СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3

1 ЦЕЛИ И ЗАДАЧИ УЧЕБНОЙ ПРАКТИКИ 4

1.1 Основные требования по технике безопасности во время прохождения учебной практики 4

1.2 Нормативно-правовые документы 5

2 ПРЕДМЕТНАЯ ОБЛАСТЬ. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ 7

2.1 Описание предметной области 7

2.2 Группы пользователей информационной системы 8

2.3 Основные требования, предъявляемые к информационной системе 8

3 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ «РЕМОНТ ПАССАЖИРСКИХ ВАГОНОВ» 10

3.1 Моделирование некоторых динамических аспектов системы 10

3.1.1 Создание диаграммы вариантов использования 10

3.1.2 Создание диаграммы деятельности 10

3.2 Разработка функциональной структуры информационной системы. Карта навигации по информационной системе 11

3.3 Разработка макета информационной системы «Ремонт пассажирских вагонов» 11

4 РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ «РЕМОНТ ПАССАЖИРСКИХ ВАГОНОВ» 12

4.1 Моделирование и разработка базы данных 12

4.2 Разработка программных модулей информационной системы «Ремонт пассажирских вагонов» 15

5 РАЗРАБОТКА ТЕСТОВЫХ НАБОРОВ И ТЕСТОВЫХ СЦЕНАРИЕВ 38

5.1 Назначение эксперимента. Выбор и обоснование методики проведения тестирования 38

5.2 Технология тестирования 38

5.3 Результаты проведения тестирования 40

ЗАКЛЮЧЕНИЕ 43

БИБЛИОГРАФИЧЕСКИЙ СПИСОК 44

ПРИЛОЖЕНИЯ А 45

ПРИЛОЖЕНИЯ Б 64

# ВВЕДЕНИЕ

Пассажирские вагоны играют ключевую роль в сфере железнодорожного транспорта, обеспечивая комфорт и безопасность путешествий. С течением времени эта отрасль претерпела значительное развитие, внедряя новые технологии и улучшая условия для пассажиров.

Начиная с классических вагонов с деревянными отделениями, современные пассажирские поезда предлагают разнообразные варианты, от купейных до открытых салонов. Инновации в дизайне и строительстве включают использование легких и прочных материалов, улучшенные системы вентиляции и климат контроля.

С развитием высокоскоростных поездов, появлением беспроводного интернета и интеллектуальных систем управления, пассажирские вагоны становятся не только местом транспортировки, но и платформой для передовых технологий. Интеграция умных решений, таких как системы безопасности и развлечений, сделала поездки более современными и удобными.

В целом, развитие пассажирских вагонов продолжает стремительно продвигаться вперед, объединяя комфорт, технологии и ответственность перед окружающей средой.

# ЦЕЛИ И ЗАДАЧИ УЧЕБНОЙ ПРАКТИКИ

Разработка программных модулей для системы ремонта пассажирских вагонов направлена на повышение эффективности, надежности и прозрачности процессов технического обслуживания. Целью является разработка и проектирование информационной системы, способствующей оптимизации работы и обеспечивающей высокий уровень безопасности и качества обслуживания пассажирского транспорта.

Задачи:

1. Автоматизация Учета Технического Состояния: Разработка модулей для автоматизации мониторинга и учета технического состояния пассажирских вагонов.
2. Управление Ремонтными Работами: Реализация функционала по управлению ремонтными работами, включая планирование, выделение ресурсов, контроль сроков и мониторинг качества выполненных работ.
3. Обеспечение Безопасности Данных: Разработка модулей для защиты конфиденциальности данных, связанных с техническим состоянием вагонов и ремонтными работами, чтобы предотвратить несанкционированный доступ.

Цель и задачи в совокупности направлены на повышение эффективности обслуживания и обеспечение безопасности и надежности пассажирского транспорта через современные информационные технологии.

## Основные требования по технике безопасности во время прохождения учебной практики

К работе в компьютерном классе допускаются лица, ознакомленные с данной инструкцией по технике безопасности и правилам поведения. Необходимо неукоснительно соблюдать правила по технике безопасности, т.к. нарушение этих правил может привести к поражению электрическим током, вызвать возгорание и навредить вашему здоровью.

* Работа обучающегося в компьютерном классе разрешается только в присутствии преподавателя (лаборанта).
* Во время занятий посторонние лица могут находиться в классе только с разрешения преподавателя.
* Во время перемен между уроками проводится обязательное проветривание компьютерного кабинета с обязательным выходом обучающегося из класса.

Каждый обучающийся в ответе за состояние своего рабочего места и сохранность размещенного на нем оборудования.

Обучающегося, допустившего нарушение по технике безопасности и правилам поведения в компьютерном классе, может быть привлечен к дисциплинарной ответственности. Если нарушение техники безопасности связано с причинением имущественного ущерба, он несет и материальную ответственность в установленном законом порядке.

## Нормативно-правовые документы

Оформление и содержание технического задания должно соответствовать требованиям стандарта «ГОСТ 19.201-78. ЕСПД. Общие положения.

Техническое задание оформляют в соответствии с ГОСТ 19.106-78 на листах формата 11 и 12 по ГОСТ 2.301-68, как правило, без заполнения полей листа. Номера листов (страниц) проставляются в верхней части листа над текстом.

Лист утверждения и титульный лист оформляют в соответствии с ГОСТ 19.104-78.

Информационную часть (аннотацию и содержание), лист регистрации изменений допускается в документ не включать.

Для внесения изменений или дополнений в техническое задание на последующих стадиях разработки про1раммы или программного изделия выпускают дополнение к нему.

Согласование и утверждение дополнения к техническому заданию проводят в том же порядке, который установлен для технического задания.

Техническое задание должно содержать следующие разделы:

* введение;
* основания для разработки;
* назначение разработки;
* требования к программе или программному изделию;
* требования к программной документации;
* технико-экономические показатели;
* стадии и этапы разработки;
* порядок контроля и приемки;
* в техническое задание допускается включать приложения.

В зависимости от особенностей программы или программного изделия допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них.

Вывод по главе: были описаны, техника безопасности во время прохождения учебной практики; нормативно-правовые документы, которые отвечают за стандарты разработки; цели и задачи учебной практики, включая вводную часть, где описана тематика отрасли и её тенденции развития.

# ПРЕДМЕТНАЯ ОБЛАСТЬ. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ

Предметная область информационной системы — это материальная система или система, характеризующая элементы материального мира, информация о которой хранится и обрабатывается. Предметная область рассматривается как некоторая совокупность реальных объектов и связей между ними.

## Описание предметной области

Необходимо разработать и спроектировать информационную систему для эффективного управления деятельностью гостиницы.

В любой уважающей себя гостинице существует большое количество возможных вариантов заселения гостей: все номера различаются все номера различаются по категориям (суперлюкс, люкс и т.д.), по количеству комнат в номере, количеству мест в каждом номере, а также по обустройству комнат – учитывается, например, наличие телевизора, холодильника, телефона.

Информационная система "Деятельность гостиницы" призвана содействовать повышению эффективности и безопасности в процессах обслуживания, а также уменьшению затрат за счет оптимизации ресурсов и раннего выявления потенциальных проблем.

## Группы пользователей информационной системы

Сотрудники, в зависимости от своих функциональных обязанностей, могут выполнять действия, направленные на мониторинг, управление, анализ данных и обеспечение безопасности в рамках проектируемой информационной системы "Деятельность гостинцы".

Роли и их Обязанности:

* + Гость: просмотр доступных номеров, возможность забронировать номер.
  + Администратор: управление системами аутентификации, авторизации и шифрования; оптимизация работы системы для обеспечения высокой производительности; полный доступ ко всей системе.
  + Менеджер: оформление гостя, подтверждение бронирования номера, прием платы за проживание, оформление квитанций.

## Основные требования, предъявляемые к информационной системе

* + Гибкость: система должна обладать высокой степенью редактирования, чтобы пользователи могли адаптировать её под различные потребности в рамках своих обязанностей; возможность легкого масштабирования системы при увеличении объема данных или числа пользователей.
  + Надежность: обеспечение бесперебойной работы системы с минимальным количеством сбоев и перерывов; регулярное создание резервных копий данных для предотвращения потери информации в случае сбоев.
  + Эффективность: обеспечение высокой производительности при обработке данных, особенно в условиях одновременного использования множества пользователей; эффективное использование вычислительных ресурсов, чтобы система функционировала быстро и эффективно.
  + Безопасность: реализация надежной системы аутентификации для проверки личности пользователей и установление строгих прав доступа; и использование шифрования для защиты передаваемой и хранимой чувствительной информации.

Вывод по главе: было описано, что такое предметная область и её описание; группы пользователей, которые будут пользоваться информационной системой и основные системные требования к ней.

# ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ «ДЕЯТЕЛЬНОСТЬ ГОСТИНИЦЫ»

Проектированием информационных систем называется многоступенчатый процесс их создания и/или модернизации путём применения упорядоченной совокупности методологий и инструментария. Проектирование (в отличие от моделирования) предполагает работу с пока несуществующим объектом и направлено на создание информационной системы.

## Моделирование некоторых динамических аспектов системы

Диаграммы деятельности применяются для моделирования динамических аспектов поведения системы. Эти динамические аспекты могут включать деятельность на любом уровне абстракции в любом представлении системной архитектуры, включая классы, интерфейсы, компоненты и узлы.

При разработке программного обеспечения заблаговременное планирование и моделирование значительно упрощают программирование. На этапе создания концептуальной модели для описания функционального назначения используется диаграмма вариантов использования, а для описания логики процедур и бизнес-процессов используется диаграмма деятельности.

### Создание диаграммы вариантов использования

В приложении А на рисунке 1 представлен пример диаграммы вариантов использования информационной системы «Деятельность гостиницы».

### Создание диаграммы деятельности

В приложении А на рисунках 2 – 5 представлена диаграмма деятельности (по ролям) информационной системы «Деятельность гостиницы».

## Разработка функциональной структуры информационной системы. Карта навигации по информационной системе

Карта переходов по информационной системе «Деятельность гостиницы» представлена в приложении А на рисунке 6.

## Разработка макета информационной системы «Деятельность гостиницы»

В приложении А на рисунке 7 представлено окно авторизации.

На этапе авторизации пользователь может ввести логин и пароль, чтобы продолжить работу в информационной системе.

В случае если пользователь является потенциальным гостем, то он нажимает на кнопку войти как гость. В этом случае откроется окно выбора номера, представленное в приложении А на рисунке 8.

После того, как пользователь успешно прошел авторизацию, он может продолжить работу с таблицами на главной форме. В приложении А на рисунках 9 - 12 представлена главная форма, которая будет отображена, в зависимости от прав пользователя в системе. Например, для гостя будут скрыты кнопки редактирования, добавления и удаления записи, так как данной группе пользователей доступен только просмотр данных и бронирование номера.

Дизайн выполнен в простом стиле для того, чтобы пользователь с любым уровнем владения компьютером мог без препятствий работать с программным обеспечением.

Вывод по главе: была спроектирована диаграмма вариантов использования, диаграмма деятельности и макеты будущей информационной системы.

# РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ «ДЕЯТЕЛЬНОСТЬ ГОСТИНИЦЫ»

Прототип — это один из этапов разработки, который заключается в продумывании содержания и расположения важных элементов интерфейса. Прототип — это моделирование конечного продукта. Это интерактивный макет, который может иметь любую степень точности. Основная цель создания прототипов – проверить, насколько последователен путь пользователя, и выявить препятствия, которые могут возникнуть в процессе его взаимодействия с продуктом.

## Моделирование и разработка базы данных

Исходя из анализа предметной области, можно выделить пять сущностей: «Rooms», «Admins», «Managers», «Guests», «Reservations».

В приложении А на рисунке 13 представлена логическая модель данных базы данных «Деятельность гостиницы».

Создание базы данных происходит в MS SQL, в примере представлено создание базы данных в MS SQL Server.

С помощью раздела «Создание» и конструктора таблиц MS SQL созданы все таблицы и поля, представленные на логической модели данных.

Названия полей и их типы данных представлены в таблицах 1-8.

Таблица 1 – Поля таблицы «Rooms»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Number | Числовой |
| Rooms | Числовой |
| Storey | Числовой |
| TV | Логический |
| Fridge | Логический |
| Bed | Числовой |
| Type | Числовой |
| Balcony | Логический |

Таблица 2 – Поля таблицы «Admins»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| AdminID | Числовой |
| Username | Текстовый |
| Password | Текстовый |

Таблица 3 – Поля таблицы «Managers»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| ManagerID | Числовой |
| Username | Текстовый |
| Password | Текстовый |

Таблица 4 – Поля таблицы «Guests»

| Название поля | Тип данных |
| --- | --- |
| GuestID | Числовой |
| LastName | Текстовый |
| FirstName | Текстовый |
| MiddleName | Текстовый |
| Address | Текстовый |
| Town | Текстовый |
| Aim | Текстовый |
| Passport | Текстовый |
| PassportDate | Дата |
| Region | Текстовый |
| Work | Текстовый |
| BirthYear | Дата |
| Pictures | OLE |
| PhoneNumber | Текстовый |

Таблица 5 – Поля таблицы «Reservations»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| ReservationID | Числовой |
| GuestID | Числовой |
| RoomNumber | Числовой |
| LastName | Текстовый |
| FirstName | Текстовый |
| MiddleName | Текстовый |
| Come | Дата |
| Leave | Дата |
| Money | Денежный |
| Cash | Логический |
| Receipt | Числовой |
| Comment | Текстовый |
| Registrar | Текстовый |

В приложении А на рисунке 14 представлены все созданные таблицы в обозревателе объектов базы данных.

После того, как таблицы созданы, необходимо создать между ними связи.

В приложении А на рисунке 15 представлена физическая модель данных, отражающая все таблицы и их поля, а также связи между таблицами.

## Разработка программных модулей информационной системы «Деятельность гостиницы»

Разработанные в MS Visual Studio формы продемонстрированы в приложении А на рисунках 16-21.

Свойства элементов формы Авторизация представлены в таблице 9.

Таблица 9 – Свойства элементов формы Авторизация

| Объект | Свойство | Значение |
| --- | --- | --- |
| Авторизация | FormBorderStyle | FixedSingle |
| Text | Авторизация |
| StartPosition | CenterScreen |
| Size | 470; 400 |
| ShowIcon | True |

Событийно-управляемые процедуры в форме Авторизация представлены в таблице 10

Таблица 10 – Событийно-управляемые процедуры в форме «Авторизация»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| Класс Авторизация | public partial class Form1 : Form  {  private const string ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True;";  public Form1()  {  InitializeComponent();  } |
| Кнопка Войти | private void button1\_Click(object sender, EventArgs e)  {  string username = textBox1.Text;  string password = textBox2.Text;  if (CheckAdminCredentials(username, password))  {  AdminForm adminForm = new AdminForm();  adminForm.Show();  this.Hide();  }  else if (CheckManagerCredentials(username, password))  {  GuestForm guestForm = new GuestForm();  ManagerForm managerForm = new ManagerForm(guestForm);  managerForm.Show();  this.Hide();  }  else  {  MessageBox.Show("Неверные учетные данные. Пожалуйста, повторите попытку.", "Ошибка входа");  }  }  private bool CheckAdminCredentials(string username, string password)  {  using (SqlConnection connection = new SqlConnection(ConnectionString))  {  connection.Open();  string query = "SELECT COUNT(\*) FROM Admins WHERE Username = @username AND Password = @password";  using (SqlCommand command = new SqlCommand(query, connection))  {  command.Parameters.AddWithValue("@username", username);  command.Parameters.AddWithValue("@password", password);  int result = (int)command.ExecuteScalar();  return result > 0;  }  }  }  private bool CheckManagerCredentials(string username, string password)  {  using (SqlConnection connection = new SqlConnection(ConnectionString))  {  connection.Open();  string query = "SELECT COUNT(\*) FROM Managers WHERE Username = @username AND Password = @password";  using (SqlCommand command = new SqlCommand(query, connection))  {  command.Parameters.AddWithValue("@username", username);  command.Parameters.AddWithValue("@password", password);  int count = (int)command.ExecuteScalar();  return count > 0;  }  }  } |
| Кнопка Войти как гость | private void button2\_Click(object sender, EventArgs e)  {  GuestForm guestForm = new GuestForm();  guestForm.Show();  this.Hide();  } |

Свойства элементов формы Администратор представлены в таблице 11.

Таблица 11 – Свойства элементов формы Администратор

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Администратор | FormBorderStyle | None |
| Text | Администрирование |
| StartPosition | FixedSingle |
| Size | 815; 490 |
| ShowIcon | False |

Событийно-управляемые процедуры в форме Администратор представлены в таблице 12.

Таблица 12 – Событийно-управляемые процедуры в форме «Администратор»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| Класс Администратор | public partial class AdminForm : Form  {  private const string ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True;";  private string selectedTable;  public AdminForm()  {  InitializeComponent();  } |
| Событие LoadTables | private void LoadTables()  {  using (SqlConnection connection = new SqlConnection(ConnectionString))  {  connection.Open();  DataTable schema = connection.GetSchema("Tables");  foreach (DataRow row in schema.Rows)  {  string tableName = row["TABLE\_NAME"].ToString();  comboBox1.Items.Add(tableName);  }  }  if (comboBox1.Items.Count > 0)  {  comboBox1.SelectedIndex = 0;  LoadTableData(comboBox1.SelectedItem.ToString());  }  } |
| Событие LoadTableData | private void LoadTableData(string tableName)  {  using (SqlConnection connection = new SqlConnection(ConnectionString))  {  connection.Open();  string query = $"SELECT \* FROM {tableName}";  using (SqlCommand command = new SqlCommand(query, connection))  {  SqlDataAdapter adapter = new SqlDataAdapter(command);  DataTable dataTable = new DataTable();  adapter.Fill(dataTable);  dataGridView1.AllowUserToAddRows = true;  dataGridView1.DataSource = dataTable;  }  }  } |
| Событие Combobox | private void comboBox1\_SelectedIndexChanged\_1(object sender, EventArgs e)  {  selectedTable = comboBox1.SelectedItem?.ToString();  if (!string.IsNullOrEmpty(selectedTable))  {  LoadTableData(selectedTable);  }  } |
| Кнопка Удалить | private void button1\_Click(object sender, EventArgs e)  {  if (comboBox1.SelectedItem != null && dataGridView1.SelectedRows.Count > 0)  {  string selectedTable = comboBox1.SelectedItem.ToString();  int selectedIndex = dataGridView1.SelectedRows[0].Index;  int idToDelete = Convert.ToInt32(dataGridView1.Rows[selectedIndex].Cells[0].Value);  using (SqlConnection connection = new SqlConnection(ConnectionString))  {  connection.Open();  string deleteQuery = $"DELETE FROM {selectedTable} WHERE {dataGridView1.Columns[0].HeaderText} = @ID";  using (SqlCommand command = new SqlCommand(deleteQuery, connection))  {  command.Parameters.AddWithValue("@ID", idToDelete);  command.ExecuteNonQuery();  }  }  LoadTableData(selectedTable);  }  else  {  MessageBox.Show("Пожалуйста, выберите строку для удаления.");  }  } |
| Кнопка Сохранить | private void button2\_Click(object sender, EventArgs e)  {  if (!string.IsNullOrEmpty(selectedTable))  {  DataTable dataTable = (DataTable)dataGridView1.DataSource;  DataTable changes = dataTable.GetChanges();  if (changes != null)  {  SaveChangesToDatabase(selectedTable, changes);  }  }  else  {  MessageBox.Show("Пожалуйста, выберите таблицу для сохранения изменений.");  }  } |
| Событие SaveChangesToDatabase | private void SaveChangesToDatabase(string tableName, DataTable changes)  {  using (SqlConnection connection = new SqlConnection(ConnectionString))  {  connection.Open();  using (SqlDataAdapter adapter = new SqlDataAdapter($"SELECT \* FROM {tableName}", connection))  {  using (SqlCommandBuilder commandBuilder = new SqlCommandBuilder(adapter))  {  adapter.Update(changes);  }  }  }  LoadTableData(tableName);  } |

Свойства элементов формы Менеджера представлены в таблице 15.

Таблица 15 – Свойства элементов формы Менеджера

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Бригадир | FormBorderStyle | FixedSingle |
| Text | Регистрация гостя |
| StartPosition | CenterScreen |
| Size | 815; 490 |
| ShowIcon | False |

Событийно-управляемые процедуры в форме Менеджер представлены в таблице 16.

Таблица 16 – Событийно-управляемые процедуры в форме «Менеджер»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| Класс Менеджер | public partial class ManagerForm : Form  {  private static GuestForm guestFormInstance;  private byte[] imageData;  private int currentReceiptNumber = 1;  public ManagerForm(GuestForm guestForm)  {  InitializeComponent();  guestFormInstance = guestForm;  textBox14.Text = currentReceiptNumber.ToString("D4");  maskedTextBox1.Mask = "9999-00-00";  maskedTextBox2.Mask = "9999-00-00";  maskedTextBox3.Mask = "9999-00-00";  maskedTextBox4.Mask = "9999-00-00";  } |
| Событие ManagerForm\_Load | private void ManagerForm\_Load(object sender, EventArgs e)  {  timer1.Interval = 1000;  timer1.Tick += timer1\_Tick;  timer1.Start();  GuestForm.ReservationRequested += GuestForm\_ReservationRequested;  } |
| Событие GuestForm\_ReservationRequested | private void HandleReservationData(int roomNumber, string phoneNumber)  {  textBox1.Text = roomNumber.ToString();  textBox2.Text = phoneNumber;  } |
| Событие timer1\_Tick | private void timer1\_Tick(object sender, EventArgs e)  {  string filePath = "ReservationData.txt";  if (File.Exists(filePath))  {  string[] lines = File.ReadAllLines(filePath);  if (lines.Length > 0)  {  string[] data = lines[0].Split(',');  int roomNumber = int.Parse(data[0]);  string phoneNumber = data[1];  HandleReservationData(roomNumber, phoneNumber);  File.Delete(filePath);  }  }  } |
| Кнопка Загрузить фото | private void button1\_Click(object sender, EventArgs e)  {  OpenFileDialog openFileDialog = new OpenFileDialog();  openFileDialog.Filter = "Image Files |\*.jpg;\*.jpeg;\*.png;\*.gif;\*.bmp";  pictureBox1.SizeMode = PictureBoxSizeMode.Zoom;  if (openFileDialog.ShowDialog() == DialogResult.OK)  {  string imagePath = openFileDialog.FileName;  pictureBox1.Image = Image.FromFile(imagePath);  imageData = File.ReadAllBytes(imagePath);  }  } |
| Кнопка Внести информацию | private void button2\_Click(object sender, EventArgs e)  {  if (ValidateInputs())  {  SaveGuestToDatabase();  MessageBox.Show("Данные успешно сохранены!");  }  else  {  MessageBox.Show("Пожалуйста, заполните все обязательные поля.");  }  } |
| Событие ValidateInputs | private bool ValidateInputs()  {  return !string.IsNullOrEmpty(textBox3.Text)  && !string.IsNullOrEmpty(textBox2.Text)  && !string.IsNullOrEmpty(textBox4.Text)  && !string.IsNullOrEmpty(textBox5.Text)  && !string.IsNullOrEmpty(maskedTextBox2.Text)  && !string.IsNullOrEmpty(textBox7.Text)  && !string.IsNullOrEmpty(textBox8.Text)  && !string.IsNullOrEmpty(textBox9.Text)  && !string.IsNullOrEmpty(textBox10.Text)  && !string.IsNullOrEmpty(maskedTextBox1.Text)  && !string.IsNullOrEmpty(textBox12.Text)  && !string.IsNullOrEmpty(textBox13.Text);  } |
| Событие SaveGuestToDatabase | private void SaveGuestToDatabase()  {  string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True";  string query = @"  INSERT INTO Guests  (LastName, FirstName, MiddleName, Address, Town, Aim, Passport, PassportDate, Region, Work, BirthYear, Picture, PhoneNumber)  VALUES  (@LastName, @FirstName, @MiddleName, @Address, @Town, @Aim, @Passport, @PassportDate, @Region, @Work, @BirthYear, @Picture, @PhoneNumber)";  using (SqlConnection connection = new SqlConnection(connectionString))  {  using (SqlCommand command = new SqlCommand(query, connection))  {  connection.Open();  command.Parameters.AddWithValue("@LastName", textBox3.Text);  command.Parameters.AddWithValue("@FirstName", textBox4.Text);  command.Parameters.AddWithValue("@MiddleName", textBox5.Text);  command.Parameters.AddWithValue("@Address", textBox8.Text);  command.Parameters.AddWithValue("@Town", textBox7.Text);  command.Parameters.AddWithValue("@Aim", textBox9.Text);  command.Parameters.AddWithValue("@Passport", textBox10.Text);  DateTime passportDate;  if (DateTime.TryParseExact(maskedTextBox1.Text, "yyyy-MM-dd", CultureInfo.InvariantCulture, DateTimeStyles.None, out passportDate))  {  command.Parameters.AddWithValue("@PassportDate", passportDate);  }  else  {  MessageBox.Show("Ошибка ввода даты паспорта.");  return;  }  command.Parameters.AddWithValue("@Region", textBox12.Text);  command.Parameters.AddWithValue("@Work", textBox13.Text);  DateTime BirthYear;  if (DateTime.TryParseExact(maskedTextBox2.Text, "yyyy-MM-dd", CultureInfo.InvariantCulture, DateTimeStyles.None, out BirthYear))  {  command.Parameters.AddWithValue("@BirthYear", BirthYear);  }  else  {  MessageBox.Show("Ошибка ввода даты рождения.");  return;  }  SqlParameter pictureParameter = new SqlParameter("@Picture", SqlDbType.VarBinary);  if (imageData != null && imageData.Length > 0)  {  pictureParameter.Value = imageData;  }  else  {  pictureParameter.Value = DBNull.Value;  }  command.Parameters.Add(pictureParameter);  command.Parameters.AddWithValue("@PhoneNumber", textBox2.Text);  command.ExecuteNonQuery();  }  }  } |
| Кнопка Забронировать | private void button3\_Click(object sender, EventArgs e)  {  if (ValidateReservationInputs())  {  MakeReservation();  MessageBox.Show("Бронирование успешно завершено!");  }  else  {  MessageBox.Show("Пожалуйста, заполните все обязательные поля.");  }  } |
| Событие ValidateReservationInput | private bool ValidateReservationInputs()  {  return !string.IsNullOrEmpty(textBox3.Text)  && !string.IsNullOrEmpty(textBox4.Text)  && !string.IsNullOrEmpty(textBox5.Text)  && !string.IsNullOrEmpty(maskedTextBox3.Text)  && !string.IsNullOrEmpty(maskedTextBox4.Text)  && !string.IsNullOrEmpty(textBox15.Text)  && !string.IsNullOrEmpty(textBox14.Text)  && !string.IsNullOrEmpty(textBox16.Text)  && !string.IsNullOrEmpty(textBox17.Text);  } |
| Событие MakeReservation | private void MakeReservation()  {  int selectedRoomNumber = int.Parse(textBox1.Text);  string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True";  string query = @"  INSERT INTO Reservations  (RoomNumber, LastName, FirstName, MiddleName, Come, Leave, Money, Cash, Receipt, Comment, Registrar)  VALUES  (@RoomNumber, @LastName, @FirstName, @MiddleName, @Come, @Leave, @Money, @Cash, @Receipt, @Comment, @Registrar)";  using (SqlConnection connection = new SqlConnection(connectionString))  {  using (SqlCommand command = new SqlCommand(query, connection))  {  connection.Open();  command.Parameters.AddWithValue("@RoomNumber", selectedRoomNumber);  command.Parameters.AddWithValue("@LastName", textBox3.Text);  command.Parameters.AddWithValue("@FirstName", textBox4.Text);  command.Parameters.AddWithValue("@MiddleName", textBox5.Text);  DateTime ComeDate;  if (DateTime.TryParseExact(maskedTextBox3.Text, "yyyy-MM-dd", CultureInfo.InvariantCulture, DateTimeStyles.None, out ComeDate))  {  command.Parameters.AddWithValue("@Come", ComeDate);  }  else  {  MessageBox.Show("Ошибка ввода даты.");  return;  }  DateTime LeaveDate;  if (DateTime.TryParseExact(maskedTextBox4.Text, "yyyy-MM-dd", CultureInfo.InvariantCulture, DateTimeStyles.None, out LeaveDate))  {  command.Parameters.AddWithValue("@Leave", LeaveDate);  }  else  {  MessageBox.Show("Ошибка ввода даты.");  return;  }  command.Parameters.AddWithValue("@Money", textBox15.Text);  command.Parameters.AddWithValue("@Cash", checkBox1.Checked);  command.Parameters.AddWithValue("@Receipt", textBox14.Text);  command.Parameters.AddWithValue("@Comment", textBox16.Text);  command.Parameters.AddWithValue("@Registrar", textBox17.Text);  command.ExecuteNonQuery();  guestFormInstance.RemoveReservedRoom(selectedRoomNumber);  currentReceiptNumber++;  textBox14.Text = currentReceiptNumber.ToString("D4");  }  }  } |

Свойства элементов формы Гость представлены в таблице 17.

Таблица 17 – Свойства элементов формы Гость

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Сотрудник | FormBorderStyle | FixedSingle |
| Text | Выбор номера |
| StartPosition | CenterScreen |
| Size | 780; 490 |
| ShowIcon | True |

Событийно-управляемые процедуры в форме Гость представлены в таблице 18.

Таблица 18 – Событийно-управляемые процедуры в форме «Гость»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| Класс Гость | public partial class GuestForm : Form  {  public static event EventHandler<ReservationEventArgs> ReservationRequested;  string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True;";  private string query = "SELECT Number, Rooms, Storey, TV, Fridge, Bed, Type, Balcony FROM Rooms WHERE 1=1 "; |
| Событие GuestForm\_Load | private void GuestForm\_Load(object sender, EventArgs e)  {  dataGridView1.ReadOnly = true;  this.roomsTableAdapter.Fill(this.dataSet2.Rooms);  foreach (DataGridViewRow row in dataGridView1.Rows)  {  int roomNumber = Convert.ToInt32(row.Cells[0].Value.ToString());  comboBox1.Items.Add(roomNumber);  maskedTextBox1.Mask = "+7 (999) 000-0000";  }  } |
| Событие RemoveReservedRoom | public void RemoveReservedRoom(int roomNumberToRemove)  {  if (dataGridView1.Columns.Contains("Number"))  {  DataGridViewColumn numberColumn = dataGridView1.Columns["Number"];  int columnIndex = numberColumn.Index;  foreach (DataGridViewRow row in dataGridView1.Rows)  {  if (row.Cells[columnIndex].Value != null && Convert.ToInt32(row.Cells[columnIndex].Value) == roomNumberToRemove)  {  dataGridView1.Rows.Remove(row);  UpdateComboBox(roomNumberToRemove);  break;  }  }  }  else  {  MessageBox.Show("Столбец 'Number' не найден в DataGridView.", "Ошибка удаления");  }  } |
| Событие UpdateComboBox | private void UpdateComboBox(int roomNumberToRemove)  {  if (comboBox1.InvokeRequired)  {  comboBox1.Invoke(new Action(() => UpdateComboBox(roomNumberToRemove)));  }  else  {  if (comboBox1.Items.Contains(roomNumberToRemove))  {  comboBox1.Items.Remove(roomNumberToRemove);  }  }  } |
| Кнопка Отсортировать | private void button1\_Click(object sender, EventArgs e)  {  int rooms, storey, beds;  bool tv, fridge, balcony;  int.TryParse(textBox1.Text, out rooms);  int.TryParse(textBox2.Text, out storey);  int.TryParse(textBox3.Text, out beds);  tv = checkBox1.Checked;  fridge = checkBox2.Checked;  balcony = checkBox3.Checked;  if (rooms <= 0 && storey <= 0 && beds <= 0 && !tv && !fridge && !balcony)  {  string defaultQuery = "SELECT Number, Rooms, Storey, TV, Fridge, Bed, Type, Balcony FROM Rooms";  using (SqlConnection connection = new SqlConnection(connectionString))  {  connection.Open();  SqlCommand defaultCommand = new SqlCommand(defaultQuery, connection);  SqlDataAdapter defaultAdapter = new SqlDataAdapter(defaultCommand);  DataTable defaultDataTable = new DataTable();  defaultAdapter.Fill(defaultDataTable);  dataGridView1.DataSource = defaultDataTable;  }  }  else  {  string query = "SELECT Number, Rooms, Storey, TV, Fridge, Bed, Type, Balcony FROM Rooms WHERE 1=1 ";  if (storey > 0)  {  query += "AND Storey = @Storey ";  }  if (rooms > 0)  {  query += "AND Rooms = @Rooms ";  }  if (tv)  {  query += "AND TV = @TV ";  }  if (fridge)  {  query += "AND Fridge = @Fridge ";  }  if (beds > 0)  {  query += "AND Bed = @Beds ";  }  if (balcony)  {  query += "AND Balcony = @Balcony ";  }  query += "ORDER BY ";  if (storey > 0)  {  query += "Storey ASC, ";  }  if (rooms > 0)  {  query += "Rooms ASC, ";  }  if (tv)  {  query += "TV DESC, ";  }  if (fridge)  {  query += "Fridge DESC, ";  }  if (beds > 0)  {  query += "Bed ASC, ";  }  if (balcony)  {  query += "Balcony DESC, ";  }  query = query.TrimEnd(',', ' ');  using (SqlConnection connection = new SqlConnection(connectionString))  {  connection.Open();  SqlCommand command = new SqlCommand(query, connection);  if (storey > 0)  {  command.Parameters.AddWithValue("@Storey", storey);  }  if (rooms > 0)  {  command.Parameters.AddWithValue("@Rooms", rooms);  }  if (tv)  {  command.Parameters.AddWithValue("@TV", tv);  }  if (fridge)  {  command.Parameters.AddWithValue("@Fridge", fridge);  }  if (beds > 0)  {  command.Parameters.AddWithValue("@Beds", beds);  }  if (balcony)  {  command.Parameters.AddWithValue("@Balcony", balcony);  }  SqlDataAdapter adapter = new SqlDataAdapter(command);  DataTable dataTable = new DataTable();  adapter.Fill(dataTable);  dataGridView1.DataSource = dataTable;  }  }  } |
| Кнопка Забронировать | private void button2\_Click(object sender, EventArgs e)  {  int selectedRoomNumber = Convert.ToInt32(comboBox1.SelectedItem);  string phoneNumber = maskedTextBox1.Text;  ReservationRequested?.Invoke(this, new ReservationEventArgs(selectedRoomNumber, phoneNumber));  SaveReservationData(selectedRoomNumber, phoneNumber);  } |
| Событие OnReservationRequeste | protected virtual void OnReservationRequested(ReservationEventArgs e)  {  ReservationRequested?.Invoke(this, e);  } |
| Класс ReservationEventArgs | public class ReservationEventArgs : EventArgs  {  public int RoomNumber { get; }  public string PhoneNumber { get; }  public ReservationEventArgs(int roomNumber, string phoneNumber)  {  RoomNumber = roomNumber;  PhoneNumber = phoneNumber;  }  } |
| Событие ReservationEventArgs | public ReservationEventArgs(int roomNumber, string phoneNumber)  {  RoomNumber = roomNumber;  PhoneNumber = phoneNumber;  } |
| Событие SaveReservationData | private void SaveReservationData(int roomNumber, string phoneNumber)  {  string filePath = "ReservationData.txt";  using (StreamWriter writer = new StreamWriter(filePath))  {  writer.WriteLine($"{roomNumber},{phoneNumber}");  }  } |

Вывод по главе: была полностью спроектирована и разработана система базы данных, были полностью спроектированы и разработаны все программные модули информационной системы “Ремонт пассажирских вагонов”.

# РАЗРАБОТКА ТЕСТОВЫХ НАБОРОВ И ТЕСТОВЫХ СЦЕНАРИЕВ

Тестирование программного обеспечения – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом.

Тестовый сценарий – это неавтоматизированный или автоматизированный сценарий, содержащий инструкции по реализации тестового набора. Тестовый сценарий может быть написан вручную (для выполнения человеком) либо полностью или частично автоматизирован.

## Назначение эксперимента. Выбор и обоснование методики проведения тестирования

В данном разделе проведены два вида тестирования. Провести анализ тестирования и выбрать, как будет проходить тестирование ИС.

Модульное тестирование. Цель: проверить, что код работает именно так, как должен (при заданных входных параметрах выдает предполагаемый результат).

Анализ покрытия кода тестами. Цель: проверить, что весь наш код отрабатывает при модульном тестировании, что нет, не участвующих в тестировании участков кода.

Анализ стилистики кода. Цель: проверка кода на соблюдение стандартов разработки (отраслевых, корпоративных, принятых на проекте).

Анализ производительности. Цель: проанализировать производительность ключевых операций, потребление памяти, утечку памяти.

## Технология тестирования

Рассмотрим, какие технологии можно применить для модульного тестирования и покрытия кода тестами.

Как в ходе выполнения проекта, так и при развитии системы, изменения в ее алгоритмы вносятся в соответствии с техническими заданиями, хотя данный документ может иметь различные названия. В технических заданиях, как правило, описываются критерии приемки: как мы и заказчик убедимся в том, что заказанный функционал работает верно.

В качестве примера можно привести такой упрощенный критерий:

* Исходные данные:
* Действия пользователя:
* В результате:
* Вручную такое тестирование, как правило, выполняется единожды, поскольку выполнять каждый раз полный комплект тестов очень трудоемко.
* Нет гарантии того, что тесты, описанные в техническом задании, покрывают функциональность полностью. Т.е. даже если тесты описаны и проводятся регулярно, вероятно будет существовать наличие не протестированного функционала.

Автоматизированное тестирование лишено данных недостатков. Конечно, возникает отдельная задача кодирования данного теста на основании критериев приемки, но это скорее дисциплинирующий фактор…

Прежде чем начинать юзабилити-тестирование необходимо собрать группу пользователей, которые будут тестировать данную информационную систему. Количество привлеченных пользователей должно быть не менее пяти человек. Пользователям будет представлен список некоторых вопросов, относящихся к графическому интерфейсу.

В таблице 21 представлены сведения о пользователях.

Таблица 21 – Информация о пользователях

|  |  |  |  |
| --- | --- | --- | --- |
| ФИО | Пол | Возраст | Образование |
| Аверьянов Николай Владимирович | Мужской | 30 | Среднее специальное |
| Туркина Надежда Михайловна | Женский | 30 | Среднее специальное |
| Окладников Антон Александрович | Мужской | 30 | Среднее специальное |
| Петрова Надежда Михайловна | Женский | 30 | Среднее специальное |
| Фурман Ксения Михайловна | Женский | 30 | Среднее специальное |

## Результаты проведения тестирования

В ходе тестирования был применен метод нагрузочного тестирования и юзабилити-тестирования, которые проверяют работу при нагрузке системы и сделать простым и удобным в пользовании информационной системы.

Суть тестирования заключается в проверке работы всех обязательных функций.

Результаты проведения нагрузочного тестирования

Для проведения нагрузочного тестирования составлена таблица 22, в которой указано количество запущенных копий программного продукта, нагрузка на процесс и оперативную память.

Таблица 22 – Результаты нагрузочного тестирования

|  |  |  |
| --- | --- | --- |
| Число копий | Нагрузка на процессор (%) | Нагрузка на память (%) |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |

Исходя из результатов нагрузочного тестирования, можно прийти к выводу, что нагрузка, оказываемая на центральный процессор и память, не имеет критичных значений. Это значит, что показатель производительности информационной системы на приемлемом уровне, то есть сбои и длительная загрузка не должны помешать пользователю при работе с данным программным продуктом.

Таблица 23 – Результаты тестирования

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вопрос | Аверьянов Николай Владимирович | Туркина Надежда Михайловна | Окладников Антон Александрович | Петрова Надежда Михайловна | Фурман Ксения Михайловна |
| Присутствуют ли такие дизайн-элементы, которые неприятны для зрения? | нет | нет | нет | нет | нет |
| Присутствует ли отвлекающая информация? | нет | нет | нет | нет | нет |
| Понятна ли структура приложения на интуитивном уровне? | да | да | да | да | да |
| Смогли ли вы найти интересующую вас информацию без затруднений? | да | да | да | нет | да |
| Понятен ли смысл приложения? | да | да | да | да | да |
| Общая оценка | 10/10 | 10/10 | 10/10 | 10/10 | 10/10 |

Исходя из средней оценки информационной системы, рассчитанной на основе общих оценок всех пользователей с учетом ответов на все поставленные вопросы, можно сделать вывод о том, что внешние характеристики графического интерфейса находятся на довольно высоком уровне. Также в информационной системе нет помех для пользователя в виде отвлекающей информации, так как все отображаемые данные находятся в рамках предметной области.

Вывод по главе: были проведены тесты путём проведения модульного тестирования, анализа покрытия кода тестами, анализа стилистики кода и анализом производительности. Также было проведено нагрузочное тестирование, в результате чего нагрузка на процессор и память не имела критичных значений.

# ЗАКЛЮЧЕНИЕ

В результате учебной практики был проведен анализ предметной области, спроектирована и разработана информационная система «Ремонт пассажирских вагонов», серверная часть которой была реализована в MS SQL, а клиентская часть в MS Visual Studio.

В MS SQL была создана база данных для хранения информации о вагонах, персонале, должностях, видов ремонта и о самом ремонте вагонов.

Клиентская часть, реализованная в MS Visual Studio, отражает все хранящиеся в базе данных таблицы. Также разработаны некоторые запросы, позволяющие редактировать и удалять существующие данные, добавлять новые и осуществлять поиск по таблицам.

Цели учебной практики были достигнуты путем проведения анализа предметной области, проектирования информационной системы и разработки программных модулей.

Это позволяет сделать вывод, что поставленные цели и задачи выполнены в полном объеме.

В дальнейшем данная информационная система может быть модернизирована путем добавления формы личного кабинета, создания версии для слабовидящих людей или улучшения и изменения дизайна.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. ГОСТ 20886-85. Организация данных в системах обработки данных. Термины и определения [Электронный доступ]. — Режим доступа: [ГОСТ 20886-85. Организация данных в системах обработки данных. Термины и определения (rostest.info)](https://rostest.info/gost/001.001.040.035/gost-20886-85/#:~:text=%D0%93%D0%9E%D0%A1%D0%A2%2020886%2D85.%20%D0%9E%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F%20%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85%20%D0%B2,%D0%BD%D0%B0%D1%83%D1%87%D0%BD%D0%BE%2D%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D1%87%D0%B5%D1%81%D0%) (Дата обращения: 22.05.2013 г.).
2. ГОСТ Р. 59793–2021. Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания [Электронный доступ]. — Режим доступа: [ГОСТ Р. 59793-2021. Информационные технологии. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания (internet-law.ru)](https://internet-law.ru/gosts/gost/77858/?ysclid=lifap277mg519765664) (Дата обращения: 30.04.2022 г.).
3. ГОСТ 19.201-78. ЕСПД. Общие положения. [Электронный доступ]. — Режим доступа: [ГОСТ 19.201-78.ЕСПД. Техническое задание. Требования к содержанию и оформлению (swrit.ru)](https://www.swrit.ru/doc/espd/19.201-78.pdf?ysclid=lpi5u6naom706376064) (Дата обращения: 01.10.1981 г.).
4. Роберт Э. Уолтерс, Майкл Коулс. SQL Server 2008: ускоренный курс для профессионалов = Accelerated SQL Server 2008. — М: [«Вильямс»](https://ru.wikipedia.org/w/index.php?title=%D0%92%D0%B8%D0%BB%D1%8C%D1%8F%D0%BC%D1%81_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)&action=edit&redlink=1), 2008. — С. 768.
5. Microsoft SQL Server [Электронный доступ]. — Режим доступа: [Microsoft SQL Server — Википедия (wikipedia.org)](https://ru.wikipedia.org/wiki/Microsoft_SQL_Server) (Дата обращения: 10.09.2023 г.).
6. Дебби Валковски. Microsoft Office Visio 2003 для «чайников» = Visio 2003 For Dummies. — М: [«Диалектика»](https://ru.wikipedia.org/w/index.php?title=%D0%94%D0%B8%D0%B0%D0%BB%D0%B5%D0%BA%D1%82%D0%B8%D0%BA%D0%B0_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)&action=edit&redlink=1), 2006. — С. 336.
7. Microsoft Visio [Электронный доступ]. — Режим доступа: [Microsoft Visio — Википедия (wikipedia.org)](https://ru.wikipedia.org/wiki/Microsoft_Visio) (Дата обращения: 07.08.2023 г.).
8. Майо Д. Самоучитель Microsoft Visual Studio 2010 = Microsoft Visual Studio 2010: A Beginner's Guide (A Beginners Guide). — C: [«БХВ-Петербург»](https://ru.wikipedia.org/w/index.php?title=%D0%91%D0%A5%D0%92-%D0%9F%D0%B5%D1%82%D0%B5%D1%80%D0%B1%D1%83%D1%80%D0%B3_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)&action=edit&redlink=1), 2010. — С. 464.
9. Microsoft Visual Studio [Электронный доступ]. — Режим доступа: [Руководство по разработке модулей расширений на C# — Павел Еремеев (pvs-studio.ru)](https://pvs-studio.ru/ru/blog/posts/csharp/a0082/) (Дата обращения: 29.08.2013 г.).

# ПРИЛОЖЕНИЯ А

Рисунок 1 – Диаграмма вариантов использования

Рисунок 2 – Диаграмма деятельности (Администратор)

Рисунок 3 – Диаграмма деятельности (Бригадир)

Рисунок 4 – Диаграмма деятельности (Сотрудник)

Рисунок 5 – Диаграмма деятельности (Гость)

Рисунок 6 – Карта переходов

Рисунок 7 – Макет Авторизации

Рисунок 8 – Макет Регистрации

Рисунок 9 – Макет Администратора

Рисунок 10 – Макет Бригадира

Рисунок 11 – Макет Сотрудника

Рисунок 12 – Макет Гостя

Рисунок 13 – Логическая модель данных

Рисунок 14 – Созданные таблицы

Рисунок 15 – Физическая модель базы данных

Рисунок 16 – Форма “Авторизация”

Рисунок 17 – Форма “Регистрация”

Рисунок 18 – Форма “Администратора”

Рисунок 19 – Форма “Бригадира”

Рисунок 20 – Форма “Сотрудника”

Рисунок 21 – Форма “Гостя”

# ПРИЛОЖЕНИЯ Б

Листинг кода формы “Авторизация”:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace Деятельность\_гостинцы

{

public partial class Form1 : Form

{

private const string ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True;";

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

string username = textBox1.Text;

string password = textBox2.Text;

if (CheckAdminCredentials(username, password))

{

AdminForm adminForm = new AdminForm();

adminForm.Show();

this.Hide();

}

else if (CheckManagerCredentials(username, password))

{

GuestForm guestForm = new GuestForm();

ManagerForm managerForm = new ManagerForm(guestForm);

managerForm.Show();

this.Hide();

}

else

{

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

}

}

private bool CheckAdminCredentials(string username, string password)

{

using (SqlConnection connection = new SqlConnection(ConnectionString))

{

connection.Open();

string query = "SELECT COUNT(\*) FROM Admins WHERE Username = @username AND Password = @password";

using (SqlCommand command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("@username", username);

command.Parameters.AddWithValue("@password", password);

int result = (int)command.ExecuteScalar();

return result > 0;

}

}

}

private bool CheckManagerCredentials(string username, string password)

{

using (SqlConnection connection = new SqlConnection(ConnectionString))

{

connection.Open();

string query = "SELECT COUNT(\*) FROM Managers WHERE Username = @username AND Password = @password";

using (SqlCommand command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("@username", username);

command.Parameters.AddWithValue("@password", password);

int count = (int)command.ExecuteScalar();

return count > 0;

}

}

}

private void button2\_Click(object sender, EventArgs e)

{

GuestForm guestForm = new GuestForm();

guestForm.Show();

this.Hide();

}

}

}

Листинг кода формы “Администратор”:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace Деятельность\_гостинцы

{

public partial class AdminForm : Form

{

private const string ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True;";

private string selectedTable;

public AdminForm()

{

InitializeComponent();

}

private void AdminForm\_Load(object sender, EventArgs e)

{

LoadTables();

}

private void LoadTables()

{

using (SqlConnection connection = new SqlConnection(ConnectionString))

{

connection.Open();

DataTable schema = connection.GetSchema("Tables");

foreach (DataRow row in schema.Rows)

{

string tableName = row["TABLE\_NAME"].ToString();

comboBox1.Items.Add(tableName);

}

}

if (comboBox1.Items.Count > 0)

{

comboBox1.SelectedIndex = 0;

LoadTableData(comboBox1.SelectedItem.ToString());

}

}

private void LoadTableData(string tableName)

{

using (SqlConnection connection = new SqlConnection(ConnectionString))

{

connection.Open();

string query = $"SELECT \* FROM {tableName}";

using (SqlCommand command = new SqlCommand(query, connection))

{

SqlDataAdapter adapter = new SqlDataAdapter(command);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

dataGridView1.AllowUserToAddRows = true;

dataGridView1.DataSource = dataTable;

}

}

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void comboBox1\_SelectedIndexChanged\_1(object sender, EventArgs e)

{

selectedTable = comboBox1.SelectedItem?.ToString();

if (!string.IsNullOrEmpty(selectedTable))

{

LoadTableData(selectedTable);

}

}

private void button1\_Click(object sender, EventArgs e)

{

if (comboBox1.SelectedItem != null && dataGridView1.SelectedRows.Count > 0)

{

string selectedTable = comboBox1.SelectedItem.ToString();

int selectedIndex = dataGridView1.SelectedRows[0].Index;

int idToDelete = Convert.ToInt32(dataGridView1.Rows[selectedIndex].Cells[0].Value);

using (SqlConnection connection = new SqlConnection(ConnectionString))

{

connection.Open();

string deleteQuery = $"DELETE FROM {selectedTable} WHERE {dataGridView1.Columns[0].HeaderText} = @ID";

using (SqlCommand command = new SqlCommand(deleteQuery, connection))

{

command.Parameters.AddWithValue("@ID", idToDelete);

command.ExecuteNonQuery();

}

}

LoadTableData(selectedTable);

}

else

{

MessageBox.Show("Пожалуйста, выберите строку для удаления.");

}

}

private void button2\_Click(object sender, EventArgs e)

{

if (!string.IsNullOrEmpty(selectedTable))

{

DataTable dataTable = (DataTable)dataGridView1.DataSource;

DataTable changes = dataTable.GetChanges();

if (changes != null)

{

SaveChangesToDatabase(selectedTable, changes);

}

}

else

{

MessageBox.Show("Пожалуйста, выберите таблицу для сохранения изменений.");

}

}

private void SaveChangesToDatabase(string tableName, DataTable changes)

{

using (SqlConnection connection = new SqlConnection(ConnectionString))

{

connection.Open();

using (SqlDataAdapter adapter = new SqlDataAdapter($"SELECT \* FROM {tableName}", connection))

{

using (SqlCommandBuilder commandBuilder = new SqlCommandBuilder(adapter))

{

adapter.Update(changes);

}

}

}

LoadTableData(tableName);

}

}

}

Листинг кода формы “Менеджер”:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using static Деятельность\_гостинцы.GuestForm;

using System.Data.SqlClient;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

using System.Globalization;

using QRCoder;

using System.Runtime.Remoting.Messaging;

namespace Деятельность\_гостинцы

{

public partial class ManagerForm : Form

{

private static GuestForm guestFormInstance;

private byte[] imageData;

private int currentReceiptNumber = 1;

public ManagerForm(GuestForm guestForm)

{

InitializeComponent();

guestFormInstance = guestForm;

textBox14.Text = currentReceiptNumber.ToString("D4");

maskedTextBox1.Mask = "9999-00-00";

maskedTextBox2.Mask = "9999-00-00";

maskedTextBox3.Mask = "9999-00-00";

maskedTextBox4.Mask = "9999-00-00";

}

private void ManagerForm\_Load(object sender, EventArgs e)

{

timer1.Interval = 1000;

timer1.Tick += timer1\_Tick;

timer1.Start();

GuestForm.ReservationRequested += GuestForm\_ReservationRequested;

}

private void GuestForm\_ReservationRequested(object sender, GuestForm.ReservationEventArgs e)

{

HandleReservationData(e.RoomNumber, e.PhoneNumber);

}

private void HandleReservationData(int roomNumber, string phoneNumber)

{

textBox1.Text = roomNumber.ToString();

textBox2.Text = phoneNumber;

}

private void timer1\_Tick(object sender, EventArgs e)

{

string filePath = "ReservationData.txt";

if (File.Exists(filePath))

{

string[] lines = File.ReadAllLines(filePath);

if (lines.Length > 0)

{

string[] data = lines[0].Split(',');

int roomNumber = int.Parse(data[0]);

string phoneNumber = data[1];

HandleReservationData(roomNumber, phoneNumber);

File.Delete(filePath);

}

}

}

private void button1\_Click(object sender, EventArgs e)

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Image Files |\*.jpg;\*.jpeg;\*.png;\*.gif;\*.bmp";

pictureBox1.SizeMode = PictureBoxSizeMode.Zoom;

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

string imagePath = openFileDialog.FileName;

pictureBox1.Image = Image.FromFile(imagePath);

imageData = File.ReadAllBytes(imagePath);

}

}

private void button2\_Click(object sender, EventArgs e)

{

if (ValidateInputs())

{

SaveGuestToDatabase();

MessageBox.Show("Данные успешно сохранены!");

}

else

{

MessageBox.Show("Пожалуйста, заполните все обязательные поля.");

}

}

private bool ValidateInputs()

{

return !string.IsNullOrEmpty(textBox3.Text)

&& !string.IsNullOrEmpty(textBox2.Text)

&& !string.IsNullOrEmpty(textBox4.Text)

&& !string.IsNullOrEmpty(textBox5.Text)

&& !string.IsNullOrEmpty(maskedTextBox2.Text)

&& !string.IsNullOrEmpty(textBox7.Text)

&& !string.IsNullOrEmpty(textBox8.Text)

&& !string.IsNullOrEmpty(textBox9.Text)

&& !string.IsNullOrEmpty(textBox10.Text)

&& !string.IsNullOrEmpty(maskedTextBox1.Text)

&& !string.IsNullOrEmpty(textBox12.Text)

&& !string.IsNullOrEmpty(textBox13.Text);

}

private void SaveGuestToDatabase()

{

string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True";

string query = @"

INSERT INTO Guests

(LastName, FirstName, MiddleName, Address, Town, Aim, Passport, PassportDate, Region, Work, BirthYear, Picture, PhoneNumber)

VALUES

(@LastName, @FirstName, @MiddleName, @Address, @Town, @Aim, @Passport, @PassportDate, @Region, @Work, @BirthYear, @Picture, @PhoneNumber)";

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand command = new SqlCommand(query, connection))

{

connection.Open();

command.Parameters.AddWithValue("@LastName", textBox3.Text);

command.Parameters.AddWithValue("@FirstName", textBox4.Text);

command.Parameters.AddWithValue("@MiddleName", textBox5.Text);

command.Parameters.AddWithValue("@Address", textBox8.Text);

command.Parameters.AddWithValue("@Town", textBox7.Text);

command.Parameters.AddWithValue("@Aim", textBox9.Text);

command.Parameters.AddWithValue("@Passport", textBox10.Text);

DateTime passportDate;

if (DateTime.TryParseExact(maskedTextBox1.Text, "yyyy-MM-dd", CultureInfo.InvariantCulture, DateTimeStyles.None, out passportDate))

{

command.Parameters.AddWithValue("@PassportDate", passportDate);

}

else

{

MessageBox.Show("Ошибка ввода даты паспорта.");

return;

}

command.Parameters.AddWithValue("@Region", textBox12.Text);

command.Parameters.AddWithValue("@Work", textBox13.Text);

DateTime BirthYear;

if (DateTime.TryParseExact(maskedTextBox2.Text, "yyyy-MM-dd", CultureInfo.InvariantCulture, DateTimeStyles.None, out BirthYear))

{

command.Parameters.AddWithValue("@BirthYear", BirthYear);

}

else

{

MessageBox.Show("Ошибка ввода даты рождения.");

return;

}

SqlParameter pictureParameter = new SqlParameter("@Picture", SqlDbType.VarBinary);

if (imageData != null && imageData.Length > 0)

{

pictureParameter.Value = imageData;

}

else

{

pictureParameter.Value = DBNull.Value;

}

command.Parameters.Add(pictureParameter);

command.Parameters.AddWithValue("@PhoneNumber", textBox2.Text);

command.ExecuteNonQuery();

}

}

}

private void button3\_Click(object sender, EventArgs e)

{

if (ValidateReservationInputs())

{

MakeReservation();

MessageBox.Show("Бронирование успешно завершено!");

}

else

{

MessageBox.Show("Пожалуйста, заполните все обязательные поля.");

}

}

private bool ValidateReservationInputs()

{

return !string.IsNullOrEmpty(textBox3.Text)

&& !string.IsNullOrEmpty(textBox4.Text)

&& !string.IsNullOrEmpty(textBox5.Text)

&& !string.IsNullOrEmpty(maskedTextBox3.Text)

&& !string.IsNullOrEmpty(maskedTextBox4.Text)

&& !string.IsNullOrEmpty(textBox15.Text)

&& !string.IsNullOrEmpty(textBox14.Text)

&& !string.IsNullOrEmpty(textBox16.Text)

&& !string.IsNullOrEmpty(textBox17.Text);

}

private void MakeReservation()

{

int selectedRoomNumber = int.Parse(textBox1.Text);

string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True";

string query = @"

INSERT INTO Reservations

(RoomNumber, LastName, FirstName, MiddleName, Come, Leave, Money, Cash, Receipt, Comment, Registrar)

VALUES

(@RoomNumber, @LastName, @FirstName, @MiddleName, @Come, @Leave, @Money, @Cash, @Receipt, @Comment, @Registrar)";

using (SqlConnection connection = new SqlConnection(connectionString))

{

using (SqlCommand command = new SqlCommand(query, connection))

{

connection.Open();

command.Parameters.AddWithValue("@RoomNumber", selectedRoomNumber);

command.Parameters.AddWithValue("@LastName", textBox3.Text);

command.Parameters.AddWithValue("@FirstName", textBox4.Text);

command.Parameters.AddWithValue("@MiddleName", textBox5.Text);

DateTime ComeDate;

if (DateTime.TryParseExact(maskedTextBox3.Text, "yyyy-MM-dd", CultureInfo.InvariantCulture, DateTimeStyles.None, out ComeDate))

{

command.Parameters.AddWithValue("@Come", ComeDate);

}

else

{

MessageBox.Show("Ошибка ввода даты.");

return;

}

DateTime LeaveDate;

if (DateTime.TryParseExact(maskedTextBox4.Text, "yyyy-MM-dd", CultureInfo.InvariantCulture, DateTimeStyles.None, out LeaveDate))

{

command.Parameters.AddWithValue("@Leave", LeaveDate);

}

else

{

MessageBox.Show("Ошибка ввода даты.");

return;

}

command.Parameters.AddWithValue("@Money", textBox15.Text);

command.Parameters.AddWithValue("@Cash", checkBox1.Checked);

command.Parameters.AddWithValue("@Receipt", textBox14.Text);

command.Parameters.AddWithValue("@Comment", textBox16.Text);

command.Parameters.AddWithValue("@Registrar", textBox17.Text);

command.ExecuteNonQuery();

guestFormInstance.RemoveReservedRoom(selectedRoomNumber);

currentReceiptNumber++;

textBox14.Text = currentReceiptNumber.ToString("D4");

}

}

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

}

}

Листинг кода формы “Гость”:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

using System.Collections;

using System.IO;

namespace Деятельность\_гостинцы

{

public partial class GuestForm : Form

{

public static event EventHandler<ReservationEventArgs> ReservationRequested;

string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;Initial Catalog=HotelManagement;Integrated Security=True;";

private string query = "SELECT Number, Rooms, Storey, TV, Fridge, Bed, Type, Balcony FROM Rooms WHERE 1=1 ";

public GuestForm()

{

InitializeComponent();

}

private void GuestForm\_Load(object sender, EventArgs e)

{

dataGridView1.ReadOnly = true;

this.roomsTableAdapter.Fill(this.dataSet2.Rooms);

foreach (DataGridViewRow row in dataGridView1.Rows)

{

int roomNumber = Convert.ToInt32(row.Cells[0].Value.ToString());

comboBox1.Items.Add(roomNumber);

maskedTextBox1.Mask = "+7 (999) 000-0000";

}

}

public void RemoveReservedRoom(int roomNumberToRemove)

{

if (dataGridView1.Columns.Contains("Number"))

{

DataGridViewColumn numberColumn = dataGridView1.Columns["Number"];

int columnIndex = numberColumn.Index;

foreach (DataGridViewRow row in dataGridView1.Rows)

{

if (row.Cells[columnIndex].Value != null && Convert.ToInt32(row.Cells[columnIndex].Value) == roomNumberToRemove)

{

dataGridView1.Rows.Remove(row);

UpdateComboBox(roomNumberToRemove);

break;

}

}

}

else

{

MessageBox.Show("Столбец 'Number' не найден в DataGridView.", "Ошибка удаления");

}

}

private void UpdateComboBox(int roomNumberToRemove)

{

if (comboBox1.InvokeRequired)

{

comboBox1.Invoke(new Action(() => UpdateComboBox(roomNumberToRemove)));

}

else

{

if (comboBox1.Items.Contains(roomNumberToRemove))

{

comboBox1.Items.Remove(roomNumberToRemove);

}

}

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

int rooms, storey, beds;

bool tv, fridge, balcony;

int.TryParse(textBox1.Text, out rooms);

int.TryParse(textBox2.Text, out storey);

int.TryParse(textBox3.Text, out beds);

tv = checkBox1.Checked;

fridge = checkBox2.Checked;

balcony = checkBox3.Checked;

if (rooms <= 0 && storey <= 0 && beds <= 0 && !tv && !fridge && !balcony)

{

string defaultQuery = "SELECT Number, Rooms, Storey, TV, Fridge, Bed, Type, Balcony FROM Rooms";

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand defaultCommand = new SqlCommand(defaultQuery, connection);

SqlDataAdapter defaultAdapter = new SqlDataAdapter(defaultCommand);

DataTable defaultDataTable = new DataTable();

defaultAdapter.Fill(defaultDataTable);

dataGridView1.DataSource = defaultDataTable;

}

}

else

{

string query = "SELECT Number, Rooms, Storey, TV, Fridge, Bed, Type, Balcony FROM Rooms WHERE 1=1 ";

if (storey > 0)

{

query += "AND Storey = @Storey ";

}

if (rooms > 0)

{

query += "AND Rooms = @Rooms ";

}

if (tv)

{

query += "AND TV = @TV ";

}

if (fridge)

{

query += "AND Fridge = @Fridge ";

}

if (beds > 0)

{

query += "AND Bed = @Beds ";

}

if (balcony)

{

query += "AND Balcony = @Balcony ";

}

query += "ORDER BY ";

if (storey > 0)

{

query += "Storey ASC, ";

}

if (rooms > 0)

{

query += "Rooms ASC, ";

}

if (tv)

{

query += "TV DESC, ";

}

if (fridge)

{

query += "Fridge DESC, ";

}

if (beds > 0)

{

query += "Bed ASC, ";

}

if (balcony)

{

query += "Balcony DESC, ";

}

query = query.TrimEnd(',', ' ');

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand(query, connection);

if (storey > 0)

{

command.Parameters.AddWithValue("@Storey", storey);

}

if (rooms > 0)

{

command.Parameters.AddWithValue("@Rooms", rooms);

}

if (tv)

{

command.Parameters.AddWithValue("@TV", tv);

}

if (fridge)

{

command.Parameters.AddWithValue("@Fridge", fridge);

}

if (beds > 0)

{

command.Parameters.AddWithValue("@Beds", beds);

}

if (balcony)

{

command.Parameters.AddWithValue("@Balcony", balcony);

}

SqlDataAdapter adapter = new SqlDataAdapter(command);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

dataGridView1.DataSource = dataTable;

}

}

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void button2\_Click(object sender, EventArgs e)

{

int selectedRoomNumber = Convert.ToInt32(comboBox1.SelectedItem);

string phoneNumber = maskedTextBox1.Text;

ReservationRequested?.Invoke(this, new ReservationEventArgs(selectedRoomNumber, phoneNumber));

SaveReservationData(selectedRoomNumber, phoneNumber);

}

protected virtual void OnReservationRequested(ReservationEventArgs e)

{

ReservationRequested?.Invoke(this, e);

}

public class ReservationEventArgs : EventArgs

{

public int RoomNumber { get; }

public string PhoneNumber { get; }

public ReservationEventArgs(int roomNumber, string phoneNumber)

{

RoomNumber = roomNumber;

PhoneNumber = phoneNumber;

}

}

private void SaveReservationData(int roomNumber, string phoneNumber)

{

string filePath = "ReservationData.txt";

using (StreamWriter writer = new StreamWriter(filePath))

{

writer.WriteLine($"{roomNumber},{phoneNumber}");

}

}

private void maskedTextBox1\_MaskInputRejected(object sender, MaskInputRejectedEventArgs e)

{

}

}

}