

УДК 004

СИСТЕМА КООРДИНАЦИИ ВЗАИМОДЕЙСТВИЯ СЕРВИСОВ РАЗРАБОТКИ ПРОГРАММНЫХ ПРИЛОЖЕНИЙ

Кудрявцев Д.С., магистрант гр. ПИМ-231, II курс,
Ларин Н.М., магистрант гр. ПИМ-231, II курс
Научный руководитель: Пимонов А.Г., д.т.н., профессор
Кузбасский государственный технический университет
имени Т.Ф. Горбачева, г. Кемерово

Введение

Современные предприятия активно внедряют разнообразные сервисы для разработки программных приложений (GitLab [1], Passwork [2], Active Directory [3]), что приводит к фрагментации управления доступом. Отсутствие централизованной координации увеличивает риски утечек данных, усложняет аудит прав и требует значительных ресурсов для администрирования.

Предположим, имеется диджитал-агентство, использующее в своей инфраструктуре Active Directory, GitLab и веб-сервис для хранения паролей Passwork. Выявлена необходимость синхронизации пользовательских данных между этими сервисами. Решением этой задачи будет создание универсальной системы, обеспечивающей следующие возможности:

- 1) централизованное управление вышеперечисленными сервисами;
- 2) гибкое управление группами и пользователями;
- 3) интеграцию с локальными и облачными инструментами.

Актуальность работы подтверждается растущим спросом на гибридные системы, сочетающие преимущества локальных и облачных решений в условиях цифровой трансформации предприятий.

Цель исследования – разработка системы координации сервисов на основе микросервисной архитектуры [7] с централизованным управлением пользователями и их правами, интеграцией с Active Directory и Passwork, а также масштабируемостью для новых инструментов. Для достижения этой цели были поставлены такие задачи, как проектирование архитектуры с использованием Next.js [4], Laravel [5] и Go [6], разработка модулей для управления пользователями и группами и интеграция с Passwork.

Научная новизна обусловлена следующими свойствами системы:

- 1) Гибридная архитектура. Предложено сочетание локальных (Active Directory) и облачных (Passwork) сервисов, а также использование микросервисов, что обеспечивает гибкость и снижение зависимости от внешних платформ.
- 2) Автоматизация синхронизации. Разработан алгоритм автоматического обновления прав пользователей при изменении членства в группах, что исключает ручное вмешательство.
- 3) Расширенная интеграция. Реализован модуль взаимодействия с Passwork, включающий генерацию паролей с настраиваемыми параметрами.

Проектирование системы

Система реализована на основе микросервисной архитектуры, что обеспечивает модульность, масштабируемость и отказоустойчивость. Основные компоненты и их взаимодействие представлены на рисунке 1.

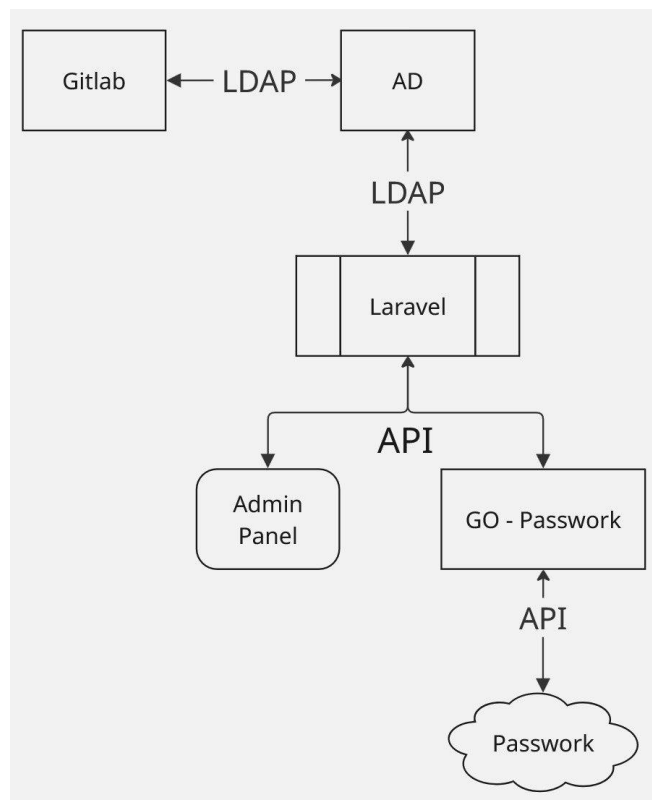


Рисунок 1 – Схема взаимодействия микросервисов

Описание компонентов:

1. Админ-панель (Frontend):

- Интерфейс для управления пользователями, группами и паролями из Password;
- Технологии: Next.js (SSR), React, UI Kit.

2. Сервис авторизации (Backend, Laravel):

- Обработывает аутентификацию, синхронизирует данные с Active Directory, реализует управление пользователями и группами;
- Технологии: Laravel (PHP), JWT-токены, PostgreSQL;
- Взаимодействие:
 - Получает данные пользователей из Active Directory с помощью LDAP [10];
 - Отправляет запросы в Go-Passwork для работы с паролями Password.

3. Go-Passwork:

- Обеспечивает интеграцию между хранилищем паролей Password и сервисом авторизации;
- Технологии: Go;

- Взаимодействие: Использует REST API [9] Passwork для управления паролями.

4. Active Directory:

- Централизованное хранилище пользователей и групп;
- Протокол: LDAP для синхронизации данных.

Управление пользователями

Модуль управления пользователями обеспечивает просмотр, создание, редактирование и удаление пользователей. На рисунке 2 представлен раздел для просмотра списка пользователей.

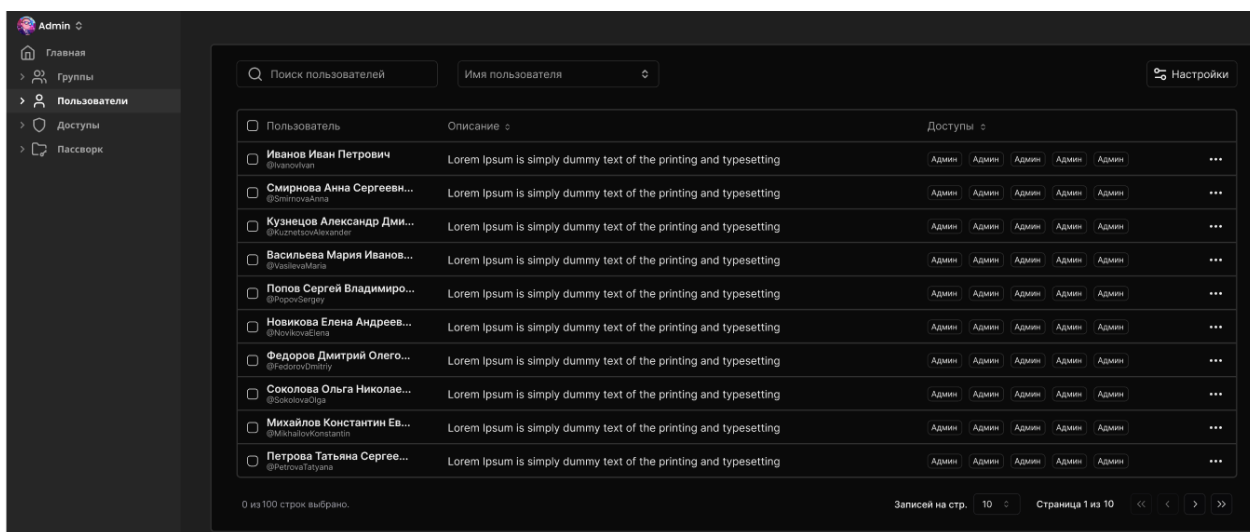


Рисунок 2 – Список пользователей

Управление группами

Модуль управления группами обеспечивает гибкое распределение прав доступа к корпоративным сервисам (рис. 3).

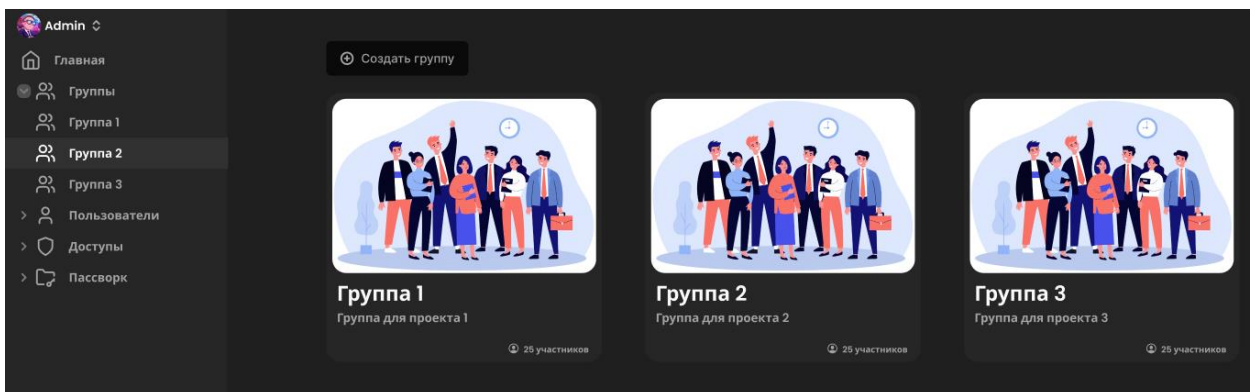


Рисунок 3 – Список групп

Его функционал включает в себя просмотр, создание, редактирование, удаление, а также выдачу доступов для групп, благодаря чему все пользователи внутри группы получают аналогичные доступы (рис. 4).

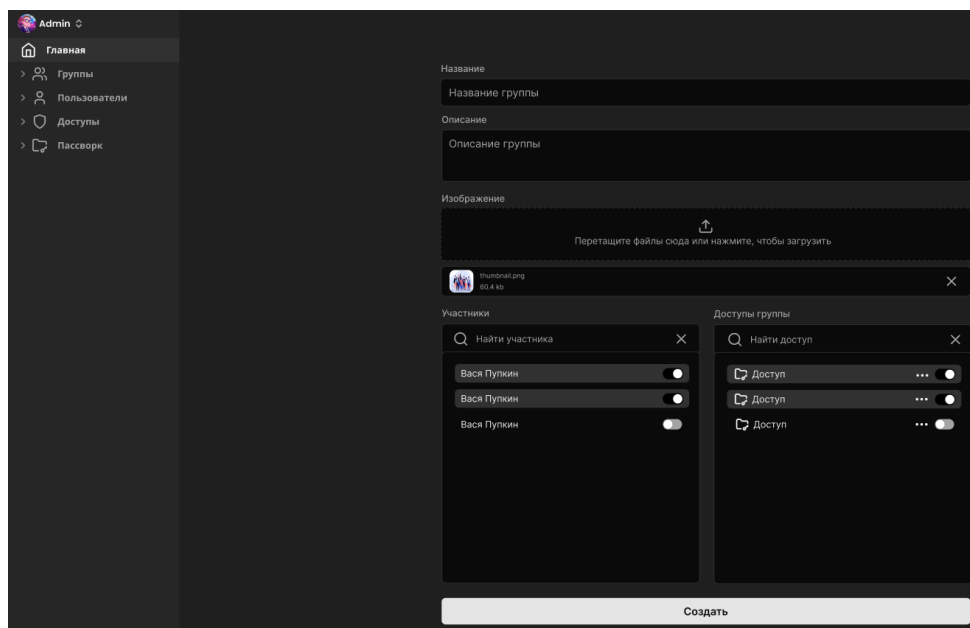


Рисунок 4 – Форма редактирования группы

Управление паролями

Модуль для управления паролями предназначен для работы с менеджером паролей Passwork и повторяет часть его функционала, например, просмотр, создание и генерацию паролей. Пароли отображаются в древовидной структуре, позволяющей работать с ними по аналогии с директориями и файлами в редакторах кода (рис. 5).

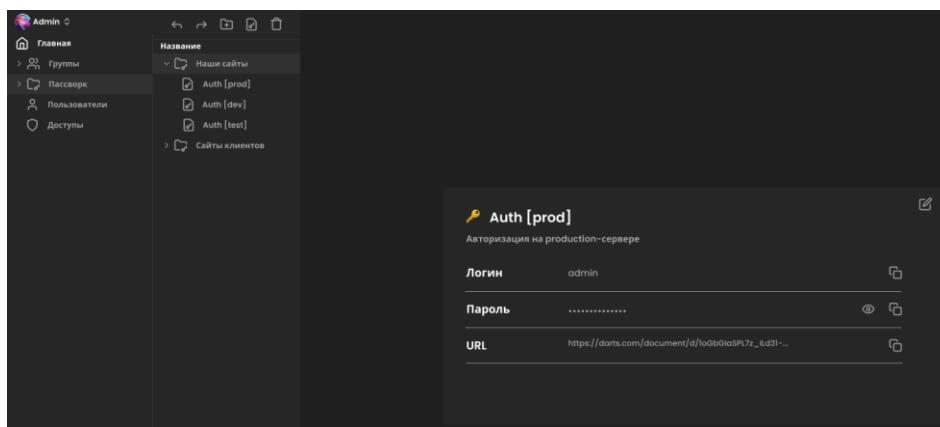


Рисунок 5 – Интерфейс модуля для управления паролями

Интеграция с сервисами

Модуль интеграции с внешними сервисами обеспечивает взаимодействие системы с GitLab и Passwork.

Интеграция GitLab с системой управления доступом осуществляется с помощью протокола LDAP, что позволяет синхронизировать пользователей и группы напрямую с Active Directory, благодаря чему группы автоматически отображаются в GitLab и права доступа к репозиториям назначаются на уровне групп. При изменении состава группы в Active Directory GitLab обновляет

доступы в течение 5 минут. Удаление пользователя из группы блокирует его доступ к репозиториям GitLab.

Для управления паролями используется REST API Password, что обеспечивает безопасное хранение и автоматизацию процессов.

Реализованные функции:

1. Создание паролей.

2. Поиск и копирование:

- Интеграция с админ-панелью позволяет искать пароли по названию или тегам.

- При копировании пароля в буфер обмена система журналирует операцию.

3. Автоматическая ротация:

- Пароли обновляются по расписанию (например, каждые 90 дней).

- Уведомления отправляются администратору за 7 дней до истечения срока.

На рисунке 6 представлена диаграмма последовательности, описывающая взаимодействие сервисов для работы с Password на примере действия «перенос директории».



Рисунок 6 – Взаимодействие с Password через API

Frontend

Клиентская часть системы реализована с помощью Next.js с использованием современных инструментов для создания адаптивных интерфейсов и динамических элементов. Клиентская часть демонстрирует интуитивно понятный интерфейс, обеспечивающий удобство работы с сервисом для администраторов.

Next.js выбран как основа фронтенда благодаря следующим преимуществам:

- CSR (Client-Side Rendering): динамическая загрузка контента на клиентской стороне с использованием состояний и эффектов.
- Компонентная архитектура: модульность и повторное использование компонентов для упрощения разработки.

- Адаптивность: использование CSS-модулей и медиазапросов для отображения на любых устройствах.

Сервис авторизации (Laravel, JWT)

Сервис авторизации отвечает за аутентификацию пользователей, реализует управление пользователями и группами и синхронизацию данных с Active Directory. Для реализации был использован фреймворк Laravel [5] и протокол JWT (JSON Web Token) [8], что обеспечивает безопасность и производительность.

Аутентификация через JWT

1. Генерация токенов: при успешной аутентификации система генерирует два токена:

- Access Token: срок действия – 15 минут. Содержит информацию о пользователе и правах.
- Refresh Token: срок действия – 7 дней. Используется для обновления Access Token.

2. Валидация токенов:

- Каждый запрос к API содержит Access Token в заголовке Authorization.
- Сервис проверяет подпись токена и его срок действия.

3. Интеграция с Active Directory: Синхронизация данных происходит через LDAP-запросы. Общая схема взаимодействия между сервисами приложения при авторизации представлена на рисунке 7.

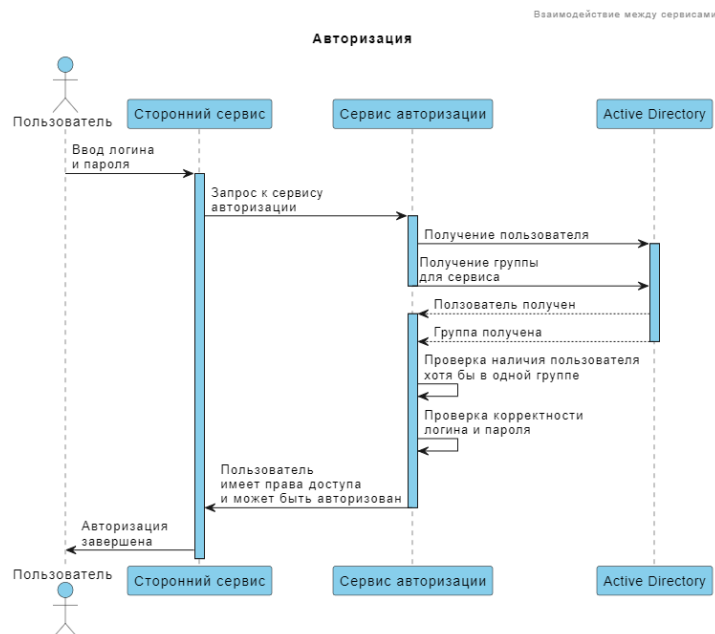


Рисунок 7 – Схема взаимодействия между сервисами приложения при авторизации

Go-Password

Go-Password реализован на языке Go и отвечает за взаимодействие между сервисом авторизации и API Password, включая шифрование данных и управление паролями. Сервис отправляет HTTPS-запросы к Password API с учётом работы с шифрованием для выполнения таких операций:

- добавление, редактирование и удаление паролей;
- копирование паролей;
- перенос пароля из директории в директорию.

Заключение

В рамках настоящей работы была создана система координации взаимодействия сервисов разработки программных приложений. В результате проведённого исследования были решены следующие задачи:

1) спроектирована архитектура на базе микросервисов с использованием Next.js, Laravel и Go;

2) разработаны модули для управления пользователями и группами;

3) выполнена интеграция с Password с помощью микросервиса на Go.

Разработанная система может быть применена в корпоративных средах для повышения эффективности управления ИТ-инфраструктурой.

Список литературы:

1. Документация GitLab. – URL: <https://about.gitlab.com/> (дата обращения: 21.03.2025).
2. Документация Password: [Электронный ресурс]. – Режим доступа: <https://password.app> (дата обращения: 21.03.2025).
3. Документация Microsoft Azure Active: Directory. – URL: <https://azure.microsoft.com/ru-ru/services/active-directory> (дата обращения: 21.03.2025).
4. Документация Next.js. – URL: <https://nextjs.org> (дата обращения: 21.03.2025).
5. Документация Laravel. – URL: <https://laravel.com> (дата обращения: 21.03.2025).
6. Документация Golang. – URL: <https://golang.org> (дата обращения: 21.03.2025).
7. Статья «Просто о микросервисах». – URL: <https://habr.com/ru/companies/raiffeisenbank/articles/346380> (дата обращения: 22.03.2025).
8. Официальный репозиторий Tymon JWT на GitHub. – URL: <https://github.com/tymon/jwt-auth> (дата обращения: 22.03.2025).
9. Статья «Что такое REST API?». – URL: <https://ru.hexlet.io/blog/posts/rest-api> (дата обращения: 29.03.2025).
10. Статья «Понимание LDAP-протокола, иерархии данных и компонентов записей». – URL: <https://habr.com/ru/articles/538662> (дата обращения: 29.03.2025).