**Содержание**

|  |  |
| --- | --- |
|  | Стр. |
| Введение | 2 |
| 1. Общие сведения | 3 |
| 2. Назначение и цели разработки | 4 |
| 2.1.Назначения системы | 4 |
| 2.2.Цель системы | 4 |
| 3. Характеристика объектов автоматизации | 5 |
| 3.1. Краткое содержание об объекте автоматизации | 5 |
| 3.2. Сведения об условиях эксплуатации объектов автоматизации | 5 |
| 4. Требования к программе или программному изделию | 6 |
| 4.1. Функциональные требования | 6 |
| 4.2. Требования к надёжности | 6 |
| 4.3. Требования к условиям эксплуатации | 6 |
| 4.4. Требования к составу и параметрам технических средств | 7 |
| 4.5. Требования к информационной и программной совместимости | 7 |
| 4.6. Требование к маркировке и упаковке | 7 |
| 4.7. Требования к транспортировке и хранению | 7 |
| 5. Требования к технической документации | 8 |
| 5.1. Предварительный состав программной документации | 8 |
| 5.2. Специальные требования к программной документации | 8 |
| 6. Технико-экономические показатели | 9 |
| 7. Стадии и этапы разработки | 10 |
| 8. Порядок контроля и приемки | 12 |
| 9. Список использованных источников | 13 |

▎Техническое задание на разработку системы управления заявками на ремонт оргтехники

▎Введение

Полное наименование темы курсового проекта — «Разработка программного модуля для учета заявок на ремонт оргтехники». Разработчик программного продукта — студент группы 81/2021.

Актуальность данного проекта обусловлена необходимостью оптимизации процессов учета и обработки заявок на ремонт оргтехники в организациях. В условиях, когда количество используемой офисной техники постоянно растет, а требования к ее работоспособности становятся все более высокими, эффективное управление процессами ремонта и обслуживания становится критически важным. Система учета заявок, разработанная в рамках данного проекта, позволит автоматизировать процесс регистрации, обработки и мониторинга заявок на ремонт оргтехники, что значительно упростит работу сотрудников и повысит оперативность реагирования на возникающие проблемы.

В рамках проекта применены такие методологии исследования, как анализ, проектирование, кодирование и тестирование. Эти этапы обеспечивают системный подход к разработке программного обеспечения и позволяют создать надежный и функциональный продукт.

Основным назначением программного продукта является автоматизация учета заявок на ремонт оргтехники. Программное обеспечение будет иметь невысокие требования к характеристикам компьютера, что позволит использовать его на стандартных рабочих станциях без необходимости в дорогостоящем оборудовании.

Цель курсового проекта заключается в разработке программного модуля для учета заявок на ремонт оргтехники, который позволит упростить и ускорить выполнение рутинных задач, связанных с ремонтом и обслуживанием техники. Для достижения этой цели сформированы следующие задачи:

1. Изучить предметную область и определить ключевые процессы учета заявок на ремонт оргтехники.

2. Изучить спецификации и требования к функционалу системы.

3. Провести анализ существующих аналогичных программных решений.

4. Разработать проект системы с учетом выявленных требований.

5. Реализовать программное обеспечение согласно разработанному проекту.

6. Провести тестирование продукта для выявления и устранения возможных ошибок.

7. Подготовить сопроводительную документацию для пользователей и администраторов системы.

8. Внедрить систему в работу организации и провести обучение сотрудников.

Разработка данного программного модуля позволит значительно повысить эффективность работы службы технической поддержки, улучшить взаимодействие между сотрудниками и техническим персоналом, а также обеспечить надежное хранение данных о заявках на ремонт оргтехники и их статусах.

▎1. Общие сведения

Система предназначена для регистрации, обработки и отслеживания заявок на ремонт оргтехники. Она будет использоваться в организациях, где требуется быстрое и качественное управление заявками.

▎2. Назначение и цели разработки

▎2.1. Назначение системы

Система предназначена для автоматизации процессов управления заявками на ремонт оргтехники, повышения качества обслуживания клиентов и упрощения взаимодействия между пользователями.

▎2.2. Цель системы

Целью системы является создание удобного интерфейса для пользователей, позволяющего им регистрировать заявки, отслеживать их статус и получать отчеты о выполненных работах.

▎3. Характеристика объектов автоматизации

▎3.1. Краткое содержание об объекте автоматизации

Объектом автоматизации являются заявки на ремонт оргтехники, включая информацию о клиенте, типе оборудования, описании проблемы и статусе заявки.

▎3.2. Сведения об условиях эксплуатации объектов автоматизации

Система будет использоваться в офисных условиях с доступом к сети Интернет и локальной сети.

▎4. Требования к программе или программному изделию

▎4.1. Функциональные требования

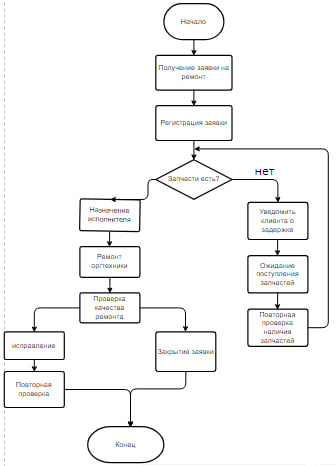
• Регистрация новых заявок.

• Возможность редактирования существующих заявок.

• Отслеживание статуса заявки (например: "Новая", "В работе", "Закрыта").

• Обработка заявки с выводом информации в отдельном окне.

• Формирование отчетов по выполненным заявкам.



▎4.2. Требования к надежности

Система должна обеспечивать сохранность данных о заявках и их статусах. В случае сбоя система должна восстанавливать данные до последнего сохраненного состояния.

▎4.3. Требования к условиям эксплуатации

Система должна функционировать на компьютерах с установленной ОС Windows и .NET Framework.

▎4.4. Требования к составу и параметрам технических средств

Необходимы стандартные ПК с минимальными параметрами:

• Процессор: не ниже 1 ГГц.

• Оперативная память: не менее 2 ГБ.

• Свободное место на диске: не менее 100 МБ.

▎4.5. Требования к информационной и программной совместимости

Система должна быть совместима с последними версиями Windows и поддерживать работу с базами данных MS SQL Server.

▎4.6. Требование к маркировке и упаковке

Документация должна быть оформлена согласно стандартам ГОСТ.

▎4.7. Требования к транспортировке и хранению

Программное обеспечение должно быть доступно для загрузки через интернет, а также иметь резервные копии для хранения на внешних носителях.

▎5. Требования к технической документации

▎5.1. Предварительный состав программной документации

• Руководство пользователя.

• Техническое описание системы.

• Документация по API (если необходимо).

▎5.2. Специальные требования к программной документации

Документация должна быть написана на русском языке и содержать примеры использования системы.

▎6. Технико-экономические показатели

Ожидается, что система сократит время обработки заявок на 30% и повысит удовлетворенность клиентов.

▎7. Стадии и этапы разработки

1. Анализ требований.

2. Проектирование системы.

3. Разработка интерфейса.

4. Реализация функционала.

5. Тестирование.

6. Внедрение и обучение пользователей.

▎8. Порядок контроля и приемки

Приемка будет проводиться поэтапно с тестированием каждой функциональной части системы.

▎9. Список использованных источников

• ГОСТ Р 19.001-96 "Единая система программной документации".

• ГОСТ Р 2.105-95 "Управление проектированием".

---

▎Пример реализации системы на C# (Windows Forms)

▎Шаги выполнения:

1. Создание проекта:

• Откройте Visual Studio.

• Создайте новый проект: Windows Forms App (.NET Framework).

2. Создание классов:

ChatGPT 4 | Midjourney | Claude | Suno, [16.12.2024 23:44]

• Создайте класс Request для представления заявки.

• Создайте класс RequestManager для управления заявками.

3. Создание форм:

• Создайте форму MainForm для отображения списка заявок и кнопок управления.

• Создайте форму RequestForm для добавления и редактирования заявки.

▎Код классов

// Request.cs

public class Request

{

public int Id { get; set; }

public string ClientName { get; set; }

public string EquipmentType { get; set; }

public string Description { get; set; }

public string Status { get; set; } // "New", "In Progress", "Closed"

}

// RequestManager.cs

using System.Collections.Generic;

public class RequestManager

{

private List<Request> requests = new List<Request>();

private int nextId = 1;

public void AddRequest(Request request)

{

request.Id = nextId++;

requests.Add(request);

}

public List<Request> GetRequests()

{

return requests;

}

public void UpdateRequest(Request request)

{

var existingRequest = requests.Find(r => r.Id == request.Id);

if (existingRequest != null)

{

existingRequest.ClientName = request.ClientName;

existingRequest.Description = request.Description;

existingRequest.Status = request.Status;

}

}

}

▎Код для форм

// MainForm.cs

using System;

using System.Windows.Forms;

public partial class MainForm : Form

{

private RequestManager requestManager = new RequestManager();

public MainForm()

{

InitializeComponent();

LoadRequests();

}

private void LoadRequests()

{

// Загрузка заявок в DataGridView (предполагается, что у вас есть DataGridView с именем dataGridViewRequests)

dataGridViewRequests.DataSource = requestManager.GetRequests();

}

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

{

var requestForm = new RequestForm(requestManager);

requestForm.ShowDialog();

LoadRequests();

}

}

// RequestForm.cs

using System;

using System.Windows.Forms;

public partial class RequestForm : Form

{

private RequestManager requestManager;

public RequestForm(RequestManager manager)

{

InitializeComponent();

requestManager = manager;

}

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

{

var request = new Request

{

ClientName = textBoxClientName.Text,

EquipmentType = textBoxEquipmentType.Text,

Description = textBoxDescription.Text,

Status = "New"

};

requestManager.AddRequest(request);

this.Close();

}

}

▎Код для Designer

// MainForm.Designer.cs

private void InitializeComponent()

{

this.dataGridViewRequests = new System.Windows.Forms.DataGridView();

this.buttonAddRequest = new System.Windows.Forms.Button();

((System.ComponentModel.ISupportInitialize)(this.dataGridViewRequests)).BeginInit();

this.SuspendLayout();

//

// dataGridViewRequests

//

this.dataGridViewRequests.ColumnHeadersHeightSizeMode = System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;

this.dataGridViewRequests.Location = new System.Drawing.Point(12, 12);

this.dataGridViewRequests.Name = "dataGridViewRequests";

this.dataGridViewRequests.Size = new System.Drawing.Size(776, 350);

//

// buttonAddRequest

//

this.buttonAddRequest.Location = new System.Drawing.Point(12, 368);

this.buttonAddRequest.Name = "buttonAddRequest";

this.buttonAddRequest.Size = new System.Drawing.Size(75, 23);

this.buttonAddRequest.Text = "Добавить заявку";

this.buttonAddRequest.Click += new System.EventHandler(this.buttonAddRequest\_Click);

//

// MainForm

//

this.ClientSize = new System.Drawing.Size(800, 450);

this.Controls.Add(this.dataGridViewRequests);

this.Controls.Add(this.buttonAddRequest);

this.Name = "MainForm";

}

// RequestForm.Designer.cs

private void InitializeComponent()

{

this.textBoxClientName = new System.Windows.Forms.TextBox();

this.textBoxEquipmentType = new System.Windows.Forms.TextBox();

this.textBoxDescription = new System.Windows.Forms.TextBox();

this.buttonSave = new System.Windows.Forms.Button();

//

// textBoxClientName

//

this.textBoxClientName.Location = new System.Drawing.Point(12, 12);

//

// textBoxEquipmentType

//

this.textBoxEquipmentType.Location = new System.Drawing.Point(12, 38);

//

// textBoxDescription

//

this.textBoxDescription.Location = new System.Drawing.Point(12, 64);

//

// buttonSave

//

this.buttonSave.Location = new System.Drawing.Point(12, 90);

this.buttonSave.Text = "Сохранить";

//

// RequestForm

//

this.ClientSize = new System.Drawing.Size(300, 150);

}

▎Блок-схема функции добавления заявки

Для создания блок-схемы можно использовать программное обеспечение для рисования (например, Microsoft Visio или онлайн-сервисы). Пример блок-схемы может включать следующие шаги:

1. Начало процесса.

2. Ввод данных заявки (Имя клиента, Тип оборудования, Описание).

3. Проверка корректности введенных данных.

4. Если данные корректны:

• Добавление заявки в систему.

• Сообщение об успешном добавлении.

5. Если данные некорректны:

• Сообщение об ошибке.

6. Завершение процесса.

Эта блок-схема может быть оформлена в соответствии с ГОСТ Р 2.105-95.

