МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

(ГУАП)

Кафедра 44 Вычислительных систем и сетей

(наименование)

ОТЧЁТ ПО ПРАКТИКЕ

ЗАЩИЩЁН С ОЦЕНКОЙ

Руководитель

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент, канд. техн. наук |  |  |  | Н. В. Кучин |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

ОТЧЁТ ПО ПРАКТИКЕ

|  |  |  |
| --- | --- | --- |
| вид практики | производственная | |
| тип практики | вычислительная | |
| на тему индивидуального задания | | Автоматизированная информационная система |
| первичной профсоюзной организации студентов и аспирантов ГУАП. Клиентская часть. | | | |
|  | | | |

|  |  |
| --- | --- |
| выполнен | Горбуновым Никитой Сергеевичем |
| фамилия, имя, отчество обучающегося в творительном падеже | |

|  |  |  |  |
| --- | --- | --- | --- |
| по направлению подготовки | 09.03.01 |  | Информатика и вычислительная техника |
|  | код |  | наименование направления |
|  | | | |
| наименование направления | | | |
| направленности | 02 |  | Вычислительные машины, комплексы, |
|  | код |  | наименование направленности |
| системы и сети | | | |
| наименование направленности | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Обучающийся группы № | 4941 |  | 09.04.2023 |  | Н.С. Горбунов |
|  | номер |  | подпись, дата |  | инициалы, фамилия |

Санкт–Петербург 2023

Содержание

[ВВЕДЕНИЕ 3](#_Toc131946723)

[1. Аналоги и сравнение 5](#_Toc131946724)

[2. Техническое задание 6](#_Toc131946725)

[2.1. Работа с данными студентов 6](#_Toc131946726)

[2.2. Пользователи 6](#_Toc131946727)

[2.3. Ролевая модель 7](#_Toc131946728)

[2.4. Статистика 7](#_Toc131946729)

[2.5. Генератор документов 8](#_Toc131946730)

[3. Используемые технологии 9](#_Toc131946731)

[3.1. C# 9](#_Toc131946732)

[3.2. Фреймворк Avalonia 10](#_Toc131946733)

[3.3. Безопасность 11](#_Toc131946734)

[4. Проектирование системы 13](#_Toc131946735)

[5. Разработка системы 13](#_Toc131946736)

[5.3. Реализация ролевого разграничения 13](#_Toc131946737)

[6. Развертывание системы 13](#_Toc131946738)

[7. Демонстрация работы системы 13](#_Toc131946739)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 14](#_Toc131946740)

# ВВЕДЕНИЕ

Одно из популярных направления развития IT технологий – автоматизация информационных систем различных сфер деятельности человека. Потому как это позволяет сократить ручной труд, минимизировать ошибки и «человеческий фактор». Именно поэтому автоматизированная информационная система ППОСА ГУАП является актуальной разработкой в существующем положении.

Такая система позволит автоматизировать работу профсоюзной организации во многих аспектах: учет студентов, в том числе студентов, имеющих социальные льготы, выплата материальной помощи, предоставление бонусов, сбор членских взносов. Все это позволит профсоюзу эффективнее выполнять свою работу, исключая ошибки: как умышленные, так и случайные.

Также в число достоинств можно включить инструмент по автоматизированной генерации служебной документации, что значительно сократит труд членов профсоюза по работе как внутри: протоколы собраний, решения профсоюзного комитета; так и снаружи: служебные записки в бухгалтерию и т.д.

АИС также обеспечивает более высокую точность обработки информации и улучшенный контроль за финансовыми потоками. Данная система также позволяет повысить качество обслуживания членов профсоюзной организации, предоставляя им доступ к необходимой информации и услугам.

Кроме того, АИС также может повысить уровень прозрачности и демократичности профсоюзной организации, обеспечивая членам доступ к информации о работе организации и возможность участия в принятии решений.

В результате, данная бакалаврская выпускная работа имеет большое практическое значение, так как позволяет проанализировать эффективность использования АИС в управлении профсоюзной организацией и предложить рекомендации по ее дальнейшему совершенствованию.

# Аналоги и сравнение

В данный момент сложно найти ПО, узкоспециализированое для профсоюзных организаций, однако существуют автоматизированные системы широкого профиля.

Системы управления взаимоотношениями с клиентами (CRM-системы). Они позволяют собирать, анализировать и управлять информацией о членах профсоюза, их потребностях и предпочтениях, что является важным элементом управления отношениями с членами профсоюза и повышения качества обслуживания.

Использование CRM системы в рамках автоматизированной системы для профсоюзной организации позволяет автоматизировать процессы взаимодействия с членами профсоюза, такие как обработка заявок, учет участия в мероприятиях, а также организация рассылок и оповещений. CRM системы позволяют вести учет контактной информации о членах профсоюза, информации об их предпочтениях и интересах, истории взаимодействия и другой полезной информации.

Кроме того, CRM системы обеспечивают возможность анализа данных и формирования отчетов о деятельности профсоюзной организации, что позволяет управлять эффективностью работы организации, определять наиболее перспективные направления развития и принимать обоснованные решения на основе данных.

# Техническое задание

## Работа с данными студентов

Система должна хранить информацию о студентах состоящих в профсоюзе, а именно следующие данные:

* ФИО
* Группа
* Номер телефона
* Электронная почта
* Номер профсоюзного билета
* Дата вступления
* Дата выхода
* Наличие социальных льгот (с датой начала и окончания действия)
* Наличие стипендии
* Должность в профсоюзе
* Должность в группе
* Бонусы, которыми пользовался студент

Следующие данные должны храниться за каждый семестр:

* Получение матпомощи
* Продление профсоюзного билета
* Наличие социальных льгот
* Наличие стипендии

Необходимо разработать удобный интерфейс для работы с данными в клиентском приложении.

## Пользователи

Для работы с системой нужно иметь учетную запись пользователя и авторизоваться с помощью неё.

Система должна хранить следующую информацию о пользователе:

* Логин
* Пароль
* Электронная почта
* Информацию о студенте, которому принадлежит данная запись (необязательно)
* Дата последнего входа в систему
* Дата создания записи
* Дата окончания действия записи

## Ролевая модель

Для разграничения прав в системе должна присутствовать ролевая модель. Она должна позволять гибко изменять доступный пользователю функционал. Права доступа задает администратор системы.

В системе должны присутствовать следующие базовые права:

* Доступ к просмотру данных студентов (разделение по институтам/факультетам)
* Доступ к изменению данных студентов (разделение по институтам/факультетам)
* Доступ к журналу действий
* Создание пользователей

Предполагается, что автоматизированные рабочие места (далее АРМ) будут установлены в офисах профсоюзной организации. Поэтому необходимо разработать единое приложение с изменяемым интерфейсом согласно доступным действиям для текущего пользователя. Таким образом реализуется возможность ограничения доступа к функционалу на уровне клиентского приложения.

## Статистика

Для контроля работы профсоюза необходимо собирать статистику по необходимым критериям, при этом предоставляя форматированные отчеты:

* Анализ вступления/выхода из профсоюза
* Анализ по бонусам, которыми пользовались студенты
* Анализ по продлению профсоюзных билетов

## Генератор документов

Клиентское приложение должно генерировать документы по запросу пользователя, например: служебная записка для бухгалтерии, или отчет для вышестоящей профсоюзной организации.

# Используемые технологии

Для разработки клиентского приложения выбран язык C# с использованием фреймворка Avalonia.

## C#

C# — это кроссплатформенный объектно-ориентированный язык программирования, созданный компанией Microsoft, а теперь переведен в Open Source[1].

Одной из главных особенностей C# является его способность к кроссплатформенной разработке. Это означает, что один и тот же код C# может быть скомпилирован и запущен на различных операционных системах, не требуя дополнительной модификации или переписывания.

Эта особенность была достигнута благодаря платформе .NET, которая предоставляет общую среду выполнения для всех языков .NET, в том числе C#. Среда выполнения, называемая Common Language Runtime (CLR), обеспечивает взаимодействие между кодом и операционной системой, а также управление памятью, безопасностью и другими аспектами выполнения приложений.

C# также поддерживает многопоточность, что делает его удобным для создания высокопроизводительных приложений. Он обладает широким набором библиотек и инструментов, которые упрощают создание и отладку приложений, а также обеспечивают доступ к ресурсам операционной системы, таким как файловая система, базы данных и сетевые соединения.

В целом, C# является мощным языком программирования, который позволяет создавать приложения для различных платформ и операционных систем. Его простой синтаксис и мощные возможности делают его популярным выбором для разработчиков, работающих в различных областях, от мобильной разработки до создания веб-сайтов и игр. Благодаря своей кроссплатформенной природе, C# является языком, который остается актуальным и перспективным в настоящее время.

## Фреймворк Avalonia

Avalonia[2] — это кроссплатформенный фреймворк для создания графических интерфейсов пользователя на языке C#. Он предназначен для разработки приложений, которые могут быть запущены на различных платформах, таких как Windows, macOS и Linux.

Основным преимуществом Avalonia является его способность к созданию нативных приложений с использованием единой кодовой базы. Он использует собственный визуальный язык, называемый XAML (Extensible Application Markup Language), который обеспечивает быстрое создание интерфейсов, а также расширяемость и гибкость в проектировании пользовательских элементов управления.

Avalonia также обеспечивает богатую функциональность, такую как многопоточность, анимации, события и другие возможности для удобной и эффективной разработки пользовательских интерфейсов. Он поддерживает различные стили оформления, которые могут быть применены к приложению, а также множество готовых элементов управления, которые можно использовать для создания интерфейсов.

Фреймворк Avalonia был создан для обеспечения простоты использования и быстрой разработки приложений. Его кроссплатформенность позволяет разработчикам создавать приложения для различных операционных систем, используя единую кодовую базу. Он также обеспечивает высокую производительность и эффективность, что делает его привлекательным выбором для создания современных приложений с богатыми пользовательскими интерфейсами.

В целом, Avalonia является кроссплатформенным фреймворком, который обеспечивает быструю разработку пользовательских интерфейсов для приложений на C#. Он предоставляет множество возможностей для создания современных приложений с богатыми пользовательскими интерфейсами, а его кроссплатформенность позволяет разработчикам создавать приложения, которые могут быть запущены на различных операционных системах.

## Безопасность

Передача данных между серверной и клиентской частями будет реализовано по протоколу HTTPS.

HTTPS (Hypertext Transfer Protocol Secure) - это протокол для передачи данных в интернете с использованием шифрования, который обеспечивает безопасность и конфиденциальность передаваемых данных. HTTPS использует SSL / TLS (Secure Sockets Layer / Transport Layer Security) протоколы для шифрования информации перед ее отправкой по сети.

Ключевой механизм безопасности HTTPS - это шифрование данных, которое происходит на стороне клиента (браузера) и сервера, перед тем как данные будут отправлены в интернет. Для этого используется криптографический алгоритм, который защищает данные от несанкционированного доступа и перехвата.

HTTPS также обеспечивает проверку подлинности и целостности данных, что делает его еще более безопасным. Проверка подлинности данных происходит путем проверки, что отправитель является действительным, а не злоумышленником, который пытается подменить данные. Проверка целостности данных гарантирует, что данные не были изменены в процессе передачи.

Другим важным механизмом безопасности HTTPS является цифровая подпись. Она используется для подтверждения подлинности источника данных и защиты от внесения изменений в передаваемые данные. Цифровая подпись создается путем шифрования данных, используя приватный ключ, который только у отправителя. Получатель может расшифровать цифровую подпись, используя открытый ключ отправителя и убедиться, что данные были отправлены именно этим отправителем.

Для идентификации пользователя в системе будут использованы механизмы авторизации и аутентификации на основе JWT токена. JWT токен[3] состоит из трех частей: заголовка, полезной нагрузки и подписи. Заголовок содержит информацию о типе токена и используемом алгоритме шифрования, полезная нагрузка - данные, которые нужно передать, а подпись - защищает токен от несанкционированного доступа.

С точки зрения безопасности, JWT токен имеет несколько преимуществ. Один из них — это невозможность изменения данных в токене. При создании токена, полезная нагрузка включает в себя информацию, которая не может быть изменена. При проверке подписи, сервер может убедиться, что данные не были изменены в процессе передачи.

Еще одним преимуществом JWT токена является возможность аутентификации пользователей без необходимости хранения информации о сессии на сервере. Вместо этого, токен содержит всю необходимую информацию о пользователе, которая может быть проверена без запроса к серверу. Это уменьшает нагрузку на сервер и увеличивает производительность.

Также стоит отметить, что JWT токен может быть защищен от подмены или изменения данных, путем использования секретного ключа для подписи токена. Ключ хранится на сервере, и только сервер имеет доступ к нему. Это обеспечивает надежную защиту от несанкционированного доступа и гарантирует, что токен был создан именно сервером.

Для исключения атаки через перехват JWT токена использован механизм его обновления, так называемый Refresh Token. Его смысл заключается в использовании токена сессии, как основного, предоставляющего доступ в систему, но имеющего короткий срок валидности. После окончания этого периода клиенту необходимо получить новый, передавая Refresh Token. Сервер контролирует использование Refresh Token, позволяя использовать его лишь единожды. Таким образом, если один и тот же токен обновления используется повторно, то сервер предполагает попытку атаки и блокирует оба токена, заставляя реального клиента повторно пройти аутентификацию через отправку логина и пароля, а злоумышленника без валидных токенов сессии и обновления.

# Проектирование системы

* 1. **Разработка и описание API для взаимодействия с сервером**
  2. **Разработка дизайна графического интерфейса клиентского приложения**
  3. **Разработка системы распределения прав и ролей в системе**
  4. **Разработка обновления приложения**

# Разработка системы

* 1. **Реализация API**
  2. **Реализация графического интерфейса**

## Реализация ролевого разграничения

* 1. **Реализация генерации документов**
  2. **Реализация обновления системы**

# Развертывание системы

# Демонстрация работы системы

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. The .NET Compiler Platform: [Электронный ресурс]. URL: <https://github.com/dotnet/roslyn> (Дата обращения: 09.04.2023)
2. ООП в картинках: [Электронный ресурс]. URL: <https://github.com/AvaloniaUI/Avalonia> (Дата обращения: 09.04.2023)
3. Пять простых шагов для понимания JSON Web Tokens (JWT): [Электронный ресурс]. // Хабр URL: <https://habr.com/ru/post/340146/> (Дата обращения: 09.04.2023)