::White;

this->guna2Button3->IndicateFocus = true;

this->guna2Button3->Location = System::Drawing::Point(118, 232);

this->guna2Button3->Name = L"guna2Button3";

this->guna2Button3->Size = System::Drawing::Size(236, 60);

this->guna2Button3->TabIndex = 9;

this->guna2Button3->Text = L"Войти";

this->guna2Button3->Click += gcnew System::EventHandler(this, &MyForm::guna2Button3\_Click);

//

// guna2TextBox1

//

this->guna2TextBox1->Animated = true;

this->guna2TextBox1->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox1->BorderRadius = 20;

this->guna2TextBox1->BorderThickness = 3;

this->guna2TextBox1->Cursor = System::Windows::Forms::Cursors::IBeam;

this->guna2TextBox1->DefaultText = L"";

this->guna2TextBox1->DisabledState->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(208)),

static\_cast<System::Int32>(static\_cast<System::Byte>(208)), static\_cast<System::Int32>(static\_cast<System::Byte>(208)));

this->guna2TextBox1->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(226)),

static\_cast<System::Int32>(static\_cast<System::Byte>(226)), static\_cast<System::Int32>(static\_cast<System::Byte>(226)));

this->guna2TextBox1->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox1->DisabledState->PlaceholderForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox1->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox1->FocusedState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox1->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold));

this->guna2TextBox1->ForeColor = System::Drawing::Color::White;

this->guna2TextBox1->HoverState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox1->IconLeft = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"guna2TextBox1.IconLeft")));

this->guna2TextBox1->IconLeftSize = System::Drawing::Size(30, 30);

this->guna2TextBox1->Location = System::Drawing::Point(59, 49);

this->guna2TextBox1->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->guna2TextBox1->Name = L"guna2TextBox1";

this->guna2TextBox1->PasswordChar = '\0';

this->guna2TextBox1->PlaceholderText = L"";

this->guna2TextBox1->SelectedText = L"";

this->guna2TextBox1->ShadowDecoration->Color = System::Drawing::Color::Transparent;

this->guna2TextBox1->Size = System::Drawing::Size(347, 51);

this->guna2TextBox1->TabIndex = 2;

this->guna2TextBox1->Click += gcnew System::EventHandler(this, &MyForm::guna2TextBox1\_Click);

//

// guna2TextBox2

//

this->guna2TextBox2->Animated = true;

this->guna2TextBox2->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox2->BorderRadius = 20;

this->guna2TextBox2->BorderThickness = 3;

this->guna2TextBox2->Cursor = System::Windows::Forms::Cursors::IBeam;

this->guna2TextBox2->DefaultText = L"";

this->guna2TextBox2->DisabledState->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(208)),

static\_cast<System::Int32>(static\_cast<System::Byte>(208)), static\_cast<System::Int32>(static\_cast<System::Byte>(208)));

this->guna2TextBox2->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(226)),

static\_cast<System::Int32>(static\_cast<System::Byte>(226)), static\_cast<System::Int32>(static\_cast<System::Byte>(226)));

this->guna2TextBox2->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox2->DisabledState->PlaceholderForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox2->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox2->FocusedState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox2->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2TextBox2->ForeColor = System::Drawing::Color::White;

this->guna2TextBox2->HoverState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox2->IconLeft = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"guna2TextBox2.IconLeft")));

this->guna2TextBox2->IconLeftSize = System::Drawing::Size(30, 30);

this->guna2TextBox2->Location = System::Drawing::Point(59, 140);

this->guna2TextBox2->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->guna2TextBox2->Name = L"guna2TextBox2";

this->guna2TextBox2->PasswordChar = '\*';

this->guna2TextBox2->PlaceholderText = L"";

this->guna2TextBox2->SelectedText = L"";

this->guna2TextBox2->ShadowDecoration->Color = System::Drawing::Color::Transparent;

this->guna2TextBox2->Size = System::Drawing::Size(347, 51);

this->guna2TextBox2->TabIndex = 1;

this->guna2TextBox2->Click += gcnew System::EventHandler(this, &MyForm::guna2TextBox2\_Click);

//

// tabPage3

//

this->tabPage3->BackColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(50)), static\_cast<System::Int32>(static\_cast<System::Byte>(109)),

static\_cast<System::Int32>(static\_cast<System::Byte>(108)));

this->tabPage3->Controls->Add(this->guna2Button4);

this->tabPage3->Controls->Add(this->guna2TextBox5);

this->tabPage3->Controls->Add(this->guna2TextBox3);

this->tabPage3->Controls->Add(this->guna2TextBox4);

this->tabPage3->Location = System::Drawing::Point(4, 4);

this->tabPage3->Name = L"tabPage3";

this->tabPage3->Padding = System::Windows::Forms::Padding(3);

this->tabPage3->Size = System::Drawing::Size(452, 396);

this->tabPage3->TabIndex = 2;

this->tabPage3->Text = L"tabPage3";

//

// guna2Button4

//

this->guna2Button4->Animated = true;

this->guna2Button4->BackColor = System::Drawing::Color::Transparent;

this->guna2Button4->BorderRadius = 20;

this->guna2Button4->BorderStyle = System::Drawing::Drawing2D::DashStyle::Dot;

this->guna2Button4->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button4->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button4->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button4->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button4->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2Button4->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold));

this->guna2Button4->ForeColor = System::Drawing::Color::White;

this->guna2Button4->IndicateFocus = true;

this->guna2Button4->Location = System::Drawing::Point(59, 317);

this->guna2Button4->Name = L"guna2Button4";

this->guna2Button4->Size = System::Drawing::Size(347, 59);

this->guna2Button4->TabIndex = 10;

this->guna2Button4->Text = L"Зарегистрироваться";

this->guna2Button4->Click += gcnew System::EventHandler(this, &MyForm::guna2Button4\_Click);

//

// guna2TextBox5

//

this->guna2TextBox5->Animated = true;

this->guna2TextBox5->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox5->BorderRadius = 20;

this->guna2TextBox5->BorderThickness = 3;

this->guna2TextBox5->Cursor = System::Windows::Forms::Cursors::IBeam;

this->guna2TextBox5->DefaultText = L"";

this->guna2TextBox5->DisabledState->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(208)),

static\_cast<System::Int32>(static\_cast<System::Byte>(208)), static\_cast<System::Int32>(static\_cast<System::Byte>(208)));

this->guna2TextBox5->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(226)),

static\_cast<System::Int32>(static\_cast<System::Byte>(226)), static\_cast<System::Int32>(static\_cast<System::Byte>(226)));

this->guna2TextBox5->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox5->DisabledState->PlaceholderForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox5->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox5->FocusedState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox5->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2TextBox5->ForeColor = System::Drawing::Color::White;

this->guna2TextBox5->HoverState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox5->HoverState->FillColor = System::Drawing::Color::Transparent;

this->guna2TextBox5->HoverState->ForeColor = System::Drawing::Color::Transparent;

this->guna2TextBox5->HoverState->PlaceholderForeColor = System::Drawing::Color::Transparent;

this->guna2TextBox5->IconLeft = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"guna2TextBox5.IconLeft")));

this->guna2TextBox5->IconLeftSize = System::Drawing::Size(30, 30);

this->guna2TextBox5->Location = System::Drawing::Point(59, 228);

this->guna2TextBox5->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->guna2TextBox5->Name = L"guna2TextBox5";

this->guna2TextBox5->PasswordChar = '\*';

this->guna2TextBox5->PlaceholderText = L"";

this->guna2TextBox5->SelectedText = L"";

this->guna2TextBox5->Size = System::Drawing::Size(347, 51);

this->guna2TextBox5->TabIndex = 10;

this->guna2TextBox5->Click += gcnew System::EventHandler(this, &MyForm::guna2TextBox5\_Click);

//

// guna2TextBox3

//

this->guna2TextBox3->Animated = true;

this->guna2TextBox3->AutoValidate = System::Windows::Forms::AutoValidate::EnablePreventFocusChange;

this->guna2TextBox3->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox3->BorderRadius = 20;

this->guna2TextBox3->BorderThickness = 3;

this->guna2TextBox3->Cursor = System::Windows::Forms::Cursors::IBeam;

this->guna2TextBox3->DefaultText = L"";

this->guna2TextBox3->DisabledState->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(208)),

static\_cast<System::Int32>(static\_cast<System::Byte>(208)), static\_cast<System::Int32>(static\_cast<System::Byte>(208)));

this->guna2TextBox3->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(226)),

static\_cast<System::Int32>(static\_cast<System::Byte>(226)), static\_cast<System::Int32>(static\_cast<System::Byte>(226)));

this->guna2TextBox3->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox3->DisabledState->PlaceholderForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox3->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox3->FocusedState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox3->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2TextBox3->ForeColor = System::Drawing::Color::White;

this->guna2TextBox3->HoverState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox3->IconLeft = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"guna2TextBox3.IconLeft")));

this->guna2TextBox3->IconLeftSize = System::Drawing::Size(30, 30);

this->guna2TextBox3->Location = System::Drawing::Point(59, 49);

this->guna2TextBox3->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->guna2TextBox3->Name = L"guna2TextBox3";

this->guna2TextBox3->PasswordChar = '\0';

this->guna2TextBox3->PlaceholderForeColor = System::Drawing::Color::LightGray;

this->guna2TextBox3->PlaceholderText = L"";

this->guna2TextBox3->SelectedText = L"";

this->guna2TextBox3->Size = System::Drawing::Size(347, 51);

this->guna2TextBox3->TabIndex = 9;

this->guna2TextBox3->Click += gcnew System::EventHandler(this, &MyForm::guna2TextBox3\_Click);

//

// guna2TextBox4

//

this->guna2TextBox4->Animated = true;

this->guna2TextBox4->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox4->BorderRadius = 20;

this->guna2TextBox4->BorderThickness = 3;

this->guna2TextBox4->Cursor = System::Windows::Forms::Cursors::IBeam;

this->guna2TextBox4->DefaultText = L"";

this->guna2TextBox4->DisabledState->BorderColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(208)),

static\_cast<System::Int32>(static\_cast<System::Byte>(208)), static\_cast<System::Int32>(static\_cast<System::Byte>(208)));

this->guna2TextBox4->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(226)),

static\_cast<System::Int32>(static\_cast<System::Byte>(226)), static\_cast<System::Int32>(static\_cast<System::Byte>(226)));

this->guna2TextBox4->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox4->DisabledState->PlaceholderForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(138)),

static\_cast<System::Int32>(static\_cast<System::Byte>(138)), static\_cast<System::Int32>(static\_cast<System::Byte>(138)));

this->guna2TextBox4->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2TextBox4->FocusedState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox4->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2TextBox4->ForeColor = System::Drawing::Color::White;

this->guna2TextBox4->HoverState->BorderColor = System::Drawing::Color::White;

this->guna2TextBox4->IconLeft = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"guna2TextBox4.IconLeft")));

this->guna2TextBox4->IconLeftSize = System::Drawing::Size(30, 30);

this->guna2TextBox4->Location = System::Drawing::Point(59, 140);

this->guna2TextBox4->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->guna2TextBox4->Name = L"guna2TextBox4";

this->guna2TextBox4->PasswordChar = '\*';

this->guna2TextBox4->PlaceholderText = L"";

this->guna2TextBox4->SelectedText = L"";

this->guna2TextBox4->Size = System::Drawing::Size(347, 51);

this->guna2TextBox4->TabIndex = 9;

this->guna2TextBox4->Click += gcnew System::EventHandler(this, &MyForm::guna2TextBox4\_Click);

//

// guna2Button2

//

this->guna2Button2->Animated = true;

this->guna2Button2->BorderRadius = 20;

this->guna2Button2->BorderStyle = System::Drawing::Drawing2D::DashStyle::DashDotDot;

this->guna2Button2->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button2->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button2->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button2->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button2->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2Button2->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold));

this->guna2Button2->ForeColor = System::Drawing::Color::White;

this->guna2Button2->Location = System::Drawing::Point(343, 92);

this->guna2Button2->Name = L"guna2Button2";

this->guna2Button2->Size = System::Drawing::Size(236, 61);

this->guna2Button2->TabIndex = 7;

this->guna2Button2->Text = L"Регистрация";

this->guna2Button2->Click += gcnew System::EventHandler(this, &MyForm::guna2Button2\_Click\_1);

//

// guna2Button1

//

this->guna2Button1->Animated = true;

this->guna2Button1->BackColor = System::Drawing::Color::Transparent;

this->guna2Button1->BorderRadius = 20;

this->guna2Button1->BorderStyle = System::Drawing::Drawing2D::DashStyle::Custom;

this->guna2Button1->CheckedState->CustomBorderColor = System::Drawing::Color::Transparent;

this->guna2Button1->CheckedState->FillColor = System::Drawing::Color::Transparent;

this->guna2Button1->CheckedState->ForeColor = System::Drawing::Color::Transparent;

this->guna2Button1->DisabledState->BorderColor = System::Drawing::Color::Transparent;

this->guna2Button1->DisabledState->CustomBorderColor = System::Drawing::Color::Transparent;

this->guna2Button1->DisabledState->FillColor = System::Drawing::Color::Transparent;

this->guna2Button1->DisabledState->ForeColor = System::Drawing::Color::Transparent;

this->guna2Button1->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(37)), static\_cast<System::Int32>(static\_cast<System::Byte>(80)),

static\_cast<System::Int32>(static\_cast<System::Byte>(80)));

this->guna2Button1->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold));

this->guna2Button1->ForeColor = System::Drawing::Color::White;

this->guna2Button1->IndicateFocus = true;

this->guna2Button1->Location = System::Drawing::Point(101, 93);

this->guna2Button1->Name = L"guna2Button1";

this->guna2Button1->Size = System::Drawing::Size(236, 60);

this->guna2Button1->TabIndex = 6;

this->guna2Button1->Text = L"Вход";

this->guna2Button1->Click += gcnew System::EventHandler(this, &MyForm::guna2Button1\_Click\_1);

//

// guna2DragControl1

//

this->guna2DragControl1->DockIndicatorTransparencyValue = 0.6;

this->guna2DragControl1->TargetControl = this;

this->guna2DragControl1->UseTransparentDrag = true;

//

// guna2Elipse1

//

this->guna2Elipse1->BorderRadius = 40;

this->guna2Elipse1->TargetControl = this;

//

// guna2AnimateWindow1

//

this->guna2AnimateWindow1->TargetForm = this;

//

// guna2DragControl2

//

this->guna2DragControl2->DockIndicatorTransparencyValue = 0.6;

this->guna2DragControl2->TargetControl = this->panel1;

this->guna2DragControl2->UseTransparentDrag = true;

//

// MyForm

//

this->BackColor = System::Drawing::Color::White;

this->ClientSize = System::Drawing::Size(1095, 639);

this->Controls->Add(this->panel1);

this->Controls->Add(this->bunifuLabel1);

this->Controls->Add(this->guna2PictureBox1);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::None;

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"MyForm";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Load += gcnew System::EventHandler(this, &MyForm::MyForm\_Load);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->guna2PictureBox1))->EndInit();

this->panel1->ResumeLayout(false);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->EndInit();

this->bunifuPages1->ResumeLayout(false);

this->tabPage2->ResumeLayout(false);

this->tabPage3->ResumeLayout(false);

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void bunifuButton1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void bunifuDropdown1\_SelectedIndexChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

//private: System::Void bunifuDropdown2\_SelectedIndexChanged(System::Object^ sender, System::EventArgs^ e) {

// if (bunifuDropdown2->Text == "Хоррор") { DROPDOWN

// MessageBox::Show("u");

//}

//}

private: System::Void bunifuLabel1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void panel1\_Paint(System::Object^ sender, System::Windows::Forms::PaintEventArgs^ e) {

}

private: System::Void guna2Button1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void guna2Button2\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void guna2Button1\_Click\_1(System::Object^ sender, System::EventArgs^ e) {

bunifuPages1->SetPage(1);

this->guna2Button1->Focus();

}

private: System::Void guna2Button2\_Click\_1(System::Object^ sender, System::EventArgs^ e) {

bunifuPages1->SetPage(2);

this->guna2Button1->Focus();

}

private: System::Void guna2TextBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void guna2TextBox2\_Click(System::Object^ sender, System::EventArgs^ e) {

this->guna2TextBox2->Text="";

}

private: System::Void guna2TextBox1\_Click(System::Object^ sender, System::EventArgs^ e) {

this->guna2TextBox1->Text = "";

}

private: System::Void MyForm\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void guna2TextBox3\_Click(System::Object^ sender, System::EventArgs^ e) {

this->guna2TextBox3->Text = "";

}

private: System::Void guna2TextBox4\_Click(System::Object^ sender, System::EventArgs^ e) {

this->guna2TextBox4->Text = "";

}

private: System::Void guna2TextBox5\_Click(System::Object^ sender, System::EventArgs^ e) {

this->guna2TextBox5->Text = "";

}

private: System::Void guna2Button3\_Click(System::Object^ sender, System::EventArgs^ e) {

String^ enteredUsername = guna2TextBox1->Text;

String^ enteredPassword = guna2TextBox2->Text;

try {

String^ fileName = enteredUsername + ".txt";

StreamReader^ reader = gcnew StreamReader(fileName);

String^ savedUsername = reader->ReadLine();

String^ savedPassword = reader->ReadLine();

reader->Close();

if (enteredPassword == savedPassword) {

katalogformh^ form2 = gcnew katalogformh();

form2->Show();

form2->Login = enteredUsername;

}

else {

guna2TextBox1->Text = "";

guna2TextBox2->Text = "";

MessageBox::Show("Неверные учетные данные. Пожалуйста, повторите попытку.");

}

}

catch (Exception^ ex) {

MessageBox::Show("Данного аккаунта не существует ");

return;

}

}

private: System::Void guna2Button4\_Click(System::Object^ sender, System::EventArgs^ e) {

String^ username = guna2TextBox3->Text;

String^ password = guna2TextBox4->Text;

String^ confirmPassword = guna2TextBox5->Text;

if (!String::IsNullOrEmpty(username) && !String::IsNullOrEmpty(password) && !String::IsNullOrEmpty(confirmPassword)) {

guna2Button4->Enabled = true;

if (password == confirmPassword)

{

try

{

array<String^>^ files = Directory::GetFiles(".", "\*.txt");

for each (String ^ file in files)

{

StreamReader^ reader = gcnew StreamReader(file);

String^ fileUsername = reader->ReadLine();

reader->Close();

if (fileUsername == username)

{

MessageBox::Show("Такой аккаунт уже зарегистрирован.");

return;

}

}

String^ fileName = username + ".txt";

StreamWriter^ writer = gcnew StreamWriter(fileName);

writer->WriteLine(username);

writer->WriteLine(password);

writer->Close();

MessageBox::Show("Вы успешно зарегистрировались!");

bunifuPages1->SetPage(1);

}

catch (Exception^ ex)

{

MessageBox::Show("Произошла ошибка при сохранении учетных данных: " + ex->Message);

}

}

else

{

guna2TextBox3->Text = "";

guna2TextBox4->Text = "";

guna2TextBox5->Text = "";

MessageBox::Show("Пароли не совпадают. Пожалуйста, повторите попытку.");

}

}

else

{

MessageBox::Show("Пожалуйста, заполните все поля!");

}

}

private: System::Void pictureBox1\_Click(System::Object^ sender, System::EventArgs^ e) {

this->Close();

}

};

}

**broni.h**

#pragma once

#include <cliext/list>

#include <cliext/vector>

#include <cliext/vector>

#include <fstream>

#include"ClassConfig.h"

#include "TimeSlotSelectionForm.h"

namespace Project9 {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

using namespace cliext;

/// <summary>

/// Summary for broni

/// </summary>

public ref class broni : public System::Windows::Forms::Form

{

public:

String^ Login;

broni(String^ L) {

Login = L;

InitializeComponent();

MessageBox::Show("Вы вошли как " + Login);

}

broni(void)

{

InitializeComponent();

//

//TODO: Add the constructor code here

//

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~broni()

{

if (components)

{

delete components;

}

}

private: Guna::UI2::WinForms::Guna2Button^ guna2Button1;

protected:

private: Guna::UI2::WinForms::Guna2Button^ guna2Button2;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button3;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button5;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button6;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button7;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button8;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button9;

private: Guna::UI2::WinForms::Guna2MessageDialog^ MessageWarning;

private: System::Windows::Forms::PictureBox^ pictureBox1;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Panel^ panel2;

private: System::Windows::Forms::Panel^ panel1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Panel^ panel4;

private: System::Windows::Forms::Panel^ panel3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: Guna::UI2::WinForms::Guna2Button^ guna2Button4;

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(broni::typeid));

this->guna2Button1 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2Button2 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2Button3 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2Button5 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2Button6 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2Button7 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2Button8 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->guna2Button9 = (gcnew Guna::UI2::WinForms::Guna2Button());

this->MessageWarning = (gcnew Guna::UI2::WinForms::Guna2MessageDialog());

this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->panel1 = (gcnew System::Windows::Forms::Panel());

this->panel2 = (gcnew System::Windows::Forms::Panel());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->panel3 = (gcnew System::Windows::Forms::Panel());

this->panel4 = (gcnew System::Windows::Forms::Panel());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->guna2Button4 = (gcnew Guna::UI2::WinForms::Guna2Button());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->BeginInit();

this->SuspendLayout();

//

// guna2Button1

//

this->guna2Button1->BorderRadius = 30;

this->guna2Button1->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button1->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button1->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button1->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button1->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(128)),

static\_cast<System::Int32>(static\_cast<System::Byte>(0)));

this->guna2Button1->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2Button1->ForeColor = System::Drawing::Color::White;

this->guna2Button1->Location = System::Drawing::Point(233, 216);

this->guna2Button1->Name = L"guna2Button1";

this->guna2Button1->Size = System::Drawing::Size(196, 75);

this->guna2Button1->TabIndex = 0;

this->guna2Button1->Text = L"9:00";

this->guna2Button1->Click += gcnew System::EventHandler(this, &broni::guna2Button1\_Click);

//

// guna2Button2

//

this->guna2Button2->BorderRadius = 30;

this->guna2Button2->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button2->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button2->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button2->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button2->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(128)),

static\_cast<System::Int32>(static\_cast<System::Byte>(0)));

this->guna2Button2->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2Button2->ForeColor = System::Drawing::Color::White;

this->guna2Button2->Location = System::Drawing::Point(462, 216);

this->guna2Button2->Name = L"guna2Button2";

this->guna2Button2->Size = System::Drawing::Size(196, 75);

this->guna2Button2->TabIndex = 1;

this->guna2Button2->Text = L"10:00";

this->guna2Button2->Click += gcnew System::EventHandler(this, &broni::guna2Button2\_Click);

//

// guna2Button3

//

this->guna2Button3->BorderRadius = 30;

this->guna2Button3->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button3->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button3->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button3->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button3->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(128)),

static\_cast<System::Int32>(static\_cast<System::Byte>(0)));

this->guna2Button3->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2Button3->ForeColor = System::Drawing::Color::White;

this->guna2Button3->Location = System::Drawing::Point(233, 349);

this->guna2Button3->Name = L"guna2Button3";

this->guna2Button3->Size = System::Drawing::Size(196, 75);

this->guna2Button3->TabIndex = 2;

this->guna2Button3->Text = L"13:00";

this->guna2Button3->Click += gcnew System::EventHandler(this, &broni::guna2Button3\_Click);

//

// guna2Button5

//

this->guna2Button5->BorderRadius = 30;

this->guna2Button5->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button5->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button5->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button5->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button5->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(128)),

static\_cast<System::Int32>(static\_cast<System::Byte>(0)));

this->guna2Button5->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2Button5->ForeColor = System::Drawing::Color::White;

this->guna2Button5->Location = System::Drawing::Point(462, 349);

this->guna2Button5->Name = L"guna2Button5";

this->guna2Button5->Size = System::Drawing::Size(196, 75);

this->guna2Button5->TabIndex = 4;

this->guna2Button5->Text = L"16:00";

this->guna2Button5->Click += gcnew System::EventHandler(this, &broni::guna2Button5\_Click);

//

// guna2Button6

//

this->guna2Button6->BorderRadius = 30;

this->guna2Button6->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button6->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button6->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button6->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button6->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(192)),

static\_cast<System::Int32>(static\_cast<System::Byte>(128)));

this->guna2Button6->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2Button6->ForeColor = System::Drawing::Color::White;

this->guna2Button6->Location = System::Drawing::Point(233, 291);

this->guna2Button6->Name = L"guna2Button6";

this->guna2Button6->Size = System::Drawing::Size(196, 47);

this->guna2Button6->TabIndex = 5;

this->guna2Button6->Text = L"Отмена";

this->guna2Button6->Click += gcnew System::EventHandler(this, &broni::guna2Button6\_Click);

//

// guna2Button7

//

this->guna2Button7->BorderRadius = 30;

this->guna2Button7->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button7->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button7->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button7->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button7->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(192)),

static\_cast<System::Int32>(static\_cast<System::Byte>(128)));

this->guna2Button7->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2Button7->ForeColor = System::Drawing::Color::White;

this->guna2Button7->Location = System::Drawing::Point(462, 291);

this->guna2Button7->Name = L"guna2Button7";

this->guna2Button7->Size = System::Drawing::Size(196, 47);

this->guna2Button7->TabIndex = 6;

this->guna2Button7->Text = L"Отмена";

this->guna2Button7->Click += gcnew System::EventHandler(this, &broni::guna2Button7\_Click);

//

// guna2Button8

//

this->guna2Button8->BorderRadius = 30;

this->guna2Button8->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button8->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button8->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button8->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button8->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(192)),

static\_cast<System::Int32>(static\_cast<System::Byte>(128)));

this->guna2Button8->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2Button8->ForeColor = System::Drawing::Color::White;

this->guna2Button8->Location = System::Drawing::Point(233, 423);

this->guna2Button8->Name = L"guna2Button8";

this->guna2Button8->Size = System::Drawing::Size(196, 47);

this->guna2Button8->TabIndex = 7;

this->guna2Button8->Text = L"Отмена";

this->guna2Button8->Click += gcnew System::EventHandler(this, &broni::guna2Button8\_Click);

//

// guna2Button9

//

this->guna2Button9->BorderRadius = 30;

this->guna2Button9->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button9->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button9->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button9->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button9->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(255)), static\_cast<System::Int32>(static\_cast<System::Byte>(192)),

static\_cast<System::Int32>(static\_cast<System::Byte>(128)));

this->guna2Button9->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->guna2Button9->ForeColor = System::Drawing::Color::White;

this->guna2Button9->Location = System::Drawing::Point(462, 423);

this->guna2Button9->Name = L"guna2Button9";

this->guna2Button9->Size = System::Drawing::Size(196, 47);

this->guna2Button9->TabIndex = 8;

this->guna2Button9->Text = L"Отмена";

this->guna2Button9->Click += gcnew System::EventHandler(this, &broni::guna2Button9\_Click);

//

// MessageWarning

//

this->MessageWarning->Buttons = Guna::UI2::WinForms::MessageDialogButtons::OK;

this->MessageWarning->Caption = nullptr;

this->MessageWarning->Icon = Guna::UI2::WinForms::MessageDialogIcon::Warning;

this->MessageWarning->Parent = this;

this->MessageWarning->Style = Guna::UI2::WinForms::MessageDialogStyle::Light;

this->MessageWarning->Text = nullptr;

//

// pictureBox1

//

this->pictureBox1->Image = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox1.Image")));

this->pictureBox1->Location = System::Drawing::Point(37, 71);

this->pictureBox1->Name = L"pictureBox1";

this->pictureBox1->Size = System::Drawing::Size(50, 50);

this->pictureBox1->TabIndex = 9;

this->pictureBox1->TabStop = false;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label1->Location = System::Drawing::Point(93, 76);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(336, 39);

this->label1->TabIndex = 10;

this->label1->Text = L"Забронировать квест";

//

// panel1

//

this->panel1->BackColor = System::Drawing::SystemColors::ActiveCaptionText;

this->panel1->Location = System::Drawing::Point(37, 122);

this->panel1->Name = L"panel1";

this->panel1->Size = System::Drawing::Size(505, 3);

this->panel1->TabIndex = 11;

//

// panel2

//

this->panel2->BackColor = System::Drawing::SystemColors::ActiveCaptionText;

this->panel2->Location = System::Drawing::Point(37, 62);

this->panel2->Name = L"panel2";

this->panel2->Size = System::Drawing::Size(505, 3);

this->panel2->TabIndex = 12;

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label2->Location = System::Drawing::Point(33, 139);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(626, 23);

this->label2->TabIndex = 13;

this->label2->Text = L"Внимание, мы не берём комиссию с игроков за бронирование игры на сайте.";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(0)));

this->label3->Location = System::Drawing::Point(33, 171);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(469, 23);

this->label3->TabIndex = 14;

this->label3->Text = L"Нажмите на желаемое время, чтобы забронировать игру.";

//

// panel3

//

this->panel3->BackColor = System::Drawing::SystemColors::ActiveCaptionText;

this->panel3->Location = System::Drawing::Point(37, 476);

this->panel3->Name = L"panel3";

this->panel3->Size = System::Drawing::Size(661, 3);

this->panel3->TabIndex = 12;

//

// panel4

//

this->panel4->BackColor = System::Drawing::SystemColors::ActiveCaptionText;

this->panel4->Location = System::Drawing::Point(37, 207);

this->panel4->Name = L"panel4";

this->panel4->Size = System::Drawing::Size(660, 3);

this->panel4->TabIndex = 13;

//

// label4

//

this->label4->AutoSize = true;

this->label4->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 12.75F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label4->Location = System::Drawing::Point(58, 295);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(78, 46);

this->label4->TabIndex = 15;

this->label4->Text = L"25 июня\r\nВторник";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Font = (gcnew System::Drawing::Font(L"Trebuchet MS", 23.25F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label5->Location = System::Drawing::Point(30, 9);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(177, 39);

this->label5->TabIndex = 16;

this->label5->Text = L"Квест ОНО";

//

// guna2Button4

//

this->guna2Button4->DisabledState->BorderColor = System::Drawing::Color::DarkGray;

this->guna2Button4->DisabledState->CustomBorderColor = System::Drawing::Color::DarkGray;

this->guna2Button4->DisabledState->FillColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(169)),

static\_cast<System::Int32>(static\_cast<System::Byte>(169)), static\_cast<System::Int32>(static\_cast<System::Byte>(169)));

this->guna2Button4->DisabledState->ForeColor = System::Drawing::Color::FromArgb(static\_cast<System::Int32>(static\_cast<System::Byte>(141)),

static\_cast<System::Int32>(static\_cast<System::Byte>(141)), static\_cast<System::Int32>(static\_cast<System::Byte>(141)));

this->guna2Button4->Font = (gcnew System::Drawing::Font(L"Segoe UI", 9));

this->guna2Button4->ForeColor = System::Drawing::Color::White;

this->guna2Button4->Location = System::Drawing::Point(259, 9);

this->guna2Button4->Name = L"guna2Button4";

this->guna2Button4->Size = System::Drawing::Size(180, 45);

this->guna2Button4->TabIndex = 17;

this->guna2Button4->Text = L"guna2Button4";

this->guna2Button4->Click += gcnew System::EventHandler(this, &broni::guna2Button4\_Click);

//

// broni

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(740, 628);

this->Controls->Add(this->guna2Button4);

this->Controls->Add(this->label5);

this->Controls->Add(this->label4);

this->Controls->Add(this->panel4);

this->Controls->Add(this->panel3);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->panel2);

this->Controls->Add(this->panel1);

this->Controls->Add(this->label1);

this->Controls->Add(this->pictureBox1);

this->Controls->Add(this->guna2Button9);

this->Controls->Add(this->guna2Button8);

this->Controls->Add(this->guna2Button7);

this->Controls->Add(this->guna2Button6);

this->Controls->Add(this->guna2Button5);

this->Controls->Add(this->guna2Button3);

this->Controls->Add(this->guna2Button2);

this->Controls->Add(this->guna2Button1);

this->Name = L"broni";

this->Text = L"broni";

this->Load += gcnew System::EventHandler(this, &broni::broni\_Load);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private:

bool bron1 = false;

bool bron2 = false;

bool bron3 = false;

bool bron4 = false;

String^ User1 = "";

String^ User2 = "";

String^ User3 = "";

String^ User4 = "";

String^ logFilePath = "Users\\log.txt";

String^ logDirectory = Path::GetDirectoryName(logFilePath);

String^ separator = " - ";

private: Void SaveBron() {

String^ configFilePath = "bron.xml";

ClassConfig^ bron = gcnew ClassConfig();

MessageWarning->Show("Сохранение конфигурации...");

if (bron->btn1 != bron1 || bron->UserBtn1 != User1) {

bron->UserBtn1 = bron1 ? User1 : ""; // Сохраняем логин пользователя, если кнопка была включена

bron->btn1 = bron1;

}

if (bron->btn2 != bron2 || bron->UserBtn2 != User2) {

bron->UserBtn2 = bron2 ? User2 : ""; // Сохраняем логин пользователя, если кнопка была включена

bron->btn2 = bron2;

}

if (bron->btn3 != bron3 || bron->UserBtn3 != User3) {

bron->UserBtn3 = bron3 ? User3 : ""; // Сохраняем логин пользователя, если кнопка была включена

bron->btn3 = bron3;

}

if (bron->btn4 != bron4 || bron->UserBtn4 != User4) {

bron->UserBtn4 = bron4 ? User4 : ""; // Сохраняем логин пользователя, если кнопка была включена

bron->btn4 = bron4;

}

bron->SaveConfig(); // Сохраняем измененную конфигурацию

}

private: Void LoadBron() {

if (File::Exists("bron.xml")) {

ClassConfig^ bron = bron->LoadConfig();

bron1 = bron->btn1;

bron2 = bron->btn2;

bron3 = bron->btn3;

bron4 = bron->btn4;

User1 = bron->UserBtn1;

User2 = bron->UserBtn2;

User3 = bron->UserBtn3;

User4 = bron->UserBtn4;

// Проверяем, можно ли включить каждую кнопку с учетом текущего пользователя

guna2Button1->Enabled = (!bron1 && User1 == "") || User1 == Login;

guna2Button2->Enabled = (!bron2 && User2 == "") || User2 == Login;

guna2Button3->Enabled = (!bron3 && User3 == "") || User3 == Login;

guna2Button5->Enabled = (!bron4 && User4 == "") || User4 == Login;

// Отключаем кнопки, если они забронированы

if (bron1) guna2Button1->Enabled = false;

if (bron2) guna2Button2->Enabled = false;

if (bron3) guna2Button3->Enabled = false;

if (bron4) guna2Button5->Enabled = false;

}

}

void LogAction(String^ timeSlot) {

try {

// Создаем директорию, если она не существует

if (!Directory::Exists(logDirectory)) {

Directory::CreateDirectory(logDirectory);

}

StreamWriter^ sw = gcnew StreamWriter(logFilePath, true);

sw->WriteLine(Login + separator + "ОНО забронировал время" + separator + timeSlot);

sw->Close();

}

catch (Exception^ ex) {

MessageBox::Show("Error writing to log file: " + ex->Message);

}

}

private: Void LogLogin(String^timeSlot){

try {

// Проверяем, существует ли директория для логов, и создаем ее при необходимости

if (!Directory::Exists(logDirectory)) {

Directory::CreateDirectory(logDirectory);

}

// Получаем текущую дату и время

String^ currentDate = DateTime::Now.ToString("yyyy-MM-dd HH:mm:ss");

// Формируем строку для записи в файл

String^ logMessage = "Квест ОНО - " + currentDate + " - " + timeSlot;

// Путь к файлу текущего пользователя

String^ userLogFilePath = Path::Combine(logDirectory, Login + ".txt");

// Записываем лог

StreamWriter^ sw = gcnew StreamWriter(userLogFilePath, true);

sw->WriteLine(logMessage);

sw->Close();

}

catch (Exception^ ex) {

// Обработка исключений при записи в файл

MessageBox::Show("Error writing to log file: " + ex->Message);

}

}

void RemoveLogEntry(String^ timeSlot) {

String^ lineToRemovePrefix = Login + separator + "ОНО забронировал время" + separator;

try {

// Проверяем, существует ли файл лога

if (File::Exists(logFilePath)) {

array<String^>^ lines = File::ReadAllLines(logFilePath);

cliext::list<String^>^ updatedLines = gcnew cliext::list<String^>();

// Удаляем строки, соответствующие указанному времени

for each (String ^ line in lines) {

if (!line->StartsWith(lineToRemovePrefix + timeSlot)) {

updatedLines->push\_back(line);

}

}

// Преобразуем cliext::list в array

array<String^>^ updatedLinesArray = gcnew array<String^>(updatedLines->size());

int i = 0;

if (!updatedLines->empty()) {

for each (String ^ line in updatedLines) {

updatedLinesArray[i++] = line;

}

}

// Записываем обновленный лог

File::WriteAllLines(logFilePath, updatedLinesArray);

}

// Удаляем логин из пользовательского файла

String^ userLogFilePath = Path::Combine(logDirectory, Login + ".txt");

if (File::Exists(userLogFilePath)) {

array<String^>^ userLines = File::ReadAllLines(userLogFilePath);

cliext::list<String^>^ updatedUserLines = gcnew cliext::list<String^>();

// Удаляем строки, соответствующие указанному времени

for each (String ^ line in userLines) {

if (!line->EndsWith(separator + timeSlot)) {

updatedUserLines->push\_back(line);

}

}

// Преобразуем cliext::list в array

array<String^>^ updatedUserLinesArray = gcnew array<String^>(updatedUserLines->size());

int j = 0;

if (!updatedUserLines->empty()) {

for each (String ^ line in updatedUserLines) {

updatedUserLinesArray[j++] = line;

}

}

// Записываем обновленный лог

File::WriteAllLines(userLogFilePath, updatedUserLinesArray);

}

}

catch (Exception^ ex) {

// Обработка исключений

MessageBox::Show("Error removing line from log file: " + ex->Message);

}

}

private: System::Void guna2Button1\_Click(System::Object^ sender, System::EventArgs^ e) {

if (!bron1 || User1 == Login) {

bron1 = true;

User1 = Login;

SaveBron();

guna2Button1->Enabled = false;

LogAction("9:00");

LogLogin("9:00");

}

else {

MessageWarning->Show("Вы не можете забронировать это время");

}

}

private: System::Void guna2Button2\_Click(System::Object^ sender, System::EventArgs^ e) {

if (!bron2 || User2 == Login) {

bron2 = true;

User2 = Login;

SaveBron();

guna2Button2->Enabled = false;

LogAction("10:00");

LogLogin("10:00");

}

else {

MessageWarning->Show("Вы не можете забронировать это время");

}

}

private: System::Void guna2Button3\_Click(System::Object^ sender, System::EventArgs^ e) {

if (!bron3 || User3 == Login) {

bron3 = true;

User3 = Login;

SaveBron();

guna2Button3->Enabled = false;

LogAction("13:00");

LogLogin("13:00");

}

else {

MessageWarning->Show("Вы не можете забронировать это время");

}

}

private: System::Void guna2Button5\_Click(System::Object^ sender, System::EventArgs^ e) {

if (!bron4 || User4 == Login) {

bron4 = true;

User4 = Login;

SaveBron();

guna2Button5->Enabled = false;

LogAction("16:00");

LogLogin("16:00");

}

else {

MessageWarning->Show("Вы не можете забронировать это время");

}

}

private: System::Void guna2Button6\_Click(System::Object^ sender, System::EventArgs^ e) {

if (bron1 && User1 == Login) {

bron1 = false;

User1 = ""; // Сбрасываем логин пользователя при отмене бронирования

SaveBron();

guna2Button1->Enabled = true;

MessageWarning->Show("Бронь снята");

RemoveLogEntry("9:00");

}

else {

MessageWarning->Show("Вы не можете отменить это время");

}

}

private: System::Void guna2Button7\_Click(System::Object^ sender, System::EventArgs^ e) {

if (bron2 && User2 == Login) {

bron2 = false;

User2 = ""; // Сбрасываем логин пользователя при отмене бронирования

SaveBron();

guna2Button2->Enabled = true;

MessageWarning->Show("Бронь снята");

RemoveLogEntry("10:00");

}

else {

MessageWarning->Show("Вы не можете отменить это время");

}

}

private: System::Void guna2Button8\_Click(System::Object^ sender, System::EventArgs^ e) {

if (bron3 && User3 == Login) {

bron3 = false;

User3 = ""; // Сбрасываем логин пользователя при отмене бронирования

SaveBron();

guna2Button3->Enabled = true;

MessageWarning->Show("Бронь снята");

RemoveLogEntry("13:00");

}

else {

MessageWarning->Show("Вы не можете отменить это время");

}

}

private: System::Void guna2Button9\_Click(System::Object^ sender, System::EventArgs^ e) {

if (bron4 && User4 == Login) {

bron4 = false;

User4 = ""; // Сбрасываем логин пользователя при отмене бронирования

SaveBron();

guna2Button5->Enabled = true;

MessageWarning->Show("Бронь снята");

RemoveLogEntry("16:00");

}

else {

MessageWarning->Show("Вы не можете отменить это время");

}

}

private: System::Void broni\_Load(System::Object^ sender, System::EventArgs^ e) {

LoadBron();

}

private: Void EditBronTime(String^ oldTimeSlot, String^ newTimeSlot) {

if (newTimeSlot == "9:00" && (!bron1 || User1 == Login)) {

if (oldTimeSlot == "9:00") return;

RemoveLogEntry(oldTimeSlot);

bron1 = true;

User1 = Login;

}

else if (newTimeSlot == "10:00" && (!bron2 || User2 == Login)) {

if (oldTimeSlot == "10:00") return;

RemoveLogEntry(oldTimeSlot);

bron2 = true;

User2 = Login;

}

else if (newTimeSlot == "13:00" && (!bron3 || User3 == Login)) {

if (oldTimeSlot == "13:00") return;

RemoveLogEntry(oldTimeSlot);

bron3 = true;

User3 = Login;

}

else if (newTimeSlot == "16:00" && (!bron4 || User4 == Login)) {

if (oldTimeSlot == "16:00") return;

RemoveLogEntry(oldTimeSlot);

bron4 = true;

User4 = Login;

}

else {

MessageWarning->Show("Вы не можете выбрать это время");

return;

}

LogAction(newTimeSlot);

LogLogin(newTimeSlot);

SaveBron();

// Освобождаем старое время

if (oldTimeSlot == "9:00") {

bron1 = false;

User1 = "";

guna2Button1->Enabled = true;

}

else if (oldTimeSlot == "10:00") {

bron2 = false;

User2 = "";

guna2Button2->Enabled = true;

}

else if (oldTimeSlot == "13:00") {

bron3 = false;

User3 = "";

guna2Button3->Enabled = true;

}

else if (oldTimeSlot == "16:00") {

bron4 = false;

User4 = "";

guna2Button5->Enabled = true;

}

SaveBron();

}

private: String^ ShowTimeSlotSelectionDialog() {

YourNamespace::TimeSlotSelectionForm^ form = gcnew YourNamespace::TimeSlotSelectionForm();

if (form->ShowDialog() == System::Windows::Forms::DialogResult::OK) {

return form->GetSelectedTimeSlot();

}

return nullptr;

}private: System::Void guna2Button4\_Click(System::Object^ sender, System::EventArgs^ e) {

String^ oldTimeSlot = nullptr;

if (bron1 && User1 == Login) {

oldTimeSlot = "9:00";

}

else if (bron2 && User2 == Login) {

oldTimeSlot = "10:00";

}

else if (bron3 && User3 == Login) {

oldTimeSlot = "13:00";

}

else if (bron4 && User4 == Login) {

oldTimeSlot = "16:00";

}

if (oldTimeSlot != nullptr) {

String^ newTimeSlot = ShowTimeSlotSelectionDialog();

if (newTimeSlot != nullptr) {

EditBronTime(oldTimeSlot, newTimeSlot);

}

}

else {

MessageWarning->Show("Вы не можете редактировать это время");

}

}

};

}

**Приложение Б**

**Результат работы приложения**



Рисунок Б.1 — Результат работы тест-кейса авторизации

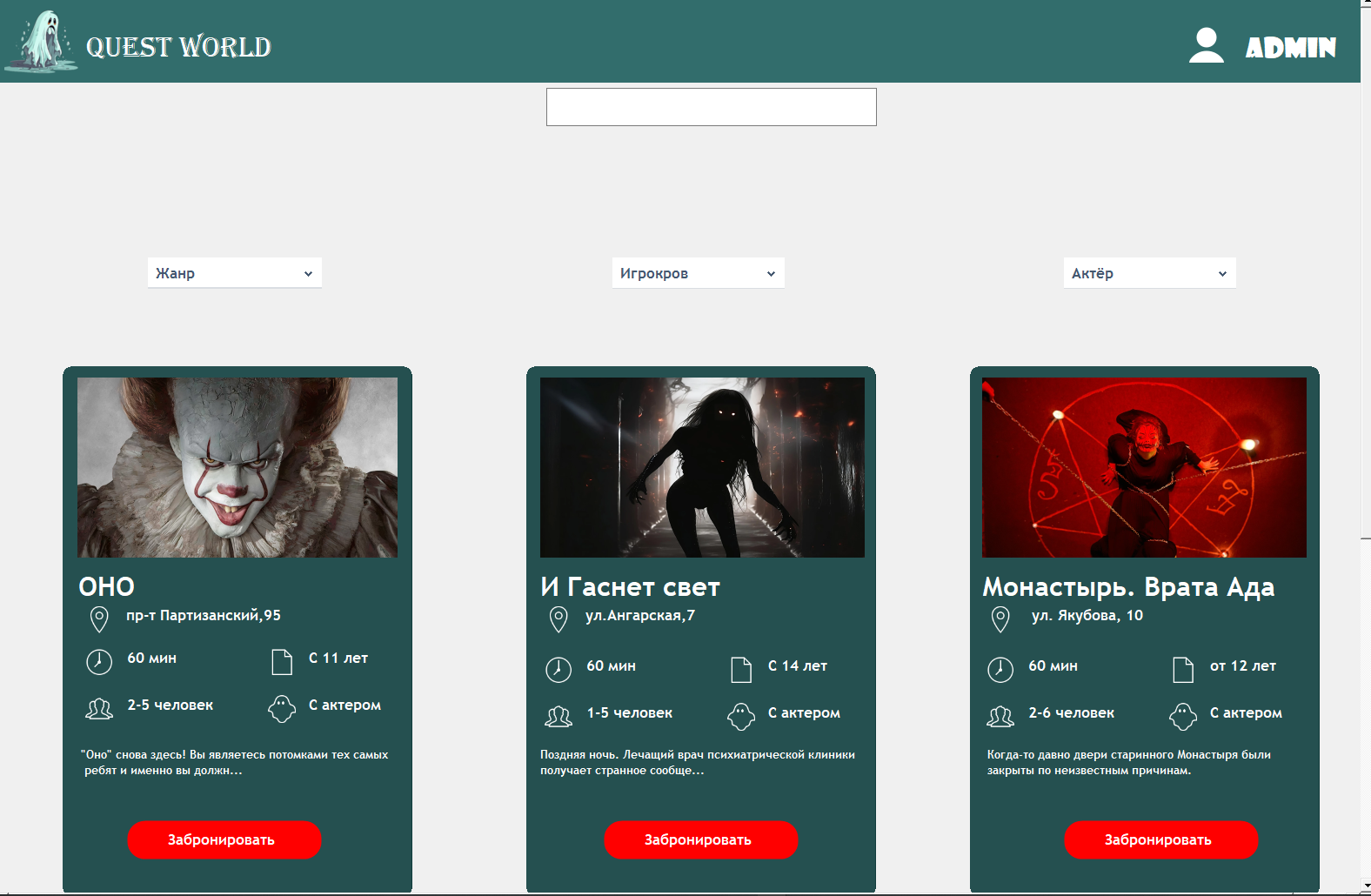


Рисунок Б.2 — Результат завершения работы тест-кейса авторизации



Рисунок Б.3 — Результат работы тест-кейса регистрации

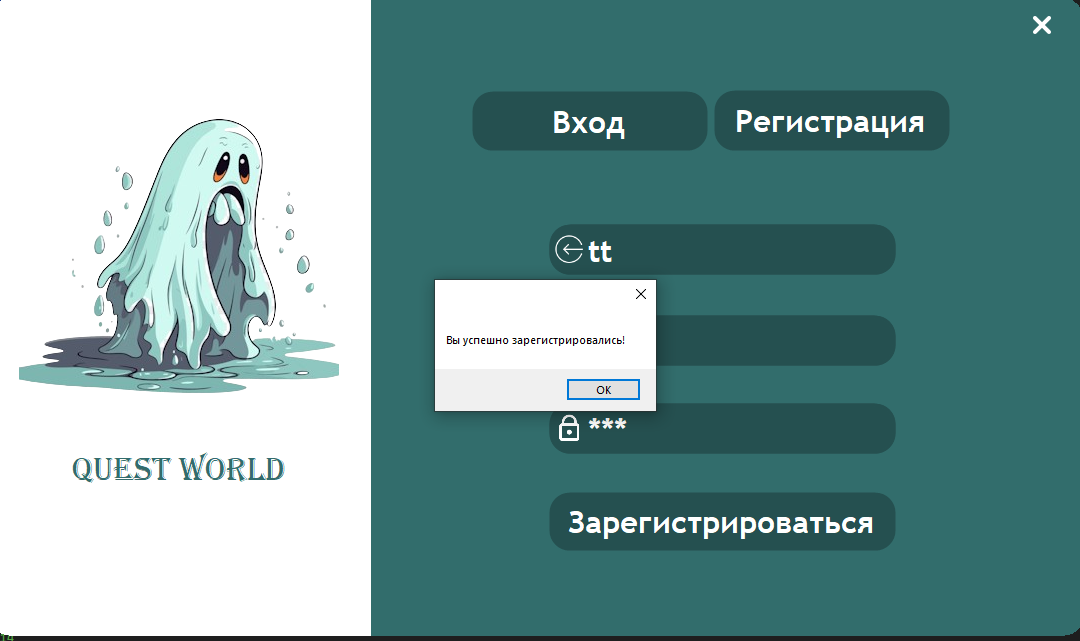


Рисунок Б.4. — Результат завершения работы тест-кейса регистрации

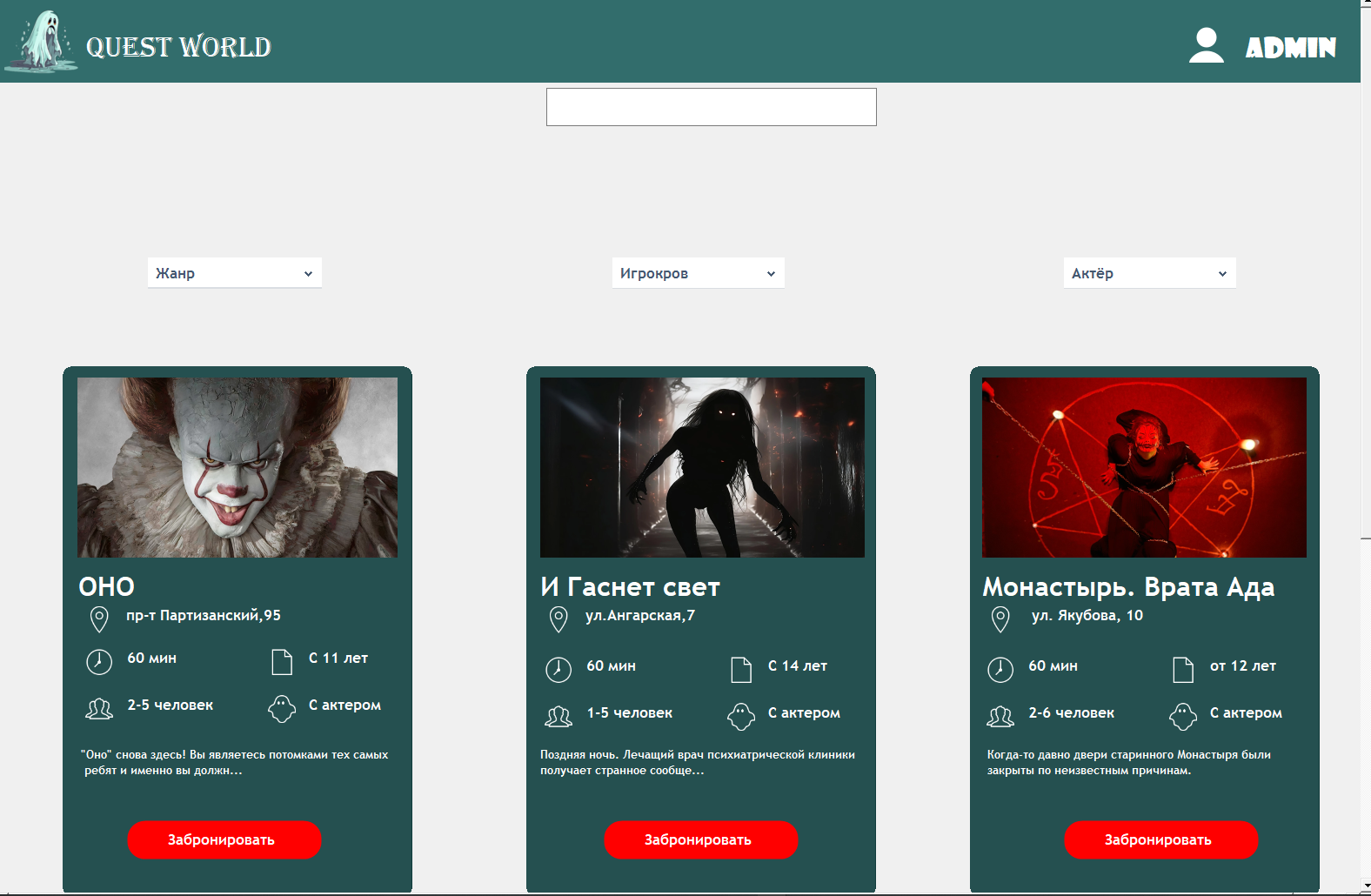


Рисунок Б.5 — Результат работы тест-кейса поиска



Рисунок Б.6— Результат завершения работы тест-кейса поиска.

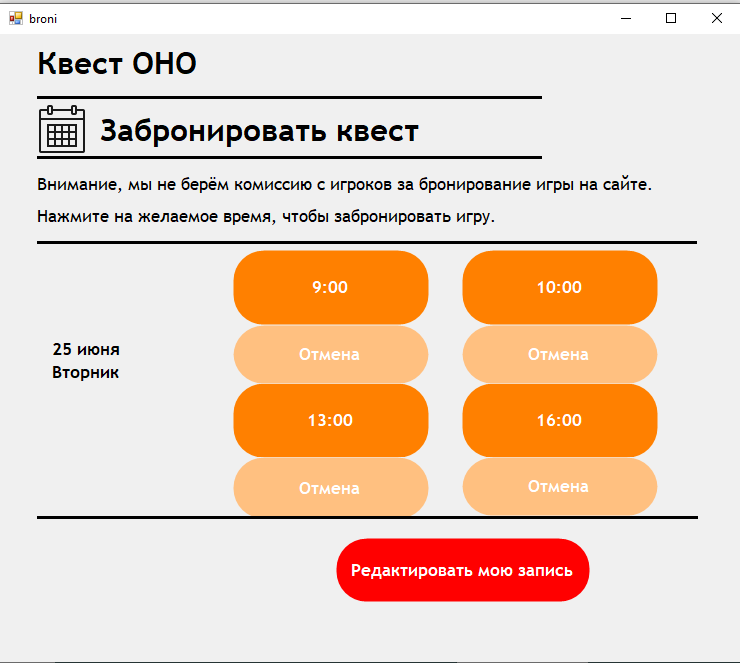


Рисунок Б.7 — Результат работы тест-кейса бронирования времени

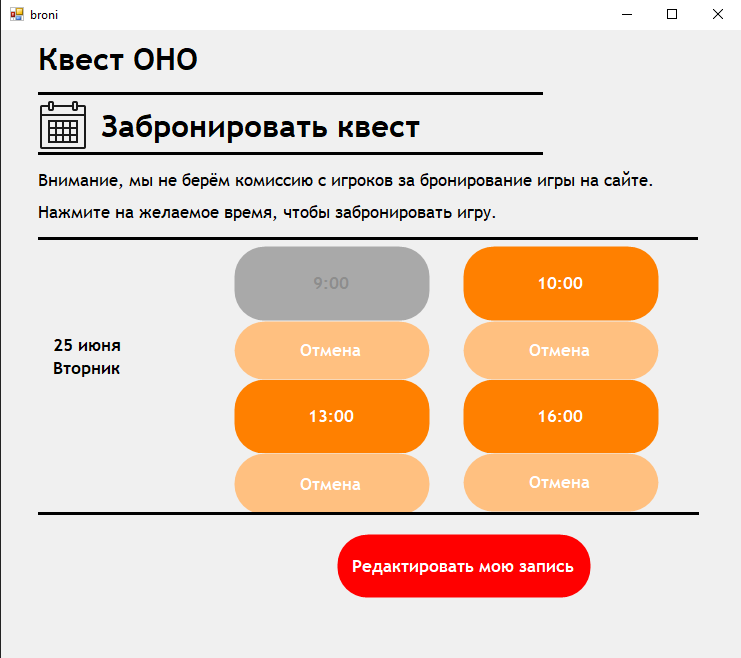


Рисунок Б.8 — Результат завершения работы тест-кейса для бронирования времени.

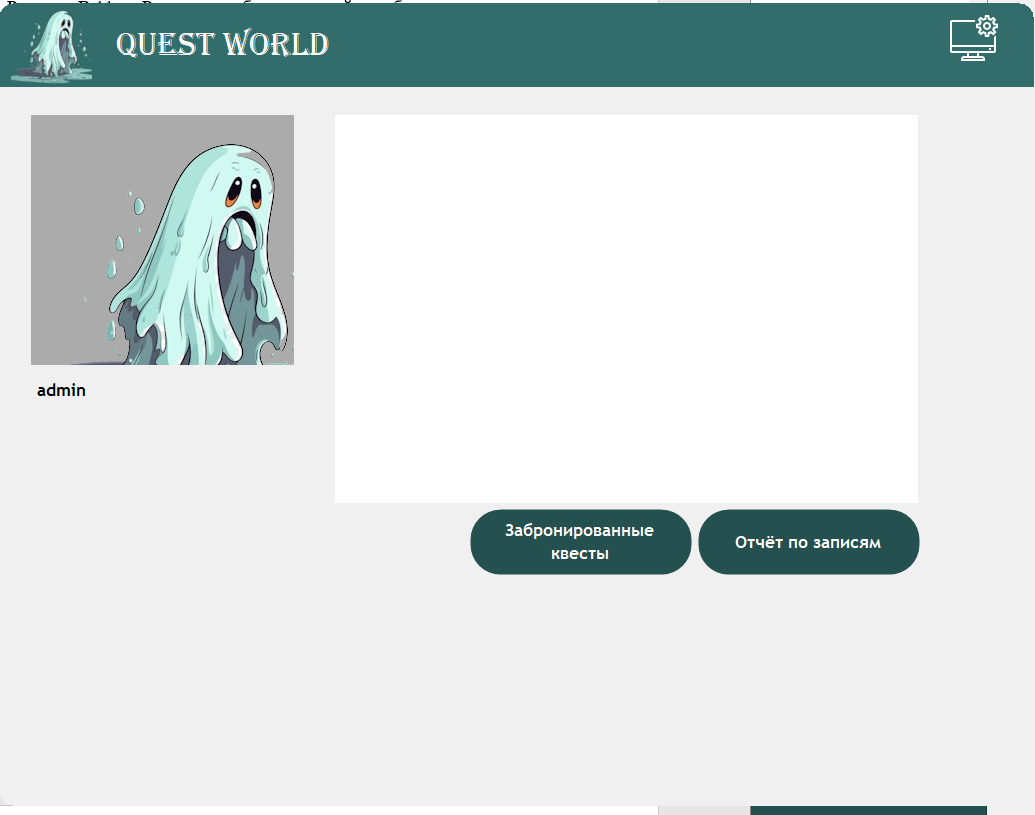


Рисунок Б.9 — Результат работы тест-кейса просмотр записей на квесты в профиле

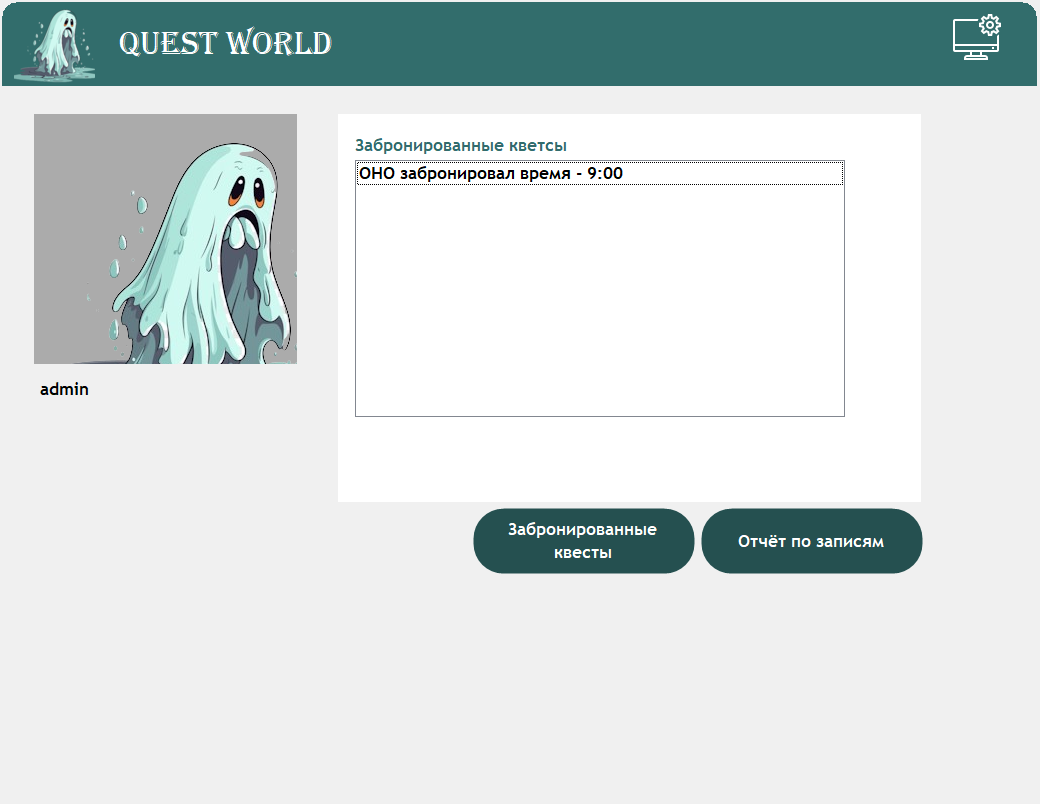


Рисунок Б.10 — Результат завершения работы тест-кейса просмотр записей на квесты в профиле

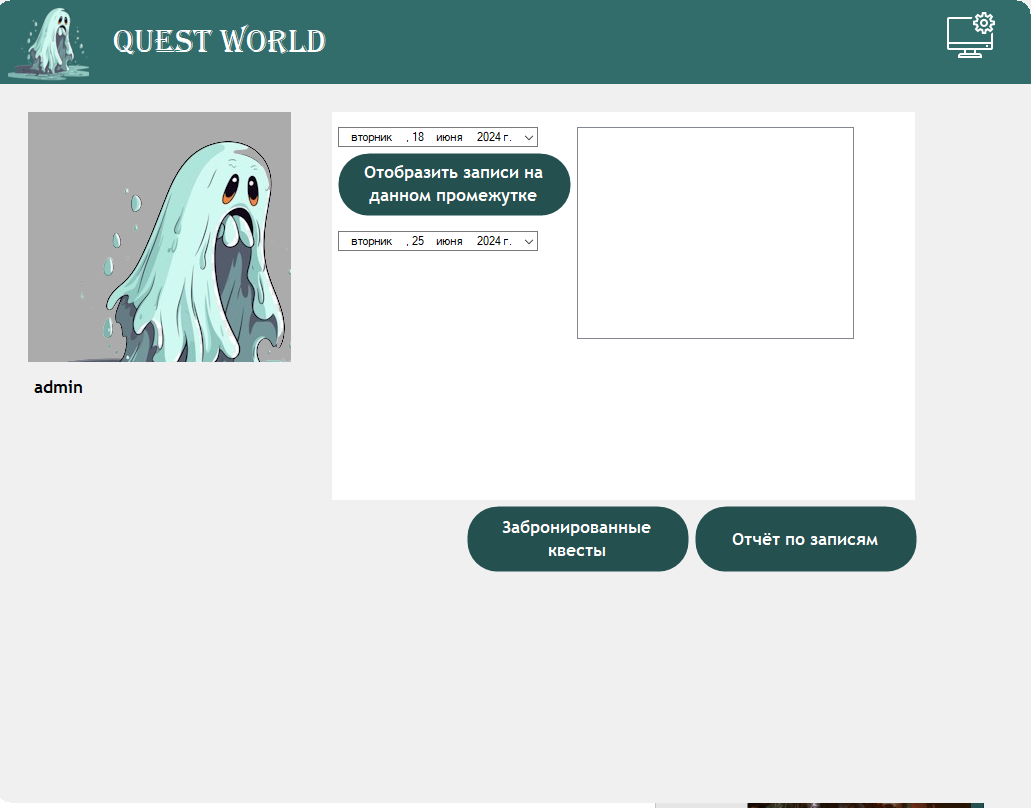


Рисунок В.11 — Результат работы тест-кейса просмотр отчёта по записям

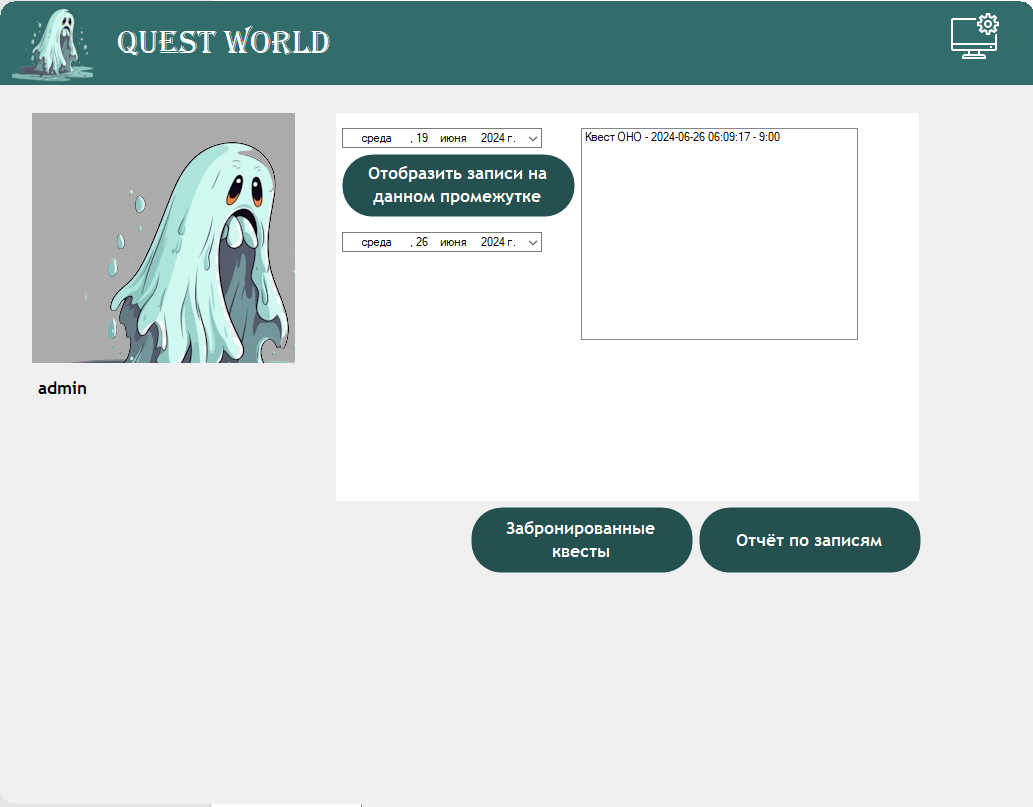


Рисунок В.12 — Результат завершения работы тест-кейса кейса просмотра отчёта по записям

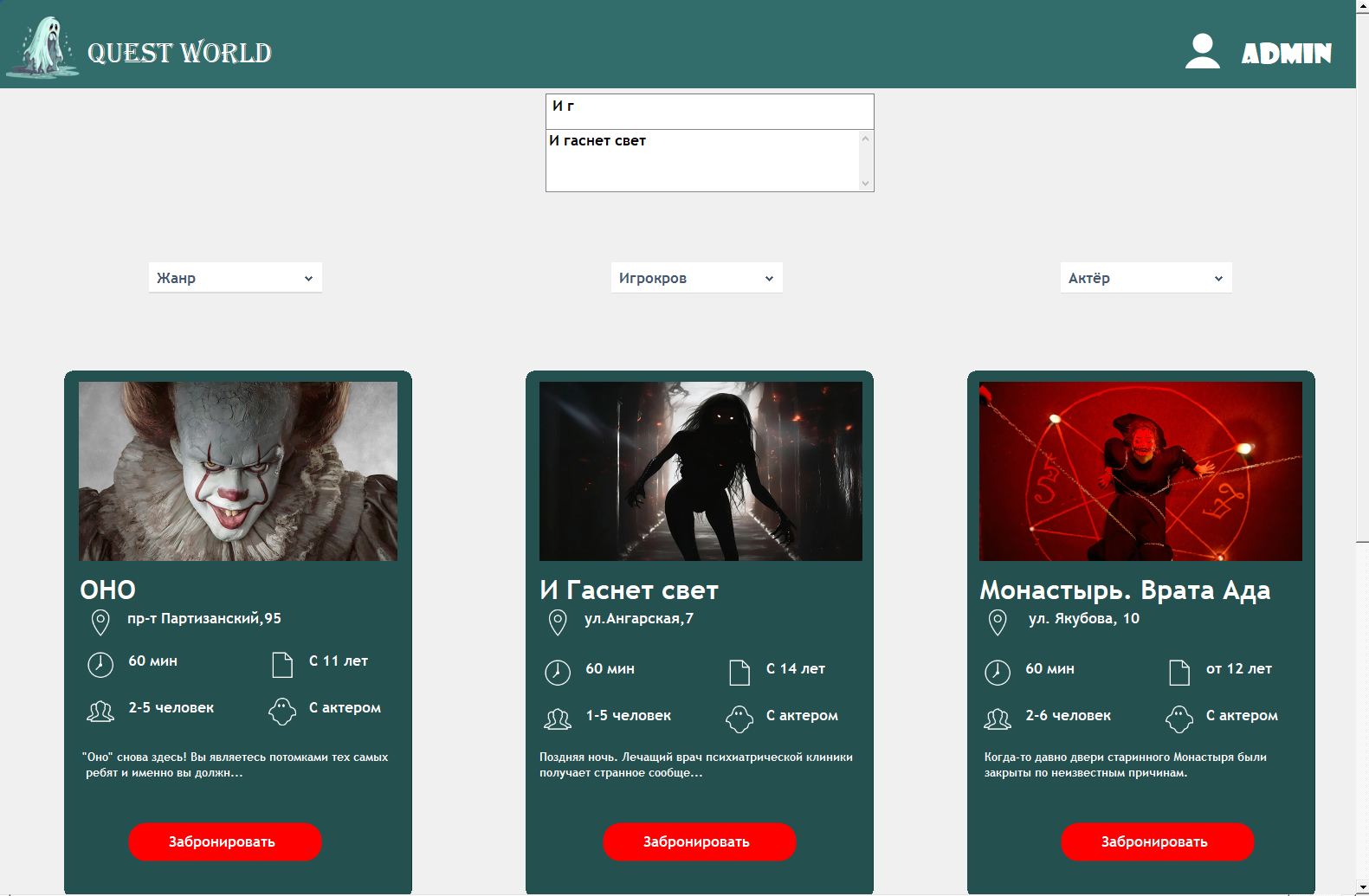


Рисунок В.13 — Результат работы тест-кейса для просмотра информации о квесте

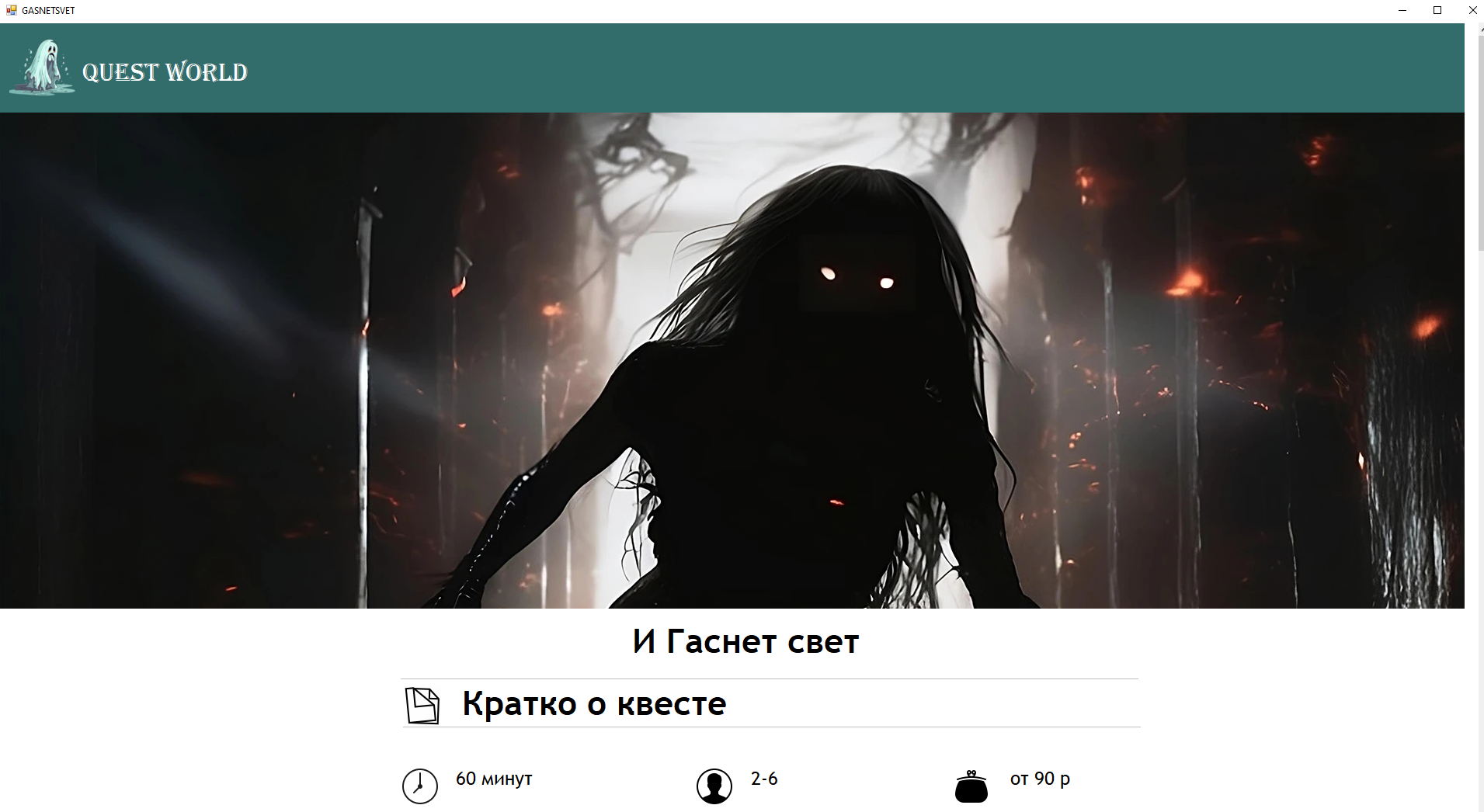


Рисунок В.14 — Результат завершения работы просмотра информации о квесте

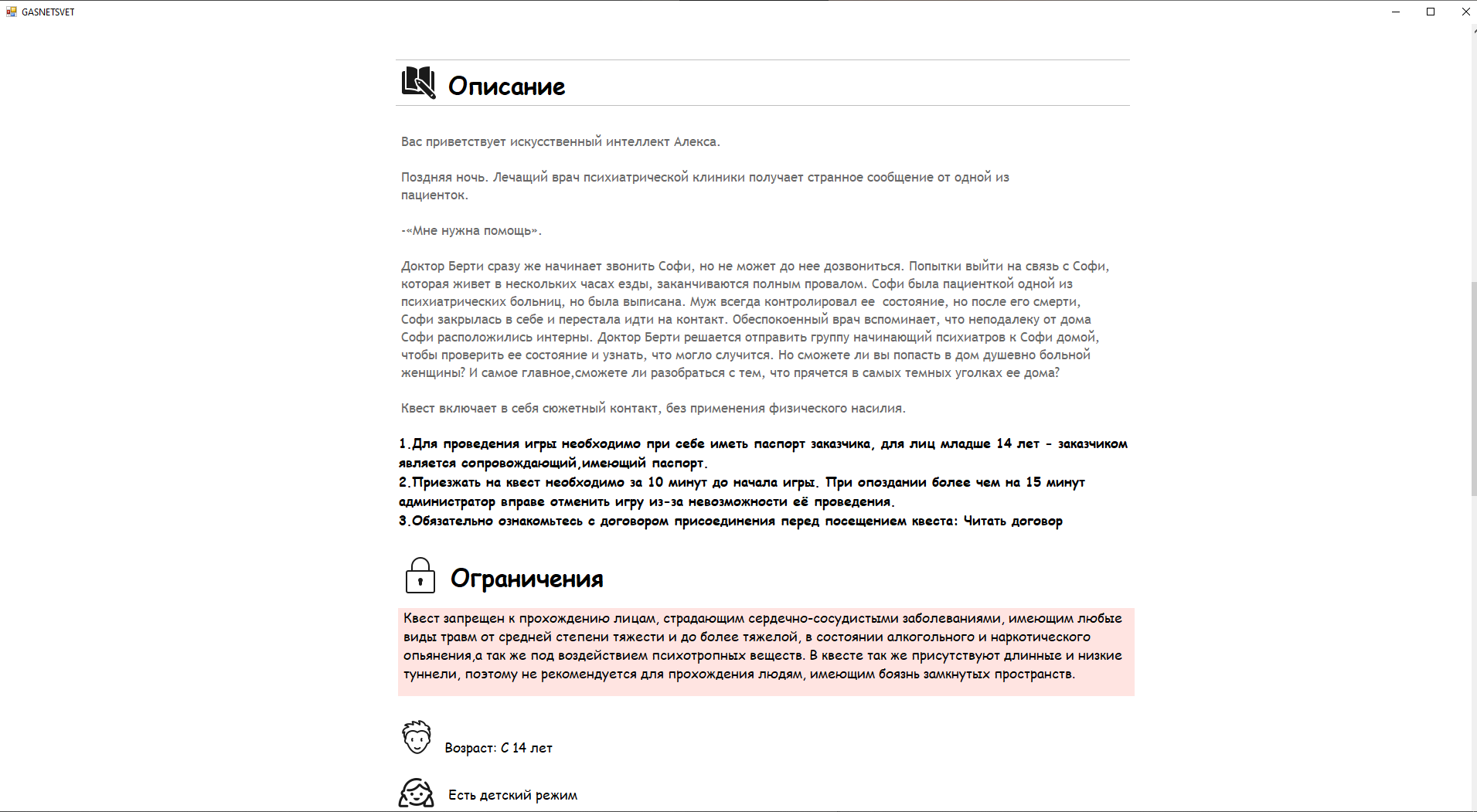


Рисунок В.15 — Результат завершения работы просмотра информации о квесте

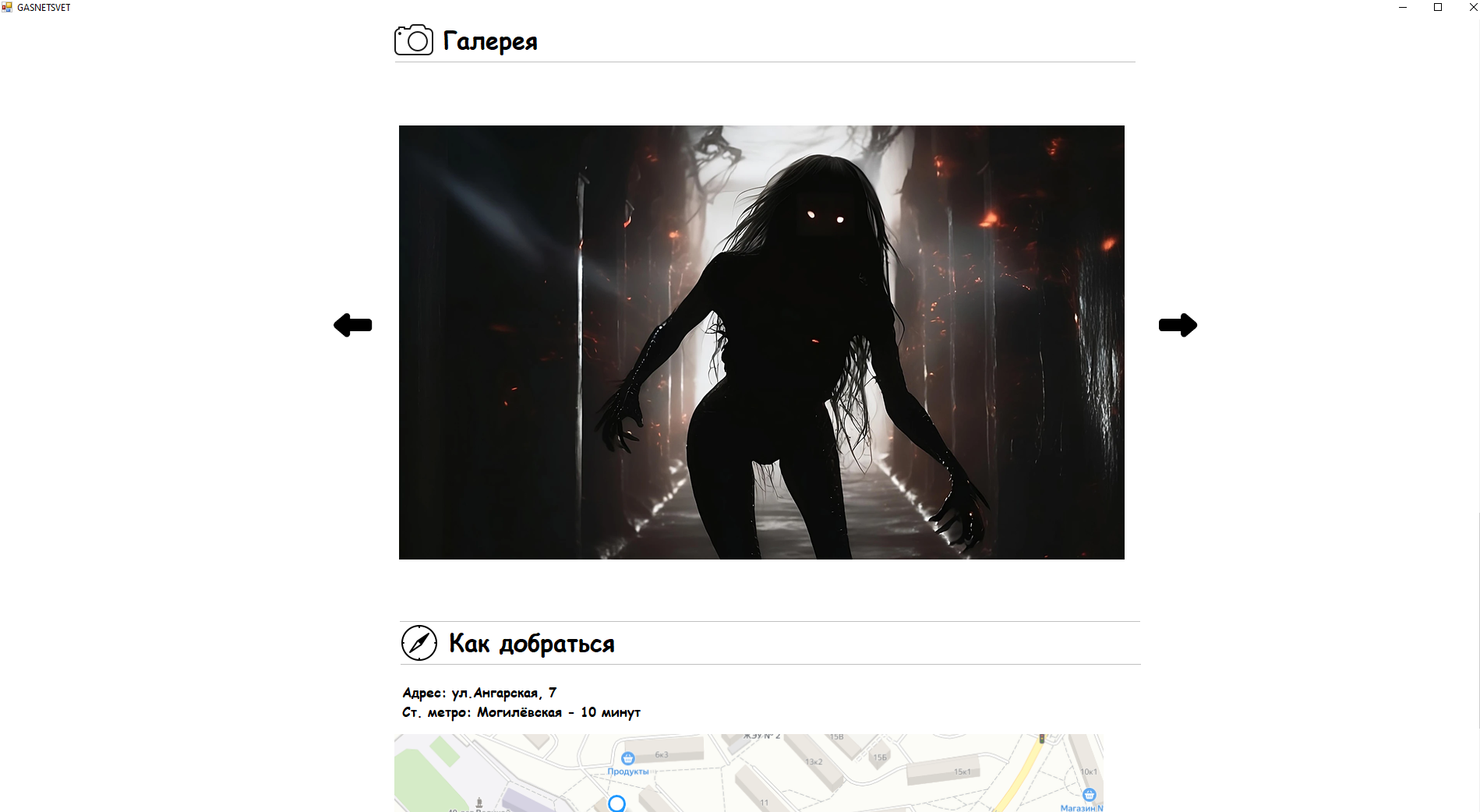


Рисунок В.16 — Результат завершения работы просмотра информации о квесте

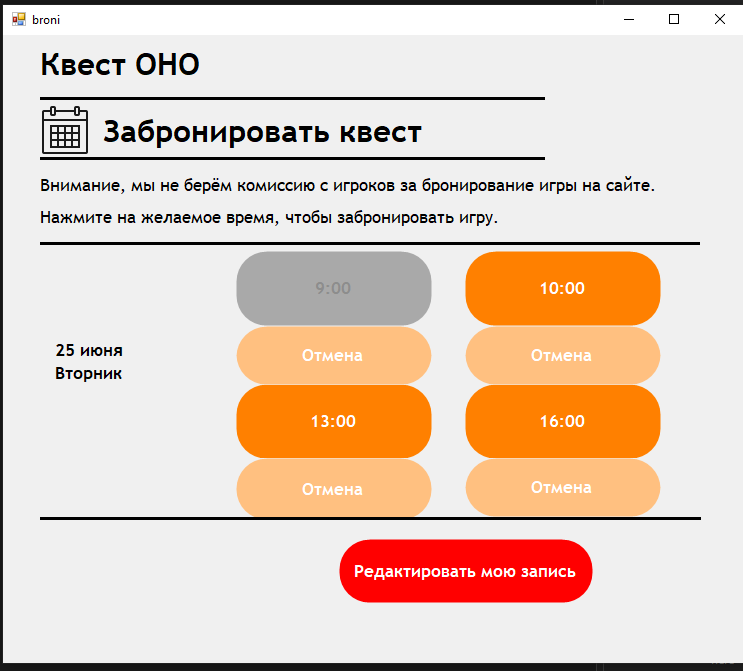


Рисунок В.17 — Результат работы тест-кейса для удаления бронирования квеста

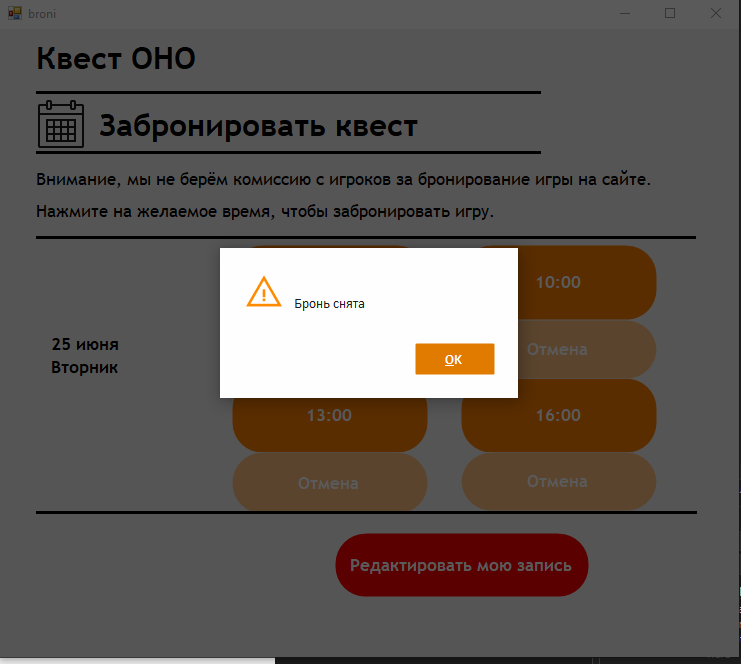


Рисунок В.18 — Результат завершения работы тест-кейса удаления бронирования квеста

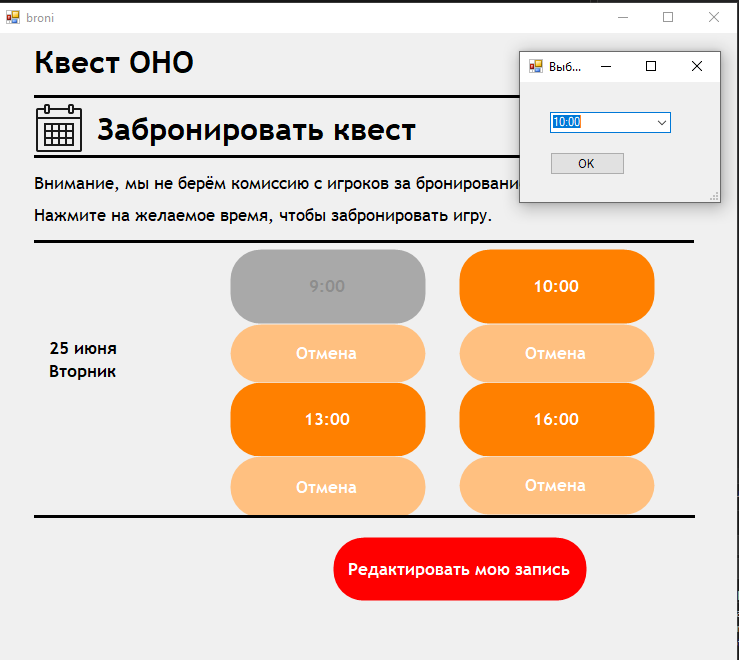


Рисунок Б.19 — Результат работы тест-кейса для редактирования забронированного времени.

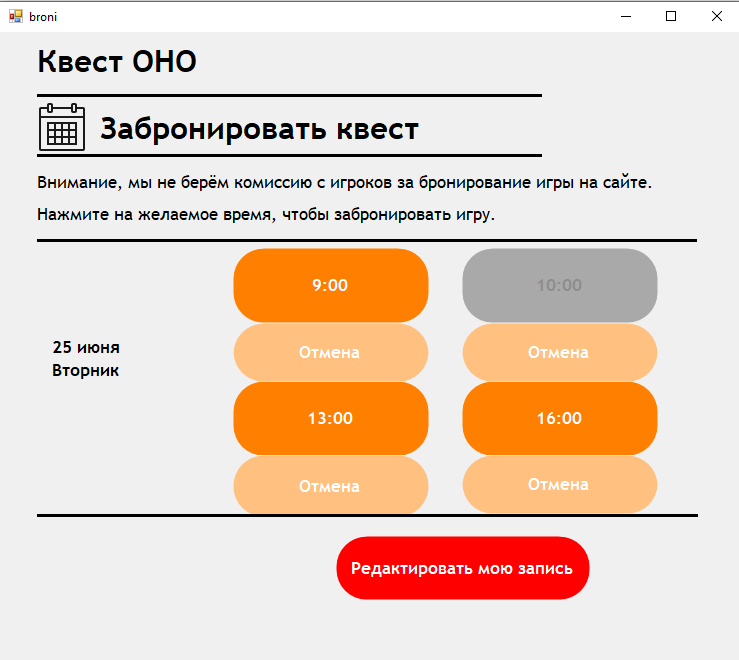


Рисунок Б.20 — Результат завершения работы тест-кейса для редактирования забронированного времени.

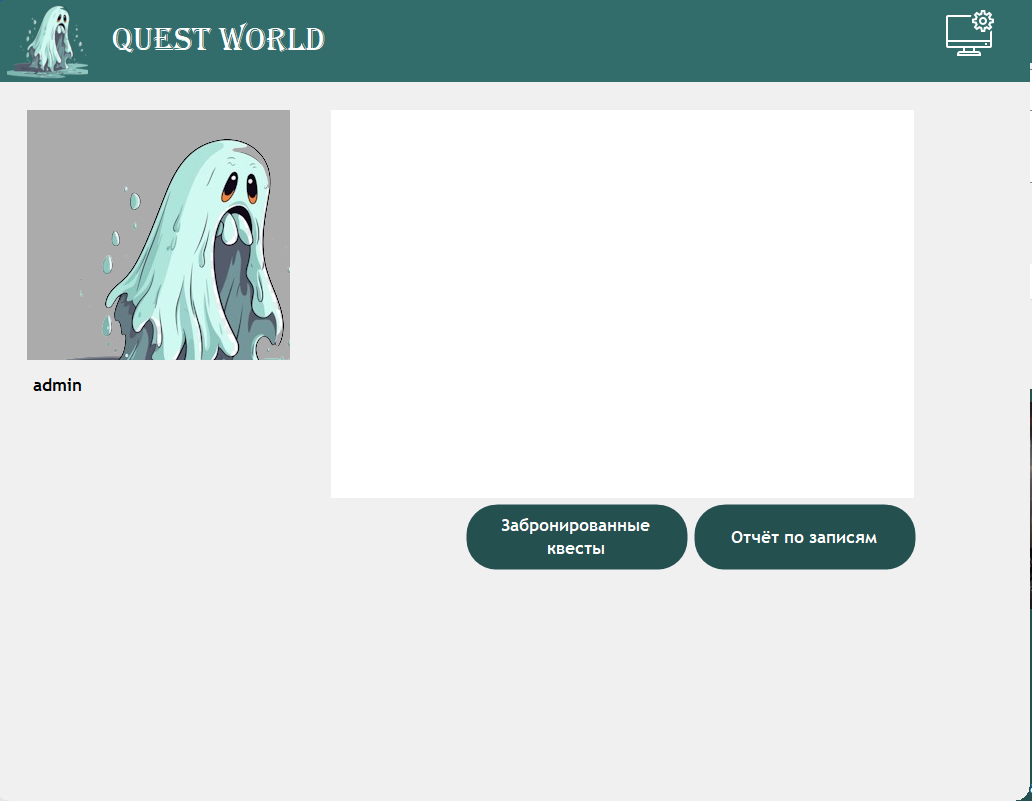


Рисунок В.21 — Результат работы тест-кейса просмотра руководства пользователя

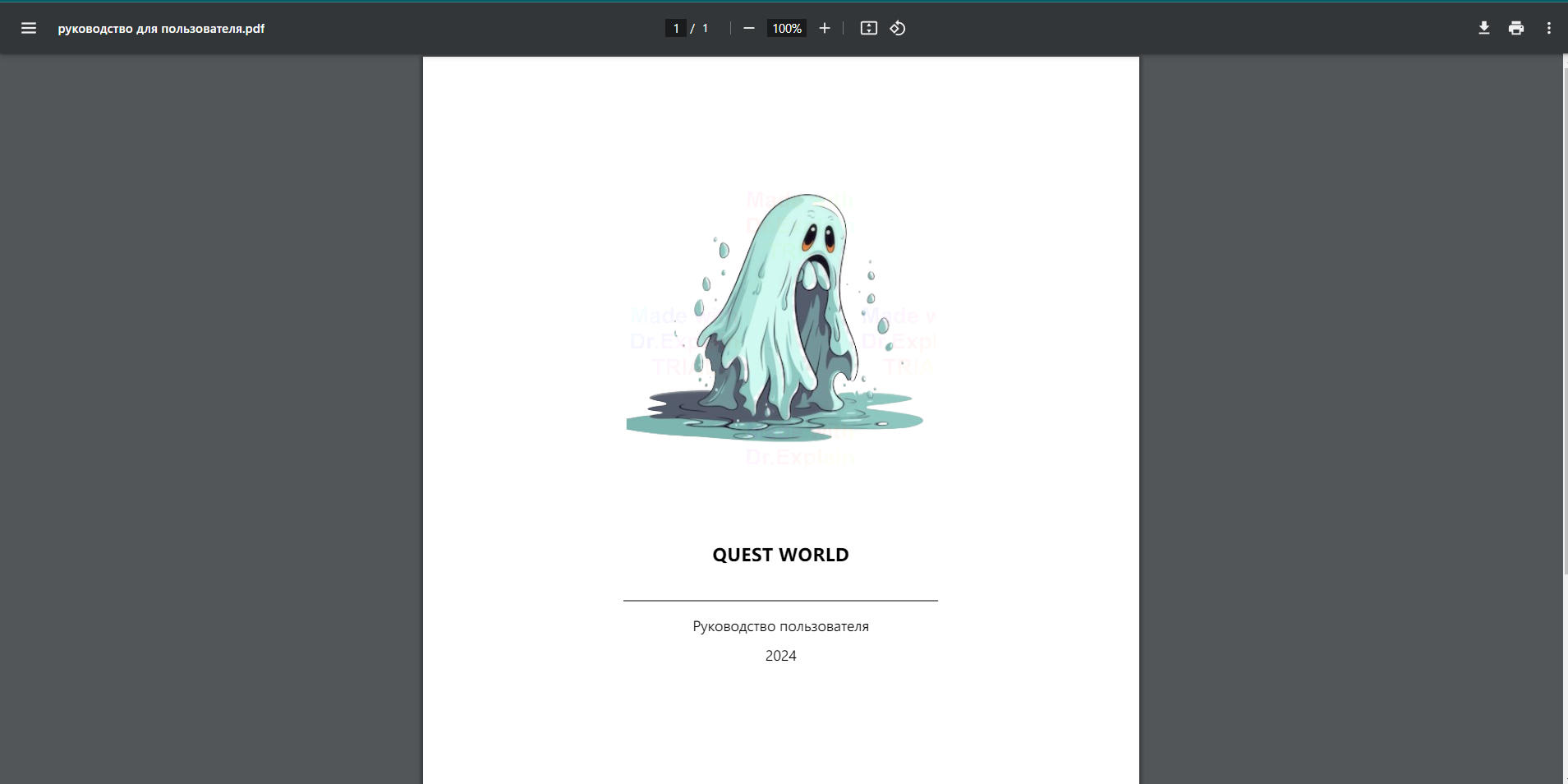


Рисунок В.22 — Результат завершения работы тест-кейса просмотра руководства пользователя

КП Т.295015.401 ГЧ

КП Т.295015.401 ГЧ

Утверд.

Багласова Т.Г.

Т. Контр.

Листов 2

Лист 1

№ докум.

Подпись

Дата

Изм.

Лист

У

Масса

Лит.

Масштаб

Реценз.

КБиП

Н. Контр.

Взам.инв.№

Подп. и дата

Инв.№подл.

Подп. и дата

Инв.№дубл.

array<String^>^ logFiles = Directory::GetFiles("Users", "\*.txt");

String^ date array<String^>^ logFiles = Directory::GetFiles("Users", "\*.txt");

String^ dateFormat = "yyyy-MM-dd HH:mm:ss";

Debug::WriteLine("Проверяем логины в диапазоне: " + startDate.ToString() + " - " + endDate.ToString());

Конец

String^ selectedTimeSlot;

public: String^ GetSelectedTimeSlot() {

return selectedTimeSlot;

}

private: Void SaveBron() {

String^ configFilePath = "bron.xml";

ClassConfig^ bron = gcnew ClassConfig();

MessageWarning->Show("Сохранение конфигурации...");

успешно

String^ oldTimeSlot = nullptr;

if (oldTimeSlot != nullptr) {

String^ newTimeSlot = ShowTimeSlotSelectionDialog();

if (newTimeSlot != nullptr) {

EditBronTime(oldTimeSlot, newTimeSlot);

}

array<String^>^ logFiles = Directory::GetFiles("Users", "\*.txt");

String^ date array<String^>^ logFiles = Directory::GetFiles("Users", "\*.txt");

String^ dateFormat = "yyyy-MM-dd HH:mm:ss";

Debug::WriteLine("Проверяем логины в диапазоне: " + startDate.ToString() + " - " + endDate.ToString());

for each (String ^ logFile in logFiles) { array<String^>^ lines = Format = "yyyy-MM-dd HH:mm:ss";

Debug::WriteLine("Проверяем логины в диапазоне: " + startDate.ToString() + " - " + endDate.ToString());

for each (String ^ logFile in logFiles) { array<String^>^ lines = File::ReadAllLines(logFile);

String^user = Path::GetFileNameWithoutExtension(logFile);

Успешно

успешно

broni^ zapisi = gcnew broni(Login);

zapisi->Show();

авторизация

Начало

Провер.

Миронова Дж.А

Разраб.

Погорельский И.О

СОЗДАНИЕ ПРОГРММНОГО СРЕДСТВА ДЛЯ ЗАПИСИ НА УЧАСТИЕ В КВЕСТАХ

А

Блок-схема алгоритма редактирования записей на квесты