

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Новгородский государственный университет имени Ярослава Мудрого»
Институт электронных и информационных систем

Кафедра «Информационных технологий и систем»

РАЗРАБОТКА КОРПОРАТИВНОГО ОБЛАЧНОГО ХРАНИЛИЩА ДАННЫХ

Техническое задание к выпускной квалификационной работе
по направлению 09.03.01 «Информатика и вычислительная техника»,
профиль «Программное обеспечение вычислительной техники и
автоматизированных систем»

НУОП.90001-01 90

Руководитель

_____ Соколова Г. Ю.

«____» _____ 2024 г.

Студент группы 0091

_____ Нечаев М. С.

«____» _____ 2024 г.

Великий Новгород, 2024 г.

1. Введение

В настоящее время технологии клиент-серверного взаимодействия являются одними из наиболее востребованных в сфере веб-разработки. Для реализации таких приложений используется целый ряд инструментов и технологий, включая MongoDB, ExpressJS, React и NodeJS.

В данном проекте я буду использовать эти инструменты для создания полноценного клиент-серверного приложения с серверной и клиентской частями. Моя цель – создать корпоративное облачное хранилище, которое будет работать по принципу Google Disk, то есть позволит пользователям хранить и обмениваться файлами в облаке.

Для реализации проекта я буду использовать MongoDB как базу данных, NodeJS и фреймворк ExpressJS для создания серверной части, React для создания клиентской части. Благодаря этим инструментам я смогу создать высокопроизводительное и масштабируемое приложение, которое будет удобно использовать и иметь широкий функционал.

2. Назначение и область применения

Назначение моего проекта заключается в создании корпоративного облачного хранилища, которое позволит пользователям сохранять и обмениваться файлами в облаке. С помощью данного приложения пользователи смогут легко хранить файлы в безопасности и иметь доступ к ним из любой точки мира.

Закрытое от публичного пространства облачное хранилище разработано специально для корпоративных клиентов и может быть успешно применено в различных секторах бизнеса, включая финансы, медицину, юриспруденцию и технологические компании.

3. Цель разработки

Основной целью разработки моего проекта является создание полноценного клиент-серверного приложения с серверной и клиентской частью, которое позволит пользователям хранить и обмениваться файлами в облаке.

4. Термины и сокращения

API - Application Programming Interface (Интерфейс программирования приложений)

CSS - Cascading Style Sheets (Каскадные таблицы стилей)

HTML - Hypertext Markup Language (Язык разметки гипертекста)

HTTP - Hypertext Transfer Protocol (Протокол передачи гипертекста)

JSON - JavaScript Object Notation (Формат обмена данными, основанный на синтаксисе объектов JavaScript)

MongoDB - NoSQL документоориентированная база данных

Node.js - JavaScript-среда выполнения на стороне сервера

React - Библиотека JavaScript для создания пользовательских интерфейсов

REST - Representational State Transfer (Архитектурный стиль взаимодействия клиента и сервера)

5. Требования к программе

5.1. Функциональные требования

- 5.1.1. Администратор должен иметь возможность в соответствии с данными в заявлении создать пользователя в системе, передав туда имя, фамилию и почту, а также реализовать пароль.
- 5.1.2. Администратор должен иметь возможность удалить пользователя из системы посредством передачи его почты в поле для удаления.
- 5.1.3. Администратор должен иметь возможность запросить все почты пользователей, зарегистрированных в системе.
- 5.1.4. Пользователь должен иметь возможность подать заявление на регистрацию аккаунта в системе.
- 5.1.5. Пользователь должен иметь возможность авторизоваться в системе, используя свои данные в виде логина (электронной почты) и пароля, в следствии он должен получить доступ к личному хранилищу.
- 5.1.6. Пользователь должен иметь возможность выхода из системы.
- 5.1.7. Пользователь должен иметь возможность загрузить файлы в систему.
- 5.1.8. Пользователь должен иметь возможность получения ссылки на скачивание файла.
- 5.1.9. Пользователь должен иметь возможность скачивания файлов на локальное устройство.
- 5.1.10. Пользователь должен иметь возможность создать директорию для организации пространства облачного хранилища.
- 5.1.11. Пользователь должен иметь возможность удалить как пустую директорию, так и с файлами и папками внутри.
- 5.1.12. Пользователь должен иметь возможность получения ссылки на скачивание директории в виде архива.

- 5.1.13. Пользователь должен иметь возможность искать файлы и папки, по ключевым словам, во всем облачном хранилище.
- 5.1.14. Пользователь должен иметь возможность наблюдать шкалу заполненного пространства в облачном хранилище, а также должна быть надпись сколько места занято и сколько в общем доступно.
- 5.1.15. Пользователь должен иметь доступ к общему пространству.
- 5.1.16. Пользователь должен иметь профиль, в котором будут отображены его имя, фамилия, аватарка и шкала занятого пространства, а также функционал для загрузки и удаления аватарки.
- 5.1.17. Приложение должно показывать загрузчик в виде модального окна в правом нижнем углу при загрузке файлов в систему с отображением у каждого файла процента загрузки и шкалы загрузки.

5.2. Требования к интерфейсу

5.2.1. Пользовательский интерфейс

- 5.2.2. Текст приложения, который предоставлен пользователю, должен быть написан на Кириллице.
- 5.2.3. На странице регистрации должна быть описана краткая инструкция по созданию аккаунта и приложено заявление на регистрацию в облачном хранилище (Рисунок А.1).
- 5.2.4. Пользователь корпоративной системы должен иметь возможность авторизоваться в облачном хранилище, используя логин (адрес электронной почты) и пароль (Рисунок А.2).
- 5.2.5. Интерфейс главной страницы должен иметь следующие возможности (Рисунок А.3):
 - 5.2.5.1. Должна быть реализована возможность выхода из системы.
 - 5.2.5.2. Должна быть реализована возможность перехода пользователя в общее пространство.
 - 5.2.5.3. Должна быть реализована поисковая строка, при помощи которой будет происходить поиск файла и папки по ключевой фразе в хранилище.
 - 5.2.5.4. Интерфейс файлов и папок должен иметь возможность переключения отображения в построчном виде или в виде плиток определенного размера (Рисунок А.4).

- 5.2.5.5. Должен быть реализован интерфейс для сортировки файлов и папок по имени, дате и типу.
- 5.2.5.6. Интерфейс должен предусматривать отображение занятого пользователем пространства в виде шкалы.
- 5.2.5.7. Должны быть отображены все файлы и папки пользователя, которые загружены в облачное хранилище.
- 5.2.5.8. Интерфейс должен иметь возможность перемещения по директориям, чтобы при нажатии на папку пользователь переходил в нее и получал все содержимое данной директории.
- 5.2.5.9. Должна быть реализована возможность выхода пользователя из вложенных папок при нажатии на кнопку «Назад».
- 5.2.5.10. Должно быть отображение веса и даты загрузки у каждого файла и папки.
- 5.2.5.11. Интерфейс папок должен предусматривать иконку на удаление и иконку для формирования ссылки на скачивание директории в виде архива.
- 5.2.5.12. Интерфейс файлов должен предусматривать иконку на удаление, скачивание файла на локальное устройство и для формирования ссылки на скачивание файла.
- 5.2.5.13. При наведении курсора на файл должны появляться иконки для загрузки и удаления файла, а также иконка для формирования ссылки на скачивание файла.
- 5.2.5.14. При наведении курсора на директорию должны появляться иконки для удаления папки и для формирования ссылки на скачивание директории в виде архива.
- 5.2.5.15. Должен быть реализован интерфейс загрузки файла в облачное хранилище при нажатии на кнопку «Загрузить файл» или перетаскивании файла в рабочую область и загрузки его при помощи «Drag'n Drop».
- 5.2.5.16. Должен быть реализован интерфейс загрузки файла с отображением процента загрузки и шкалы данной загрузки.
- 5.2.6. Интерфейс пользователя должен иметь возможность для загрузки и удаления аватарки и последующим ее отображением в системе, отображение имени, фамилии и занятого пространства пользователем в виде шкалы занятого пространства (Рисунок А.5).
- 5.2.7. Администратор системы должен иметь форму для создания пользователя с передаваемыми туда данными в виде имени, фамилии почты и пароля

пользователя, форму для удаления пользователя из системы посредством передачи в поле его почты и иметь функционал для запроса всех почт пользователей в системе (Рисунок А.6).

5.2.8. Интерфейс должен сменять тему на светлую или темную в зависимости от выбранной в браузере цветовой темы (Рисунок А.7).

5.2.9. Программный интерфейс

5.2.9.1. Должно быть разработано эффективное взаимодействие сервера с клиентом по архитектурному стилю REST API, основанному на стандартных HTTP-методах.

5.2.9.2. API должен предоставлять методы для манипуляции с файлами и папками, для файлов должно быть реализовано удаление, скачивание на локальный компьютер и получение ссылки на скачивание, для папок должно быть реализовано удаление и получение ссылки на скачивание директории в виде архива.

5.2.9.3. API должен предоставлять механизм аутентификации пользователей, а также проверку их на аутентификацию, для обеспечения безопасного доступа к данным.

5.2.9.4. API должен предоставлять возможность для загрузки или удаления аватарки пользователя.

5.2.9.5. API, в зависимости от запроса пользователя, должно возвращать данные в JSON формате, либо в виде файла или архива.

5.3. Требования к реализации

5.3.1. Реализация облачного хранилища с возможностью хранения, удаления и получения ссылок на файлы, создания и удаления папок, а также получения ссылки на скачивание директории в виде архива и с возможностью доступа в общее хранилище.

5.3.2. Программа должна использовать следующий стек технологий: React, MongoDB, Node.js и Express.

5.3.3. Реализация клиентской части происходит при помощи библиотеки React, JS, и стилей, написанных на CSS.

5.3.4. Реализация серверной части происходит при помощи Node.js и фреймворка Express.

- 5.3.5. Аутентификация пользователей с использованием технологии JSON Web Token.
- 5.3.6. Использование документоориентированной системы управления базы данных MongoDB, для хранения информации о пользователях и их файлах и папках.
- 5.3.7. Создание API для взаимодействия клиента с сервером по HTTP протоколу, клиент отправляет запросы на сервер, используя HTTP методы (GET, POST и DELETE), данные передаются в JSON формате.

5.4. Требования к окружению

5.4.1. Аппаратные требования

Минимальная конфигурация аппаратных средств для нормального функционирования программы должна содержать:

- 5.4.1.1. Процессор: 64-битный процессор с тактовой частотой не менее 2 ГГц.
- 5.4.1.2. Оперативная память: не менее 2 ГБ.
- 5.4.1.3. Свободное место на жестком диске: не менее 10 ГБ.
- 5.4.1.4. Клавиатура.
- 5.4.1.5. Манипулятор мышь.

5.4.2. Программные требования

- 5.4.2.1. Операционная система: Windows 7 или выше, macOS 10.10 или выше, Linux
- 5.4.2.2. Установленные браузеры Google 11.x или Yandex 22.x.
- 5.4.2.3. Node.js версии 20.x или выше.
- 5.4.2.4. MongoDB версии 6.x или выше.
- 5.4.2.5. Express.js версии 4.x или выше.
- 5.4.2.6. React версии 18.x или выше.

5.5. Требования к надёжности

- 5.5.1. Программа должна обрабатывать и отображать ошибки, возникающие в процессе работы.
- 5.5.2. Должны быть ограничения на вводимые данные.
- 5.5.3. При вводе неправильного логина или пароля сервер должен давать ответ об ошибке, а клиент должен выделять поля красным если пароль меньше трех символов или почта не подходит под ожидаемую маску ввода.

5.6. Требования к тестированию

- 5.6.1. Для тестирования должны быть разработаны тесты на контроль вводимых данных.
- 5.6.2. Необходимо разработать тест, который проверяет, что пользователь не сможет загрузить больше файлов, чем доступно места на диске.
- 5.6.3. Необходимо разработать тест, который будет проверять, что процесс загрузки файла работает корректно.
- 5.6.4. Необходимо разработать тест, который будет проверять, что процесс удаления файла работает корректно.

5.7. Требования к установке

- 5.7.1. Заказчику поставляется архив с кодом программы, ссылка на сайт и документы (пункт 5.10.1) на флешке.
- 5.7.2. Для успешной работы программы у пользователя должны быть установлены **Google 11.x** или **Yandex 22.x**.
- 5.7.3. Node.js версии 20.x или выше.
- 5.7.4. MongoDB версии 6.x или выше.
- 5.7.5. Express.js версии 4.x или выше.
- 5.7.6. React версии 18.x или выше.

5.8. Требования к безопасности

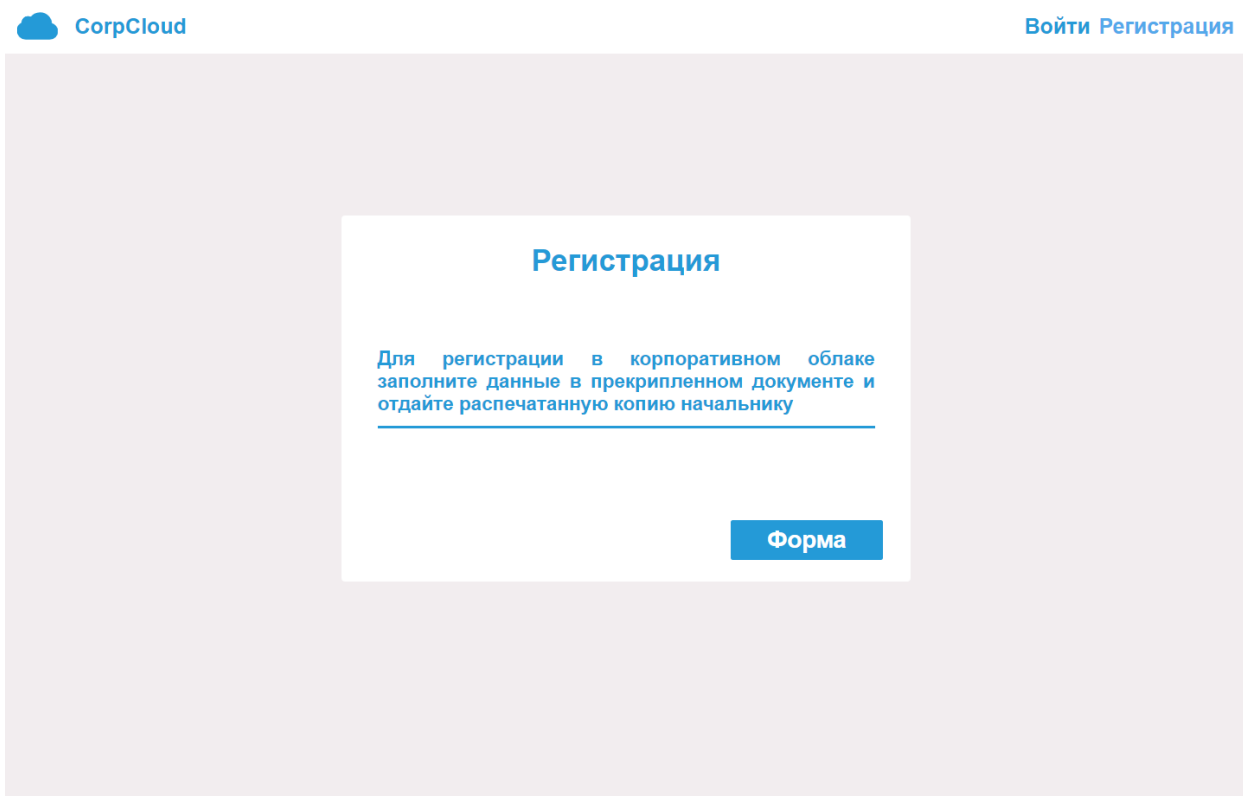
- 5.8.1. Ограничение доступа к критическим функциям программы только авторизованным пользователям.
- 5.8.2. Обеспечение конфиденциальности данных пользователей, таких как логины, пароли и другие личные данные.
- 5.8.3. При регистрации должен быть указан уникальный логин.
- 5.8.4. Должна быть учетная запись админа, через которую можно будет создавать, удалять пользователей и получать их почты.
- 5.8.5. Должно быть предусмотрено шифрование паролей в базе данных.

5.9. Требования к документации

- 5.9.1. По проекту должны быть разработаны следующие пользовательские документы:
 - Программа и методика испытаний

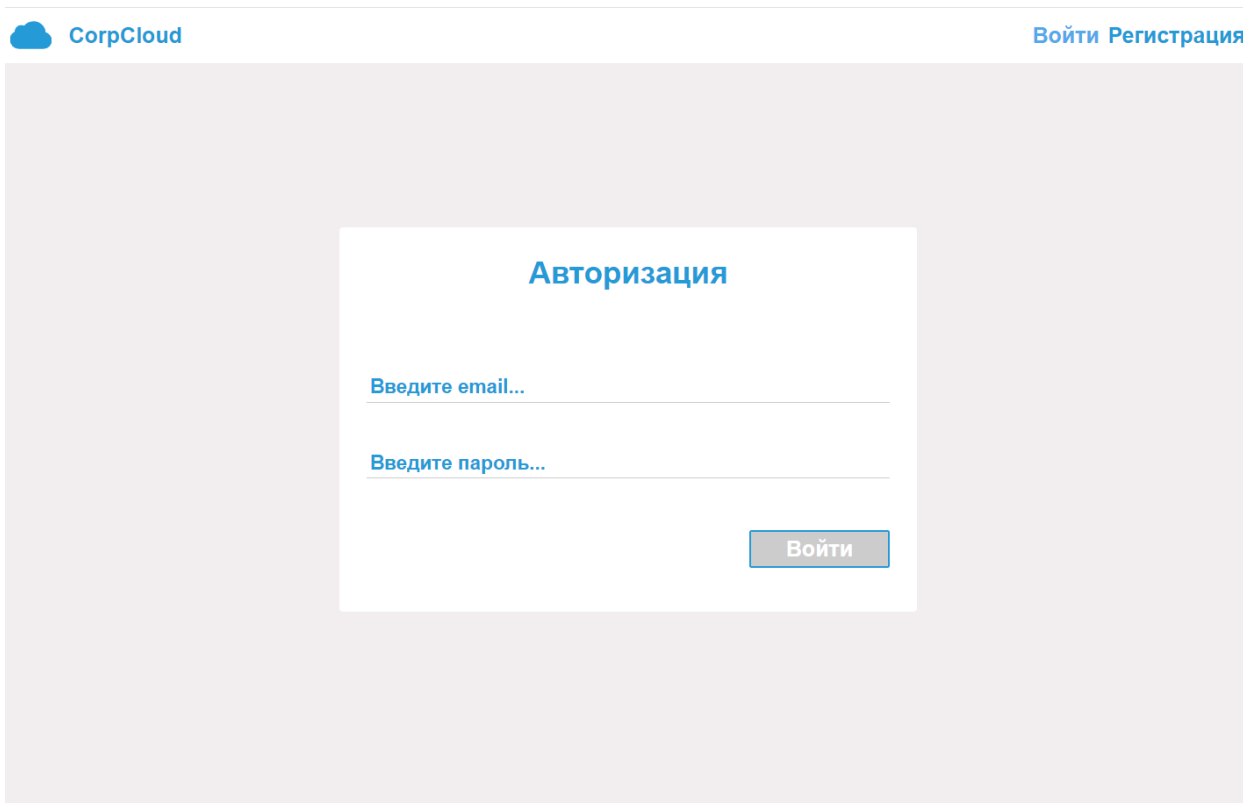
5.9.2. Вся документация представляется на русском языке в формате DOCX

5.10. Приложение А



The screenshot shows the registration page of the CorpCloud application. At the top left is the CorpCloud logo, and at the top right are links for 'Войти' (Login) and 'Регистрация' (Registration). The main content area has a light purple background. In the center is a white box titled 'Регистрация'. Inside this box, there is a blue heading 'Регистрация', followed by a paragraph of blue text: 'Для регистрации в корпоративном облаке заполните данные в прикрепленном документе и отдайте распечатанную копию начальнику'. Below this text is a blue button labeled 'Форма'.

Рисунок А.1 – страница регистрации



The screenshot shows the authorization page of the CorpCloud application. At the top left is the CorpCloud logo, and at the top right are links for 'Войти' (Login) and 'Регистрация' (Registration). The main content area has a light purple background. In the center is a white box titled 'Авторизация'. Inside this box, there are two input fields: the first is labeled 'Введите email...' and the second is labeled 'Введите пароль...'. Below these fields is a blue button labeled 'Войти'.

Рисунок А.2 – страница авторизации

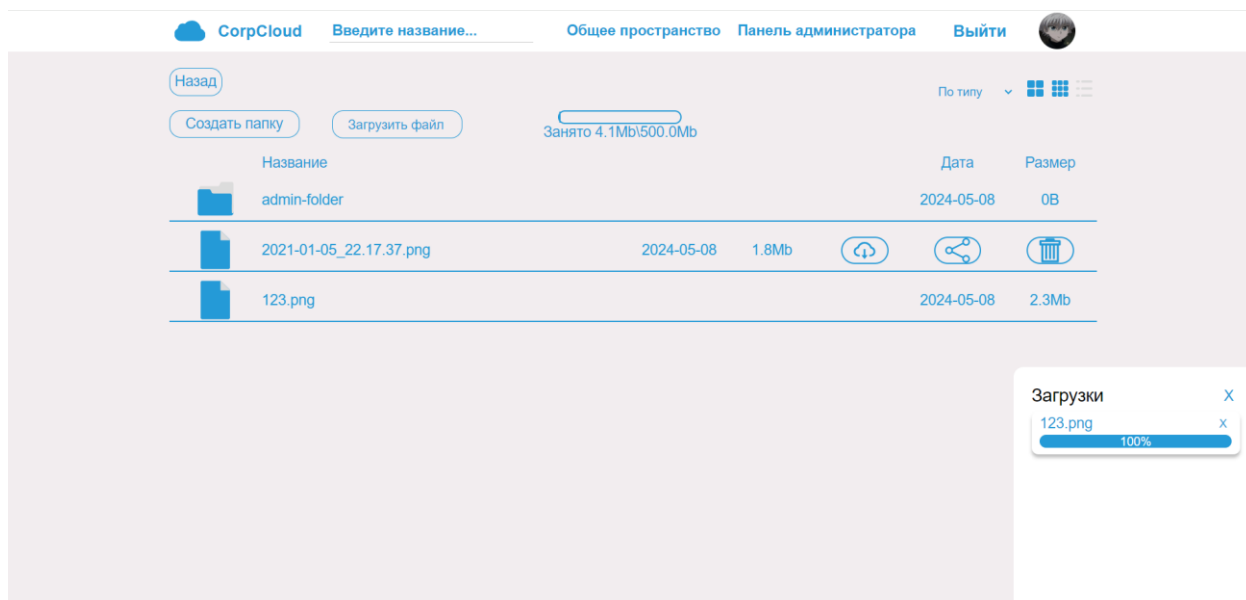


Рисунок А.3 – главная страница

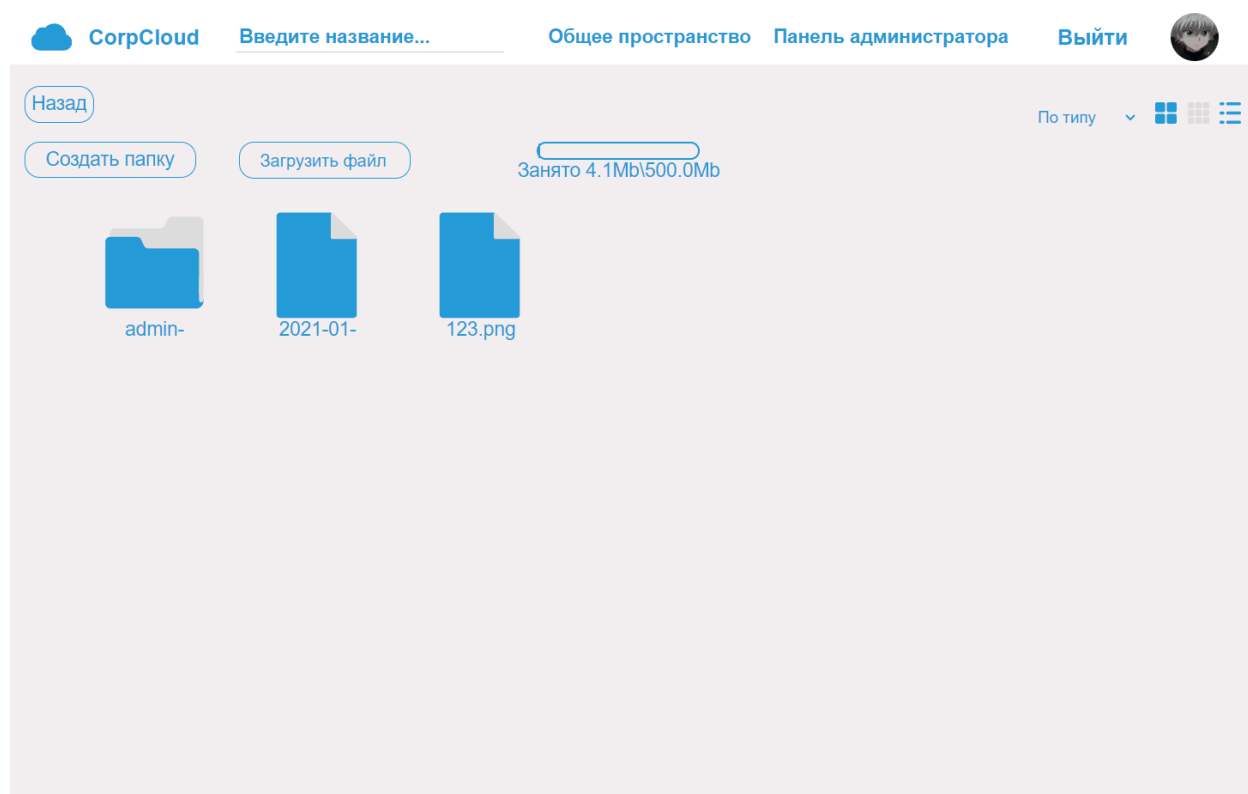


Рисунок А.4 – главная страница, расположение плитками

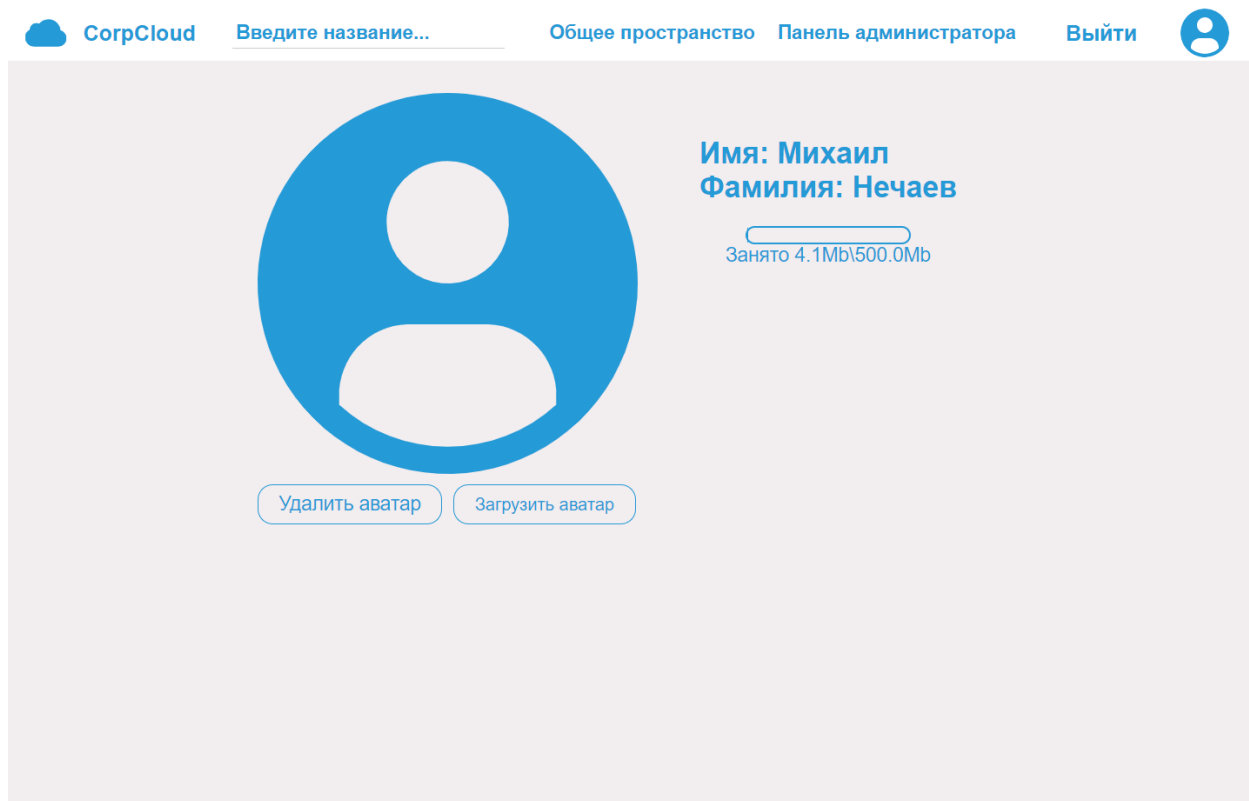


Рисунок А.5 – профиль пользователя

CorpCloud

Введите название...

Общее пространство

Панель администратора

Выйти

Создание пользователя

Введите имя...

Введите фамилию...

Введите email...

Введите пароль...

Создать

Удаление пользователя по почте

Введите email...

Удалить

Запросить пользователей

test@gmail.ru
admin@gmail.com
general@gmail.com

Рисунок А.6 – панель администратора

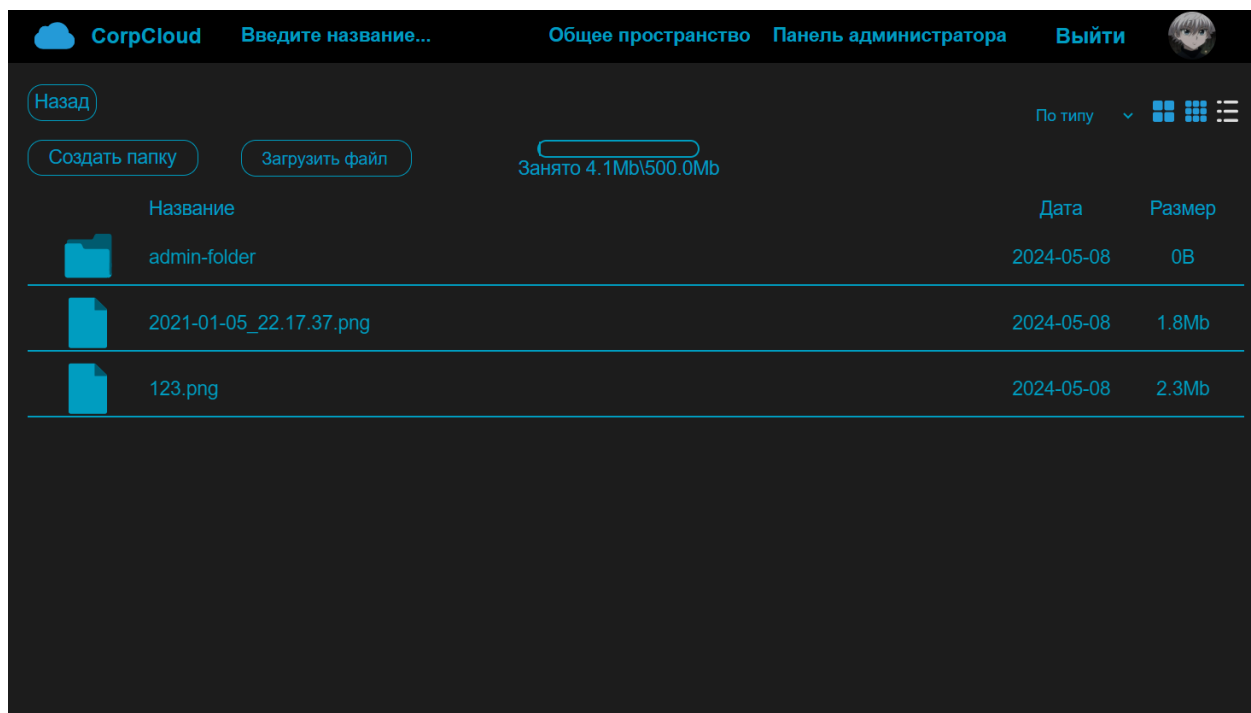


Рисунок А.7 – смена цветовой палитры