МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
Высшая школа экономики и управления
Кафедра «Информационные технологии в экономике»

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой, д.т.н., с.н.с.
/ Б.М. Суховилов /
«_____»_____2021г.

Тема ВКР: Разработка игрового приложения игра «Дежавю»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.03.2021.11-1895-215.ВКР

Руководитель, доцент
_____/ О.И. Галичин /
«_____»_____2021 г.

Автор
студент группы зЭУ – 582
_____/ А.П Даньков /
«_____» _____2021 г.

Нормоконтролер, доцент
_____/Е.А. Конова/
«_____»_____2021 г.

Челябинск 2021

АННОТАЦИЯ

Даньков А.П. − Разработка игрового приложения «Дежавю» − Челябинск: ЮУрГУ, зЭУ-582, 50 с., 42 ил., 3 табл., библиогр. список - 26 наименований.

Выпускная квалификационная работы выполнена с целью разработки демонстрационной версии игры в жанре логические миниигры.

В ходе работы проведен анализ предметной области. Смоделированы объекты игры и запрограммированы взаимодействия пользователя с игрой.

В результате создана демонстрационная версия миниигры. Приложение предназначено для внедрения в ФКУ СИЗО – 3 ГУФСИН России по Челябинской области в качестве средств релаксации и развития памяти.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В современном мире всё чаще используют информационные технологии для развития людей, что находит отражение в современной научной литературе [7; 9; 14; 17; 18]. Развивающие и логические игры воспитывают в детях умение правильно мыслить, развивают память, помогают научить ребёнка логично поступать в тех или иных ситуациях. Данный способ развития определённых качеств ребёнка является очень эффективным, так как игры вызывают интерес у людей разных возрастов и очень просто приобщить к данному методу даже самого ленивого ребёнка [7]. Одной из таких игр можно назвать игру «Дежавю».

Актуальность работы заключается в том, что игра «Дежавю» может помочь человеку любого возраста не только потренировать память и психологически разгрузить, но способствует развитию личности. С каждым годом в жизни современных людей объем информации становится больше, и чтобы лучше её усваивать, необходимо развивать свою память.

Целью дипломной работы является разработка игрового программного обеспечения для развития памяти, при реализации данной цели необходимо выполнить ряд задач:

На данном этапе происходит ознакомление со средой разработки, историей её создания, функционалом и инструментами, которые она содержит.

Разработать программный продукт. Данный пункт подразумевает создание базы данных пользователей, проектирование интерфейса игры и написание программного кода для оптимизации её работы.

Разработать сопроводительную документацию. Данный этап заключается в написании общих сведений о программном средстве и инструкций для программиста и пользователя.

Рассчитать технико-экономические показатели и выяснить срок окупаемости игрового программного обеспечения.

Объектом исследования является развитие, тренировка памяти и

психологическая разгрузка с помощью информационных технологий.

Предметом изучения является компьютерная игра.

Практическая значимость выпускной квалификационной работы заключается в возможности использования созданной игры для развития и тренировки памяти людей любого возраста.

ГЛАВА 1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ

1.1 Постановка задачи

Игра «Дежавю», направленная на развитие памяти и внимательности, разработана не только для детей, но и для людей любой возрастной группы, так как имеет различные уровни сложности и подходит даже для взрослых, которые хотят потренировать и проверить способности своей памяти. Игра может найти применение не только в дошкольных и образовательных учреждениях, но может стать хорошим помощником и для любого человека, у которого возникнет желание потренировать память, посоревноваться с другими игроками в своих достижениях и просто приятно провести время.

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением заказчиком совокупности организационно - технических мероприятий, перечень которых приведен ниже:

Организовать бесперебойное питание технических средств.

Использовать лицензионное программное обеспечение.

Регулярно выполнять рекомендации Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;

Испытывать программные средства на наличие компьютерных вирусов.

Со стороны разработчика:

Автоматическое создание резервных копий.

Организация автоматического обновления программы.

Организация автоматического восстановления системы.

Для корректной работы программы требуется один человек – программист для технической поддержки создаваемого приложения [12].

Требования к составу и параметрам технических средств:

процессор с тактовой частотой 2.0Hz, не менее;

оперативную память объемом, 1Гигабайт, не менее;

свободное дисковое пространство не менее 1гб.

Для реализации программного продукта использоуем язык программирования C# (си–шарп). Выполнение технического задания предполагает проектирование программного продукта.

1.2 Проектирование программного продукта с использованием UML

Для понимания логики приложения удобно представить ее в виде диаграмм.

Стартовое описание удобно начинать с диаграммы прецедентов, которая показывает варианты использования для каждого участника. Диаграмма, отражающая отношения между акторами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне. Диаграмма прецедентов представлена (рисунок 1).

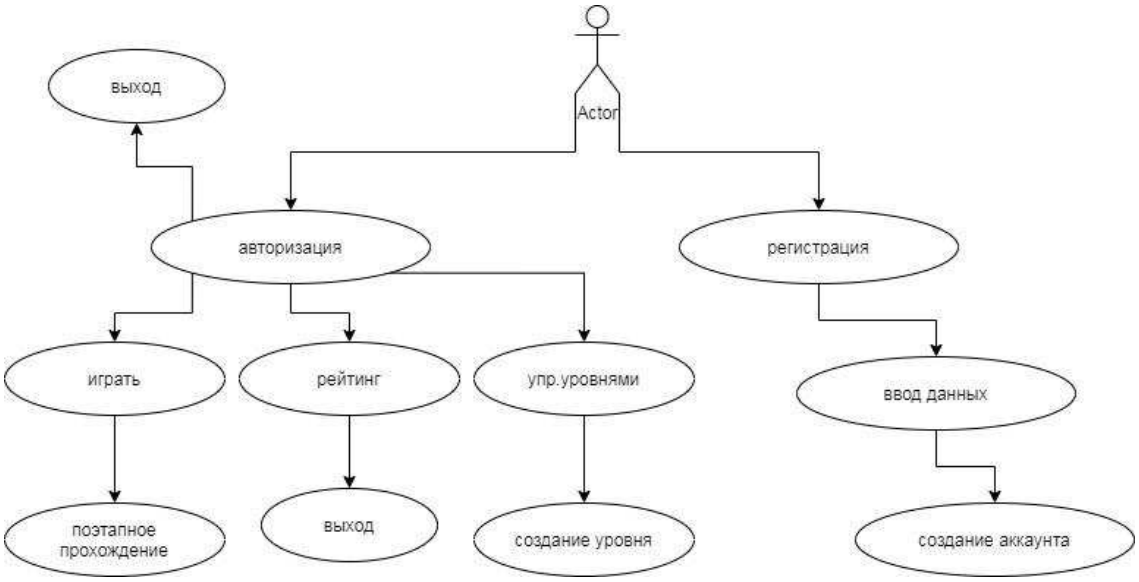


Рисунок 1 – Диаграмма прецедентов

Построена диаграмма последовательностей действий различных объектов. Диаграммы последовательностей используются для уточнения диаграмм прецедентов, более детального описания логики сценариев использования. Это удобное средство документирования проекта с точки зрения сценариев

использования.

Диаграммы последовательностей обычно содержат объекты, которые взаимодействуют в рамках сценария, сообщения, которыми они обмениваются, и возвращаемые результаты, связанные с сообщениями. Впрочем, часто возвращаемые результаты обозначают лишь в том случае, если это не очевидно из контекста. Диаграмма последовательности представлена (рисунок 2).

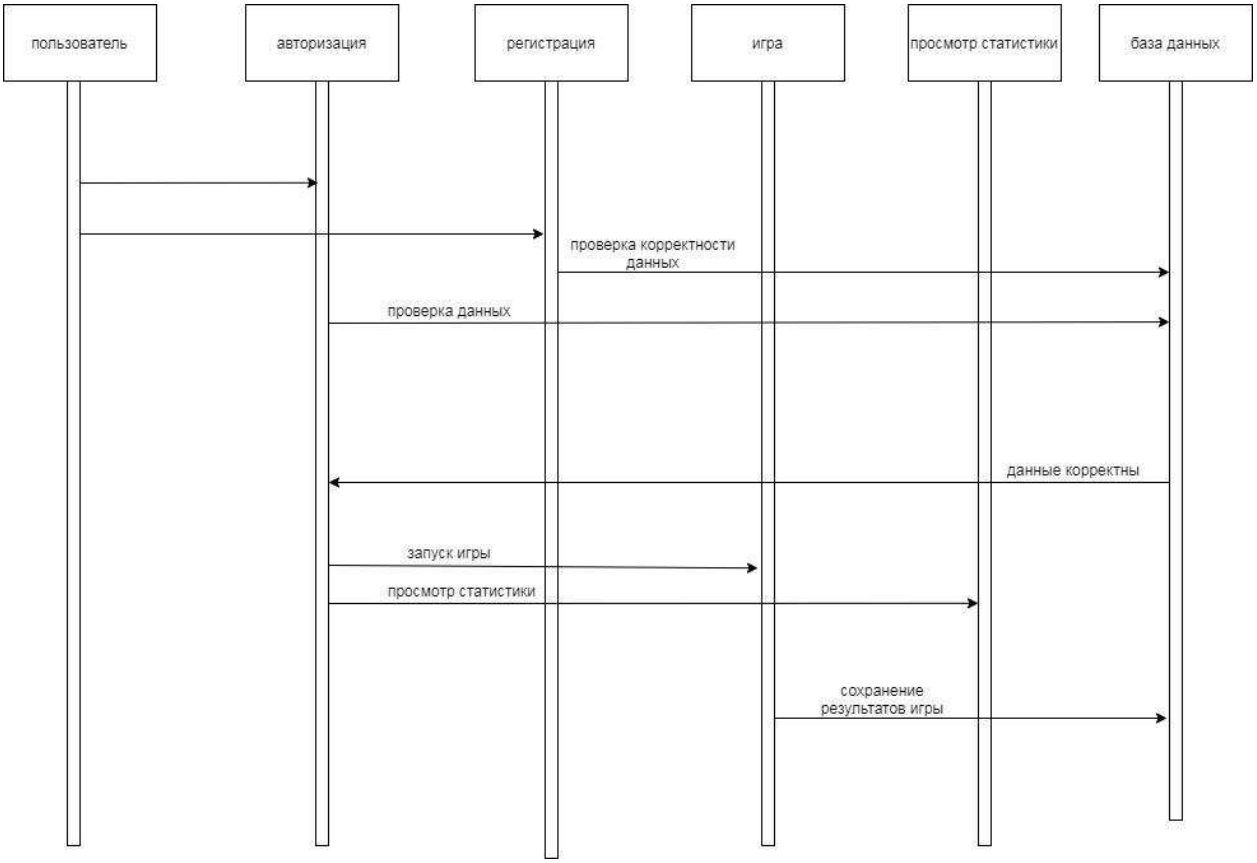


Рисунок 2 – Диаграмма последовательности.

Поскольку игра представляет собой многопользовательское приложение с отдельным сервером в качестве инструмента синхронизации, так же важно разработать диаграмму развертывания (рисунок 3).

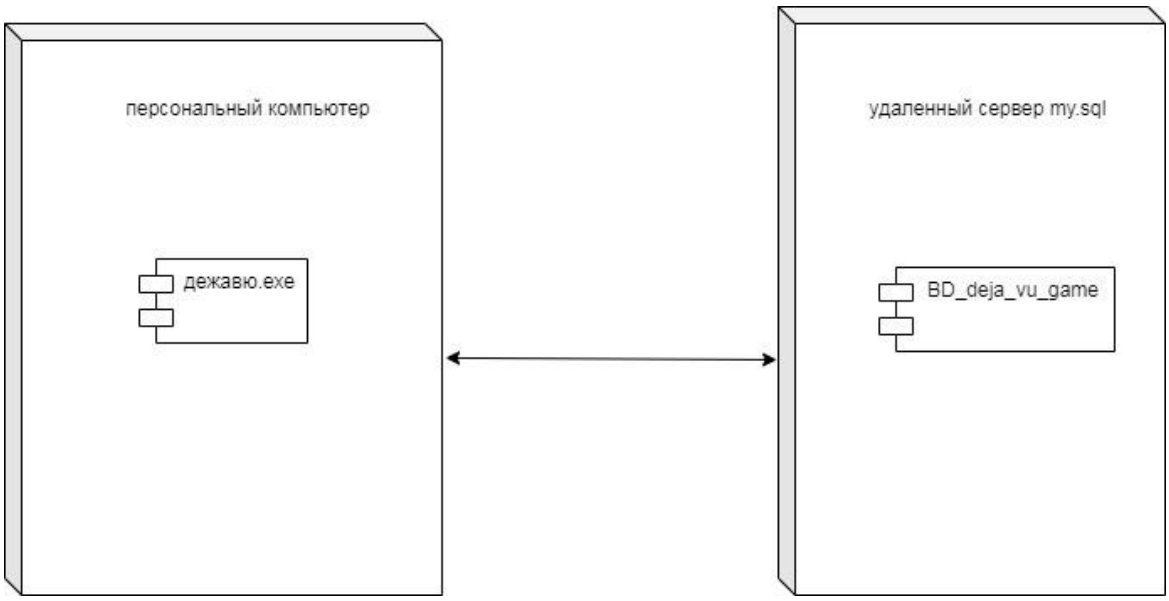


Рисунок 3 – Диаграмма развертывания.

Диаграмма компонентов показывает взаимосвязи между модулями (логическими или физическими), из которых состоит моделируемая система. (рисунок 4).

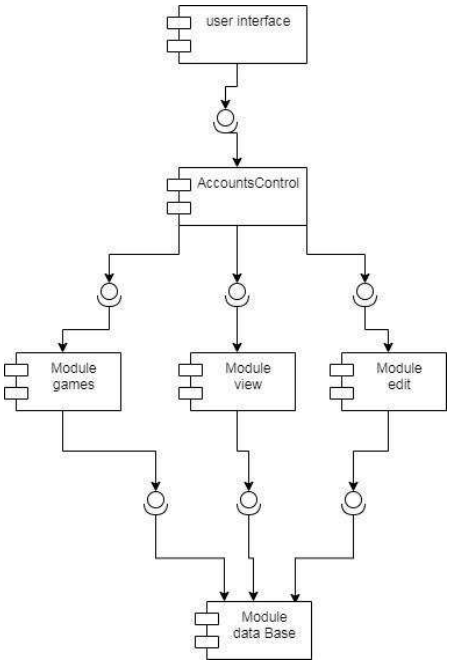


Рисунок 4 – Диаграммы компонентов.

Основной тип сущностей на диаграмме компонентов - это сами компоненты, а также интерфейсы, посредством которых указывается взаимосвязь между компонентами.

Вывод по первому разделу

В данном разделе проведен анализ предметной области, сформулировано обобщенное техническое задание, спроектированы UML диаграммы, компонентов, развертывания, последовательности, прецедентов.

ГЛАВА 2 ИНСТРУМЕНТАЛЬНОЕ СРЕДСТВО РАЗРАБОТКИ ПРОГРАММЫ

2.1 Основное функциональное назначение среды разработки

Интегрированная среда разработки Visual Studio включает в себя много функций и достоинств, которые представлены ниже.

Встроенный Web - сервер. Для обслуживания Web - приложения ASP.NET необходим Web - сервер, который будет ожидать Web - запросы и обрабатывать соответствующие страницы. Наличие в Visual Studio интегрированного Web - сервера позволяет запускать Web - сайт прямо из среды проектирования, а также повышает безопасность, исключая вероятность получения доступа к тестовому Web - сайту с какого - нибудь внешнего компьютера, поскольку тестовый сервер может принимать соединения только с локального компьютера.

Поддержка множества языков при разработке. Visual Studio позволяет писать код на своем языке или любых других предпочитаемых языках, используя все время один и тот же интерфейс (IDE). Более того, Visual Studio также еще позволяет создавать Web - страницы на разных языках, но помещать их все в одно и то же Web - приложение. Единственным ограничением является то, что в каждой Web - странице можно использовать только какой–то один язык (очевидно, что в противном случае проблем при компиляции было бы просто не избежать).

Для создания большинства приложений требуется Web - страницы ASP.NET тому не исключение. Например, добавление Web - элемента управления, присоединение обработчиков событий и корректировка форматирования требует установки в разметке страницы ряда деталей. В Visual Studio такие детали устанавливаются автоматически.

В качестве недостатка можно отметить невозможность отладчика (Microsoft Visual Studio Debugger) отслеживать в коде режима ядра. Отладка в Windows в режиме ядра в общем случае выполняется при использовании WinDbg,

KD или SoftICE.

2.2 Возможности инструментального средства

Окно панели элементов отображает элементы управления, которые вы можете добавлять в проекты Visual Studio. Чтобы открыть панель элементов, в меню «Вид» нужно выбрать пункт Панель элементов (рисунок 5).

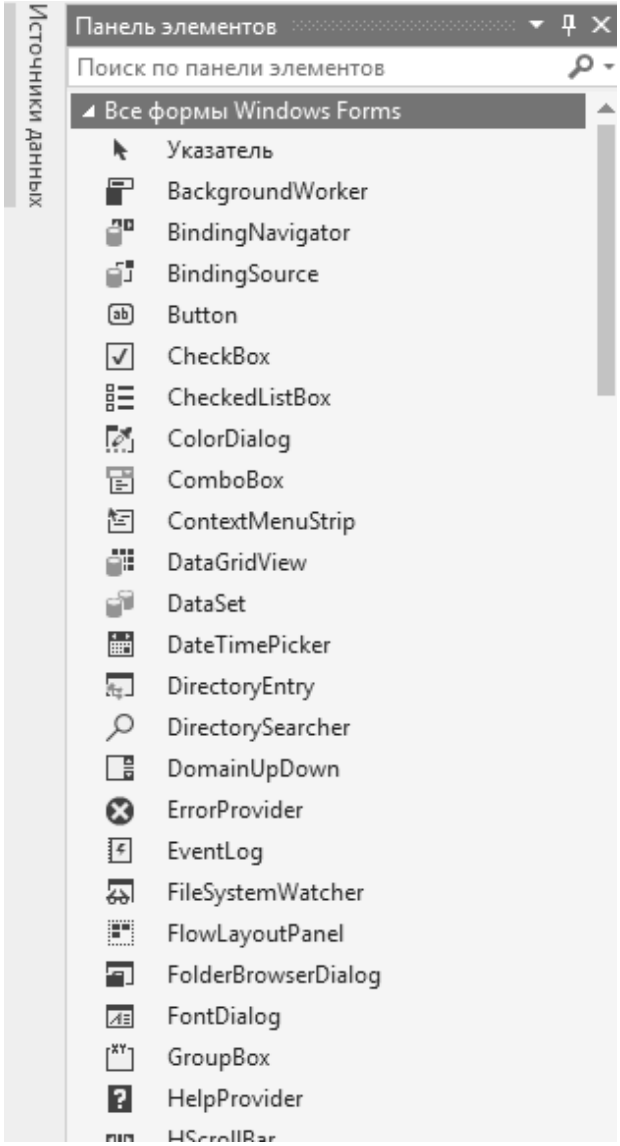


Рисунок 5 – Панель элементов

Можно перетаскивать различные элементы управления на поверхность используемого конструктора, а также изменять размер и положение элементов управления.

Панель элементов отображается вместе с представлениями конструктора,

например представлением XAML - файла. На панели элементов отображаются только те элементы управления, которые можно использовать в текущем конструкторе. Можно выполнить поиск в пределах панели элементов, чтобы отфильтровать отображаемые элементы.

По умолчанию панель элементов свернута в левой части Visual Studio. Чтобы отобразить ее, необходимо навести на нее курсор. Можно закрепить панель элементов, щелкнув на панели инструментов значок «Закрепить», чтобы она оставалась открытой. Также можно открепить окно панели элементов и перетащить его в любое место на экране. Чтобы закрепить, открепить или скрыть панель элементов, нужно щелкнуть ее правой кнопкой и выбрать нужное действие [15].

Можно изменить порядок элементов на вкладке панели элементов или добавить собственные вкладки, используя следующие команды в контекстном меню:

Переименовать элемент. Переименование выбранного элемента.

Показать все. Отображение всех возможных элементов управления (не только тех, которые можно использовать в текущем конструкторе).

Представление списка. Отображение элементов управления в вертикальном списке. Если этот флажок не установлен, элементы управления размещаются горизонтально.

Выбрать элементы. Открывает диалоговое окно Выбор элементов панели элементов, в котором можно указать элементы, отображаемые на панели элементов. Вы можете показать или скрыть элемент, установив или сняв его флажок.

Сортировать элементы по алфавиту. Сортировка элементов по имени.

Сброс панели. Восстановление параметров и элементов по умолчанию для панели элементов.

Добавить вкладку. Добавление новой вкладки на панель элементов.

Вверх. Перемещение выбранного элемента вверх.

Вниз. Перемещение выбранного элемента вниз.

Основные элементы программы:

Button. Представляет стандартную кнопку, которую пользователь может нажать для выполнения действий.

CheckBox. Указывает, включено или выключено условие.

ComboBox. Отображает данные в раскрывающемся поле со списком.

DataGrid. Отображает табличные данные из набора данных и позволяет вносить изменения в источник данных.

DataGridView. Предоставляет гибкую, расширяемую систему для отображения и редактирования табличных данных.

DateTimePicker. Позволяет пользователю выбрать один элемент из списка дат или времени.

Label. Отображает текст, который не может быть изменен пользователем.

RadioButton. Представляет набор из двух или более взаимоисключающих вариантов выбора для пользователя.

TableLayoutPanel. Представляет панель, в которой содержимое динамически отображается в сетке, состоящей из строк и столбцов.

TextBox. Позволяет пользователю вводить изменяемый текст из нескольких строк.

Timer. Вызывает событие через определенные интервалы [6, с. 194].

В данном разделе были описаны возможности инструментальных средств. Они являются весьма важными составляющими практически любого программного обеспечения.

Выводы по второму разделу

Во втором разделе описана программная структура среды разработки функционал и возможности инструментального средства.

ГЛАВА 3 РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА

3.1 Разработка алгоритма решения задачи

Приложение разработано с помощью технологии Windows Forms на языке C# (си–шарп).

Проблема увеличения объема информации в современном обществе и составлено представление о том, чем данное программное средство будет полезно в данной ситуации.

Следующим этапом создания базы данных и проектирование внешнего вида главного меню программы, создание каркаса для будущей игры, создан удобный и понятный интерфейс.

После создания интерфейса написан код программы, проведено тестирование программы и устранены все ошибки. Проанализирована актуальность созданного программного средства.

3.2 Описание разработки программного продукта

Создания Базы Данных (БД) в PhpMyAdmin:

Для создания БД, необходимо открыть в браузере страницу PhpMyAdmin, в поле «Пользователь» нужно ввести «root» и нажать «Ок» (рисунок 6).

Для создания БД необходимо перейти во вкладку «Databases». В открывшемся окне, в поле под надписью: «Create database» указать имя БД и нажать кнопку «Create» (рисунок 7).

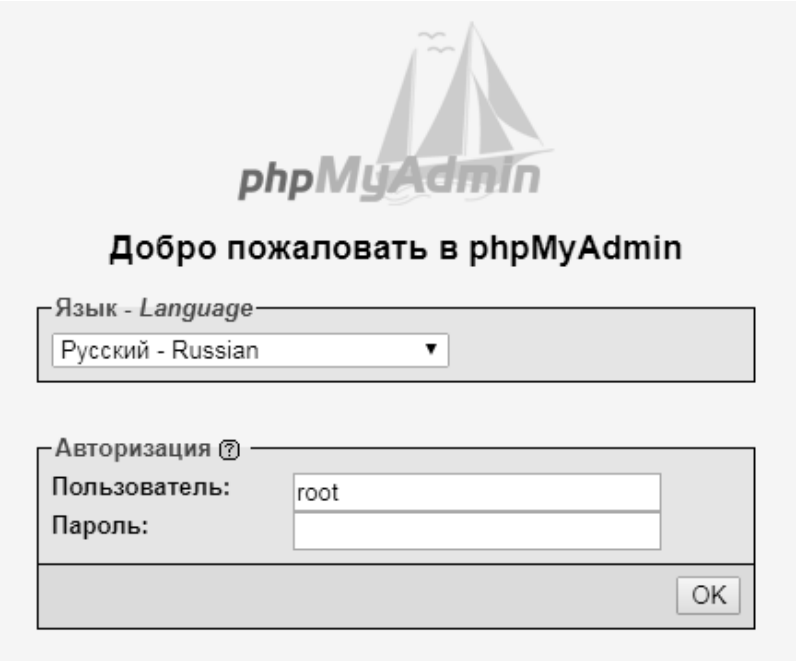


Рисунок 6 – Окно входа в PhpMyAdmin

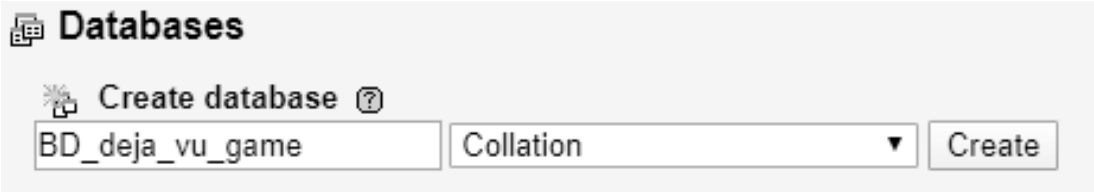


Рисунок 7 – Создание БД

Для создания таблицы в БД необходимо перейти в неё в левом меню phpMyAdmin. В открывшемся окне в «Create table», в поле «Name» указать имя (Рисунок 8), количество столбцов и нажать кнопку «Go».

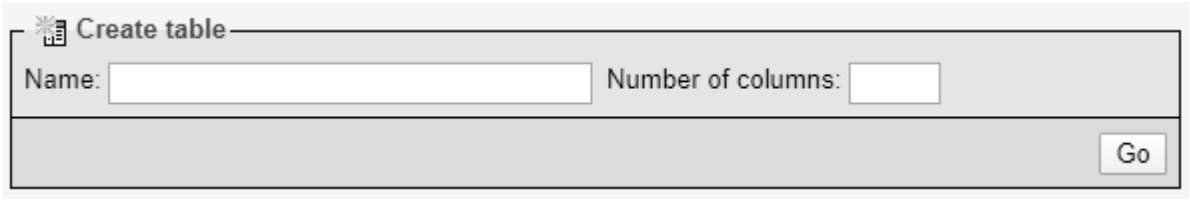


Рисунок 8 – Создание таблицы

В открывшемся окне следует указать наименование столбца, в поле «Name», его тип, поле «Type», и количество символов, поле «Length/Values». После создания структуры необходимо нажать кнопку «Save» (рисунок 9).

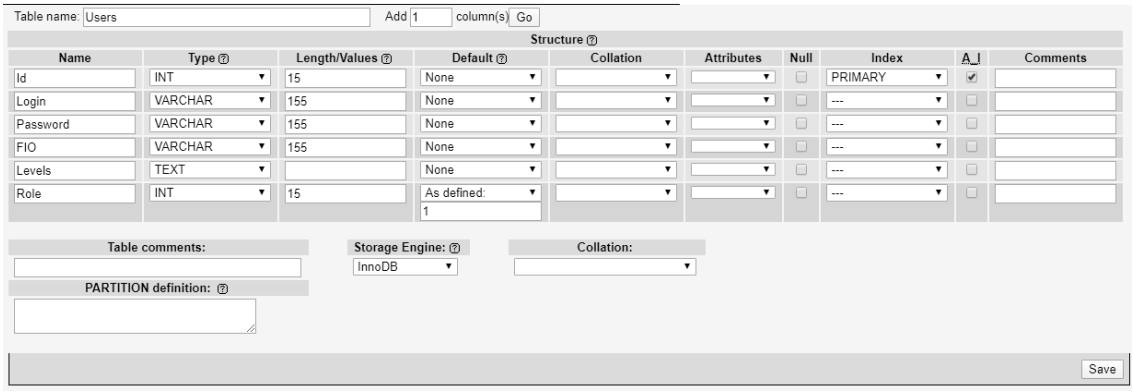


Рисунок 9 – Создание структуры, создаваемой таблицы

Последующие таблицы создаются таким же образом, после их создания БД готова к работе и можно приступать в разработке программного обеспечения.

Создание пользовательского интерфейса в Visual Studio. Первым делом следует создать проект: «Файл» – «Создать» – «Проект…». В отрывшемся окне выбрать «Visual C#» – «Классическое приложение Windows» – «Приложение Windows Forms (.NET Framework)» (рисунок 10).

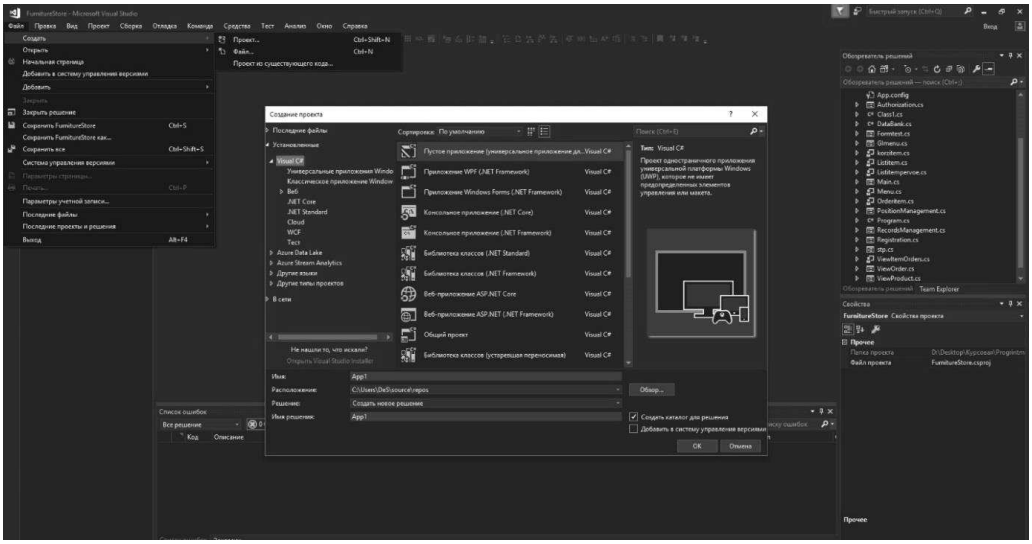


Рисунок 10 – Главное меню Visual Studio

Для начала необходимо создать интерфейс. После создания проекта первое что встречает – это пустая форма. На этой форме размещаются такие элементы как: кнопки, текст, таблицы картинки и многое другое. Все эти элементы расположены в левой части программы «Toolbox» (рисунок 11), и для дальнейшей работы с ними достаточно просто перенести их левой кнопкой мыши на форму.

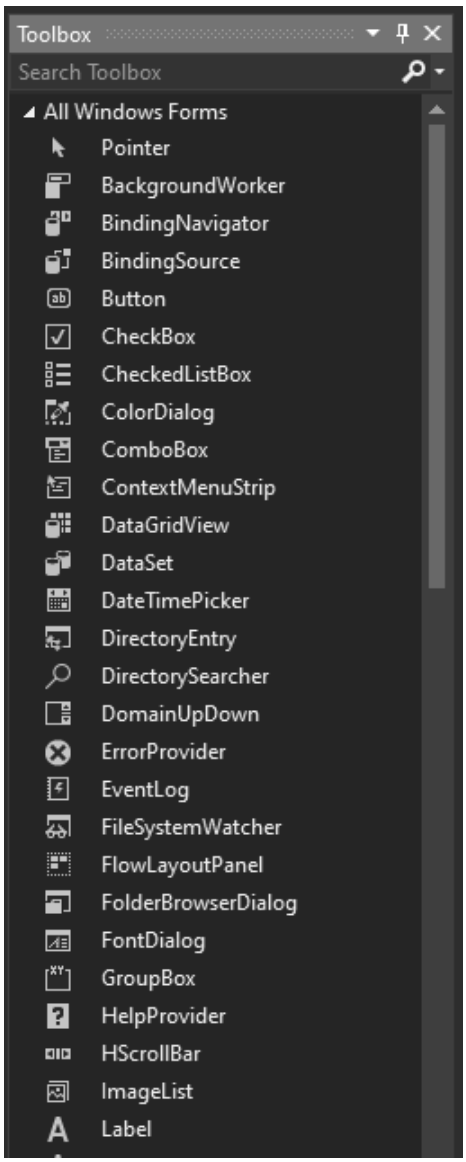


Рисунок 11 – Toolbox

Для изменения размера формы и элементов необходимо выделить объект и потянуть за края, или же в правой нижней части программы, в окне свойств элемента «Properties» (рисунок 12). В данном окне помимо размеров изменяются и другие свойства элемента, это могут быть: цвет, видимость, шрифт, содержание, отступы и так далее.

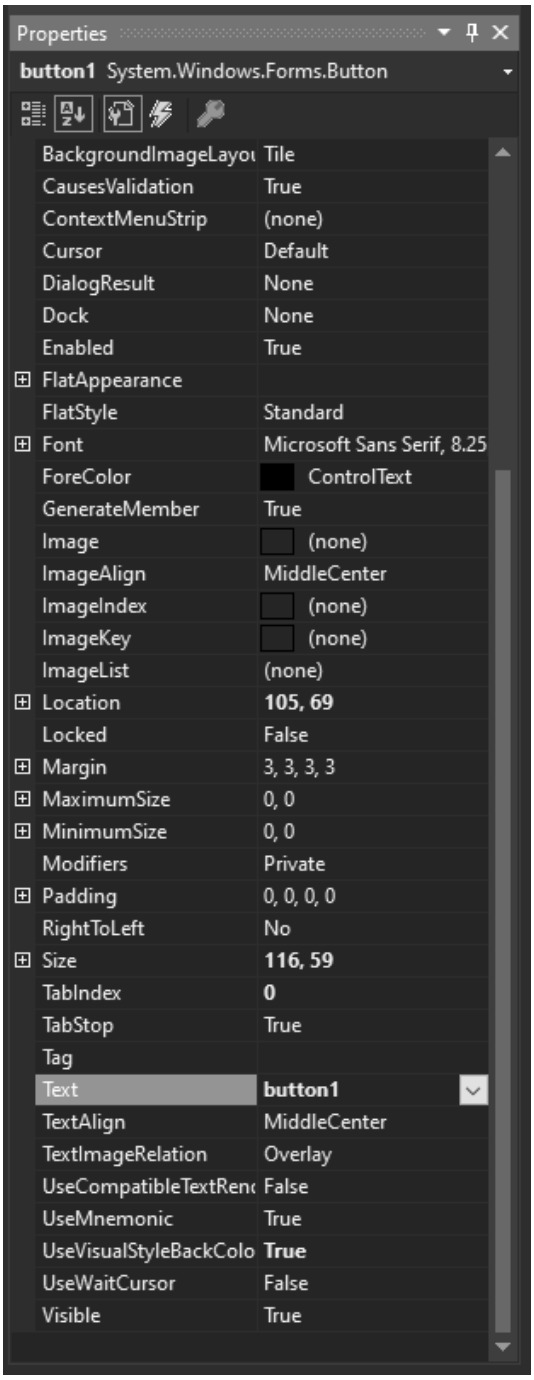


Рисунок 12 – Окно «Properties», вкладка «Properties»

Таким образом создаются визуальные составляющие windows forms. В данной программе необходимо создать следующие окна, окно авторизации в программу (рисунки 13 – Дизайн окна «Авторизация»).

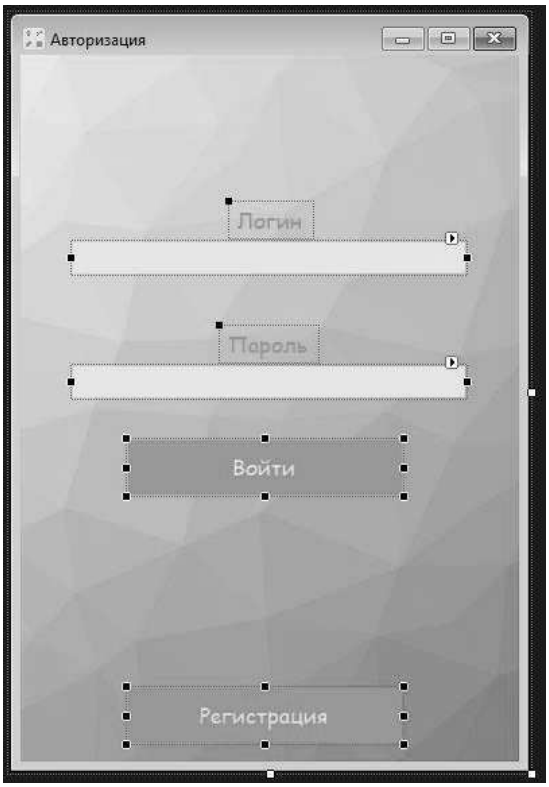


Рисунок 13 – Дизайн окна «Авторизация»

После авторизации мы попадаем в главное меню (Рисунок 14 – Дизайн окна «Главное меню»).

Рисунок 14 – Дизайн окна «Главное меню»

Выбрав в меню функцию играть, мы оказываемся в (рисунок 15 – Дизайн окна уровней).

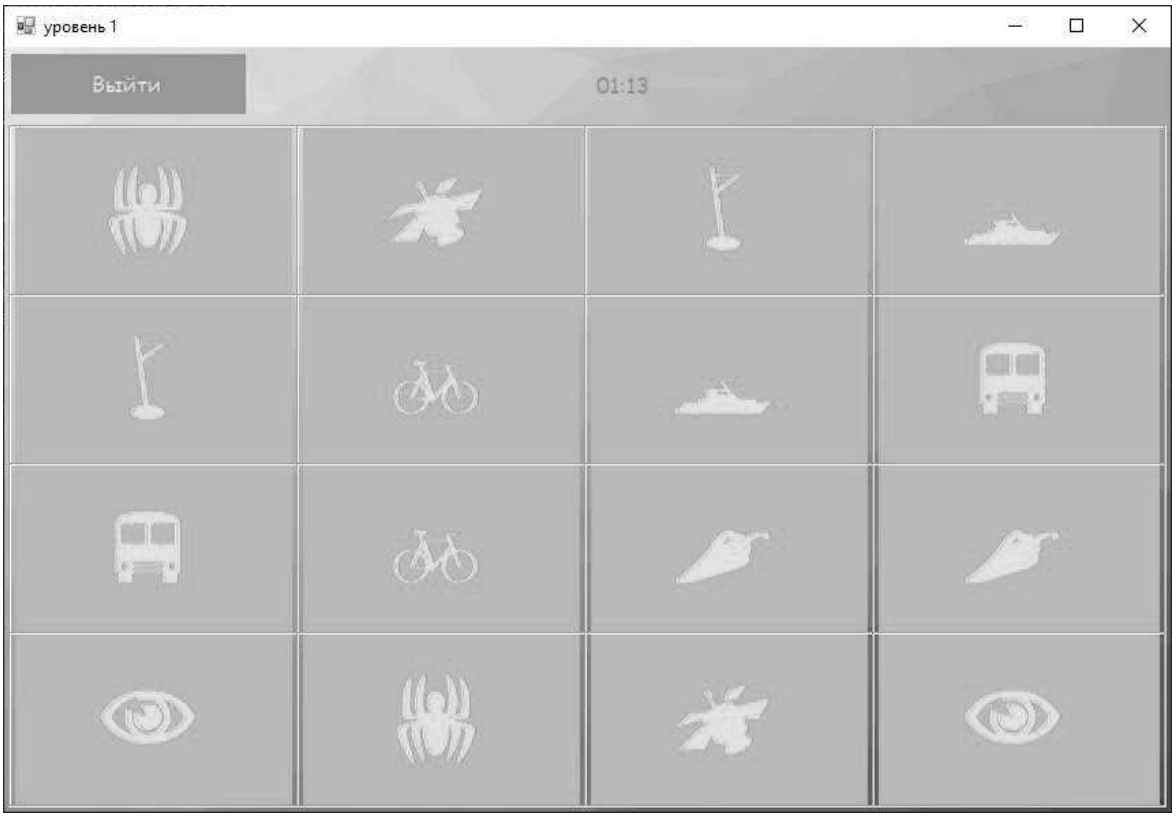


Рисунок 15 – Дизайн окна уровней

Выбирая меню управления уровнями, попадаем в настройку создания своего уровня(рисунок 16 – Дизайн окна «Управления уровнями»)

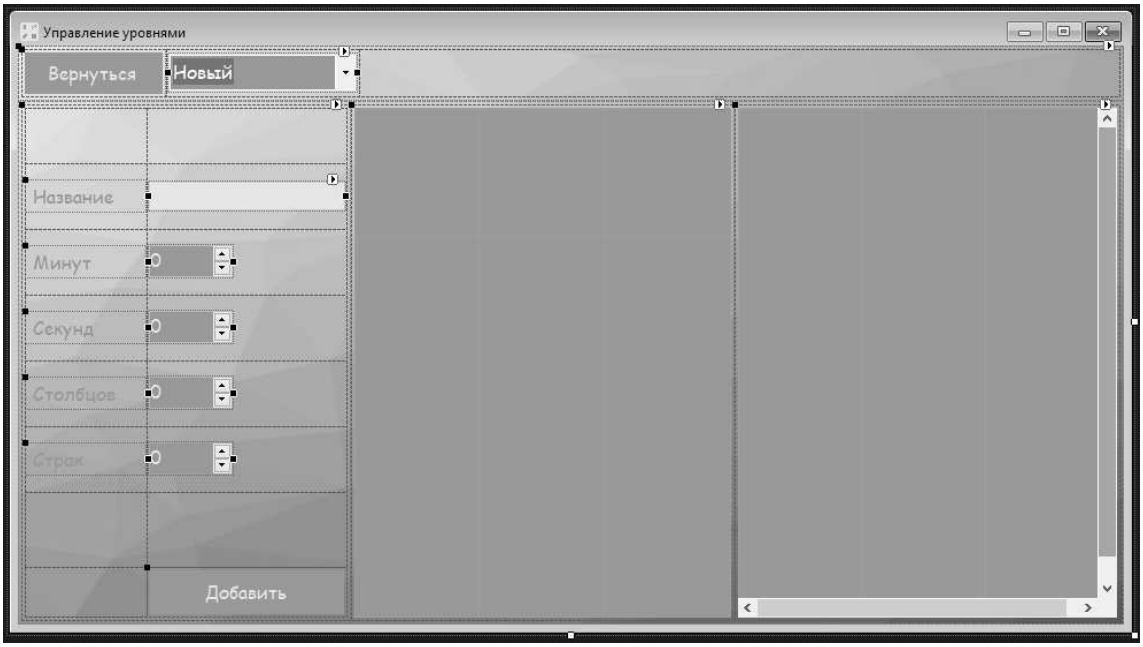


Рисунок 16 – Дизайн окна «Управления уровнями»

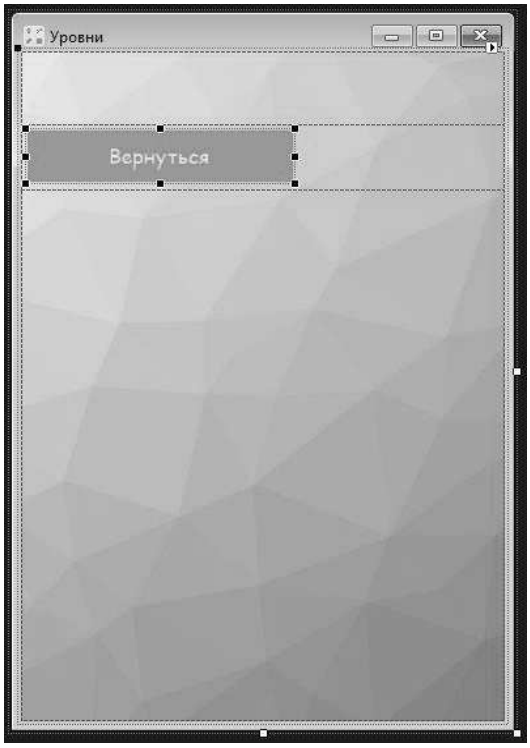


Рисунок 17 – Дизайн окна «Уровни»

Кликнув по меню рейтинг, мы оказываемся в таблице рейтингов всех пользователей (рисунок 18 – Дизайн окна «Рейтинг»)

Рисунок 18 – Дизайн окна «Рейтинг»

При входе в игру, не имея своего аккаунта, нам предоставляется возможность зарегистрироваться и авторизоваться (рисунок 19 – Дизайн окна «Регистрация»)

Рисунок 19 – Дизайн окна «Регистрация»

После создания интерфейса можно приступить к его программированию. Для создания обработчиков события, например, события клика по кнопке, необходимо выбрать элемент, в окне «Properties» выбрать вкладку «Events» (рисунок 20) и дважды нажать по необходимому действию, в данном случае «Click».

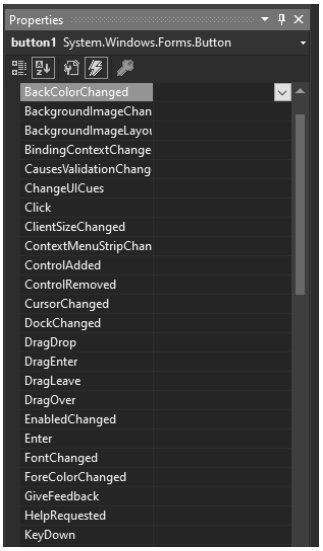


Рисунок 20 – Окно «Properties», вкладка «Events»

Visual Studio автоматически создаст обработчик события, выглядеть он должен следующим образом (рисунок 21).

```
private void button1_Click(object sender, EventArgs e)
{

}
```

Рисунок 21 – Пример обработчика события

Переход между формами осуществляется при помощи кода (рисунок 22 – код перехода на форму «Регистрация»), на нем представлен переход на форму регистрации.

```
private void button_reg_Click(object sender, EventArgs e)
{
    this.Hide();
    Reg f = new Reg();
    f.ShowDialog();
    this.Close();
}
```

Рисунок 22 – Код перехода на форму «Регистрация»

При нажатии на кнопку должно проверяться существование пользователя в системе по введенным логину и паролю, если пользователя нет, то пользователю программы должно выдавать окно с текстом о неправильно введенном логине и пароле. Если же такой пользователь есть, то в программе запоминаются его Id, ФИО, Login и роль посредством запроса в БД. Таким образом код авторизации выглядит следующим образом (рисунок 23 – код авторизации).

```
private void button_signin_Click(object sender, EventArgs e)
{
    if (Convert.ToInt16(MySQLQueryes.MySQLQueryRead("SELECT COUNT(*) FROM `Users` WHERE `Login` = '" +
        textBox_login.Text + "' AND `Password` = '" + textBox_pass.Text + "'")) != 0)
    {
        DataBank.IdAuthUser = Convert.ToInt16(MySQLQueryes.MySQLQueryRead("SELECT `Id` FROM `Users` WHERE `Login` = '" +
            textBox_login.Text + "' AND `Password` = '" + textBox_pass.Text + "'"));
        DataBank.RoleAuthUser = Convert.ToInt16(MySQLQueryes.MySQLQueryRead("SELECT `Role` FROM `Users` WHERE `Login` = '" +
            textBox_login.Text + "' AND `Password` = '" + textBox_pass.Text + "'"));
        DataBank.FIOAuthUser = MySQLQueryes.MySQLQueryRead("SELECT `FIO` FROM `Users` WHERE `Login` = '" + textBox_login.Text +
            "' AND `Password` = '" + textBox_pass.Text + "'");
        DataBank.LoginAuthUser = MySQLQueryes.MySQLQueryRead("SELECT `Login` FROM `Users` WHERE `Login` = '" + textBox_login.Text +
            "' AND `Password` = '" + textBox_pass.Text + "'");
        this.Hide();
        Glavnaya f = new Glavnaya();
        f.ShowDialog();
        this.Close();
    }
    else
    {
        MessageBox.Show("Логин или пароль введен неверно!");
    }
}
```

Рисунок 23 – Код авторизации

Все запросы в БД осуществляются при помощи заранее написанных функций, выведенных в отдельный класс. Так запрос в БД без ожидания ответа выглядит как представлено на рисунке (рисунок 24 – запрос в БД без ожидания ответа).

```
public static void MySQLQuery(string command)
{
    try
    {
        DataBank.Conn.Open();
        MySqlCommand cmd = new MySqlCommand(command, DataBank.Conn);
        cmd.ExecuteScalar();
        DataBank.Conn.Close();
    }
    catch
    { }
}
```

Рисунок 24 – Запрос в БД без ожидания ответа

Данный запрос уже считывает, но возвращает одно значение (рисунок 25 – запрос в БД с возвращением одного значения).

```
public static string MySQLQueryRead(string command)
{
    string ret = "";
    try
    {
        DataBank.Conn.Open();
        MySqlCommand cmd = new MySqlCommand(command, DataBank.Conn);
        MySqlDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
            ret = reader[0].ToString();
        DataBank.Conn.Close();
    }
    catch
    { }
    return (ret);
}
```

Рисунок 25 – Запрос в БД с возвращением одного значения

Запрос возвращает таблицу, которую можно разобрать по названиям столбцов и индексированию строк (рисунок 26 – запрос в БД с возвращением одного значения).

```
public static DataTable MySQLQueryReadDT(MySqlCommand command)
{
    DataBank.Conn.Open();
    command.Connection = DataBank.Conn;
    DbDataReader reader = command.ExecuteReader();
    DataTable result = new DataTable();
    result.Load(reader);
    DataBank.Conn.Close();
    return result;
}
```

Рисунок 26 – Запрос в БД с возвращением одного значения

При нажатии на кнопку «Зарегистрироваться» начала необходимо проверить наполненность полей и отсутствие пользователя в системе с таким логином, после чего осуществить запрос в БД с добавлением нового пользователя. Код регистрации выглядит следующим образом (рисунок 27 – код регистрации нового пользователя).

```
private void button_reg_Click(object sender, EventArgs e)
{
    if (textBox_login.Text != "" && textBox_login.Text != " ")
    {
        MessageBox.Show("Логин введен некорректно");
        return;
    }
        if (textBox_pass.Text != "" && textBox_pass.Text != " ")
        {
            MessageBox.Show("Пароль введен некорректно");
            return;
        }
            if (Convert.ToInt16(MySQLQueryes.MySQLQueryRead("SELECT COUNT(*) FROM `Users` WHERE `Login` = '" +
                textBox_login.Text + "'")) == 0)
            {
                MessageBox.Show("Данны логин уже занят!");
                return;
            }
                if (textBox_pass.Text == textBox_sucPass.Text)
                {
                    MessageBox.Show("Пароли не совпадают!");
                    return;
                }
                    if (textBox_FIO.Text != "" && textBox_FIO.Text != " ")
                    {
                        MessageBox.Show("ФИО введено некорректно!");
                        return;
                    }
                        MySQLQueryes.MySQLQuery("INSERT INTO `Users`(`Login`, `Password`, `FIO`, `Role`) VALUES ('" +
                            textBox_login.Text + "', '" + textBox_pass.Text + "', '" + textBox_FIO.Text + "', '0');");
                        MessageBox.Show("Регистрация прошла успешно!");
                        this.Hide();
                        Auth f = new Auth();
                        f.ShowDialog();
                        this.Close();
}
```

Рисунок 27 – Код регистрации нового пользователя

Кнопки запуска уровней должны загружаться динамически, на основе данных из БД. Необходимо в первую очередь загрузить все уровни из БД

(рисунок 28 – загрузка уровней из БД) и загрузить время уровней пользователя (рисунок 29 – загрузка времени уровней пользователя).

```
public static void loadLevels()
{
    dataLevels.Clear();
    DataTable dt = MySQLQueryes.MySQLQueryReadDT("SELECT * FROM `Levels` WHERE 1");
    foreach(DataRow row in dt.Rows)
    {
        dataLevels.Add((int)row[0], new levels { name = row[1].ToString(), min = (int)row[2],
            sec = (int)row[3], icons = JsonConvert.DeserializeObject<List<String>>(row[4].ToString()),
            columnsCount = (int)row[5], rowsCount = (int)row[6]});
    }
}
```

Рисунок 28 – Загрузка уровней из БД

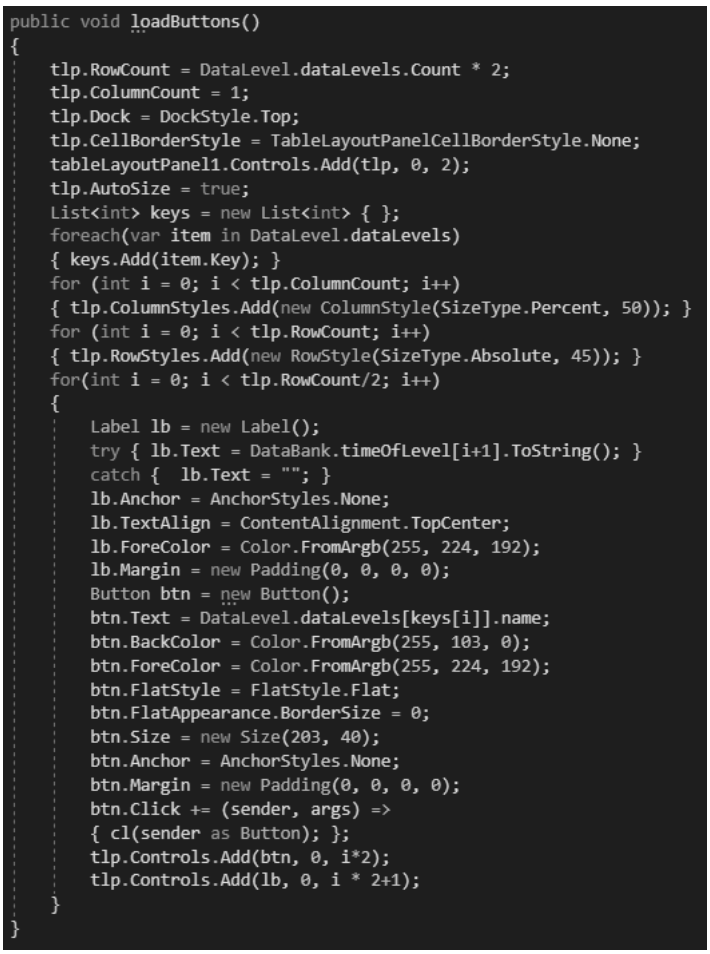
```
public static void load_timeOfLevel()
{
    string json = MySQLQueryes.MySQLQueryRead("SELECT `Levels` FROM `Users` WHERE `Id` = '"+IdAuthUser+"';");
    Dictionary<int, string> timeOfLevelBD = JsonConvert.DeserializeObject<Dictionary<int, string>>(json);
    timeOfLevel = timeOfLevelBD;
}
```

Рисунок 29 – Загрузка времени уровней пользователя

На основе списка всех уровней необходимо создать кнопку перехода на форму, каждой из которых присваивается следующая функция клика (рисунок 30 – обработчики события клика по кнопке на окне «Уровни»), задается цвет, размер и так далее. Также помимо кнопки необходимо создать текст с текущим рекордом уровня. Код для появления данного текста представлен на рисунке (рисунок 31 – создание кнопок и надписей на окне «Уровни»).

```
private void cl(Button bt)
{
    DataLevel.selectedLevel = DataLevel.getKeyDataLevels(bt.Text);
    this.Hide();
    Level_load f = new Level_load();
    f.ShowDialog();
    this.Close();
}
```

Рисунок 30 – Обработчики события клика по кнопке на окне «Уровни»



```
public void loadButtons()
{
    tlp.RowCount = DataLevel.dataLevels.Count * 2;
    tlp.ColumnCount = 1;
    tlp.Dock = DockStyle.Top;
    tlp.CellBorderStyle = TableLayoutPanelCellBorderStyle.None;
    tableLayoutPanel1.Controls.Add(tlp, 0, 2);
    tlp.AutoSize = true;
    List<int> keys = new List<int> { };
    foreach(var item in DataLevel.dataLevels)
    { keys.Add(item.Key); }
    for (int i = 0; i < tlp.ColumnCount; i++)
    { tlp.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 50)); }
    for (int i = 0; i < tlp.RowCount; i++)
    { tlp.RowStyles.Add(new RowStyle(SizeType.Absolute, 45)); }
    for(int i = 0; i < tlp.RowCount/2; i++)
    {
        Label lb = new Label();
        try { lb.Text = DataBank.timeOfLevel[i+1].ToString(); }
        catch {  lb.Text = ""; }
        lb.Anchor = AnchorStyles.None;
        lb.TextAlign = ContentAlignment.TopCenter;
        lb.ForeColor = Color.FromArgb(255, 224, 192);
        lb.Margin = new Padding(0, 0, 0, 0);
        Button btn = new Button();
        btn.Text = DataLevel.dataLevels[keys[i]].name;
        btn.BackColor = Color.FromArgb(255, 103, 0);
        btn.ForeColor = Color.FromArgb(255, 224, 192);
        btn.FlatStyle = FlatStyle.Flat;
        btn.FlatAppearance.BorderSize = 0;
        btn.Size = new Size(203, 40);
        btn.Anchor = AnchorStyles.None;
        btn.Margin = new Padding(0, 0, 0, 0);
        btn.Click += (sender, args) =>
        { cl(sender as Button); };
        tlp.Controls.Add(btn, 0, i*2);
        tlp.Controls.Add(lb, 0, i * 2+1);
    }
}
```

Рисунок 31 – Создание кнопок и надписей на окне «Уровни»

После выбора кнопки уровня игрока переводит на новую форму, на которой и должны загрузиться символы на основе списка уровней из БД. В начале следует задать время таймера для уровня, количество строк и столбцов таблицы и разместить там элемент формы «Label» с обработчиком события (рисунки 32–33 код загрузки информации об уровне и генерации таблицы), в котором будет проверяться соответствие картинок и отмечены ли все картинки. Как только они будут отмечены, то будет сохранен результат, если он лучше предыдущего. Далее следует случайное заполнение таблицы символами из списка (рисунок 34 – код клика по картинке). По завершению загрузки уровня запустится таймер, который тоже имеет свой обработчик события (рисунок 35 – код клика по картинке). Код загрузки уровня выглядит следующим образом (рисунок 36 – создание кнопок и надписей на окне «Уровни»).

```
public Level_load()
{
    InitializeComponent();
    min = DataLevel.dataLevels[DataLevel.selectedLevel].min;
    sec = DataLevel.dataLevels[DataLevel.selectedLevel].sec;
    icons = DataLevel.dataLevels[DataLevel.selectedLevel].icons;
    this.Text = DataLevel.dataLevels[DataLevel.selectedLevel].name;
    tlp.ColumnCount = DataLevel.dataLevels[DataLevel.selectedLevel].columnsCount;
    tlp.RowCount = DataLevel.dataLevels[DataLevel.selectedLevel].rowsCount;
    tlp.Dock = DockStyle.Fill;
    tlp.CellBorderStyle = TableLayoutPanelCellBorderStyle.Inset;
    tableLayoutPanel1.Controls.Add(tlp, 0, 1);
    for(int i = 0; i < tlp.ColumnCount; i++)
    { tlp.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 100 / tlp.ColumnCount)); }
    for(int i = 0; i < tlp.RowCount; i++)
    { tlp.RowStyles.Add(new RowStyle(SizeType.Percent, 100 / tlp.RowCount)); }
    for (int i = 0; i < DataLevel.dataLevels[DataLevel.selectedLevel].rowsCount; i++)
    {
        for (int j = 0; j < DataLevel.dataLevels[DataLevel.selectedLevel].columnsCount; j++)
        {
            Label addLabel = new Label();
            addLabel.Font = new Font("Webdings", 62, FontStyle.Bold, GraphicsUnit.Point, 2, false);
            addLabel.Text = "c";
            addLabel.Dock = DockStyle.Fill;
            addLabel.TextAlign = ContentAlignment.MiddleCenter;
            addLabel.BackColor = Color.Orange;
            addLabel.ForeColor = Color.Orange;
            tlp.Controls.Add(addLabel, j, i);
        }
    }
    AssignIconsToSquares();
    timer.Tick += OnTimerEvent;
    timer.Start();
}
```

Рисунок 32 – Код загрузки информации об уровне и генерации таблицы

```
private void AssignIconsToSquares()
{
    Label label;
    int randomNumber;
    for (int i = 0; i < tlp.Controls.Count; i++)
    {
        if (tlp.Controls[i] is Label) label = (Label)tlp.Controls[i];
        else continue;
        randomNumber = random.Next(0, icons.Count);
        try
        { label.Text = icons[randomNumber];
        icons.RemoveAt(randomNumber); } catch { }
        label.Click += label_Click;
    }
}
```

Рисунок 33 – Код генерации картинок

```
private void label_Click(object sender, EventArgs e)
{
    if (firstClicked != null && secondClicked != null) return;
    Label clickedLabel = sender as Label;
    if (clickedLabel == null)  return;
    if (clickedLabel.ForeColor == Color.Black)  return;
    if (firstClicked == null)
    { firstClicked = clickedLabel;
        firstClicked.ForeColor = Color.Black;
        return; }
    secondClicked = clickedLabel;
    secondClicked.ForeColor = Color.Black;
    CheckForWinner();
    if (firstClicked.Text == secondClicked.Text)
    { firstClicked.ForeColor = Color.Red;
        secondClicked.ForeColor = Color.Red;
        firstClicked = null;
        secondClicked = null;
        SoundPlayer Simple = new SoundPlayer(@"C:\Windows\Media\tada.wav");
        Simple.Play(); }
    else
    { timer1.Tick += timer1_Tick;
        timer1.Start(); }
}
```

Рисунок 34 – Код клика по картинке

```
private void CheckForWinner()
{
    Label label;
    for (int i = 0; i < tlp.Controls.Count; i++)
    {
        label = tlp.Controls[i] as Label;
        if (label != null && label.ForeColor == label.BackColor) return;
    }
    if(DataBank.timeOfLevel != null)
    {
        if (DataBank.timeOfLevel.ContainsKey(DataLevel.selectedLevel))
        {
            var numbers = DataBank.timeOfLevel[DataLevel.selectedLevel];
            var secOnDT = Convert.ToInt16(numbers.Substring(numbers.IndexOf(':') + 1));
            var minOnDT = Convert.ToInt16(numbers.Substring(0, numbers.IndexOf(':')));
            if(min > minOnDT || (min == minOnDT && sec > secOnDT)) DataBank.timeOfLevel[DataLevel.selectedLevel] = min + ":" + sec;
        }
        else
        { DataBank.timeOfLevel.Add(DataLevel.selectedLevel, (min + ":" + sec)); }
    }
    else
    { DataBank.timeOfLevel = new Dictionary<int, string> { {DataLevel.selectedLevel, min+":"+sec } }; }
    DataBank.save_timeOfLevel();
    MessageBox.Show("Вы прошли уровень!");
    this.Hide();
    Glavnaya f = new Glavnaya();
    f.ShowDialog();
    this.Close();
}
```

Рисунок 35 – Код клика по картинке

```
private void OnTimerEvent(object sender, EventArgs e)
{
    if (min == 0 && sec == 0)
    {
        MessageBox.Show("Вы проиграли");
        return;
    }
    if (sec > 0)
        sec--;
    else
    {
        min--;
        sec = 59;
    }
    label_time.Text = ((min < 10) ? ("0" + min.ToString()) : min.ToString())
        + ":" + ((sec < 10) ? ("0" + sec.ToString()) : sec.ToString());
}
```

Рисунок 36 – Создание кнопок и надписей на окне «Уровни»

Для изменения уровней для начала следует загрузить уровни в элемент формы «Combobox» для их выбора и дальнейшего изменения (рисунок 37 – загрузка уровней в combobox). А также загрузить все возможные картинки для их добавления в уровни (рисунок 38 – загрузка всех возможных картинок).

```
foreach (var item in DataLevel.dataLevels)
{
    comboBox_levels.Items.Add(item.Value.name);
}
```

Рисунок 37 – Загрузка уровней в combobox

```
public void loadAllIcons()
{
    int column = 0;
    int row = 0;
    foreach (var item in valuess)
    {
        Label addLabel = new Label();
        addLabel.Font = new Font("Webdings", 62, FontStyle.Bold, GraphicsUnit.Point, 2, false);
        addLabel.Text = item;
        addLabel.Dock = DockStyle.Fill;
        addLabel.TextAlign = ContentAlignment.MiddleCenter;
        addLabel.BackColor = Color.Orange;
        addLabel.ForeColor = Color.Aqua;
        addLabel.Click += label_Click;
        tlp_allIcons.Controls.Add(addLabel, column, row);
        column++;
        if (column == 3)
        { tlp_allIcons.RowCount = tlp_allIcons.RowCount + 1;
            column = 0;
            row++; }
    }
    for (int i = 0; i < tlp_allIcons.RowCount; i++)
    { tlp_allIcons.RowStyles.Add(new RowStyle(SizeType.Absolute, 110));
        tlp_allIcons.RowStyles[i] = new RowStyle(SizeType.Absolute, 110); }
}
```

Рисунок 38 – Загрузка всех возможных картинок

Также необходимо выполнить загрузку картинок выбранного уровня, код представлен на рисунке (рисунок 39 – загрузка картинок выбранного уровня).

```
public void loadIconsSelectedLevel()
{
    tlp_selectedIcons.Controls.Clear();
    tlp_selectedIcons.RowCount = 1;
    column_iconSelectedLevel = 0;
    row_iconSelectedLevel = 0;
    foreach (var item in icons)
    {
        if (column_iconSelectedLevel == 3)
        { tlp_selectedIcons.RowCount = tlp_selectedIcons.RowCount + 1;
            column_iconSelectedLevel = 0;
            row_iconSelectedLevel++; }
        Label addLabel = new Label();
        addLabel.Font = new Font("Webdings", 62, FontStyle.Bold, GraphicsUnit.Point, 2, false);
        addLabel.Text = item;
        addLabel.Dock = DockStyle.Fill;
        addLabel.TextAlign = ContentAlignment.MiddleCenter;
        addLabel.BackColor = Color.Orange;
        addLabel.ForeColor = Color.Aqua;
        addLabel.Click += label_Click;
        tlp_selectedIcons.Controls.Add(addLabel, column_iconSelectedLevel, row_iconSelectedLevel);

        column_iconSelectedLevel++;
    }
    tlp_selectedIcons.RowCount++;
    for (int i = 0; i < tlp_selectedIcons.RowCount; i++)
    { tlp_selectedIcons.RowStyles.Add(new RowStyle(SizeType.Absolute, 110));
        tlp_selectedIcons.RowStyles[i] = new RowStyle(SizeType.Absolute, 110); }
}
```

Рисунок 39 – Загрузка картинок выбранного уровня

К каждой картинке следует привязать обработчик события, который будет определять: картинка относится к уровню или ко всем картинкам. В зависимости от того, к чему она относится, картинка будет удаляться или добавляться. Код выглядит следующим образом (рисунок 40 – обработчик события клика по картинке, в окне управления уровнями).

```
private void label_Click(object sender, EventArgs e)
{
    Label label = sender as Label;
    if(label.Parent.Name == "tlp_selectedIcons")
    {
        tlp_selectedIcons.Controls.Remove(label);
        icons.Remove(label.Text);
        loadIconsSelectedLevel();
        tlp_selectedIcons.RowCount++;
    }
    else
    {
        if (column_iconSelectedLevel == 3)
        { tlp_selectedIcons.RowCount = tlp_selectedIcons.RowCount + 1;
            column_iconSelectedLevel = 0;
            row_iconSelectedLevel++; }
        icons.Add(label.Text);
        Label addLabel = new Label();
        addLabel.Font = new Font("Webdings", 62, FontStyle.Bold, GraphicsUnit.Point, 2, false);
        addLabel.Text = label.Text;
        addLabel.Dock = DockStyle.Fill;
        addLabel.TextAlign = ContentAlignment.MiddleCenter;
        addLabel.BackColor = Color.Orange;
        addLabel.ForeColor = Color.Aqua;
        addLabel.Click += label_Click;
        tlp_selectedIcons.Controls.Add(addLabel, column_iconSelectedLevel, row_iconSelectedLevel);
        column_iconSelectedLevel++;
        for (int i = 0; i < tlp_selectedIcons.RowCount; i++)
        { tlp_selectedIcons.RowStyles.Add(new RowStyle(SizeType.Absolute, 110));
            tlp_selectedIcons.RowStyles[i] = new RowStyle(SizeType.Absolute, 110); }
        tlp_selectedIcons.AutoScrollMinSize = new Size(270, 50);
    }
}
```

Рисунок 40 – Обработчик события клика по картинке, в окне управления уровнями

В завершение необходимо создать обработчик события для кнопки, сохраняющей изменения и добавляющей новые уровни, код данной кнопки

представлен на рисунке (рисунок 41 – Обработчик события клика по кнопке сохранения добавления уровней).

```
private void button_save_Click(object sender, EventArgs e)
{
    if(button_save.Text == "Добавить")
    {
        if((icons.Count * 2) != (nud_columns.Value * nud_rows.Value))
        {
            MessageBox.Show("Кол-во элементов не совпадает с кол-вом ячеек таблицы");
            return;
        }
        DataLevel.dataLevels.Add(DataLevel.dataLevels.Keys.Max() + 1, new levels { name = textBox_name.Text,
            min = (int)nud_min.Value, sec = (int)nud_sec.Value, icons = this.icons,
            columnsCount = (int)nud_columns.Value, rowsCount = (int)nud_rows.Value });
        DataLevel.addLevel(DataLevel.getKeyDataLevels(textBox_name.Text));
        MessageBox.Show("Уровень успешно добавлен!");
        textBox_name.Text = "";
        nud_min.Value = 0;
        nud_sec.Value = 0;
        nud_columns.Value = 0;
        nud_rows.Value = 0;
        tlp_selectedIcons.Controls.Clear();
        tlp_selectedIcons.RowCount = 1;
        if (icons != null) icons.Clear();
    }
}
```

Рисунок 41 – Обработчик события клика по кнопке сохранения добавления уровней

3.3 Пример тестовой проверки программы

При запуске игры появляется главное меню игры с возможностью авторизации в игре или для регистрации нового игрока.

При нажатии на кнопку «Регистрация» открывается окно, которое позволяет пользователю зарегистрироваться в системе. После авторизации в системе пользователь видит главное меню, которое даёт возможность начать игру, посмотреть рейтинг, настроить уровни (для администратора) или покинуть игру.

Если выбрать любой уровень игры (к примеру, первый уровень), пользователя переносит на форму с игрой. Во время игры таймер отсчитывает время. Любую ячейку формы можно перевернуть, нажав на неё однократно левой кнопкой мыши. После открытия первой картинки, игрок открывает вторую, при их несовпадении, они обе переворачиваются обратно через одну секунду. Если картинки совпадают, то они остаются открытыми, и так происходит до конца игры, пока все ячейки не будут перевёрнуты (рисунок 42).

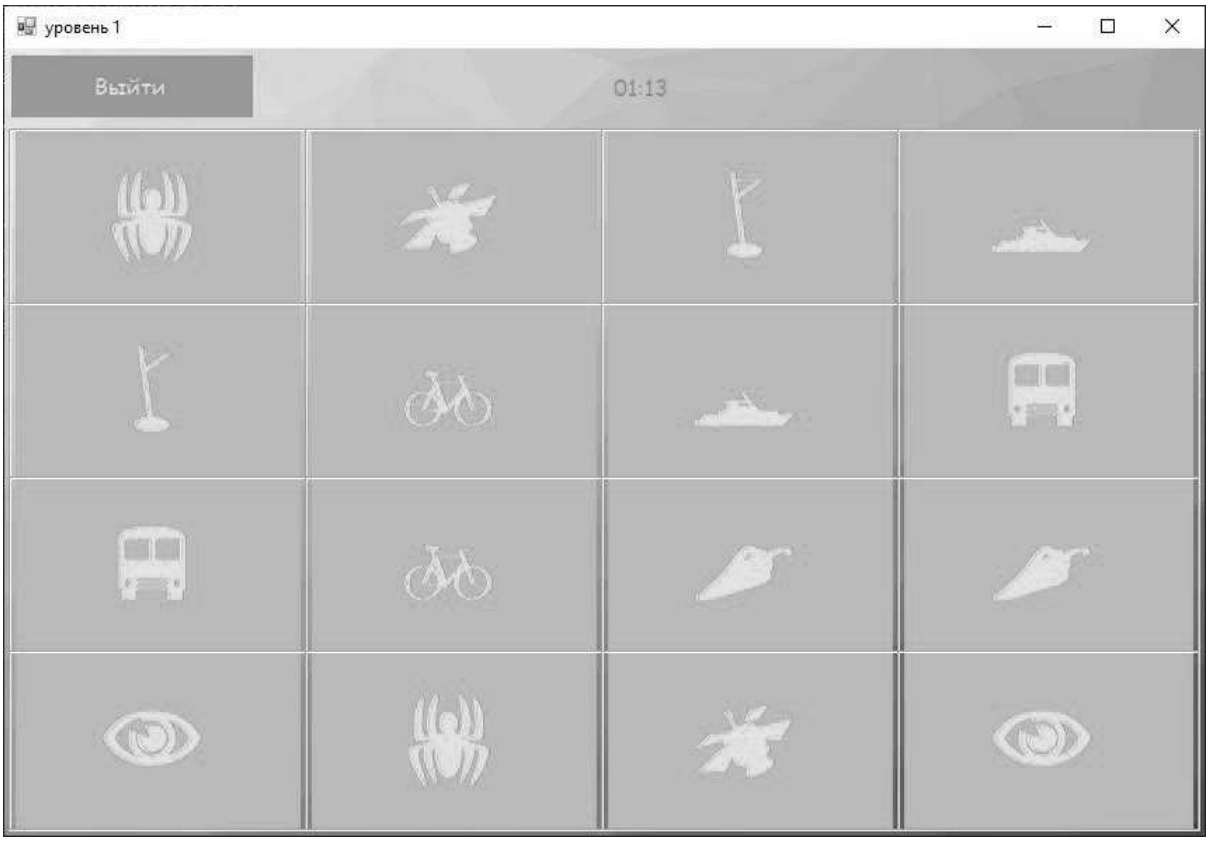


Рисунок 42 – Первый уровень игры

В ходе тестовой проверки программы проведена проверка работы приложения. Таким образом, нами выявлено, что программный продукт работает без сбоев и готов к использованию.

Выводы по третьему разделу

В третьем разделе описан программный код игры с помощью рисунков, тестовая проверка программы, описание разработки программного продукта и дизайн игрового приложения.

ГЛАВА 4 ЭКОНОМИЧЕСКАЯ ЧАСТЬ

Разрабатываемая игра является достаточно простой, поэтому разработчик выполняет функции всех необходимых членов команды – дизайнера и программиста. Для реализации любого подобного проекта, необходим компьютер средней мощности. Готовый проект размещен в ФКУ СИЗО – 3 ГУФСИН России. Далее подсчитаем расходы.

4.1 Расчет трудозатрат

На разработку такого небольшого проекта необходимо около четырех недель. Сколько бы стоило нанять одного специалиста из областей гейм-дизайна, программирования приведено ниже в таблице 2, взята средняя зарплата по региону Челябинской области с сайта поиска работ headhanter.ru .

Таблица 1 − Расчет трудозатрат

Показатель	1 месяц
Заработная плата программиста C#, руб.	50000
Итого, руб.	50000

4.2 Расчет стоимости необходимого оборудования

Для работы с выбранными программами использовалось государственное оборудование ФКУ СИЗО – 3 ГУФСИН России, затраты на оборудование для нас составляют 0 рублей. Стоимость необходимого оборудования представлена в таблице 2.

Таблица 2 − Стоимость оборудования

Показатель	Стоимость
Стоимость ПК для программиста, руб.	0

4.3. Расчет ожидаемой прибыли

Работая в ФКУ СИЗО – 3 ГУФСИН России, затраты на разработку программы составили 0 рублей, так как создание игры «Дежавю» выполнялась в рабочее время в помещении учреждения. Наша прибыль составила 42000 рублей

исходя из заработной платы сотрудника учреждения.

Выводы по четвертому разделу

В разделе описывается экономическая часть, приведены расчеты по затратам и прибыли от реализации игры.

ЗАКЛЮЧЕНИЕ

В данной дипломной работе выявлено, что любому человеку необходимо развивать и периодически тренировать свою память в любом возрасте. Созданное программное средство может помочь решить данную проблему, так как в данном случае тренировка памяти производится при помощи увлекательной игры.

Главной целью данной дипломной работы является разработка игрового программного средства «Дежавю», направленного на развитие и тренировку памяти. Его создание выполнялось в несколько этапов, один из них – это написание технического задания на создание программного продукта. На данном этапе были выявлены требования к структуре и надёжности программного средства, к составу и параметрам технических средств и к информационной и программной совместимости.

Далее следовал этап изучения инструментального средства разработки программы, при котором мы ознакомились с историей создания среды разработки, изучили её функционал и инструменты, которые она содержит.

Затем был разработан игровой программный продукт, который в первую очередь подразумевал создание базы данных пользователей для отслеживания рекордов каждого игрока, спроектирован пользовательский интерфейс, который предполагал подбор шрифта, стиля и цветового набора для удобства работы с программой. К тому же, разработан код программного продукта, благодаря которому игра начала свою работу. Проведена и описана тестовая проверка игры, в которой продемонстрировано прохождение одного из уровней и представлен его интерфейс. Выявлено, что программный продукт работает без сбоев и готов к использованию.

Разработана сопроводительная документация, которая включает в себя общие сведения о программном средстве и инструкции для программиста и пользователя.

Рассчитаны технико-экономические показатели и срок окупаемости игрового программного продукта. Сделан вывод, что судя по расчетам разработка

программного продукта эффективна, так как так как нормативный показатель окупаемости ниже фактического, а фактический рассчитанный срок окупаемости ниже нормативного.

Изучены правила безопасности на предприятии для устойчивого функционирования системы, защищенной разработки и дальнейшей технической поддержки приложения.

Уникальность разработанного программного продукта заключается в том, что каждый игрок после прохождения уровня может сравнить свой результат среди других пользователей в общей таблице рейтинга, при желании улучшить его и стать лидером.

Созданная программа может найти своё применение не только в учебных учреждениях, но подходит и для людей любого возраста, которые решили потренировать свою память и проверить свои возможности в соревновании с другими игроками.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

Нормативные правовые акты

1. ГОСТ 19.402–78 ЕСПД. Описание программы. Единая система программной документации: межгосударственный стандарт: дата введения 1980–01–01 / Государственный комитет СССР по стандартам. – Изд. Официальное. – М.: Стандартинформ, 2010. – 74 с.

2. ГОСТ 19.502–78 ЕСПД. Общее описание. Требования к содержанию и оформлению: межгосударственный стандарт: дата введения 1980–01–01 / Государственный комитет СССР по стандартам. – Изд. Официальное. – М.: Стандартинформ, 2010. – 90 с.

3. ГОСТ 19.505–79 ЕСПД. Руководство оператора: Требования к содержанию и оформлению : межгосударственный стандарт: дата введения 1980–01–01 / Государственный комитет СССР по стандартам. – Изд. Официальное. – М.: Стандартинформ, 2010. – 99 с.

4. IEEE Std 1063–2001, «IEEE Standard for Software User Documentation». For software user documentation: International Standard: Date of introduction 2001-03-19. – New York: The Institute of Electrical and Electronics Engineers, Inc. [Электронный ресурс]. – Режим доступа: https://standards.ieee.org/standard/1063-2001.html (время обращения 28.05.2021 г.).

Специальная литература

5. Агуров, П. C.#. Разработка компонентов в MS Visual Studio: учебное пособие / П. Агуров. – СПб.: БХВ-Петербург, 2017. – 466 с.

6. Биллинг, В. А. Основы объектного программирования на C#: учебное пособие / В.А. Биллинг. – М.: Бином, 2015. – 582 с.

7. Бурлаков, И. Психология компьютерных игр / И. Бурлаков. – М.: Независимая фирма «Класс», 2000. – 84 с.

8. Евдокимов, П. В. C# на примерах: учебное пособие / П.В. Евдокимов. – СПб.: Наука и Техника, 2017. – 320 с.

9. Игра со всех сторон (Книга о том, как играют дети и прочие люди). Современные исследования, междисциплинарный подход, практические рекомендации, взгляд в будущее. – М.: Фонд «Прагматика культуры», 2003. – 432 с.

10. Кумагина, Е. А. Модели жизненного цикла и технологии проектирования программного обеспечения: учебно-методическое пособие / Е.А Кумагина. – Нижний Новгород: ННГУ, 2016. – 41 с.

11. Ломакин, В. В. Базы данных и базы знаний: учебное пособие / В.В. Ломакин. – Белгород: издательство БелГУ, 2014. – 216 с.

12. Понамарев, В. В. Программирование на C++/C# в Visual Studio : учебное пособие / В.В. Понамарев. – М.: BHV, 2015. – 352 с.

13. Пропп, В. Морфология волшебной сказки / В. Пропп. – М.: Лабиринт, 1998. – 512 с.

14. Стиллмен, Э. Изучаем C#: учебное пособие / Э. Стиллмен. – СПб.: Питер, 2016. – 816 с.

15. Ульман, Л. MySQL: учебное пособие / Л. Ульман. – М.: ДМК Пресс, 2018. – 352 с.

16. Burn A. Writing computer games: Game literacy and new-old narratives. Educational Studies in Language and Literature. – 2007. – № 7(4). – 67 p.

17. Carroll N. Philosophy of Art. A Contemporary Inroduction. – L.: Routledge, 2010.

Электронные ресурсы

18. Энциклопедия языков программирования C# [Электронный ресурс]. – Режим доступа: http://progopedia.ru/language/csharp/, свободный (дата обращения: 09.01.2021).

19. Объектно–ориентированный язык программирования C# [Электронный ресурс]. – Режим доступа: https://habr.com/ru/hub/csharp/, свободный (дата обращения: 14.01.2021).

20. Введение в Windows Forms [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/windowsforms/1.1.php, свободный (дата обращения: 22.01.2021).

21. Практическое руководство. Создание приложений Windows Forms [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru–ru/dotnet/framework/winforms/how–to–create–a–windows–forms–application–from–the–command–line, свободный (дата обращения: 23.01.2021).

22. Жизненный цикл ПО. Каскадная модель (Waterfall) [Электронный ресурс]. – Режим доступа: https://xbsoftware.ru/blog/zhiznennyj–tsykl–po–kaskadnaya–model–waterfall/, свободный (дата обращения: 07.02.2021).

23. Инструкция пользователя информационной системы [Электронный ресурс]. – Режим доступа: https://infourok.ru/instrukciya-polzovatelya-informacionnoy-sistemi-2028441.html, свободный (дата обращения 19.03.2021).

24. MySql [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/MySQL, свободный (дата обращения 14.03.2021).

25. Проектирование программного продукта [Электронный ресурс]. – Режим доступа: https://studfile.net/preview/5910911/page:3/, свободный (дата обращения 02.04.2021).

26. Каскадная модель жизненного цикла: преимущества и недостатки [Электронный ресурс]. – Режим доступа: https://vkmp.ru/dlja-rukovoditelja/954-kaskadnaja-model-zhiznennogo-cikla-preimushhestva-i-nedostatki/, свободный (дата обращения 11.03.2021).

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А
Форма авторизации пользователя в системе

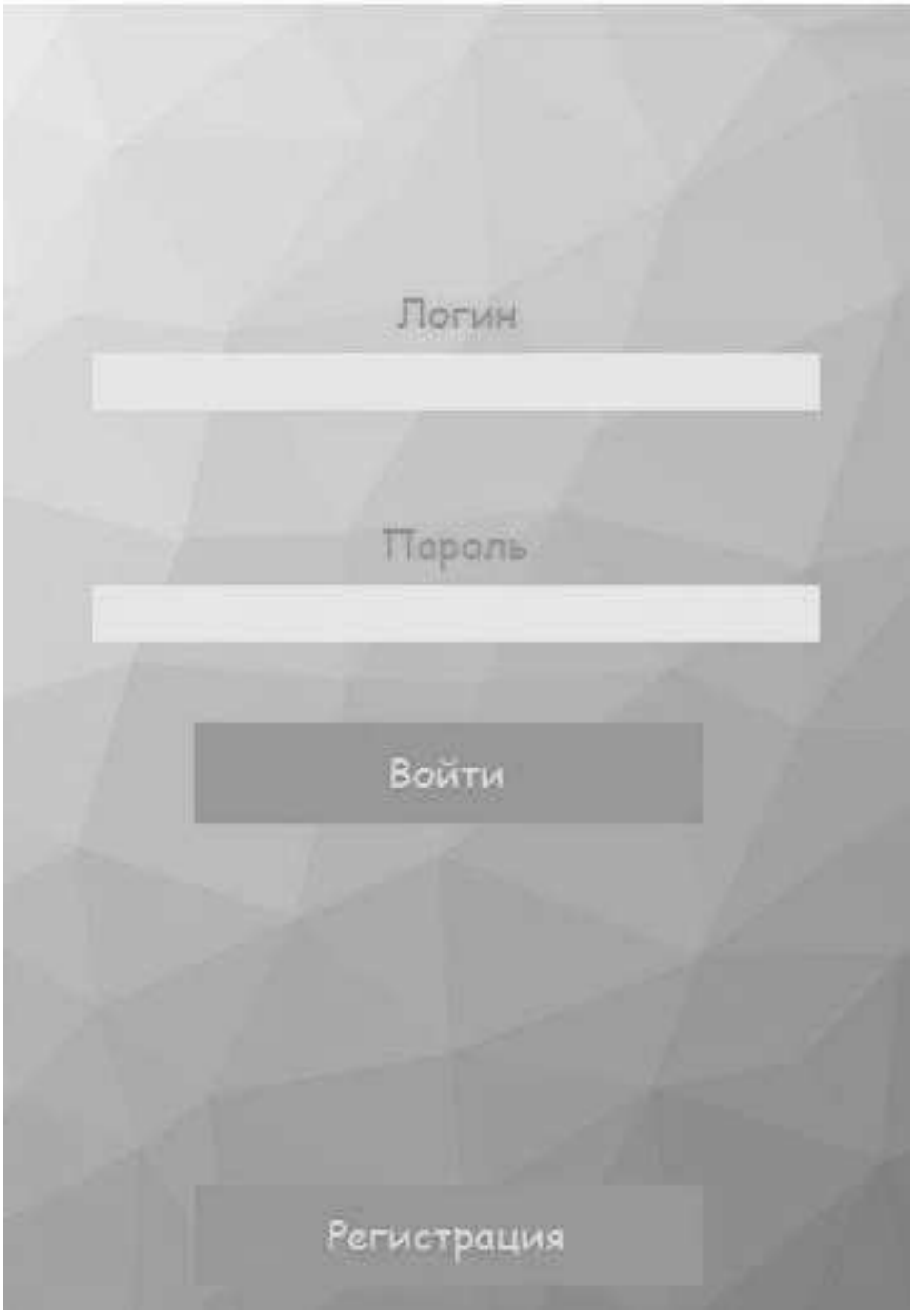


Рисунок А.1 – Форма авторизации

ПРИЛОЖЕНИЕ Б
Главное меню игры

Рисунок Б.1 – Главное меню

ПРИЛОЖЕНИЕ В
Форма регистрации пользователя в системе

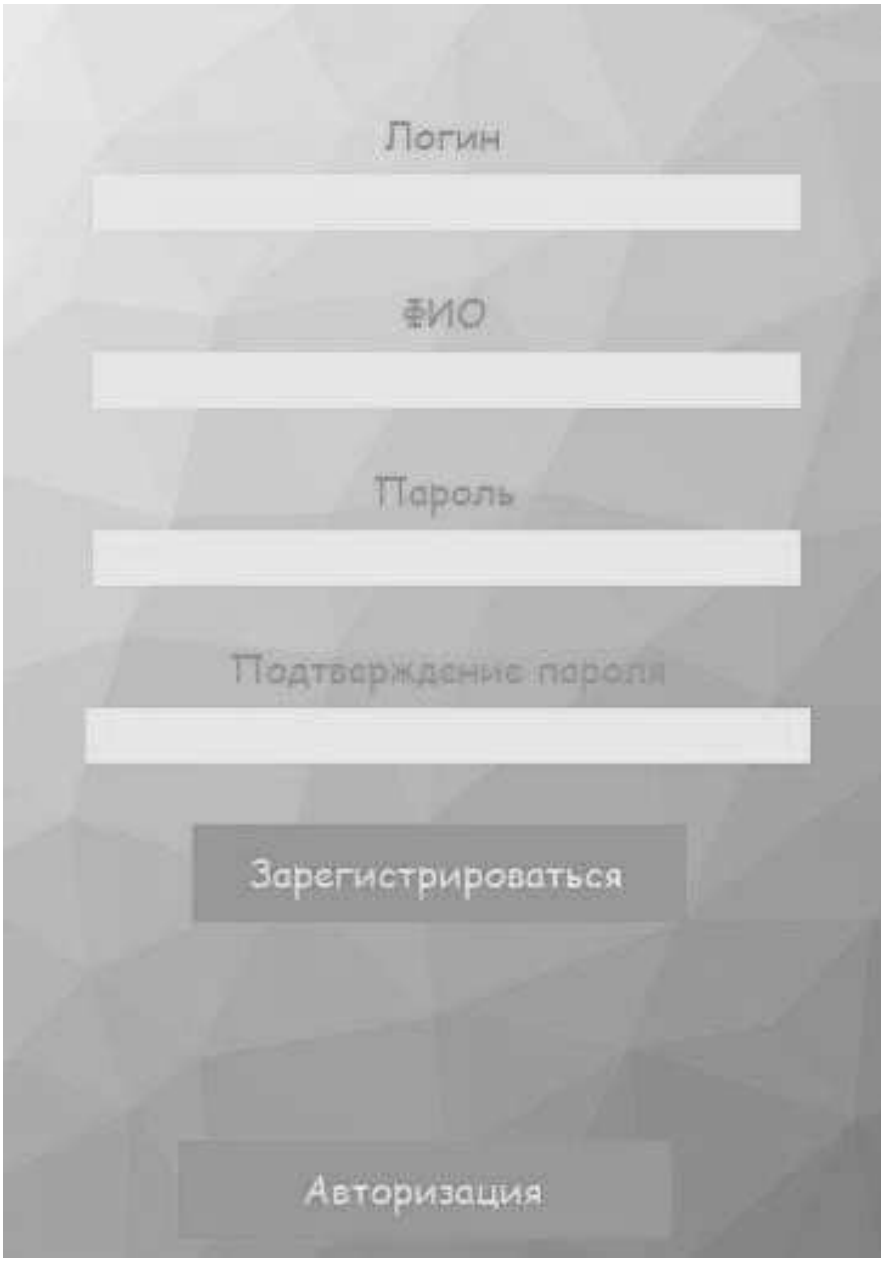


Рисунок В.1 – Форма регистрации

ПРИЛОЖЕНИЕ Г
Выбор уровня

Рисунок Г.1 – Выбор уровня

ПРИЛОЖЕНИЕ Д
Рейтинг игроков

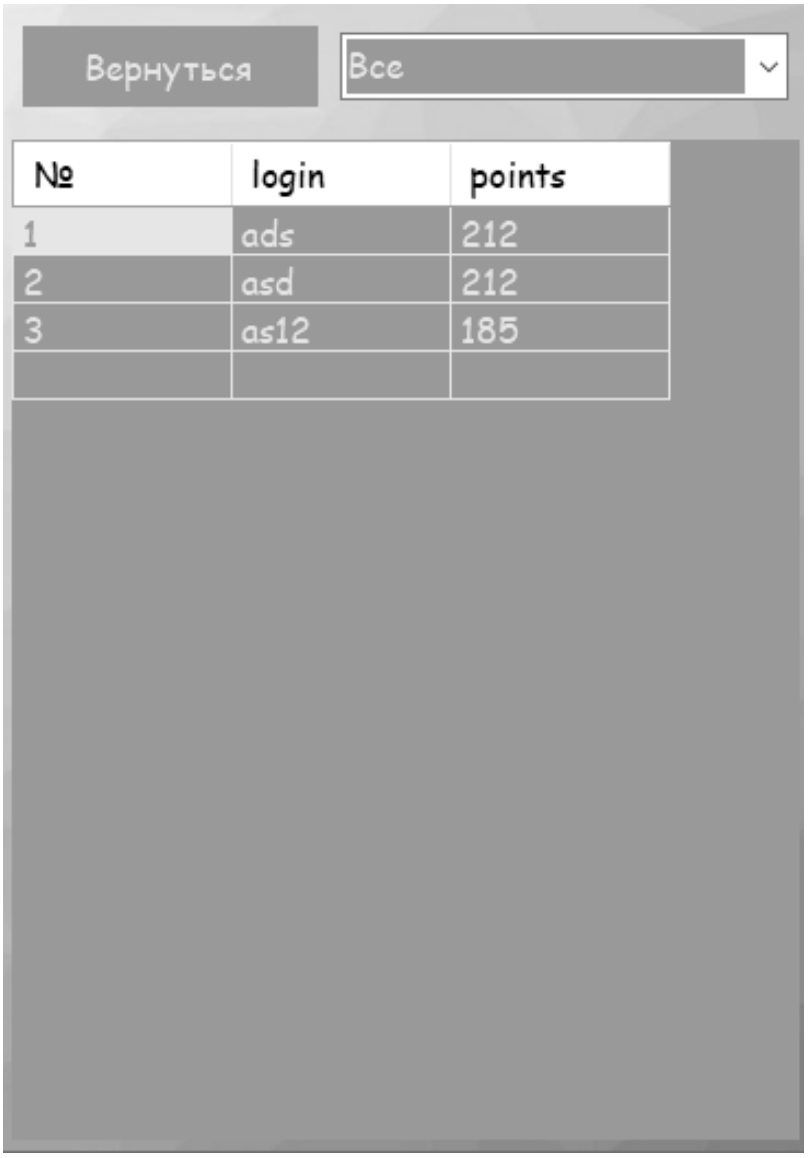


Рисунок Д.1 – Рейтинг игроков

ПРИЛОЖЕНИЕ Е
Управление уровнями для администратора игры

Рисунок Е.1 – Управление уровнями